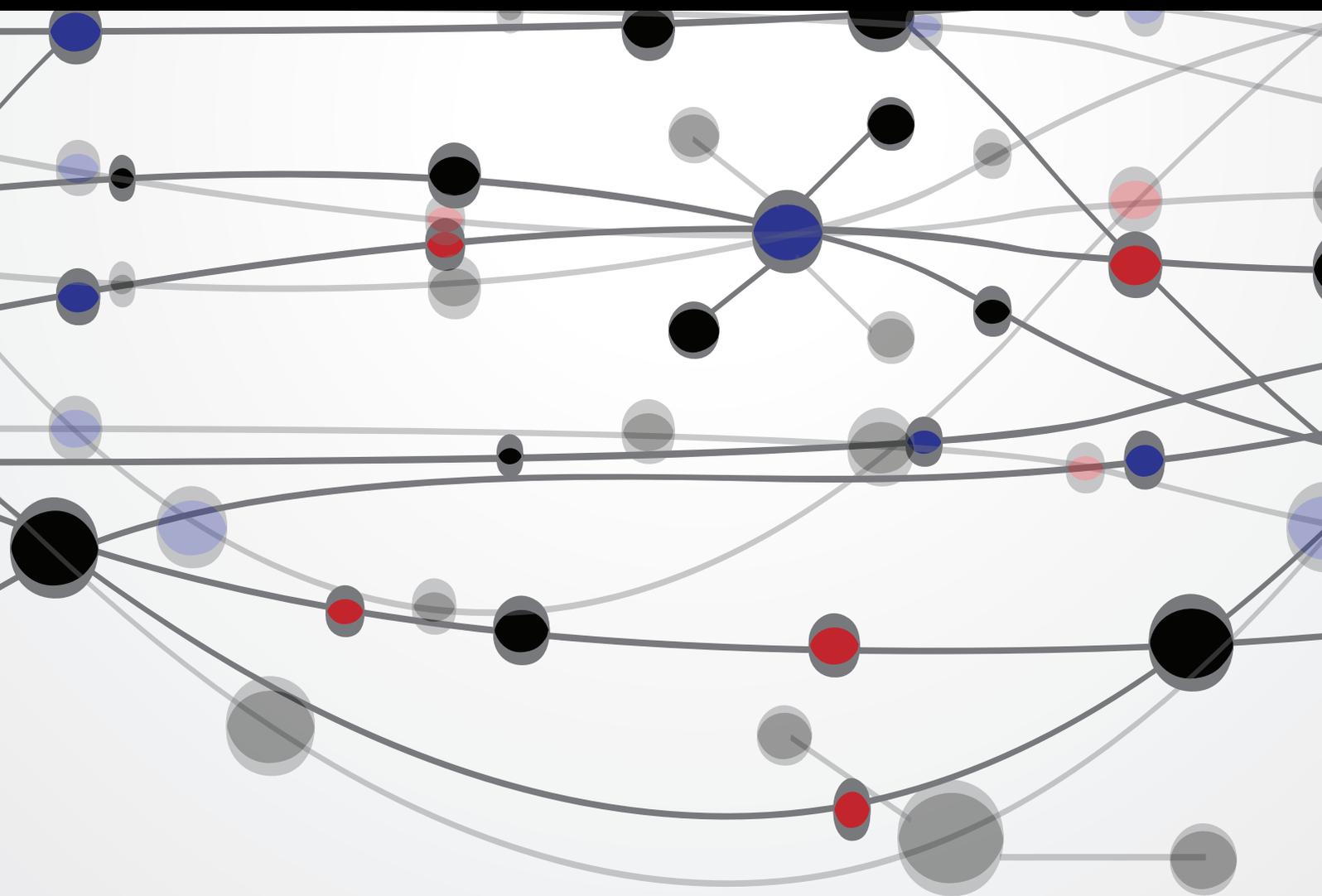


Computational Intelligence and Metaheuristic Algorithms with Applications

Guest Editors: Xin-She Yang, Su Fong Chien, and Tiew On Ting





Computational Intelligence and Metaheuristic Algorithms with Applications

The Scientific World Journal

Computational Intelligence and Metaheuristic Algorithms with Applications

Guest Editors: Xin-She Yang, Su Fong Chien,
and Tiew On Ting



Copyright © 2014 Hindawi Publishing Corporation. All rights reserved.

This is a special issue published in “The Scientific World Journal.” All articles are open access articles distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Contents

Computational Intelligence and Metaheuristic Algorithms with Applications, Xin-She Yang, Su Fong Chien, and Tiew On Ting
Volume 2014, Article ID 425853, 4 pages

Gene Network Biological Validity Based on Gene-Gene Interaction Relevance, Francisco Gómez-Vela and Norberto Díaz-Díaz
Volume 2014, Article ID 540679, 11 pages

Comparing Evolutionary Strategies on a Biobjective Cultural Algorithm, Carolina Lagos, Broderick Crawford, Enrique Cabrera, Ricardo Soto, José-Miguel Rubio, and Fernando Paredes
Volume 2014, Article ID 745921, 10 pages

Towards Enhancement of Performance of K-Means Clustering Using Nature-Inspired Optimization Algorithms, Simon Fong, Suash Deb, Xin-She Yang, and Yan Zhuang
Volume 2014, Article ID 564829, 16 pages

Congestion Control for a Fair Packet Delivery in WSN: From a Complex System Perspective, Daniela Aguirre-Guerrero, Ricardo Marcelín-Jiménez, Enrique Rodriguez-Colina, and Michael Pascoe-Chalke
Volume 2014, Article ID 381305, 12 pages

A Variable Neighborhood Walksat-Based Algorithm for MAX-SAT Problems, Noureddine Bouhmala
Volume 2014, Article ID 798323, 11 pages

Tuning of Kalman Filter Parameters via Genetic Algorithm for State-of-Charge Estimation in Battery Management System, T. O. Ting, Ka Lok Man, Eng Gee Lim, and Mark Leach
Volume 2014, Article ID 176052, 11 pages

Improved Bat Algorithm Applied to Multilevel Image Thresholding, Adis Alihodzic and Milan Tuba
Volume 2014, Article ID 176718, 16 pages

Focusing on the Golden Ball Metaheuristic: An Extended Study on a Wider Set of Problems, E. Osaba, F. Diaz, R. Carballado, E. Onieva, and A. Perallos
Volume 2014, Article ID 563259, 17 pages

Multiobjective Memetic Estimation of Distribution Algorithm Based on an Incremental Tournament Local Searcher, Kaifeng Yang, Li Mu, Dongdong Yang, Feng Zou, Lei Wang, and Qiaoyong Jiang
Volume 2014, Article ID 836272, 21 pages

Null Steering of Adaptive Beamforming Using Linear Constraint Minimum Variance Assisted by Particle Swarm Optimization, Dynamic Mutated Artificial Immune System, and Gravitational Search Algorithm, Soodabeh Darzi, Tiong Sieh Kiong, Mohammad Tariqul Islam, Mahamod Ismail, Salehin Kibria, and Balasem Salem
Volume 2014, Article ID 724639, 10 pages

A Cuckoo Search Algorithm for Multimodal Optimization, Erik Cuevas and Adolfo Reyna-Orta
Volume 2014, Article ID 497514, 20 pages

MAC Protocol for Ad Hoc Networks Using a Genetic Algorithm, Omar Elizarraras, Marco Panduro, Aldo L. Méndez, and Alberto Reyna
Volume 2014, Article ID 670190, 9 pages

An Ant Colony Optimization Based Feature Selection for Web Page Classification, Esra Saraç and Selma Ayşe Özel
Volume 2014, Article ID 649260, 16 pages

Reinforcement Learning for Routing in Cognitive Radio Ad Hoc Networks, Hasan A. A. Al-Rawi, Kok-Lim Alvin Yau, Hafizal Mohamad, Nordin Ramli, and Wahidah Hashim
Volume 2014, Article ID 960584, 22 pages

Features Extraction of Flotation Froth Images and BP Neural Network Soft-Sensor Model of Concentrate Grade Optimized by Shuffled Cuckoo Searching Algorithm, Jie-sheng Wang, Shuang Han, Na-na Shen, and Shu-xia Li
Volume 2014, Article ID 208094, 17 pages

A Novel User Classification Method for Femtocell Network by Using Affinity Propagation Algorithm and Artificial Neural Network, Afaz Uddin Ahmed, Mohammad Tariqul Islam, Mahamod Ismail, Salehin Kibria, and Haslina Arshad
Volume 2014, Article ID 253787, 14 pages

A Synchronous-Asynchronous Particle Swarm Optimisation Algorithm, Nor Azlina Ab Aziz, Marizan Mubin, Mohd Saberi Mohamad, and Kamarulzaman Ab Aziz
Volume 2014, Article ID 123019, 17 pages

Gait Signal Analysis with Similarity Measure, Sanghyuk Lee and Seungsoo Shin
Volume 2014, Article ID 136018, 8 pages

An Improved Ant Colony Optimization Approach for Optimization of Process Planning, JinFeng Wang, XiaoLiang Fan, and Haimin Ding
Volume 2014, Article ID 294513, 15 pages

Integrated Model of Multiple Kernel Learning and Differential Evolution for EUR/USD Trading, Shangkun Deng and Akito Sakurai
Volume 2014, Article ID 914641, 12 pages

A Distributed Parallel Genetic Algorithm of Placement Strategy for Virtual Machines Deployment on Cloud Platform, Yu-Shuang Dong, Gao-Chao Xu, and Xiao-Dong Fu
Volume 2014, Article ID 259139, 12 pages

An Artificial Bee Colony Algorithm for Uncertain Portfolio Selection, Wei Chen
Volume 2014, Article ID 578182, 12 pages

Fault Detection of Aircraft System with Random Forest Algorithm and Similarity Measure, Sanghyuk Lee, Wookje Park, and Sikhang Jung
Volume 2014, Article ID 727359, 7 pages

Adaptive MANET Multipath Routing Algorithm Based on the Simulated Annealing Approach,
Sungwook Kim
Volume 2014, Article ID 872526, 8 pages

A Solution Quality Assessment Method for Swarm Intelligence Optimization Algorithms,
Zhaojun Zhang, Gai-Ge Wang, Kuansheng Zou, and Jianhua Zhang
Volume 2014, Article ID 183809, 8 pages

Application of Reinforcement Learning in Cognitive Radio Networks: Models and Algorithms,
Kok-Lim Alvin Yau, Geong-Sen Poh, Su Fong Chien, and Hasan A. A. Al-Rawi
Volume 2014, Article ID 209810, 23 pages

Firefly Algorithm for Cardinality Constrained Mean-Variance Portfolio Optimization Problem with Entropy Diversity Constraint, Nebojsa Bacanin and Milan Tuba
Volume 2014, Article ID 721521, 16 pages

Cuckoo Search with Lévy Flights for Weighted Bayesian Energy Functional Optimization in Global-Support Curve Data Fitting, Akemi Gálvez, Andrés Iglesias, and Luis Cabellos
Volume 2014, Article ID 138760, 11 pages

PSO-Based Support Vector Machine with Cuckoo Search Technique for Clinical Disease Diagnoses,
Xiaoyong Liu and Hui Fu
Volume 2014, Article ID 548483, 7 pages

An Island Grouping Genetic Algorithm for Fuzzy Partitioning Problems, S. Salcedo-Sanz, J. Del Ser, and Z. W. Geem
Volume 2014, Article ID 916371, 15 pages

On the Effectiveness of Nature-Inspired Metaheuristic Algorithms for Performing Phase Equilibrium Thermodynamic Calculations, Seif-Eddeen K. Fateen and Adrian Bonilla-Petriciolet
Volume 2014, Article ID 374510, 12 pages

Cloud Model Bat Algorithm, Yongquan Zhou, Jian Xie, Liangliang Li, and Mingzhi Ma
Volume 2014, Article ID 237102, 11 pages

Evolutionary Multiobjective Query Workload Optimization of Cloud Data Warehouses,
Tansel Dokeroglu, Seyyit Alper Sert, and Muhammet Serkan Cinar
Volume 2014, Article ID 435254, 16 pages

An Investigation of Generalized Differential Evolution Metaheuristic for Multiobjective Optimal Crop-Mix Planning Decision, Oluwole Adekanmbi, Oludayo Olugbara, and Josiah Adeyemo
Volume 2014, Article ID 258749, 8 pages

Towards the Novel Reasoning among Particles in PSO by the Use of RDF and SPARQL, Iztok Fister Jr., Xin-She Yang, Karin Ljubič, Dušan Fister, Janez Brest, and Iztok Fister
Volume 2014, Article ID 121782, 10 pages



Support Vector Machine Based on Adaptive Acceleration Particle Swarm Optimization,

Mohammed Hasan Abdulameer, Siti Norul Huda Sheikh Abdullah, and Zulaiha Ali Othman

Volume 2014, Article ID 835607, 8 pages

Novel Back Propagation Optimization by Cuckoo Search Algorithm, Jiao-hong Yi, Wei-hong Xu,
and Yuan-tao Chen

Volume 2014, Article ID 878262, 8 pages

Editorial

Computational Intelligence and Metaheuristic Algorithms with Applications

Xin-She Yang,¹ Su Fong Chien,² and Tiew On Ting³

¹*School of Science and Technology, Middlesex University, London NW4 4BT, UK*

²*Strategic Advanced Research (StAR), Mathematical Modeling Lab, MIMOS Berhad, Technology Park Malaysia, Kuala Lumpur, Malaysia*

³*Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, No. 111 Ren'ai Road, HET, SIP, Suzhou, Jiangsu 215123, China*

Correspondence should be addressed to Xin-She Yang; x.yang@mdx.ac.uk

Received 4 August 2014; Accepted 4 August 2014; Published 31 December 2014

Copyright © 2014 Xin-She Yang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

1. Introduction

Nature-inspired metaheuristic algorithms have become powerful and popular in computational intelligence and many applications. There are some important developments in recent years, and this special issue aims to provide a timely review of such developments, including ant colony optimization, bat algorithm, cuckoo search, particle swarm optimization, genetic algorithms, support vector machine, neural networks, and others. In addition, these algorithms have been applied in a diverse range of applications, and some of these latest applications are also summarized here.

Computational intelligence and metaheuristic algorithms have become increasingly popular in computer science, artificial intelligence, machine learning, engineering design, data mining, image processing, and data-intensive applications. Most algorithms in computational intelligence and optimization are based on swarm intelligence (SI) [1, 2]. For example, both particle swarm optimization [1] and cuckoo search [3] have attracted much attention in science and engineering. They both can effectively deal with continuous problems [2] and combinatorial problems [4]. These algorithms are very different from the conventional evolutionary algorithms such as genetic algorithms and simulated annealing [5, 6] and other heuristics [7].

Many new optimization algorithms are based on the so-called swarm intelligence (SI) with diverse characteristics

in mimicking natural systems [1, 2]. Consequently, different algorithms may have different features and thus may behave differently, even with different efficiencies. However, It still lacks in-depth understanding why these algorithms work well and exactly under what conditions, though there were some good studies that may provide insight into algorithms [2, 8].

This special issue focuses on the recent developments of SI-based metaheuristic algorithms and their diverse applications as well as theoretical studies. Therefore, this paper is organized as follows. Section 2 provides an introduction and comparison of the so-called infinite monkey theorem and metaheuristics, followed by the brief review of computational intelligence and metaheuristics in Section 3. Then, Section 4 touches briefly the state-of-the-art developments, and finally, Section 5 provides some open problems about some key issues concerning computational intelligence and metaheuristics.

2. Monkeys, Shakespeare, and Metaheuristics

There is a well-known thought experiment, called the infinite monkey theorem, which states that the probability of producing any given text will almost surely be one if an infinite number of monkeys randomly type for an infinitely long time [9, 10]. In other words, the infinite monkeys can be expected to reproduce the whole works of Shakespeare. For example, to reproduce the text “swarm intelligence” (18 characters

including the space), for a random typing sequence of n characters on a 101-key computer keyboard, the probability of a consecutive 18-character random string to be “swarm intelligence” is $p_s = (1/101)^{18} \approx 8.4 \times 10^{-37}$, which is extremely small. However, the importance here is that this probability is not zero. Therefore, for an infinitely long sequence $n \rightarrow \infty$, the probability of reproducing the collected works of Shakespeare is one, though the formal rigorous mathematical analysis requires Borel-Cantelli lemma [9, 11].

Conversely, we can propose a finite monkey theorem without proof. For a given finite number of monkeys typing for a fixed amount of time, what is the probability of reproducing any piece of text such as this paper?

In many ways, heuristic and metaheuristic algorithms have some similarities to the infinite monkey approach. Monkeys type randomly and, ultimately, some meaningful high-quality text may appear. Similarly, most stochastic algorithms use randomization to increase the search capability. If such algorithms are executed for a sufficiently long time with multiple runs, it can be expected that the global optimality of a given problem can be reached or found. In theory, it may take infinitely long to guarantee such optimality, but, in practice, it can take many thousands or even millions of iterations. If we consider the optimality as an important piece of work of Shakespeare, the infinite monkeys should be able to reproduce or achieve it in an infinite amount of time.

However, there are some key differences between the heuristic algorithms and the infinite monkey approach. First, monkeys randomly type without any memory or learning processing, and each key input is independent of another. Heuristic algorithms try to learn from history and the past moves so as to generate new, better moves or solutions [7]. Second, random monkeys do not select what has been typed, while algorithms try to select the best solutions or the fittest solutions [5]. Third, monkeys use purely stochastic components, while all heuristic algorithms use both deterministic and stochastic components. Finally, monkey typing at most is equivalent to a random search on a flat landscape, while heuristic algorithms are often cleverly constructed to use the landscape information in combination with history (memory) and selection. All these differences ensure that heuristic algorithms are far better than the random monkey-typing approach.

In addition, metaheuristics are usually considered as a higher level of heuristics, because metaheuristic algorithms are not simple trial-and-error approaches and metaheuristics are designed to learn from past solutions, to be biased towards better moves, to select the best solutions, and to construct sophisticated search moves. Therefore, metaheuristics can be much better than heuristic algorithms and can definitely be far more efficient than random monkey-typing approaches.

3. Computational Intelligence and Metaheuristics

Computational intelligence has been in active development for many years. Classical methods and algorithms such as machine learning methods, classifications and cluster

methods, and data mining techniques are all well established, though constant improvements and refinements are being carried out. For example, neural networks and support vector machines have been around for a few decades, and they have been applied to almost every area of science and engineering [12, 13]. However, it was mainly in the 1990s when these two methods became truly popular, when the mass computer facilities become affordable with the steady increase of the computational speed.

Nowadays computational intelligence has permeated into many applications directly or indirectly. Accompanying this expansion, nature-inspired metaheuristic algorithms begin to demonstrate promising power in computational intelligence and many other areas [14]. For example, cuckoo search has been used in optimizing truss structures [15] and other applications [3], while a hybrid approach combining a two-stage eagle strategy with differential evolution can save computational efforts [16]. New algorithms emerge almost every year with a trend of speedup.

Algorithms which appeared in the last five years include bat algorithm [17], cuckoo search [3], flower pollination algorithm [18], and others, which are in addition to the popular and well-accepted algorithms such as particle swarm optimization, ant colony optimization, firefly algorithm, differential evolution, and genetic algorithms. Different algorithms have different sources of inspiration, and they can also perform differently [1, 2]. For example, among most recent, bioinspired algorithms, flower pollination algorithm (FPA), or flower algorithm (FA) for simplicity, was developed by Xin-She Yang, which has demonstrated very good efficiency in solving both single optimization and multiobjective optimization problems [18]. Both the flower algorithm and the cuckoo search use more subtle Lévy flights instead of standard Gaussian random walks [19].

However, some efficient approaches can be based on the combination of different algorithms, and the eagle strategy is a two-stage strategy combining a coarse explorative stage and an intensive exploitative stage in an iterative manner [16].

Applications can be very diverse, from structural optimization [15] to energy efficient telecommunications [20]. Detailed list of applications can be found in recent review articles [3, 17] or books [2].

4. State-of-the-Art Developments

As the developments are active and extensive, it is not possible to cover a good part of the recent advances in a single special issue. Therefore, this special issue can only provide a timely snapshot of the state-of-the-art developments. The responses to this special issue were overwhelming, and more than 100 submissions were received. After going through the rigorous peer-review process, 32 papers have been accepted for this issue. A brief summary of these papers is given below.

E. Cuevas et al. provide a study of multimodal optimization using the cuckoo search algorithm, while E. Saraç and S. A. Özel carry out web page classification using ant colony optimization and O. Elizarraras et al. obtain better performance in ad hoc network using genetic algorithms. In

addition, K. Yang et al. provided a multiobjective memetic estimation based on incremental local search, and S. Darzi et al. solve a beam enhancement problem using particle swarm optimization and other approaches, followed by the study of routing in cognitive radio ad hoc networks by H. A. A. Al-Rawi et al. and the feature extraction of flotation froth images using a combined approach of shuffled cuckoo search and BP neural networks by J.-s. Wang et al. Furthermore, A. U. Ahmed et al. provide user categorization for closed access femtocell network and N. A. Ab Aziz et al. present a synchronous-asynchronous particle swarm optimization approach.

On the other hand, S. Lee and S. Shin carry out gait signal analysis using similarity measures, and J. Wang et al. use improved ant colony optimization for process planning, while S. Deng and A. Sakurai use multiple kernel learning approach in combination with differential evolution to model EUR/USD trading problems, followed by the optimization of virtual machine deployment by Y.-S. Dong et al. and fault detection of aircraft system by random forest algorithm and similarity measures by S. Lee et al. In addition, S. Kim presents an adaptive MANET multigraph routing approach based simulated annealing. Moreover, solution quality assessment in the context of swarm intelligence has been attempted by Z. Zhang et al. and application of model and algorithms in cognitive radio networks has been carried out by K.-L. A. Yau et al.

Further algorithm developments and enhancements include the study of the mean-variance portfolio optimization by using the firefly algorithm by N. Bacanin and M. Tuba, the global support curve data fitting via the cuckoo search with Lévy flights by A. Gálvez et al., the uncertain portfolio selection by artificial bee colony by W. Chen, and fuzzy partitioning problems by island grouping genetic algorithm approach by S. Salcedo-Sanz et al. In addition, Y. Zhou et al. present a cloud model based bat algorithm, while I. Fister Jr. et al. propose novel reasoning in the context of PSO using RDF and SPARQL, followed by J.-h. Yi et al.'s detailed study of back propagation optimization by the cuckoo search algorithm.

In addition to the above applications in networks, planning, and feature selection, more applications include the diagnosis of clinical diseases using PSO-based support vector machine with cuckoo search by X. Liu and H. Fu, phase equilibrium thermodynamic calculations using nature-inspired metaheuristic algorithms by S.-E. K. Fateen and A. Bonilla-Petriciolet, query workload optimization of cloud data warehouse by T. Dokeroglu et al., and crop-mix planning decision using multiobjective differential evolution by O. Adekanmbi et al.

S. Fong et al. propose ways to enhance performance of K-means clustering by using nature-inspired optimization algorithms, while A. Alihodzic and M. Tuba carry out multilevel image thresholding by using the improved bat algorithm. In addition, F. Gómez-Vela and N. Díaz-Díaz use gene-gene interaction for gene network biological validity, while D. Aguirre-Guerrero et al. provide a fair packet delivery method with congestion control in wireless sensor network, and N. Bouhmala solves MAX-SAT problems using a variable neighbourhood approach. In parallel with the

above developments, T. O. Ting et al. tune Kalman filter parameters using genetic algorithms for battery management, and E. Osaba et al. present a golden ball algorithm for solving routing problems. Last but not least, C. Lagos et al. compare evolutionary strategies in the text of the biobjective cultural algorithm.

As we can see from the above extensive list of papers, the current studies concern a diverse range of real-world applications as well as algorithm developments and analysis.

5. Open Questions

In fact, there is still a significant gap between theory and practice. Most metaheuristic algorithms have successful applications in practice, but their mathematical analysis lags far behind. In fact, apart from a few limited results about the convergence and stability concerning particle swarm optimization, genetic algorithms, simulated annealing, and others [21, 22], many algorithms do not have theoretical analysis in the literature. Therefore, we may know that they can work well in practice, but we hardly understand why they work and how to improve them with a good understanding of their working mechanisms.

In addition, there is a well-known “no-free-lunch” theorem which concerns the average performance for solving all problems [23]. However, this theorem is only valid under strict conditions such as the assumptions of closed under permutation of solution sequences and nonrevising assumption of the search points. In fact, for coevolutionary approaches, there are potentially free lunches [24].

There are many key issues that need to be addressed in the context of computational intelligence and metaheuristic algorithms. To list all these problems may require a lengthy article to provide sufficient details for each key issue. However, we believe that the following open problems are worth emphasizing.

- (i) It still lacks a general mathematical framework for analyzing the convergence and stability of metaheuristic algorithms. There are some good results using Markov chains, dynamic systems, and self-organization theory, but a systematic framework is yet to be developed.
- (ii) Parameter tuning is still a time-consuming process for tuning algorithms. How to best tune an algorithm so that it can work for a wide range of problems is still an unsolved problem. In fact, it is a hyperoptimization problem; that is, it is the optimization of an optimization algorithm.
- (iii) How can we solve high-dimensional problems effectively? At the moment, most case studies using metaheuristic algorithms are small-scale problems. It is not clear if these algorithms are scalable to deal with large-scale problems effectively.
- (iv) Discrete problems and combinatorial optimization, especially those NP-hard problems, are still very challenging to solve. Though studies indicate that metaheuristic algorithms can be effective alternatives [2], it is still at early stage of the research. More studies are highly needed.

Obviously, these challenges also pose great opportunities for researchers. It can be expected that any progress in the above areas will provide great insight into the understanding of metaheuristic algorithms and their capabilities in solving a diverse range of problems in real-world applications.

Xin-She Yang
Su Fong Chien
Tiew On Ting

References

- [1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, Piscataway, NJ, USA, December 1995.
- [2] X. S. Yang, *Cuckoo Search and Firefly Algorithm: Theory and Applications*, vol. 516 of *Studies in Computational Intelligence*, Springer, Heidelberg, Germany, 2014.
- [3] X.-S. Yang and S. Deb, "Cuckoo search: recent advances and applications," *Neural Computing and Applications*, vol. 24, no. 1, pp. 169–174, 2014.
- [4] M. K. Marichelvam, T. Prabaharan, and X. S. Yang, "Improved cuckoo search algorithm for hybrid flow shop scheduling problems to minimize makespan," *Applied Soft Computing*, vol. 19, no. 1, pp. 93–101, 2014.
- [5] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.
- [6] S. Kirkpatrick, C. D. Gellat, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
- [7] P. Judea, *Heuristics*, Addison-Wesley, New York, NY, USA, 1984.
- [8] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.
- [9] G. Marsaglia and A. Zaman, "Monkey tests for random number generators," *Computers & Mathematics with Applications*, vol. 26, no. 9, pp. 1–10, 1993.
- [10] A. Gut, *Probability: A Graduate Course*, Springer Texts in Statistics, Springer, Berlin, Germany, 2005.
- [11] A. V. Prokhorov, "Borel-Cantelli lemma," in *Encyclopedia of Mathematics*, M. Hazewinkel, Ed., Springer, Heidelberg, Germany, 2002.
- [12] C. M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, Oxford, UK, 1995.
- [13] V. N. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, NY, USA, 1995.
- [14] X. S. Yang, *Nature-Inspired Optimization Algorithms*, Elsevier, London, UK, 2014.
- [15] A. H. Gandomi, S. Talatahari, X.-S. Yang, and S. Deb, "Design optimization of truss structures using cuckoo search algorithm," *Structural Design of Tall and Special Buildings*, vol. 22, no. 17, pp. 1330–1349, 2013.
- [16] X.-S. Yang and S. Deb, "Two-stage eagle strategy with differential evolution," *International Journal of Bio-Inspired Computation*, vol. 4, no. 1, pp. 1–5, 2012.
- [17] X.-S. Yang, "Bat algorithm: literature review and applications," *International Journal of Bio-Inspired Computation*, vol. 5, no. 3, pp. 141–149, 2013.
- [18] X.-S. Yang, M. Karamanoglu, and X. S. He, "Flower pollination algorithm: a novel approach for multiobjective optimization," *Engineering Optimization*, vol. 46, no. 9, pp. 1222–1237, 2014.
- [19] I. Pavlyukevich, "Lévy flights, non-local search and simulated annealing," *Journal of Computational Physics*, vol. 226, no. 2, pp. 1830–1844, 2007.
- [20] T. O. Ting, S. F. Chien, X. S. Yang, and S. H. Lee, "Analysis of quality-of-service aware orthogonal frequency division multiple access system considering energy efficiency," *IET Communications*, vol. 8, no. 11, pp. 1947–1954, 2014.
- [21] D. Greenhalgh and S. Marshall, "Convergence criteria for genetic algorithms," *SIAM Journal on Computing*, vol. 30, no. 1, pp. 269–282, 2000.
- [22] W. J. Gutjahr, "Convergence analysis of metaheuristics," *Annals of Information Systems*, vol. 10, no. 1, pp. 159–187, 2010.
- [23] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.
- [24] D. H. Wolpert and W. G. Macready, "Coevolutionary free lunches," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 6, pp. 721–735, 2005.

Research Article

Gene Network Biological Validity Based on Gene-Gene Interaction Relevance

Francisco Gómez-Vela and Norberto Díaz-Díaz

School of Engineering, Pablo de Olavide University, 41013 Seville, Spain

Correspondence should be addressed to Francisco Gómez-Vela; fgomez@upo.es

Received 25 April 2014; Accepted 11 July 2014; Published 8 September 2014

Academic Editor: Su Fong Chien

Copyright © 2014 F. Gómez-Vela and N. Díaz-Díaz. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In recent years, gene networks have become one of the most useful tools for modeling biological processes. Many inference gene network algorithms have been developed as techniques for extracting knowledge from gene expression data. Ensuring the reliability of the inferred gene relationships is a crucial task in any study in order to prove that the algorithms used are precise. Usually, this validation process can be carried out using prior biological knowledge. The metabolic pathways stored in KEGG are one of the most widely used knowledgeable sources for analyzing relationships between genes. This paper introduces a new methodology, GeneNetVal, to assess the biological validity of gene networks based on the relevance of the gene-gene interactions stored in KEGG metabolic pathways. Hence, a complete KEGG pathway conversion into a gene association network and a new matching distance based on gene-gene interaction relevance are proposed. The performance of GeneNetVal was established with three different experiments. Firstly, our proposal is tested in a comparative ROC analysis. Secondly, a randomness study is presented to show the behavior of GeneNetVal when the noise is increased in the input network. Finally, the ability of GeneNetVal to detect biological functionality of the network is shown.

1. Background

Modeling process occurring in living organisms is one of the main goals in bioinformatics [1–4]. Gene networks (GNs) have become one of the most important approaches to discover which gene-gene relationships are involved in a specific biological process.

A GN can be represented as a graph where genes, proteins, and/or metabolites are represented as nodes and their relationships as edges [1].

It is important to note that GNs can vary substantially depending on the model architecture used to infer the network. These models can be categorized into four main approaches according to Hecker et al. [1]: correlation [5, 6], logical [7–9], differential equation-based, and Bayesian networks [10, 11]. These approaches have been broadly used in bioinformatics. For example, Rangel et al. [12] used linear modeling to infer T-cell activation from temporal gene expression data, or Faith et al. [13] adapted correlation and

Bayesian networks to develop a method for inferring the regulatory interactions of *Escherichia coli*.

Once a model has been generated, it is very important to assure the algorithm reliability in order to demonstrate its efficacy. The quality of the algorithm(s) can be measured by applying so-called synthetic data [14] and/or by using prior biological knowledge [15]. Synthetic data approaches can be used to analyze the performance of the GN inference algorithm, whereas a study of biological validity is supported by real data.

Synthetic data methods produce an artificial data set according to a previously known network. The values of the simulated gene expression are stored in a data set and used as input for the GN inference algorithm. Finally, the performance of the algorithm is tested comparing both GNs. Currently, this process can be carried out using different tools as GeneNetWeaver [16] or SynTREN [17].

Although this approach is commonly used for comparing inference algorithms, it can not fully reproduce the internal

features of real biological processes. This drawback means they are not suitable for the validation of the inferred models, from a biological point of view.

To address this issue, comparison with prior biological knowledge has been proposed [18, 19]. Currently, there are a number of different available biological repositories where the Kyoto encyclopedia of genes and genomes (KEGG) is one of the most widely used for analyzing relationships between genes [20, 21]. KEGG's metabolic pathways contain knowledge about different biological processes. These pathways are represented as a graph where nodes represent genes, enzymes, or compounds (i.e., carbohydrates, lipids, and amino acids) and edges encode relationships, reactions, or interactions between the nodes. The pathways contained in the KEGG database represent the actual knowledge of molecular interaction and reaction networks for metabolism, genetic information processing, environmental information processing, cellular processes, and human disease. They provide useful structured information for gene network validation. For example, C. Li and H. Li [15] used KEGG transcriptional pathways to perform a network analysis of the glioblastoma microarray data, or Ko et al. [22] tested a new Bayesian network approach using gene-gene relationships stored in KEGG. In this line we proposed a GN validation framework based on a direct comparison between a gene network and KEGG pathways [23].

The aforementioned approaches, hereafter called the classical use of KEGG, present three major shortcomings: (a) not all the biological information is used, (b) only strong gene-gene relationships are considered, and (c) the current biological knowledge is not complete.

Gene-gene relationships are only usually considered by metabolic pathways based GN validation approaches. Hence, all other biological information provided by pathways is ignored, such as gene-compound or compound-compound relationships (see Table 1). For example, Wei and Li [24] only used human gene-gene interactions stored in the KEGG pathways when performing simulation studies, excluding gene-compound and compound-compound relationships. Or Zhou and Wong [25] used the relationship between KEGG gene pairs (mainly PPrel and ECreI) to study protein-protein interaction data sets.

Furthermore, current GN validation approaches are not entirely accurate as they only consider strong relationships between genes (direct gene-gene interactions), leaving weaker relationships to one side [4].

In addition, the use of prior biological knowledge could present another important lack, the current limitations of the biological databases. As described by Dougherty and Shmulevich [2], biological knowledge has some intrinsic limitations in the sense that they depend inherently on the nature of scientific knowledge. Others are contingent depending on the present state of knowledge, including technology. Current validation methods use these biological databases in order to classify the inferred relationships as true or false positives. Due to the intrinsic problem of the biological databases, it is not possible to argue that these false positives are actually caused by a bad prediction from the inference methods or because of incomplete knowledge.

This paper proposes a new methodology, GeneNetVal, to analyze the biological validity of a gene network by utilizing the biological information stored in KEGG by weighting the gene-gene relationships. GeneNetVal uses different types of relationships contained in KEGG pathways (gene-gene, gene-compound, and compound-compound), carrying out an exhaustive and complete conversion of a pathway into a gene network. The network obtained will be used as a gold standard in comparison with the input network. Moreover, a novel matching distance is proposed. This measure, based on gene-gene interaction relevance, takes into account the concept of weak relationships between a pair of genes to present a set of nondeterministic indices with different levels of accuracy. Thus, we do not categorically accept or refuse a gene-gene relationship, but a weighted value is assigned according to distance of those genes in the pathway. Through these values we generated a new gene network validity measure and mitigate the problem of the incomplete biological knowledge.

2. Methods

In this section, the GeneNetVal methodology and also the methods used to perform the experiments will be presented. These methods will be used in Results and Discussion section.

2.1. GeneNetVal Methodology. As already stated, the two-step methodology proposed, GeneNetVal, is based on KEGG metabolic pathways and summarized in Figure 1. In the first step, a complete conversion of a metabolic pathway into a gene association network is carried out. In the second step, the biological validity of a GN is determined. In order to do this, a novel matching distance between networks is used.

2.1.1. Step One: From Metabolic Pathways to Gene Association Networks. KEGG database stores knowledge about many different organisms, but we only need the information pertaining to the network to be analyzed. Hence, only the KEGG metabolic pathways for the same organism of the input network are considered. This is represented in Figure 1, where all pathways of the organism o are extracted.

These pathways are converted into gene association networks where all types of pathway relationships (see Table 1), including gene-gene (PPrel, ECreI, and GRel), gene-compound (PCrel), and compound-compound, are used.

As stated previously, a metabolic pathway is composed of different types of nodes (genes or other compounds) while genes are only used in gene networks. This difference exhibits that direct comparison between them is unreliable based on the information containing different elements. This difference is overcome by increasing the abstraction level of the pathways. Concretely, each pathway is converted into a gene association network, the highest level of abstraction for reconstruction of gene regulatory processes as it is described by Martínez-Ballesteros et al. [30]. This conversion process is represented in Figure 2 and explained below.

Firstly, all the compound nodes presented in the pathway are removed. However, gene nodes are conserved along with

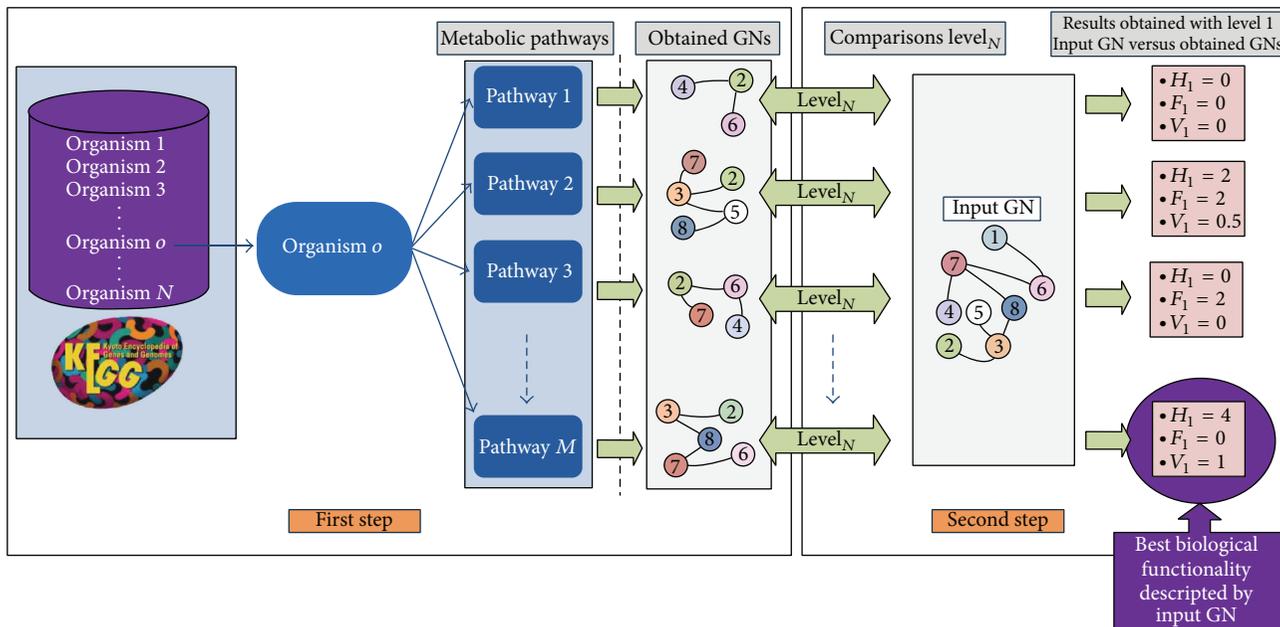


FIGURE 1: A schematic representation of GeneNetVal methodology. In the first step, organism o’s information is extracted from KEGG database. Each of the M metabolic pathways is processed to obtain M gene networks. In the second step, M evaluations of the input network are carried out. Note that the results presented were obtained by applying our approach at level 1.

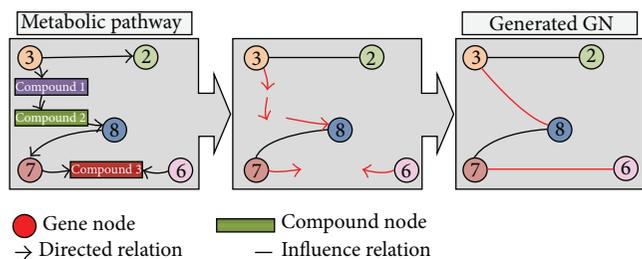


FIGURE 2: The simplest conversion example. In the first substep the compound nodes and the direction of the relationship edges are removed. In the second substep, new association relationships are established.

TABLE 1: Possible interactions presented in KEGG are classified into two groups, gene relationships and others relationships. First the four types of gene relationships are presented.

KEGG relationships	Description
Gene relationships	
ECrel	Enzyme-enzyme relation
PPrel	Protein-protein interaction
GERel	Gene expression interaction
PCrel	Protein-compound interaction
Others	
compound-compound	Compound-compound interaction

their relationships of influence (nondirect edges), be they PPrel, ECrel, or GERel. The PCrel, compound-compound, and others relationships are processed in different way.

The compound nodes located between two genes carry information from one gene to another. They act as a bridge between the genes, so these two gene nodes should be related. Based on this, after removing the compound nodes, new undirected gene-gene relationships will be created. These relationships are established between each pair of genes that were previously associated with the same compound node.

Figure 2 shows the conversion process from “Pathway M ” (Figure 1) to a gene network in detail. For example, genes 3 and 8 are associated with a compound node in the pathway but there is no direct relationship between them. However, the information pertaining to this indirect gene-gene influence should be taken into account so that a new influence relationship between genes is created. Similarly, a relationship is generated between genes 6 and 7.

The conversion presented in Figure 2 is a simple example; pathways are often more complex. In a pathway, multiple genes are likely related to the same compound node, or the chemical compounds are transferred by two or more genes/enzymes. These two cases should be considered to carry out an exhaustive conversion. In the first type, multiple genes somehow interact with the same compound (substrate of a chemical reaction, product, etc.). This biological information is preserved creating new relationships (see Figure 3(a)). In the second group, the genes responsible for transferring the compounds should be related in the new GN, since they actually interact with the chemical compounds simultaneously. Hence, new relationships between these genes are included (see Figure 3(b)).

2.1.2. Second Step: Biological Validity. In the second step, the metabolic pathways are used as biological knowledge to

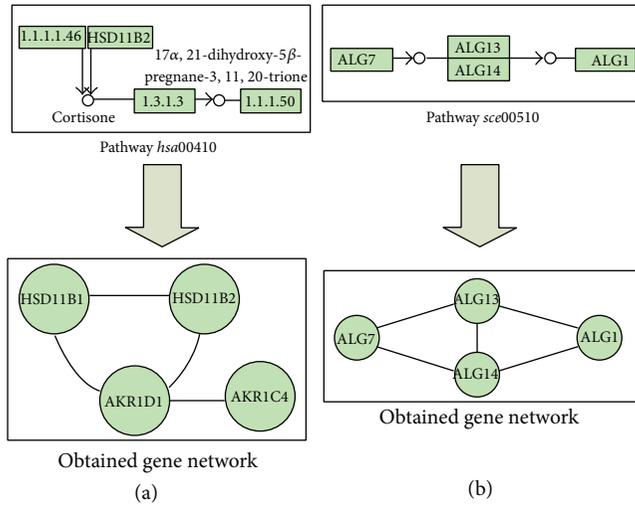


FIGURE 3: Two portions of real KEGG pathways with multiconnected nodes. (a) contains a fragment of the pathway *hsa00410* where three genes are connected to the same compound. In the process of conversion to a gene network, new relationships between these genes are created. (b) shows a fragment of the *sce00510* pathway, illustrating how a compound is transferred by two genes. When the gene network is created, these two genes must be connected (as shown in the figure). Note genes HSD11B1, AKR1D1, and AKR1C4 (a) correspond to enzymes “1.1.1.146,” “1.2.1.3,” and “1.1.1.50,” respectively.

evaluate the input network. Usually, the literature applies a scoring methodology [1, 27, 29] to evaluate an inferred model using prior knowledge, be it synthetic or biological data. Based on this idea and on the notion of the strong and weak relationships in GNs [4], the authors have developed a novel measure for evaluating the validity of an input network that is based on the relevance of the gene-gene interactions stored in KEGG.

Let $G_I = \{N_I, E_I\}$ and $G_P = \{N_P, E_P\}$ be two graphs, where (N_I, N_P) represent the nodes of the graphs and (E_I, E_P) represent the edges (gene-gene relationships). The validity of the input graph (G_I), according to the biological information of pathway P represented in the graph G_P , is measured as the difference between both graphs at certain level of distance.

Definition 1 (Level). Let a graph $G = \{N, E\}$ and two nodes $g_\alpha, g_\beta \in N$. The level of the relationship between (g_α, g_β) is calculated as the number of edges between nodes g_α and g_β in G .

For example, in Figure 4, the relationship between nodes 3 and 7 in G_P has a level of 2 because there are two edges between these nodes.

Definition 2 (Hits at level l (Hit_l)). The number of edges where the level between the nodes directly connected in G_I is l in G_P .

An example of Hit_1 and Hit_2 can be found in Figure 4, where the edge between genes 3 and 2 represents the Hit_1

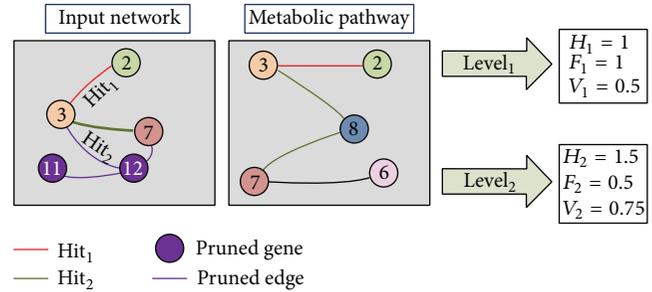


FIGURE 4: An example of the comparison using level 1 and level 2. Examples of Hit_1 and Hit_2 are presented. The purple nodes and their relationships are pruned for this specific evaluation because they do not belong to the metabolic pathway.

and the edge between 3 and 7 is Hit_2 . Obviously the greater the distance between nodes, the lower the relevance of the evaluated relationship. Thus, the new matching distance provides two weighted indices through comparison with the selected level.

Definition 3. Cumulative hits at level n , \mathcal{H}_n , can be defined as the weighted sum of correctly inferred edges at level n in G_I according to the information presented in G_P . Consider

$$\mathcal{H}_n(G_I, G_P) = \sum_{l=1}^n \frac{\text{Hit}_l}{l}, \quad (1)$$

where \mathcal{H}_n denotes the sum of edges that were correctly inferred weighted by their relevance in the network with distance (level) n .

Figure 4 presents an example of calculation of H_1 and H_2 .

Definition 4. Cumulative failures at level n , \mathcal{F}_n , can be defined as the number of incorrect inferred edges at level n in G_I

$$\mathcal{F}_n(G_I, G_P) = \|E_I\| - \mathcal{H}_n(G_I, G_P), \quad (2)$$

where $\|E_I\|$ is the number of edges in G_I . Thus, \mathcal{F}_n denotes the number of edges that were not correctly inferred in the network with distance (level) n .

Figure 4 shows an example of calculation of F_1 and F_2 . At level 1, the graph presents one cumulative failure because of the genes 3 and 7, which are directly connected in G_I and have a distance of 2 in G_P . As the interaction between 3 and 7 is weak (hit of level 2), the value of the cumulative failure level 2 is 0.5. Accordingly, the validity measure can be defined.

Definition 5. The validity (GeneNetVal measure) of graph G_I according to G_P level n , \mathcal{V}_n , is defined as the proportion of correctly inferred edges at level n in G_I . Consider

$$\mathcal{V}_n(G_I, G_P) = \frac{\mathcal{H}_n(G_I, G_P)}{\mathcal{H}_n(G_I, G_P) + \mathcal{F}_n(G_I, G_P)} = \frac{\mathcal{H}_n(G_I, G_P)}{\|E_I\|}. \quad (3)$$

This measure ranges between 0 and 1, where 0 is the lowest validity value and 1 the highest. The validity measure estimates the ratio of correctness of G_I with respect to G_P .

The biological validity is obtained as the proportion of positive prediction according to the cumulative hits and failures. This is the principal measure obtained by our methodology to rate the quality of a GN.

2.2. ROC Study. A receiver operating characteristic (ROC) analysis will be presented in the Results section. The goal of this study is to compare the performance of different gene network validity approaches, evaluating real networks against random networks (without biological sense). The three networks will be used in the experiment, attempt to encompass the regulation of a large number of functional processes in yeast. Hence, we have assumed that these networks contain biological meaning of each functional process described in the KEGG pathways (they are functionally complex networks).

Therefore, the evaluation of these networks should produce relevant validity results for each of the pathways considered. In contrast, the biological validity of random networks ought to yield poor results because, in fact, they should not contain biological meaning.

A validity threshold (T) has been used to decide if the input network has relevant information for each selected pathway. T denotes the minimum validity value for a network with a specific pathway to be considered as valid value. In order to generate the ROC curve for each experiment, we have used 101 different T values (from 0 to 1). A confusion matrix is obtained for each iteration. If the validity value obtained for a pathway exceeds the T value, the input network is classified as a positive (true positive or false positive, depending on whether the input network is a real network or a random network). If the obtained value is lower, the input network is described as a negative (true negative or false negative). With this idea, for each iteration the indexes are computed for the confusion matrix.

Hence, it is possible to calculate 101 confusion matrices and 101 true positive rates (TPR) and false positive rates (FPR) values to draw the curve.

Figure 5 provides a toy example showing the entire process (only for one random network). It offers a comparison between the results obtained by a real network and the results obtained by a random network. With the validity values obtained for both networks (Figure 5(a)), different confusion matrices were generated according to different thresholds, only 4 thresholds in this example (Figure 5(b)). Thus, for each iteration it is possible to obtain the values of TPR and FPR (Figure 5(c)). With these values, the ROC curve is finally represented (Figure 5(d)).

It is important to note that the results presented in Figure 6 are average values for a sample of random networks.

2.3. Selecting Functional Description with GeneNetVal. The specific functionality of the input network could be studied in accordance with the biological process information store

in a specific KEGG pathway. A metabolic pathway represents a model of a particular biological process. Different sets of genes are involved in different pathways. This should be considered if a functional assessment of the input network is performed. If a pathway contains a set of genes, this set is annotated to the pathway's biological function. Hence, any information from the input network that does not belong to the specific biological process will be not taken into account for this validation. Note that these relationships should not be considered as a failure because actually there is no information to classify the validity of the interactions from genes in the input network that are not present in metabolic pathways.

This pruning process, which is depicted in Algorithm 1, entails removing any edge from the input network if the corresponding genes are not present in the specific pathway. The input network will suffer a different pruning for each pathway. Through this pruning, the input network can be evaluated independently for each process. An example of this pruning is shown in Figure 4 where the purple edges are removed for the comparison with the pathway.

After pruning, the comparisons with each pathway will show the validity measure. The functionality described by the pathway with the highest value of \mathcal{V}_n (GeneNetVal measure) will be the functionality that best fits the input network. A high \mathcal{V}_n value means that the input network fully or partially describes the functionality that is described by that particular metabolic pathway.

Hence, M different comparisons have been carried out in Figure 1, where the highest value was generated by the gene network extracted from "pathwayM."

It is also possible for the input network to contain information about more than one specific biological process. Alternatively, the biological processes are usually interrelated (e.g., the cell cycle and the meiosis). An example of this situation in Figure 1 might be the comparison between the gene network from "pathway2" and the input network. In that case, the highest values of the validity measure could be considered, to determine which processes are better described.

3. Results and Discussion

The performance of our proposal was tested through three experiments using different types of networks. Firstly our proposal was compared with the classical use of KEGG. A ROC analysis of different distance level of GeneNetVal and precision measure were carried out. The behavior of the method proposed with different noise level is tested in the second experiment. Finally, the ability of GeneNetVal to detect the biological functionality encoded in a input network is analyzed in the third experiment.

3.1. ROC Analysis. The ROC analysis was performed to show the improvement achieved by our approach over those that only consider direct gene-gene relationships [24, 25], along with its robustness against information without biological meaning (see Section 2.2).

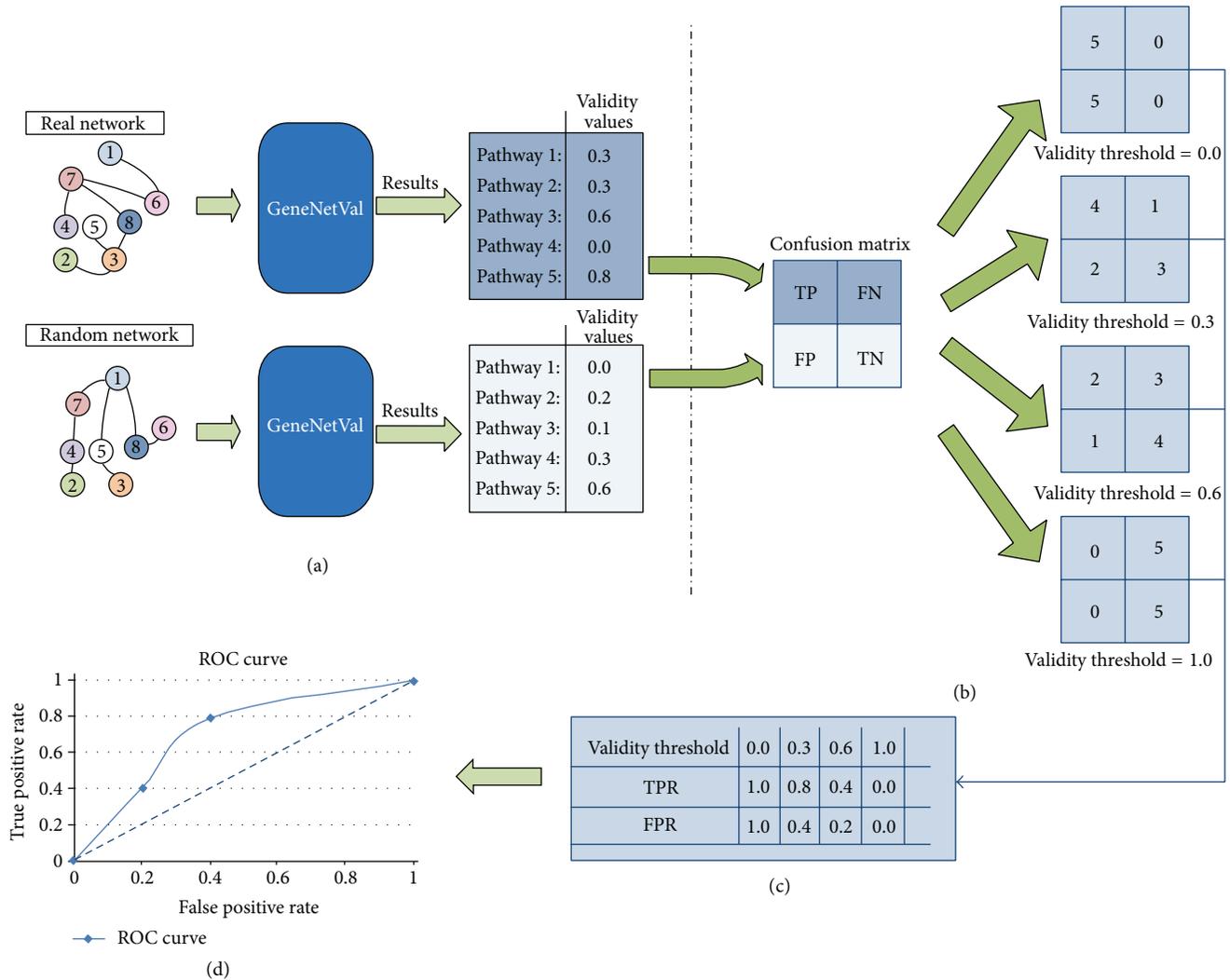


FIGURE 5: Representation of a toy example for the ROC study performed. (a) represents the GeneNetVal process, where the validity values for both networks are obtained. In (b) the confusion matrices are obtained. The TPR and FPR values are presented in (c). Finally the ROC curve is depicted in (d).

ROC analysis has been widely used in the literature [31, 32] because it is able to score the performance of classifiers and rankers as a trade-off between a true positive rate and false positive rate. Additionally, the area under the ROC curve (AUC) is presented, as it provides information about the level of randomness of the approach.

For this study three complex and contrasting yeast gene networks with different types of gene relationships were used. A protein-protein interaction network was used by Batada et al. [33] in the analysis of highly connected proteins in a network (hubs). The network resulting of selecting the protein-protein and protein-DNA interactions of the *Saccharomyces Genome Database* (SGD) [34] provides an access to the complete *Saccharomyces cerevisiae* (yeast) genomic sequence. And, finally, the network was presented by Lee et al. [35] (YeastNet v.2) which combines protein-protein, protein-DNA, coexpression, phylogenetic conservation, and literature information.

For every input network explained above, two different topologies of random networks were considered: pure random and scale-free. This latter topology is used since biological networks usually follow it [36, 37].

The sample size for each input network and topology was calculated with a confidence interval of 95% for an infinite population of networks [38]. Hence, a sample size of 385 random networks was used. Pure random networks were designed to have the same node and edge size as the input network, but gene-gene relationships were randomly generated. Scale-free networks were generated using the open source library JGraphT, with the same nodes as well. To use information stored in KEGG, we extracted the KGML files of yeast pathways using the KEGG API.

The results of the analysis are represented in Figure 6, where each row represents the study of a different input network. The left column in the figure represents the study

```

INPUT  $G_I = \{N_I, E_I\}$ : input network
 $G_P = \{N_P, E_P\}$ : network obtained from the pathway
OUTPUT  $G_R = \{N_R, E_R\}$ : network pruned
begin
 $\langle N_R, E_R \rangle := \langle N_I, E_I \rangle$ 
for each edge  $e \in E_I$  do
  for each node  $n$  of  $e$  do
    if  $n \notin N_P$  then
       $N_R = N_R - \{n\}$ 
       $E_R = E_R - \{e\}$ 
    end if
  end for
end for
end

```

ALGORITHM 1: Pruning process.

for pure random topology, and the rightmost shows the scale-free topology. Each graph contains five lines that encode the behavior of GeneNetVal considering the distance levels from one to four and the precision measure [30, 39] for the classical use of KEGG. In total, more than 11000 (3 input networks \times 2 topologies \times 5 measures/levels \times 385 networks) evaluations were carried out.

The ROC curves show that the results of the three networks follow a similar pattern for both topologies. Particularly striking is the distance between the point (1, 1) and the one above. FPR is 1 for a threshold equal to zero (see Section 2.2 for more details) but represents a very low value for the next checkpoint (threshold = 0.01). This could be due to the fact that the use of KEGG as gold standard is very effective detecting interactions with no biological meaning.

For some levels the lines do not start at point (0, 0) (Figures 6(b) and 6(d)). This is because some KEGG pathways do not contain many interactions (e.g., *sce00062* pathway contains only 5); so a random network might contain those gene relationships at a certain distance level.

Regarding the values obtained for the area under the curve (AUC), it is important to note that the major is the number of type of relationships considered in the network the better the methodology performs. The best results are obtained by Lee's network [35] which combines four different types of relationships. The second best result is generated using SGD, while Batada's network presents the worst result. This makes sense since KEGG pathways gather biological data from various contrast sources.

Comparing the classical use of KEGG with level 1 of our proposal, which only differs on how the pathways information is managed, is possible to argue that the conversion proposed produces significant improvement in AUC. Level 1 produces better result in all cases. For example, the AUC value of 0.88 is increased to 0.92 in SGD for scale-free topology (Figure 6(d)). Furthermore, it is possible to improve AUC by increasing the distance level in the comparison. The best

result is shown by level 2, while levels 3 and 4 perform worse than levels 1 and 2.

The results presented show that GeneNetVal is capable of detecting gene relationships with and without biological meaning. Furthermore, the methodology presents a significant improvement compared to the classical approach (precision) for all levels studied. In particular, the best performance is obtained by level 2 for all the experiments.

Finally, in spite of the fact that biological databases are crucial information sources for evaluating results obtained in any study, they have some limitations. These limitations are intrinsic to all of them, in the sense that they depend inherently on the nature of scientific knowledge; others are contingent, depending on the present state of knowledge, including technology [2, 40]. Such limitations can include incorrect event or entity labels, missdirections in the relationships, absence of associations, and other ambiguities. Consequently, the performance of prior knowledge-based methods could be affected by these limitations, including our approach. In particular, GeneNetVal could be affected for incorrect event or entity labels and also for the absence of association in the metabolic pathways in terms of bad classification of relationships (incorrect hit or failure). Despite this fact, it is worth mentioning that the classical approaches are also affected for the problems presented above. In this sense, GeneNetVal presents a more robust performance than the classical approaches, since the use of indirect relationships mitigates these problems. This affirmation is supported by the results presented in this ROC analysis, where GeneNetVal performs better than the classical approach even though the same databases (containing the same lacks) are used in both methods.

3.2. Randomness Study. Despite the fact that in the ROC analysis section it was shown that GeneNetVal is better distinguishing real networks from random networks than a classical approach extracted from the literature, in this section the behavior of the methodology to the progressive inclusion of noise will be shown.

Concretely, we have carried out the study for all of the yeast networks which were previously presented in the paper (Batada, Lee, and SGD networks). These input networks were changed increasing randomness in their gene relationships. Hence, in a loop process composed of 10 iterations, the random relationships added to the networks were increased in 10% at each iteration. In the same way a 10% of the original relationships were removed. To avoid bias, this was done 385 times (sample size with a confidence interval of 95% assuming an infinite population of random networks) [38]. Therefore, 15360 (385 networks \times 10 iterations \times 4 original networks) different random networks were analyzed.

According to the results presented in the ROC analysis section, the validity value level 2 was considered in this experiment. As gold standard, we have used the pathway *sce04111* (yeast cell cycle) since it is one of the most studied pathways from yeast [41–43]. The results averages are summarized in Figure 7.

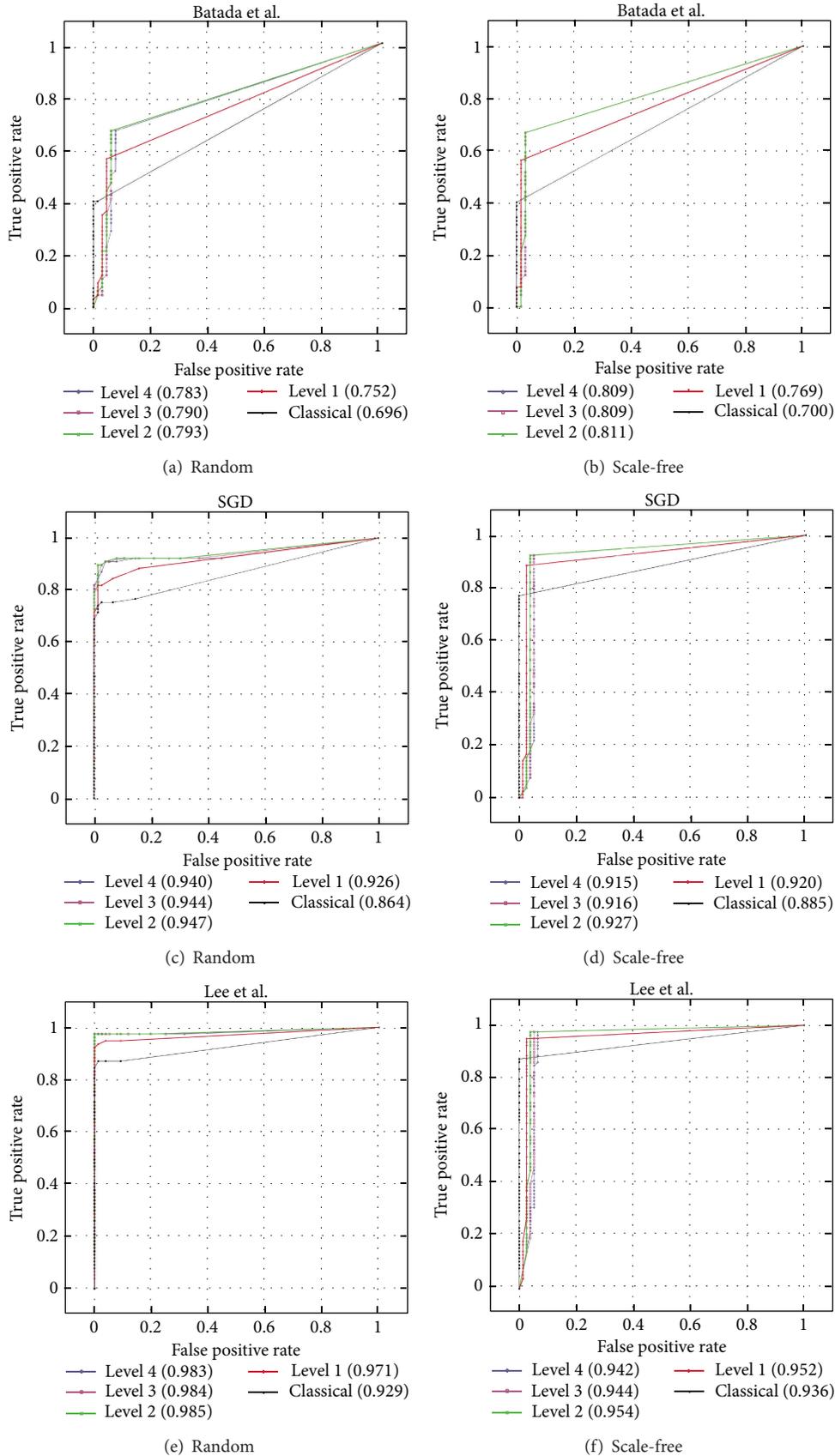


FIGURE 6: ROC analysis of our methodology using some yeast networks. For this analysis two different topologies were used: pure random and scale-free topology.

TABLE 2: Most representative $Level_2$ results found using our method assessing some yeast cell cycle networks. The results from GeneNetVal and the Hits levels 1 and 2 obtained from the evaluation are shown. Note that all similarities found with the KGN are also found for the cell cycle pathways network as expected.

	GeneNetVal ₂	Hit ₁	Hit ₂	GeneNetVal ₂	Hit ₁	Hit ₂
		Nariai et al. [26]			Gallo et al. [27]	
KGN	0.852	35	5	0.65	4	5
sce04111	0.852	35	5	0.65	4	5
sce04113	0.81	22	3	0.4	2	2
		Bulashevskaya and Eils [28]			Ponzoni et al. [29]	
KGN	0.542	4	5	0.647	6	10
sce04111	0.458	4	3	0.618	6	9
sce04113	0.2	0	2	0.417	1	1

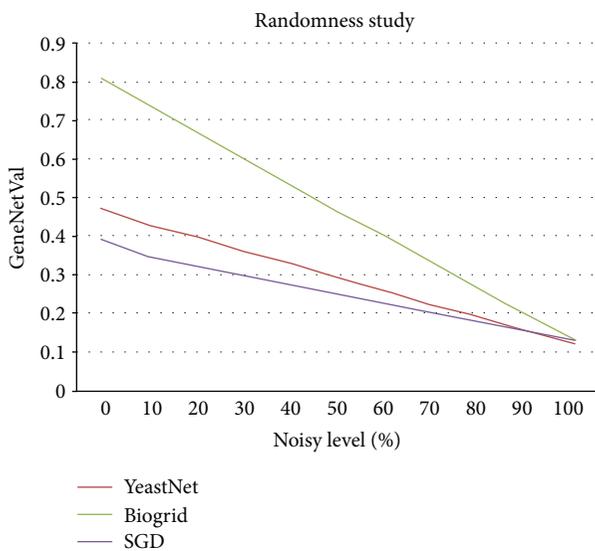


FIGURE 7: Results of the randomness study of GeneNetVal using level 2. For this study, we have used different yeast networks versus pathway *sce04111*.

Figure 7 presents the evolution of the validity values for the yeast networks. It can be observed that the different validity values follow a similar behavior. This behavior verifies that the loss of relevant information in the networks is progressive, and it increases as the randomness is increased in them as well. These results show that our method is able to detect the loss of information as the randomness increases in the networks.

3.3. A Functional Study: Yeast Cell Cycle Networks. In this section some well-known yeast networks are used to prove the usefulness of our approach by detecting specific biological functionality as it was described in Section 2.3. These networks were produced by applying different gene networks inference approaches to the same time-series yeast cell-cycle microarray [44]. Concretely, the networks were generated by applying the approaches of the network presented by Nariai et al. [26], which is obtained through a Bayesian-based

algorithm; Bulashevskaya and Eils [28] that is another Bayesian-based algorithm; Ponzoni et al. [29] whose algorithm called GRNCORP is based on a combinatorial optimization; and finally the network presented by Gallo et al. [27] (called GRNCORP2) that is a performance improvement of GRNCORP.

For this study, all the information stored in KEGG has been brought together in a single complex network. This global network (KEGG global network, KGN) is generated according to the knowledge gathered in each gene association network generated from *Saccharomyces cerevisiae* pathways. The aim of KGN is to conduct a global evaluation of the different networks to decide whether the networks contain biological knowledge or not. Specifically, the evaluation has been performed with level 2, according to the results obtained in the ROC analysis section. To compare the gene networks, only the relationships between genes contained in the input network and KGN have been considered. It is not possible to establish the quality of those interactions, because KEGG contains no information to ascertain whether the gene-gene interactions are biologically relevant or not.

In Table 2, the KGN rows, the global evaluation results, are shown. It is worth mentioning that two of the four networks obtain better validity results with the KGN because of the inclusion of a greater number of the indirect relationships (Hit₂).

Once the results of the global KGN network are obtained, a specific functional biological analysis was performed. This analysis reveals whether the cell cycle's network describes a specific biological process or if the information is dispersed among all the pathways stored in the organism.

Therefore, all input networks have been compared with each of the 105 GNs obtained from *Saccharomyces cerevisiae* pathways. The most representative results were obtained with the *sce04111* and *sce04113* pathways, which represent the cell cycle and meiosis processes, respectively. These results are presented in the *sce04111* and *sce04113* rows of Table 2. They show that the networks store the majority of its biological information regarding the cell cycle metabolic pathway. For that reason, we obtain practically the same values using the cell cycle pathway or KGN for the evaluation.

Moreover, all the hits (Hits₁ and Hits₂) obtained in comparison to KGN are also found in cell cycle pathway

sce04111. This information is summarized in Table 2 where the results of Hits₁ and Hits₂ are presented for all networks studied. As expected, the functionality found in the global evaluation is completed in relation to the yeast cell cycle process.

The input networks also contain biological information for the meiosis pathway (sce04113) (last row of Table 2). This does not contradict the previous result, since meiosis is related to the cell cycle [45], sharing gene-gene relationships.

The presented results show that our approach is able to identify the biological functionality that best describes the input network. Moreover, it is also able to find more related subprocesses.

4. Conclusions

A new methodology to score the biological validity of gene networks is proposed. Our presented approach, called GeneNetVal, entails identifying biological knowledge included in the input network. It is based on the gene-gene interaction relevance between the genes involved in the KEGG metabolic pathways. The method provides a complete and exhaustive conversion from a pathway to a gene association network. This approach also uses the concept of weak relationships between genes to present a new matching distance with different distance levels.

Three different experiments have been carried out. Firstly, our approach was compared to the classical use of KEGG to score the gene network validity. Comparisons were made for three different *Saccharomyces cerevisiae* complex networks. To demonstrate the robustness of the methodology, ROC analysis was performed for pure random and scale-free topologies. The results show that the proposal represents a significant improvement over the classical use of KEGG for assessing gene networks. Furthermore, it is possible to obtain better results by increasing the level of GeneNetVal in the comparison, where level 2 presents the best performance for all the experiments.

Secondly, a randomness study was performed. This study shows GeneNetVal is able to detect the loss of information in the input networks as the noise increases in them. Hence, our proposal distinguishes biologically correct from less correct networks.

Finally, the ability of GeneNetVal in finding a specific biological functionality was tested using some yeast cell cycle networks. The results show that our proposal is able to identify the main biological process described in an input network and other related processes.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Authors' Contribution

Francisco Gómez-Vela and Norberto Díaz-Díaz conceived the method, conducted the study, and wrote the paper. Francisco Gómez-Vela designed and implemented the algorithm.

Norberto Díaz-Díaz motivated the research problem. All authors read, edited, and approved the final paper.

Acknowledgments

This research was partially supported by the Ministry of Science and Innovation, Projects TIN2011-28956-C02-1, and Pablo de Olavide University. Francisco Gómez-Vela thanks Fernando Casares for helpful discussions about metabolic pathways and weak relationships.

References

- [1] M. Hecker, S. Lambeck, S. Toepfer, E. van Someren, and R. Guthke, "Gene regulatory network inference: data integration in dynamic models—a review," *BioSystems*, vol. 96, no. 1, pp. 86–103, 2009.
- [2] E. R. Dougherty and I. Shmulevich, "On the limitations of biological knowledge," *Current Genomics*, vol. 13, no. 7, pp. 574–587, 2012.
- [3] C. Yeh, H. Yeh, C. R. Arias, and V. Soo, "Pathway detection from protein interaction networks and gene expression data using color-coding methods and a* search algorithms," *The Scientific World Journal*, vol. 2012, Article ID 315797, 14 pages, 2012.
- [4] J. F. Poyatos, "The balance of weak and strong interactions in genetic networks," *PLoS ONE*, vol. 6, no. 2, Article ID e14598, 2011.
- [5] A. J. Butte and I. S. Kohane, "Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements," *Pacific Symposium on Biocomputing*, pp. 418–429, 2000.
- [6] P. Zoppoli, S. Morganella, and M. Ceccarelli, "Timedelay-ARACNE: reverse engineering of gene networks from time-course data by an information theoretic approach," *BMC Bioinformatics*, vol. 11, no. 1, article 154, 2010.
- [7] S. Bornholdt, "Boolean network models of cellular regulation: prospects and limitations," *Journal of the Royal Society Interface*, vol. 5, no. 1, pp. S85–S94, 2008.
- [8] L. Glass and S. A. Kauffman, "The logical analysis of continuous, non-linear biochemical control networks," *Journal of Theoretical Biology*, vol. 39, no. 1, pp. 103–129, 1973.
- [9] I. Shmulevich, E. R. Dougherty, S. Kim, and W. Zhang, "Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks," *Bioinformatics*, vol. 18, no. 2, pp. 261–274, 2002.
- [10] C. J. Needham, J. R. Bradford, A. J. Bulpitt, and D. R. Westhead, "A primer on learning in Bayesian networks for computational biology," *PLoS Computational Biology*, vol. 3, no. 8, article e129, 2007.
- [11] Q. Zhang, Y. Cao, Y. Li, Y. Zhu, S. S. M. Sun, and D. Guo, "A sub-space greedy search method for efficient Bayesian Network inference," *Computers in Biology and Medicine*, vol. 41, no. 9, pp. 763–770, 2011.
- [12] C. Rangel, J. Angus, Z. Ghahramani et al., "Modeling T-cell activation using gene expression profiling and state-space models," *Bioinformatics*, vol. 20, no. 9, pp. 1361–1372, 2004.
- [13] J. J. Faith, B. Hayete, J. T. Thaden et al., "Large-scale mapping and validation of *Escherichia coli* transcriptional regulation from a compendium of expression profiles," *PLoS Biology*, vol. 5, no. 1, article e8, 2007.

- [14] R. Gill and S. Datta, "A statistical framework for differential network analysis from microarray data," *BMC Bioinformatics*, vol. 11, article 95, 2010.
- [15] C. Li and H. Li, "Network-constrained regularization and variable selection for analysis of genomic data," *Bioinformatics*, vol. 24, no. 9, pp. 1175–1182, 2008.
- [16] T. Schaffter, D. Marbach, and D. Floreano, "GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods," *Bioinformatics*, vol. 27, no. 16, pp. 2263–2270, 2011.
- [17] T. Van den Bulcke, K. Van Leemput, B. Naudts et al., "SynTReN: a generator of synthetic gene expression data for design and analysis of structure learning algorithms," *BMC Bioinformatics*, vol. 7, article 43, 2006.
- [18] J. S. Aguilar-Ruiz, D. S. Rodriguez-Baena, N. Diaz-Diaz, and I. A. Nepomuceno-Chamorro, "CarGene: characterisation of sets of genes based on metabolic pathways analysis," *International Journal of Data Mining and Bioinformatics*, vol. 5, no. 5, pp. 558–573, 2011.
- [19] N. Díaz-Díaz and J. S. Aguilar-Ruiz, "GO-based functional dissimilarity of gene sets," *BMC Bioinformatics*, vol. 12, article 360, 2011.
- [20] J. C. Clemente, K. Ikeo, G. Valiente, and T. Gojobori, "Optimized ancestral state reconstruction using Sankoff parsimony," *BMC Bioinformatics*, vol. 10, no. 1, article 51, 2009.
- [21] H. Binder and M. Schumacher, "Incorporating pathway information into boosting estimation of high-dimensional risk prediction models," *BMC Bioinformatics*, vol. 10, article 18, 2009.
- [22] Y. Ko, C. X. Zhai, and S. Rodriguez-Zas, "Inference of gene pathways using mixture Bayesian networks," *BMC Systems Biology*, vol. 3, article 54, 2009.
- [23] N. Díaz-Díaz, F. Gómez-Vela, D. S. Rodriguez-Baena, and J. Aguilar-Ruiz, "Gene regulatory networks validation framework based in kegg," in *Proceedings of the 6th International Conference on Hybrid Artificial Intelligent Systems (HAIS '11)*, pp. 279–286, Wroclaw, Poland, May 2011.
- [24] Z. Wei and H. Li, "A Markov random field model for network-based analysis of genomic data," *Bioinformatics*, vol. 23, no. 12, pp. 1537–1544, 2007.
- [25] H. Zhou and L. Wong, "Comparative analysis and assessment of M. tuberculosis H37Rv protein-protein interaction datasets," *BMC Genomics*, vol. 12, supplement 3, p. S20, 2011.
- [26] N. Nariai, S. Kim, S. Imoto, and S. Miyano, "Using protein-protein interactions for refining gene networks estimated from microarray data by Bayesian networks," *Pacific Symposium on Biocomputing*, vol. 347, pp. 336–347, 2004.
- [27] C. A. Gallo, J. A. Carballido, and I. Ponzoni, "Discovering time-lagged rules from microarray data using gene profile classifiers," *BMC Bioinformatics*, vol. 12, article 123, 2011.
- [28] S. Bulashevskaya and R. Eils, "Inferring genetic regulatory logic from expression data," *Bioinformatics*, vol. 21, no. 11, pp. 2706–2713, 2005.
- [29] I. Ponzoni, F. J. Azuaje, J. C. Augusto, and D. H. Glass, "Inferring adaptive regulation thresholds and association rules from gene expression data through combinatorial optimization learning," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 4, no. 4, pp. 624–633, 2007.
- [30] M. Martínez-Ballesteros, I. A. Nepomuceno-Chamorro, and J. C. Riquelme, "Discovering gene association networks by multi-objective evolutionary quantitative association rules," *Journal of Computer and System Sciences*, vol. 80, no. 1, pp. 118–136, 2014.
- [31] F. M. Alakwaa, N. H. Solouma, and Y. M. Kadah, "Construction of gene regulatory networks using biclustering and bayesian networks," *Theoretical Biology and Medical Modelling*, vol. 8, article 39, 2011.
- [32] T. Xu, L. F. Du, and Y. Zhou, "Evaluation of GO-based functional similarity measures using S. cerevisiae protein interaction and expression profile data," *BMC Bioinformatics*, vol. 9, article 472, 2008.
- [33] N. N. Batada, T. Reguly, A. Breitkreutz et al., "Stratus not altocumulus: a new view of the yeast protein interaction network," *PLoS Biology*, vol. 4, no. 10, article e317, 2006.
- [34] J. M. Cherry, E. L. Hong, C. Amundsen et al., "Saccharomyces genome database: the genomics resource of budding yeast," *Nucleic Acids Research*, pp. D700–D705, 2012.
- [35] I. Lee, Z. Li, and E. M. Marcotte, "An improved, bias-reduced probabilistic functional gene network of baker's yeast, *Saccharomyces cerevisiae*," *PLoS ONE*, vol. 2, no. 10, article e988, 2007.
- [36] R. Albert, "Scale-free networks in cell biology," *Journal of Cell Science*, vol. 118, part 21, pp. 4947–4957, 2005.
- [37] R. Khanin and E. Wit, "How scale-free are biological networks," *Journal of Computational Biology*, vol. 13, no. 3, pp. 810–818, 2006.
- [38] J. E. Bartlett, J. W. Kotrlík, and C. C. Higgins, "Organizational research: determining appropriate sample size in survey research," *Information Technology, Learning, and Performance Journal*, vol. 19, no. 1, pp. 43–50, 2001.
- [39] J. C. Cuevas Tello, D. Hernández-Ramírez, and C. A. García-Sepúlveda, "Support vector machine algorithms in the search of KIR gene associations with disease," *Computers in Biology and Medicine*, vol. 43, no. 12, pp. 2053–2062, 2013.
- [40] M. L. Green and P. D. Karp, "Genome annotation errors in pathway databases due to semantic ambiguity in partial EC numbers," *Nucleic Acids Research*, vol. 33, no. 13, pp. 4035–4039, 2005.
- [41] S. Nagarajan, A. L. Kruckeberg, K. H. Schmidt et al., "Uncoupling reproduction from metabolism extends chronological lifespan in yeast," *Proceedings of the National Academy of Sciences of the United States of America*, 2014.
- [42] I. Ponzoni, M. Nueda, S. Tarazona et al., "Pathway network inference from gene expression data," *BMC Systems Biology*, vol. 8, supplement 2, p. S7, 2014.
- [43] P. Reshetova, A. Smilde, A. van Kampen, and J. Westerhuis, "Use of prior knowledge for the analysis of high-throughput transcriptomics and metabolomics data," *BMC Systems Biology*, vol. 8, supplement 2, p. S2, 2014.
- [44] P. T. Spellman, G. Sherlock, M. Q. Zhang et al., "Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization," *Molecular Biology of the Cell*, vol. 9, no. 12, pp. 3273–3297, 1998.
- [45] S. M. Honigberg and K. Purnapatre, "Signal pathway integration in the switch from the mitotic cell cycle to meiosis in yeast," *Journal of Cell Science*, vol. 116, no. 11, pp. 2137–2147, 2003.

Research Article

Comparing Evolutionary Strategies on a Biobjective Cultural Algorithm

**Carolina Lagos,¹ Broderick Crawford,^{1,2} Enrique Cabrera,³
Ricardo Soto,^{1,4} José-Miguel Rubio,^{1,5} and Fernando Paredes⁶**

¹ Escuela de Ingeniería Informática, Pontificia Universidad Católica de Valparaíso, 2362807 Valparaíso, Chile

² Universidad Finis Terrae, 7500000 Santiago, Chile

³ CIMFAV Facultad de Ingeniería, Universidad de Valparaíso, 2362735 Valparaíso, Chile

⁴ Universidad Autónoma de Chile, 7500138 Santiago, Chile

⁵ Departamento de Computación e Informática, Universidad de Playa Ancha, 33449 Valparaíso, Chile

⁶ Escuela de Ingeniería Industrial, Universidad Diego Portales, 8370109 Santiago, Chile

Correspondence should be addressed to Carolina Lagos; carolina.lagos.c@mail.pucv.cl

Received 9 April 2014; Accepted 27 June 2014; Published 31 August 2014

Academic Editor: Xin-She Yang

Copyright © 2014 Carolina Lagos et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Evolutionary algorithms have been widely used to solve large and complex optimisation problems. Cultural algorithms (CAs) are evolutionary algorithms that have been used to solve both single and, to a less extent, multiobjective optimisation problems. In order to solve these optimisation problems, CAs make use of different strategies such as normative knowledge, historical knowledge, circumstantial knowledge, and among others. In this paper we present a comparison among CAs that make use of different evolutionary strategies; the first one implements a historical knowledge, the second one considers a circumstantial knowledge, and the third one implements a normative knowledge. These CAs are applied on a biobjective uncapacitated facility location problem (BOUFLP), the biobjective version of the well-known uncapacitated facility location problem. To the best of our knowledge, only few articles have applied evolutionary multiobjective algorithms on the BOUFLP and none of those has focused on the impact of the evolutionary strategy on the algorithm performance. Our biobjective cultural algorithm, called BOCA, obtains important improvements when compared to other well-known evolutionary biobjective optimisation algorithms such as PAES and NSGA-II. The conflicting objective functions considered in this study are cost minimisation and coverage maximisation. Solutions obtained by each algorithm are compared using a hypervolume S metric.

1. Introduction

Evolutionary algorithms (EAs) are an effective alternative to (approximately) solve several large and complex optimisation problems, as they are able to find good solutions for a wide range of problems in acceptable computational time. Although less studied, EAs for multiobjective optimisation (MO) problems, called evolutionary multiobjective optimisation (EMO), have demonstrated to be very effective. In fact, during the last two decades, several authors have focused their efforts on the development of several EMO algorithms to solve a wide range of MO problems. For instance, Maravall

and de Lope [1] use a genetic algorithm (GA) to solve the multiobjective dynamic optimisation for automatic parking system. In [2] the authors propose an improvement to the well-known NSGA algorithm (called NSGA-II) based on an elitist approach. In [3] the author presents an EMO algorithm applied on a specific variation of the well-studied capacitated vehicle routing problem (CVRP), where the author includes in the EMO algorithm an explicit collective memory method, namely, the extended virtual loser (EVL). Other well-known EMO algorithms developed during the last two decades are PAES [4] and MO particle swarm optimisation [5]. More recently, hybrid techniques have been also applied to a large

number of optimisation problems (see [6]). A comprehensive literature review related to EMO algorithms can be found in [7].

EMO algorithms have some problems that must be taken into account, though. For instance, they tend to fall into premature convergence with low evolution efficiency [8]. This is because of implicit information embodied in the evolution process and domain knowledge corresponding to optimisation problems which are not fully included in the solution approach [9]. To overcome these problems, one can make use of implicit evolution information. Reynolds [10] proposes an EA, called cultural algorithm (CA), which is inspired from human culture evolution process and that makes use of the implicit evolution information generated at each iteration. The CAs have a dual evolution structure which consists of two spaces: population and belief space. On the one hand, the population space works as in any other EA, that is, using evolutionary operators such as mutation and crossover. On the other hand, in the belief space, implicit knowledge is extracted from selected individuals in the population and stored in a different way. Then, they are used to guide the whole evolution process in the population space such that they can induce the population to escape from local optimal solutions. It has been proved that CAs can effectively improve the evolution performance [9]. Although less studied, CAs have been also used to solve MO problems. Coello et al. [11] and Coello et al. [12], two remarkable surveys on CAs, only mention Coello and Landa [13] work as example of a CA application solving MO problems. More recently, Zhang et al. [14] present a CA which is enhanced by using a particle swarm optimisation algorithm. This enhanced CA is applied to fuel distribution MO problem. Srinivasan and Ramakrishnan [15] present a MO cultural algorithm that is applied on data mining domain. In [16] the authors applied a CA to a biobjective portfolio selection problem using normative knowledge in the belief space. In [17] authors present a formal framework to implement MO cultural algorithms. To the best of our knowledge, [18] is the only article that uses CAs to solve the biobjective uncapacitated facility location problem (BOUFLP). Furthermore, we did not find any article which compares the performance of CAs using different evolutionary strategies at the belief space level. Thus, in this paper we present an extension of the biobjective cultural algorithm (BOCA) developed in [18]. We use two different strategies at the belief space level and compare the performance of our new algorithms with the performance of the previous one. We also compare its results with other well-known EMO algorithms such as PAES and NSGA-II. Obtained solutions were compared using the hypervolume S metric proposed in [19].

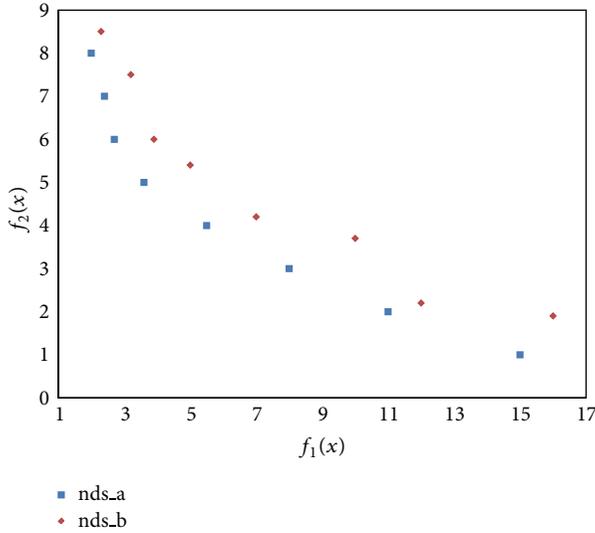
The remaining of this paper is organised as follows. Section 2 shows an overview on MO focused on EMO algorithms. Section 2.1 presents briefly the BOUFLP and some of its distinctive features. In Section 3, we describe our implementation for the BOCA algorithm. We describe the main differences between our implementation and the one in [18]. Section 3.2 presents the BOCA algorithm applied to a set of well-known instances from the literature. Finally, Section 4 presents the conclusions of this work.

2. Overview

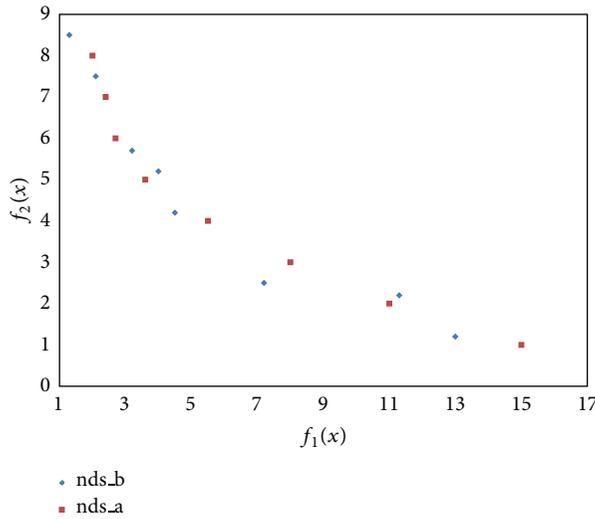
In this section we show an overview of topics related to this paper. In Section 2.1, MO concepts are presented, emphasizing EMO algorithms and its state of art. More specifically, we focus on the development of EMO algorithms for MO combinatorial optimisation (MOCO) problems. In Section 2.2 we present the BOUFLP formulation based on a cost-coverage approach. Finally, in Section 2.3 we present an overview of CAs and its multiobjective extension. Details of our CA implementation are also presented at the end of this section.

2.1. (Evolutionary) Multiobjective Optimisation. In this section we briefly introduce the main principles of MO problems and, particularly, MOCO problems. For a comprehensive review of this topic see [20, 21]. In this paper we will make use of the following notation for the comparison of vectors (solutions). Let y and $y' \in \mathbb{R}^p$. We say that $y \leq y'$ if $y_k \leq y'_k \forall k = 1, \dots, p$. Similarly, we will say that $y \leq y'$ if $y \leq y'$ but $y \neq y'$. Finally, we say that $y < y'$ if $y_k < y'_k \forall k = 1, \dots, p$. A solution $\hat{x} \in \mathbb{R}^n$, with n being equal to the number of decision variables, is called an efficient solution and its image $f(\hat{x})$, with $f: \mathbb{R}^n \rightarrow \mathbb{R}^p$, a nondominated point of the MO problem if there is no $x \in \mathbb{R}^n$, with $x \neq \hat{x}$, such that $f(x) \leq f(\hat{x})$. In [22] the author describes several excellence relations. These relations establish strict partial orders in the set of all nondominated points related to different aspects of their quality. Previously, in [23, 24] the authors consider several outperformance relations to address the closeness of the set of nondominated points found by an algorithm to the actual set of nondominated points, called Pareto Frontier (PF). In [25] a comprehensive explanation of the desirable features of an approximation to the PF is presented. In this paper, we choose the S metric which is properly explained in [24]. The S metric calculates the hypervolume of a multidimensional region [19] and allows the integration of aspects that are individually measured by other metrics. An advantage of the S metric is that each algorithm can be assessed independently of the other algorithms involved in the study. However, the S values of two sets \mathcal{A} and \mathcal{B} , namely, $S_{\mathcal{A}}$ and $S_{\mathcal{B}}$, respectively, cannot be used to derive whether either set entirely dominates the other. Figure 1(a) shows a situation where $S_{\mathcal{A}} > S_{\mathcal{B}}$ and set \mathcal{A} completely dominates set \mathcal{B} . Figure 1(b) shows a situation where $S_{\mathcal{B}} > S_{\mathcal{A}}$ but neither \mathcal{A} dominates \mathcal{B} nor \mathcal{B} dominates \mathcal{A} .

In this paper we use EAs to solve the BOUFLP. An EA is a stochastic search procedure inspired by the evolution process in nature. In this process, individuals evolve and the fitter ones have a better chance of reproduction and survival. The reproduction mechanisms favour the characteristics of the stronger parents and hopefully produce better children guaranteeing the presence of those characteristics in future generations [26]. EAs have been successfully applied to a large number of both single- and multiobjective optimisation problems. Comprehensive reviews of EMO algorithms are presented in [11, 24] and more recently in [12]. A more general



(a) Nondominated set \mathcal{A} completely dominates nondominated set \mathcal{B}



(b) Neither \mathcal{A} dominates \mathcal{B} nor \mathcal{B} dominates \mathcal{A}

FIGURE 1: S metric represented as the area that is dominated by a set of (approximately) nondominated points.

review of hybrid heuristics solving MOCO problems, where EMO algorithms are also included, is presented in [27].

2.2. Biobjective Uncapacitated Facility Location Problem. Facility location problem (FLP) is one of the most important problems for companies with the aim of distributing products to their customers. The problem consists of selecting sites to install plants, warehouses, and distribution centres, allocating customers to serving facilities and interconnecting facilities by flow assignment decisions. Comprehensive reviews and analysis of the FLP are presented in [28–30].

In this paper we consider a two-level supply chain, where a single plant serves a set of warehouses, which serve a set of end customers or retailers. Figure 2 shows this configuration.

Two main (conflicting) objectives can be identified in the FLP:

- (i) minimise the total cost associated with the facility installation and customer allocation and
- (ii) maximise the customers rate coverage.

Several works on single-objective optimisation have been carried out considering these two objectives separately. On the one hand, uncapacitated FLP (UFLP) is one of most studied FLPs in the literature. In the UFLP the main goal is to minimise the location-allocation cost of the network. On the other hand, p median FLPs are one of the most common FLPs among those that are focused on coverage maximisation. Most important FLP models are well described and formalised in [29]. MO FLPs have been also well studied in the literature during the last two decades. A survey on this topic can be found in [31].

As we mentioned before, in this paper we solve the BOUFLP. BOUFLP has been modelled with minisum and maxisum objectives (cost and coverage). The following model formulation is based on [32]. Let $I = \{1, \dots, m\}$ be the set of potential facilities and $J = \{1, \dots, n\}$ the set of customers. Let FC_i be the fixed cost of opening facility i and d_j the demand of customer j . Let c_{ij} be the cost of assigning the customer j to facility i and h_{ij} the distance between facility i and customer j . Let D_{MAX} be the maximal covering distance; that is, customers within this distance to an open facility are considered well served. Let $Q^j = \{i \in I : h_{ij} \leq D_{MAX}\}$ be the set of facilities that could serve customer j within the maximal covering distance D_{MAX} . Let x_i be 1 if facility i is open and 0, otherwise. Let y_{ij} be 1 if the whole demand of customer j is served by facility i and 0, otherwise. Consider

$$\text{minimise } f(x, y) \tag{1}$$

$$\text{s.t. } \sum_{i \in I} y_{ij} \quad \forall j \in J \tag{2}$$

$$y_{ij} \leq x_i \quad \forall j \in J, i \in I \tag{3}$$

$$y_{ij} \in \{0, 1\} \quad \forall j \in J, i \in I \tag{4}$$

$$x_i \in \{0, 1\} \quad \forall i \in I \tag{5}$$

with $f : \mathbb{R}^n \rightarrow \mathbb{R}^p$ and $p = 2$. Objective functions f_1 and f_2 are as follows:

$$f_1(x, y) = \sum_{i \in I} FC_i x_i + \sum_{i \in I} \sum_{j \in J} c_{ij} y_{ij} \tag{6}$$

$$f_2(x, y) = - \sum_{j \in J} d_j \sum_{i \in Q^j} y_{ij} \tag{7}$$

Equation (6) represents total operating cost; the first term corresponds to location cost, that is, the sum of the fixed costs of all the open facilities, and the second term represents the allocation cost, that is, the cost of attending customer demand by an open facility. Equation (7) measures coverage as the sum of the demand of customers attended by open facilities within the maximal covering distance. Equations (2) and (3)

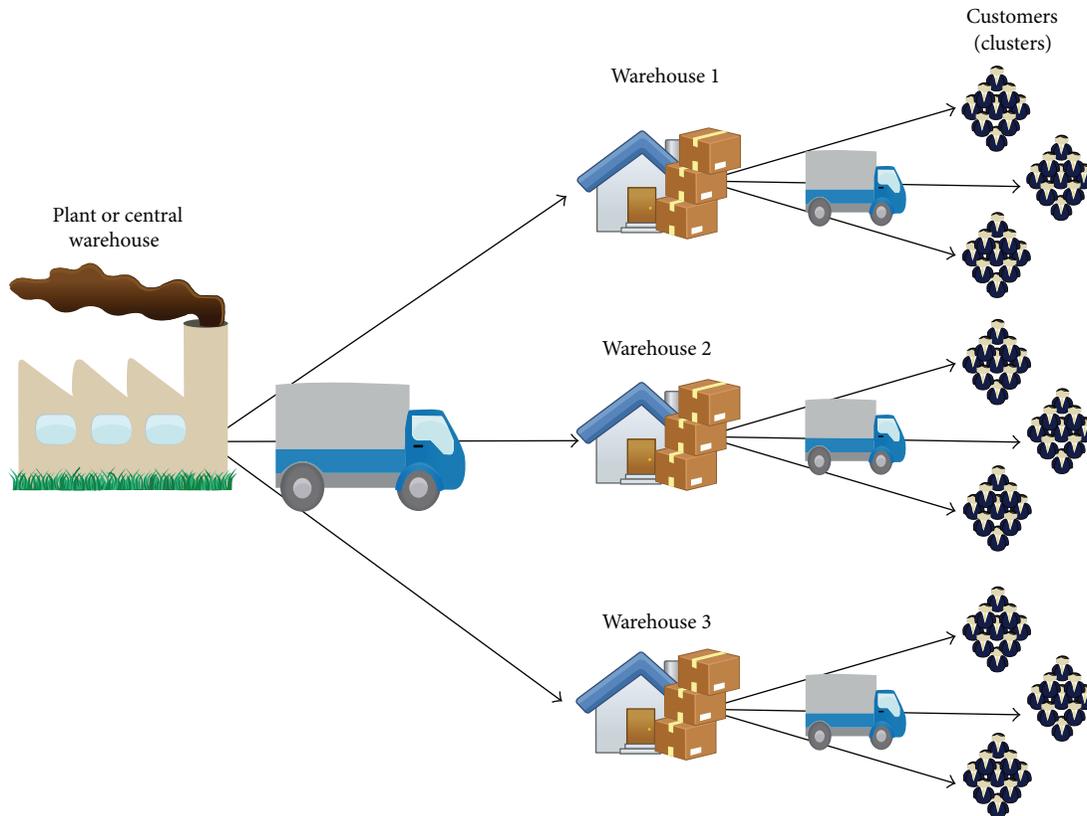


FIGURE 2: A two-level supply chain network configuration.

ensure that each customer is attended by only one facility. Equation (3) also forces customer to be assigned to an open facility. Finally, equations (4) and (5) set decision variables as binary.

2.3. Biobjective Cultural Algorithm. The experience and beliefs accepted by a community in a social system are the main motivations for the creation of the CAs. Originally proposed by Reynolds [10], CAs model the evolution of cultural systems based on the principles of human social evolution. In this case, evolution is seen as an optimisation process [10]. The CAs guide the evolution of the population based on the knowledge. Knowledge acquired during previous iterations is provided to future generations, allowing accelerating the convergence of the algorithm to good solutions [33]. Domain knowledge is modelled separately from the population, because there is certain independence between them which allows us to work and model them separately, in order to enhance the overall algorithm performance. Figure 3 shows this interaction.

CAs are mainly characterised by presenting two inheritance systems: one at population level, called *population space*, and the other at knowledge level, called *belief space*. This key feature is designed to increase the learning rates and convergence of the algorithm and thus to do a more responsive system for a number of problems [34]. Moreover, it allows us to identify two significant levels of knowledge:

a microevolutionary level (represented by the population space) and macroevolutionary level (represented by the space of beliefs) [35].

CAs have the following components: population space (set of individuals who have independent features) [35]; belief space (stored individuals acquired in previous generations) [34]; computer protocol, connecting the two spaces and defining the rules on the type of information to be exchanged between them by using the acceptance and influence functions; and finally knowledge sources which are described in terms of their ability to coordinate the distribution of individuals depending on the nature of a problem instance [35]. These knowledge sources can be of the following types: circumstantial, normative, domain, topographic, and historical.

The most distinctive feature of CAs is the use of the belief space which through an influence function affects future generations. For this reason, in this paper we focus on the effect on the algorithm performance of changes in such an influence function. To do this, we have considered results obtained previously in [18], where the authors used an influence function based on historical knowledge, and we compare those results with our BOCA implementation which considers two influence functions: the first one based on circumstantial knowledge and the second one based on normative knowledge. Algorithm 1 shows the general procedure of our BOCA algorithm.

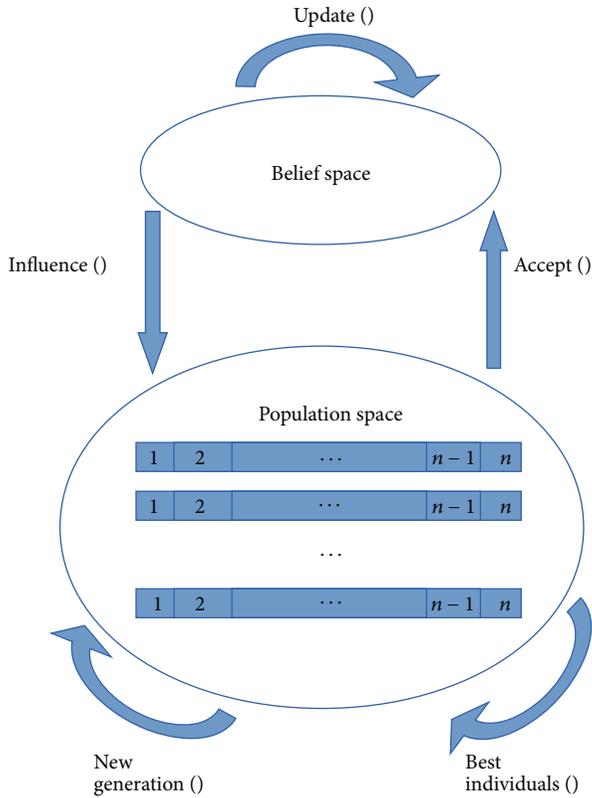


FIGURE 3: CA general diagram.

```

begin
  k = 0;
  initialise Population  $P_k$ ;
  initialise BeliefSpace  $B_k$ ;
  while  $k < iterationNumber$  do
    Evaluate( $P_k$ );
     $P_k^{best} = bestIndividuals(P_k)$ ;
    updateBeliefSpace( $B_k$ , accept( $P_k^{best}$ ));
    influence( $P_k$ ,  $B_k$ );
     $k = k + 1$ ;
     $P_k = newPopulation(P_{k-1})$ ;
  end
  
```

ALGORITHM 1: BOCA general algorithmic frame.

To initialise the population, we use a semirandom function. In its first phase, this function defines in a stochastic way the set of facilities that will be opened (selected facilities). Then, we allocate each customer to a selected facility minimising the cost function f_1 while avoiding minimising the coverage function f_2 . This strategy provides better results than using completely random initial populations, and its computational time additional cost is marginal.

To obtain the next generation, two parents are used in a recombination process. To avoid local optimal values, we do not overuse the culture. Thus, a parent is selected from the population to obtain diversity and the other parent is selected

from the belief space to influence the next generation. The belief space keeps a list of all the individuals which meet some criteria. These criteria depend on what knowledge the algorithm implements. In this paper the circumstantial knowledge selects the best individuals found so far for each objective function. Thus, one individual will give us information on the best value found for f_1 and the other will do the same for f_2 . The historical knowledge stores a list of individuals with the best fitness value found so far. The fitness value is calculated as the hypervolume S that is covered by an individual. Finally, normative knowledge considers a list of individuals which are pairwise nondominated with respect to the other individuals of their generation.

Let $|J|$ be the number of available facilities and let $|I|$ be the number of customers of our BOUFLP. In this paper decisions variables x and y are represented by a binary $|J|$ -length vector and $|J| \times |I|$ matrix, respectively. Comparing two different solutions (individuals) needs an evaluation criterion. In this paper we use the same criterion explained in [18].

3. Computational Experiments

In this section we present the set of instances that are used in this study as well as results obtained by our BOCA implementation.

3.1. Instances Presentation. The instances that were used in this paper correspond to random instances using a problem generator that follows the methodology from UfLib [36]. Previous works in the literature have also used this problem generator to create their test instances [18, 26].

The BOCA algorithm has several parameters that need to be set. As in [18], the number of generations considered in this paper is equal to 100. Population size L is set equal to 100, mutation probability in the population space P_{ps} is equal to 0.2, and probability of mutation in the belief space P_{bs} is 0.04. Both L and P_{ps} values are different from the values used in [18]. These values are chosen as they all together yield to the best performance of the algorithm given some test instances. Thus, although resulting values are different from that used in [18], the method we use to set them is the same as that used in that work. This is important in order to fairly compare the different BOCA algorithms.

3.2. Results and Discussion. In this section we compare the results obtained by the previous BOCA algorithm (see [18] for further details) and our approach. Moreover, a comparison between results obtained by well-known EMO algorithms such as NSGA-II and PAES and our BOCA algorithm is also presented in this section.

Tables 1 and 2 show the results obtained by the BOCA implementations using historical [18], circumstantial, and normative knowledge, respectively. In the same way, Tables 3 and 4 present the results obtained by the well-known NSGA-II and PAES algorithms. For each algorithm S value (%), time t (in seconds), and the number of efficient solutions $\hat{x} \in \hat{X}$ have been included in these tables. As we mentioned before,

TABLE 1: Results obtained by the BOCA implementations for instance of class A.

Instance	BOCA (historical)			BOCA (circumstantial)			BOCA (normative)		
	t_1 (sec)	S_1 %	$ \widehat{X}_1 $	t_2 (sec)	S_2 %	$ \widehat{X}_2 $	t_3 (sec)	S_3 %	$ \widehat{X}_3 $
$A_{10-25}C_1$	252	0.7057	13	167	0.7058	13	37	0.6963	11
$A_{10-25}C_2$	197	0.6136	11	176	0.6136	11	27	0.6135	10
$A_{10-25}C_3$	186	0.6993	9	184	0.6992	9	33	0.6992	9
$A_{10-25}C_4$	115	0.5729	5	193	0.5729	5	28	0.5728	3
$A_{10-25}C_5$	152	0.6351	8	168	0.6352	8	32	0.6352	8
$A_{10-25}C_6$	183	0.5982	9	186	0.5961	8	34	0.5944	8
$A_{30-75}C_1$	464	0.8071	18	793	0.7828	16	1908	0.7300	12
$A_{30-75}C_2$	1247	0.8013	47	598	0.7798	23	1201	0.7090	30
$A_{30-75}C_3$	1177	0.7552	50	611	0.7374	21	1305	0.7098	29
$A_{30-75}C_4$	450	0.7471	21	960	0.6307	10	1450	0.6048	15
$A_{30-75}C_5$	940	0.7790	46	610	0.7628	32	1725	0.7078	28
$A_{30-75}C_6$	740	0.7245	37	536	0.6656	14	446	0.7010	28
$A_{50-150}C_1$	1473	0.7878	52	2386	0.6685	27	3481	0.7140	42
$A_{50-150}C_2$	1043	0.7611	37	4266	0.6418	27	5412	0.6200	32
$A_{50-150}C_3$	1165	0.7883	53	1644	0.7171	24	3941	0.7160	27
$A_{50-150}C_4$	735	0.6345	17	1437	0.5849	17	3419	0.6010	20
$A_{50-150}C_5$	1459	0.8039	53	2333	0.6725	21	2745	0.7158	35
$A_{50-150}C_6$	1434	0.7903	43	2399	0.6535	26	2811	0.7040	48

TABLE 2: Results obtained by the BOCA implementations for instance of class B.

Instance	BOCA (historical)			BOCA (circumstantial)			BOCA (normative)		
	t_1 (sec)	S_1 %	$ \widehat{X}_1 $	t_2 (sec)	S_2 %	$ \widehat{X}_2 $	t_3 (sec)	S_3 %	$ \widehat{X}_3 $
$B_{10-25}C_1$	135	0.7795	7	196	0.7795	7	36	0.7721	8
$B_{10-25}C_2$	130	0.5856	5	128	0.5813	4	48	0.5807	4
$B_{10-25}C_3$	238	0.7410	14	229	0.7278	12	101	0.7170	13
$B_{10-25}C_4$	260	0.7261	14	222	0.7246	14	182	0.7218	12
$B_{10-25}C_5$	115	0.8117	6	185	0.7250	4	25	0.8020	5
$B_{10-25}C_6$	285	0.6517	18	245	0.6340	15	31	0.5967	12
$B_{30-75}C_1$	683	0.7650	38	736	0.6945	26	1208	0.7119	20
$B_{30-75}C_2$	944	0.7323	59	882	0.7133	30	1014	0.7115	24
$B_{30-75}C_3$	1351	0.6335	72	683	0.6263	28	1076	0.6106	28
$B_{30-75}C_4$	618	0.5214	32	857	0.4947	27	472	0.4824	29
$B_{30-75}C_5$	905	0.7473	45	917	0.6869	37	1820	0.6647	25
$B_{30-75}C_6$	800	0.8775	42	565	0.8502	24	781	0.8310	35
$B_{50-150}C_1$	802	0.8345	23	885	0.8105	18	2371	0.8090	20
$B_{50-150}C_2$	1221	0.7634	53	1405	0.7134	24	2678	0.7050	35
$B_{50-150}C_3$	1410	0.7915	58	1706	0.7169	13	2791	0.6940	45
$B_{50-150}C_4$	567	0.7498	10	604	0.6720	17	3841	0.6420	15
$B_{50-150}C_5$	939	0.6393	37	482	0.5470	27	5712	0.5870	25
$B_{50-150}C_6$	1904	0.8016	67	1745	0.7560	34	4958	0.7150	54

we want to produce a set with a large number of efficient solutions $\widehat{x} \in \widehat{X}$, a S value close to 100% (ideal), and a small t . For the sake of easy reading, we have split the set of instances into two subsets (instances type A and B).

We then compare our BOCA implementations with the one presented in [18]. Tables 5 and 6 show a comparison between those algorithms. As we can see, when compared in terms of its S value (the bigger, the better), BOCA algorithm

using historical knowledge (BOCA^H) performs consistently better than the ones using circumstantial (BOCA^C) and normative (BOCA^N) knowledge. In fact BOCA^H obtains a S value that is, in average, 5.8% bigger than the one obtained by BOCA^C and 6.5% bigger than the S value obtained by BOCA^N. When compared in terms of the CPU time needed to reach the number of iterations (generations), BOCA^H is, in average, faster than both BOCA^C and BOCA^N algorithms.

TABLE 3: Results obtained by well-known MOEA algorithms NSGA-II and PAES for instances of class A.

Instance	NSGA-II			PAES		
	t_4 (sec)	S_4 %	$ \widehat{X}_4 $	t_5 (sec)	S_5 %	$ \widehat{X}_5 $
$A_{10-25}C_1$	1344	0.7057	13	372	0.7057	13
$A_{10-25}C_2$	1511	0.6136	11	394	0.6136	11
$A_{10-25}C_3$	1859	0.6993	9	326	0.6993	9
$A_{10-25}C_4$	3186	0.5729	5	305	0.5729	5
$A_{10-25}C_5$	1748	0.6351	8	378	0.6351	8
$A_{10-25}C_6$	1650	0.5982	9	384	0.5982	9
$A_{30-75}C_1$	1633	0.8071	18	499	0.7795	16
$A_{30-75}C_2$	1345	0.8012	43	622	0.7915	41
$A_{30-75}C_3$	1394	0.7538	43	603	0.7384	29
$A_{30-75}C_4$	1874	0.7508	20	525	0.7342	20
$A_{30-75}C_5$	1413	0.7783	40	595	0.7572	33
$A_{30-75}C_6$	1474	0.6676	36	511	0.5300	12
$A_{50-150}C_1$	2385	0.7913	43	1386	0.7391	25
$A_{50-150}C_2$	2522	0.7597	39	1231	0.6729	23
$A_{50-150}C_3$	2298	0.7665	34	1269	0.6900	26
$A_{50-150}C_4$	2575	0.6344	17	1233	0.6106	15
$A_{50-150}C_5$	2446	0.8072	40	1251	0.7590	23
$A_{50-150}C_6$	2259	0.7862	38	1218	0.6461	27

TABLE 4: Results obtained by well-known MOEA algorithms NSGA-II and PAES for instances of class B.

Instance	NSGA-II			PAES		
	t_4 (sec)	S_4 %	$ \widehat{X}_4 $	t_5 (sec)	S_5 %	$ \widehat{X}_5 $
$B_{10-25}C_1$	1439	0.7795	7	397	0.7795	7
$B_{10-25}C_2$	1430	0.5856	5	405	0.5856	5
$B_{10-25}C_3$	1288	0.741	13	433	0.7410	13
$B_{10-25}C_4$	1747	0.7261	14	440	0.7261	14
$B_{10-25}C_5$	1766	0.8117	6	363	0.8117	6
$B_{10-25}C_6$	1261	0.6517	17	394	0.6517	17
$B_{30-75}C_1$	1566	0.7609	30	562	0.7134	27
$B_{30-75}C_2$	1525	0.7285	49	563	0.6899	34
$B_{30-75}C_3$	1345	0.6330	51	578	0.5991	53
$B_{30-75}C_4$	2562	0.5333	23	502	0.5211	21
$B_{30-75}C_5$	1433	0.7631	31	564	0.7360	26
$B_{30-75}C_6$	1460	0.8088	47	585	0.7602	32
$B_{50-150}C_1$	2698	0.8447	23	1280	0.6925	10
$B_{50-150}C_2$	2289	0.7515	41	1241	0.4970	13
$B_{50-150}C_3$	2284	0.7988	52	1279	0.6836	26
$B_{50-150}C_4$	2603	0.7500	10	1192	0.7253	14
$B_{50-150}C_5$	2301	0.6387	35	1212	0.5021	16
$B_{50-150}C_6$	3178	0.8029	43	1260	0.7152	32

We can note that for B instances, times required by BOCA^H and BOCA^C are, in average, quite similar (only 1.6% of difference). Finally, when we look at the number of efficient solutions found by each algorithm ($|\widehat{X}|$ column), we can see that, again, BOCA^H outperforms both BOCA^C and BOCA^N algorithms. In this case, the average number of efficient

TABLE 5: Comparison among our BOCA implementations (A instances).

Instance	$\Delta_{S_2}^{S_1}$	$\Delta_{t_2}^{t_1}$	$\Delta_{ \widehat{X}_2 }^{ \widehat{X}_1 }$	$\Delta_{S_3}^{S_1}$	$\Delta_{t_3}^{t_1}$	$\Delta_{ \widehat{X}_3 }^{ \widehat{X}_1 }$
$A_{10-25}C_1$	-0.014	33.73	0.00	1.332	85.32	15.38
$A_{10-25}C_2$	0.000	10.66	0.00	0.016	86.29	9.09
$A_{10-25}C_3$	0.014	1.08	0.00	0.014	82.26	0.00
$A_{10-25}C_4$	0.000	-67.83	0.00	0.017	75.65	40.00
$A_{10-25}C_5$	-0.016	-10.53	0.00	-0.016	78.95	0.00
$A_{10-25}C_6$	0.351	-1.64	11.11	0.635	81.42	11.11
$A_{30-75}C_1$	3.011	-70.91	11.11	9.553	-311.21	33.33
$A_{30-75}C_2$	2.683	52.04	51.06	11.519	3.69	36.17
$A_{30-75}C_3$	2.357	48.09	58.00	6.012	-10.88	42.00
$A_{30-75}C_4$	15.58	-113.33	52.38	19.047	-222.22	28.57
$A_{30-75}C_5$	2.080	35.11	30.43	9.140	-83.51	39.13
$A_{30-75}C_6$	8.130	27.57	62.16	3.244	39.73	24.32
$A_{50-150}C_1$	15.143	-61.98	48.08	9.368	-136.32	19.23
$A_{50-150}C_2$	15.675	-309.01	27.03	18.539	-418.89	13.51
$A_{50-150}C_3$	9.0320	-41.12	54.72	9.172	-238.28	49.06
$A_{50-150}C_4$	7.8170	-95.51	0.00	5.280	-365.17	-17.65
$A_{50-150}C_5$	16.345	-59.90	60.38	10.959	-88.14	33.96
$A_{50-150}C_6$	17.310	-67.29	39.53	10.920	-96.03	-11.63

TABLE 6: Comparison among our BOCA implementations.

Instance	$\Delta_{S_2}^{S_1}$	$\Delta_{t_2}^{t_1}$	$\Delta_{ \widehat{X}_2 }^{ \widehat{X}_1 }$	$\Delta_{S_3}^{S_1}$	$\Delta_{t_3}^{t_1}$	$\Delta_{ \widehat{X}_3 }^{ \widehat{X}_1 }$
$B_{10-25}C_1$	0.000	-45.19	0.00	0.949	73.33	-14.29
$B_{10-25}C_2$	0.734	1.54	20.00	0.837	63.08	20.00
$B_{10-25}C_3$	1.781	3.78	14.29	3.239	57.56	7.14
$B_{10-25}C_4$	0.207	14.62	0.00	0.592	30.00	14.29
$B_{10-25}C_5$	10.681	-60.87	33.33	1.195	78.26	16.67
$B_{10-25}C_6$	2.716	14.04	16.67	8.439	89.12	33.33
$B_{30-75}C_1$	9.216	-7.76	31.58	6.941	-76.87	47.37
$B_{30-75}C_2$	2.595	6.57	49.15	2.840	-7.42	59.32
$B_{30-75}C_3$	1.137	49.44	61.11	3.615	20.36	61.11
$B_{30-75}C_4$	5.121	-38.67	15.63	7.480	99.92	9.38
$B_{30-75}C_5$	8.082	-1.33	17.78	11.053	-101.10	44.44
$B_{30-75}C_6$	3.111	29.38	42.86	5.299	2.38	16.67
$B_{50-150}C_1$	2.876	-10.35	21.74	3.056	-195.64	13.04
$B_{50-150}C_2$	6.550	-15.07	54.72	7.650	-119.33	33.96
$B_{50-150}C_3$	9.425	-20.99	77.59	12.318	-97.94	22.41
$B_{50-150}C_4$	10.376	-6.53	-70.00	14.377	-577.43	-50.00
$B_{50-150}C_5$	14.438	48.67	27.03	8.181	-508.31	32.43
$B_{50-150}C_6$	5.689	8.35	49.25	10.803	-160.40	19.40

solutions found by the BOCA^H algorithm is about 20% bigger than the one obtained by the other two approaches.

Results above are consistent with the good performance obtained by the BOCA^H approach in [18]. Moreover, results show that performance of the BOCA algorithm depends largely on the selected knowledge and it can make the difference in terms of S value, time, and number of efficient solutions found by the algorithm. This is an important finding

TABLE 7: Comparison among our BOCA with circumstantial knowledge and NSGA-II and PAES.

Instance	$\Delta_{S_4}^{S_2}$	$\Delta_{t_4}^{t_2}$	$\Delta_{ \bar{X}_4 }^{ \bar{X}_2 }$	$\Delta_{S_5}^{S_2}$	$\Delta_{t_5}^{t_2}$	$\Delta_{ \bar{X}_5 }^{ \bar{X}_2 }$
$A_{10-25}C_1$	-1.35	-3532.43	-18.18	-1.35	-905.41	-18.18
$A_{10-25}C_2$	-0.02	-5496.30	-10.00	-0.02	-1359.26	-10.00
$A_{10-25}C_3$	-0.01	-5533.33	0.00	-0.01	-887.88	0.00
$A_{10-25}C_4$	-0.02	-11278.57	-66.67	-0.02	-989.29	-66.67
$A_{10-25}C_5$	0.02	-5362.50	0.00	0.02	-1081.25	0.00
$A_{10-25}C_6$	-0.64	-4752.94	-12.50	-0.64	-1029.41	-12.50
$A_{30-75}C_1$	-10.56	14.41	-50.00	-6.78	73.85	-33.33
$A_{30-75}C_2$	-13.00	-11.99	-43.33	-11.64	48.21	-36.67
$A_{30-75}C_3$	-6.20	-6.82	-48.28	-4.03	53.79	0.00
$A_{30-75}C_4$	-24.14	-29.24	-33.33	-21.40	63.79	-33.33
$A_{30-75}C_5$	-9.96	18.09	-42.86	-6.98	65.51	-17.86
$A_{30-75}C_6$	4.76	-230.49	-28.57	24.39	-14.57	57.14
$A_{50-150}C_1$	-10.83	31.49	-2.38	-3.52	60.18	40.48
$A_{50-150}C_2$	-22.53	53.40	-21.88	-8.53	77.25	28.13
$A_{50-150}C_3$	-7.05	41.69	-25.93	3.63	67.80	3.70
$A_{50-150}C_4$	-5.56	24.69	15.00	-1.60	63.94	25.00
$A_{50-150}C_5$	-12.77	10.89	-14.29	-6.04	54.43	34.29
$A_{50-150}C_6$	-11.68	19.64	20.83	8.22	56.67	43.75

TABLE 8: Comparison among our BOCA with circumstantial knowledge and NSGA-II and PAES.

Instance	$\Delta_{S_4}^{S_2}$	$\Delta_{t_4}^{t_2}$	$\Delta_{ \bar{X}_4 }^{ \bar{X}_2 }$	$\Delta_{S_5}^{S_2}$	$\Delta_{t_5}^{t_2}$	$\Delta_{ \bar{X}_5 }^{ \bar{X}_2 }$
$B_{10-25}C_1$	-0.96	-3897.22	12.50	-0.96	-1002.78	12.50
$B_{10-25}C_2$	-0.84	-2879.17	-25.00	-0.84	-743.75	-25.00
$B_{10-25}C_3$	-3.35	-1175.25	0.00	-3.35	-328.71	0.00
$B_{10-25}C_4$	-0.60	-859.89	-16.67	-0.60	-141.76	-16.67
$B_{10-25}C_5$	-1.21	-6964.00	-20.00	-1.21	-1352.00	-20.00
$B_{10-25}C_6$	-9.22	-3967.74	-41.67	-9.22	-1170.97	-41.67
$B_{30-75}C_1$	-6.88	-29.64	-50.00	-0.21	53.48	-35.00
$B_{30-75}C_2$	-2.39	-50.39	-104.17	3.04	44.48	-41.67
$B_{30-75}C_3$	-3.67	-25.00	-82.14	1.88	46.28	-89.29
$B_{30-75}C_4$	-10.55	-442.80	20.69	-8.02	-6.36	27.59
$B_{30-75}C_5$	-14.80	21.26	-24.00	-10.73	69.01	-4.00
$B_{30-75}C_6$	2.67	-86.94	-34.29	8.52	25.10	8.57
$B_{50-150}C_1$	-4.41	-13.79	-15.00	14.40	46.01	50.00
$B_{50-150}C_2$	-6.60	14.53	-17.14	29.50	53.66	62.86
$B_{50-150}C_3$	-15.10	18.17	-15.56	1.50	54.17	42.22
$B_{50-150}C_4$	-16.82	32.23	33.33	-12.98	68.97	6.67
$B_{50-150}C_5$	-8.81	59.72	-40.00	14.46	78.78	36.00
$B_{50-150}C_6$	-12.29	35.90	20.37	-0.03	74.59	40.74

as it points out the relevance of the choice of a specific type of knowledge.

We now compare BOCA^C and BOCA^N algorithms to the well-known NSGA-II and PAES algorithms. Tables 7 and 8 show a comparison between our BOCA^C algorithm and the NSGA-II and PAES algorithms. As we can see, although BOCA^C obtains, in average, a S value 6.8% lower than the

TABLE 9: Comparison among our BOCA with normative knowledge and NSGA-II and PAES.

Instance	$\Delta_{S_4}^{S_3}$	$\Delta_{t_4}^{t_3}$	$\Delta_{ \bar{X}_4 }^{ \bar{X}_3 }$	$\Delta_{S_5}^{S_3}$	$\Delta_{t_5}^{t_3}$	$\Delta_{ \bar{X}_5 }^{ \bar{X}_3 }$
$A_{10-25}C_1$	0.01	-704.79	0.00	0.01	-122.75	0.00
$A_{10-25}C_2$	0.00	-758.52	0.00	0.00	-123.86	0.00
$A_{10-25}C_3$	-0.01	-910.33	0.00	-0.01	-77.17	0.00
$A_{10-25}C_4$	0.00	-1550.78	0.00	0.00	-58.03	0.00
$A_{10-25}C_5$	0.02	-940.48	0.00	0.02	-125.00	0.00
$A_{10-25}C_6$	-0.35	-787.10	-12.50	-0.35	-106.45	-12.50
$A_{30-75}C_1$	-3.10	-105.93	-12.50	0.42	37.07	0.00
$A_{30-75}C_2$	-2.74	-124.92	-86.96	-1.50	-4.01	-78.26
$A_{30-75}C_3$	-2.22	-128.15	-104.76	-0.14	1.31	-38.10
$A_{30-75}C_4$	-19.04	-95.21	-100.00	-16.41	45.31	-100.00
$A_{30-75}C_5$	-2.03	-131.64	-25.00	0.73	2.46	-3.13
$A_{30-75}C_6$	-0.30	-175.00	-157.14	20.37	4.66	14.29
$A_{50-150}C_1$	-18.37	0.04	-59.26	-10.56	41.91	7.41
$A_{50-150}C_2$	-18.37	40.88	-44.44	-4.85	71.14	14.81
$A_{50-150}C_3$	-6.89	-39.78	-41.67	3.78	22.81	-8.33
$A_{50-150}C_4$	-8.46	-79.19	0.00	-4.39	14.20	11.76
$A_{50-150}C_5$	-20.03	-4.84	-90.48	-12.86	46.38	-9.52
$A_{50-150}C_6$	-20.31	5.84	-46.15	1.13	49.23	-3.85

one obtained by the NSGA-II algorithm, it is more than three times faster. Moreover, when BOCA^C is compared to PAES algorithm, the obtained S values are, in average, equivalent while BOCA^C is around 30% faster than PAES. PAES obtains, in average, more efficient points than BOCA^C though (9.32%).

Finally, Tables 9 and 10 show a comparison between BOCA^N and NSGA-II and PAES algorithms. BOCA^N performs quite similar to PAES algorithm with respect to both S value and the number of obtained efficient solutions. However, BOCA^N is faster than PAES. Similar situation occurs when BOCA^N is compared to NSGA-II algorithm. Although NSGA-II obtains better values for both S and $|\bar{X}|$, BOCA^N is much faster than NSGA-II. This situation can be explained by the very fast performance that our BOCA^N algorithm obtains for the set of small instances. When we look further at the results, we can note that if we only consider both medium and large size instances, execution times obtained by both algorithms are quite similar to each other. This result confirms what is outlined in [18] in the sense of the good performance that the BOCA algorithm shows. Furthermore, our results confirm this good performance with respect to other well-known EMO algorithms does not depend on which type of knowledge is considered. However, as we mentioned before, the choice of the knowledge used on the BOCA algorithm is an important issue and it has an impact on the algorithm performance.

4. Conclusions and Future Work

Evolutionary algorithms are a very good alternative to solve complex combinatorial optimisation problems. In this paper

TABLE 10: Comparison among our BOCA with normative knowledge and NSGA-II and PAES.

Instance	$\Delta_{S_4}^{S_3}$	$\Delta_{t_4}^{t_3}$	$\Delta_{ \widehat{X}_4 }^{ \widehat{X}_3 }$	$\Delta_{S_5}^{S_3}$	$\Delta_{t_5}^{t_3}$	$\Delta_{ \widehat{X}_5 }^{ \widehat{X}_3 }$
$B_{10-25}C_1$	0.00	-634.18	0.00	0.00	-102.55	0.00
$B_{10-25}C_2$	-0.74	-1017.19	-25.00	-0.74	-216.41	-25.00
$B_{10-25}C_3$	-1.81	-462.45	-8.33	-1.81	-89.08	-8.33
$B_{10-25}C_4$	-0.21	-686.94	0.00	-0.21	-98.20	0.00
$B_{10-25}C_5$	-11.96	-854.59	-50.00	-11.96	-96.22	-50.00
$B_{10-25}C_6$	-2.79	-414.69	-13.33	-2.79	-60.82	-13.33
$B_{30-75}C_1$	-9.56	-112.77	-15.38	-2.72	23.64	-3.85
$B_{30-75}C_2$	-2.13	-72.90	-63.33	3.28	36.17	-13.33
$B_{30-75}C_3$	-1.07	-96.93	-82.14	4.34	15.37	-89.29
$B_{30-75}C_4$	-7.80	-198.95	14.81	-5.34	41.42	22.22
$B_{30-75}C_5$	-11.09	-56.27	16.22	-7.15	38.50	29.73
$B_{30-75}C_6$	4.87	-158.41	-95.83	10.59	-3.54	-33.33
$B_{50-150}C_1$	-4.22	-204.86	-27.78	14.56	-44.63	44.44
$B_{50-150}C_2$	-5.34	-62.92	-70.83	30.33	11.67	45.83
$B_{50-150}C_3$	-11.42	-33.88	-300.00	4.64	25.03	-100.00
$B_{50-150}C_4$	-11.61	-330.96	41.18	-7.93	-97.35	17.65
$B_{50-150}C_5$	-16.76	-377.39	-29.63	8.21	-151.45	40.74
$B_{50-150}C_6$	-6.20	-82.12	-26.47	5.40	27.79	5.88

we have implemented a biobjective cultural algorithm to solve the well-known BOUFLP. We have considered two different sources of knowledge, namely, circumstantial and normative, and compare them with a previously implemented historical knowledge. Furthermore, we compare our BOCA approaches with two well-known EAs, namely, NSGA-II and PAES.

Although BOCA approaches using both normative and circumstantial knowledge could not improve the results obtained by the BOCA algorithm with the historical knowledge, results pointed out that performance of the BOCA algorithm depends largely on the selected knowledge and it can make the difference in terms of S value, time, and number of efficient solutions found by the algorithm. This is an important finding as it points out the relevance of the choice of a specific type of knowledge. Moreover, our results also confirm the good performance showed by the BOCA algorithm with respect to other well-known EMO algorithms such as NSGA-II and PAES algorithms. The BOCA algorithm is very competitive when compared to those EMO algorithms independently of the type of knowledge implemented.

As a future work we think that more investigation is needed in order to find patterns that allow us to get the right knowledge implemented depending on the problem features. As we mentioned before, the knowledge choice has an impact on the performance of the BOCA algorithm and therefore it must be studied in depth. Also, as future work, hybrid knowledge could be implemented in order to exploit the advantages of each kind of knowledge at the same time. Moreover, our BOCA algorithm can be used to solve other

interesting MOPs arising in the logistic field, such as routing or scheduling problems.

Appendix

Result Tables

In this appendix section, obtained results are presented. Columns $S_{\{\cdot\}}$ show the S value obtained by algorithm $\{\cdot\}$ as %. Algorithms are indexed as follows. The original BOCA algorithm is indexed by $\{1\}$. BOCA algorithms using circumstantial and normative knowledge are indexed by $\{2\}$ and $\{3\}$, respectively. Finally, the other EAs considered in this paper, namely, NSGA-II and PAES, are indexed by $\{4\}$ and $\{5\}$, respectively. Columns $t_{\{\cdot\}}$ show the time obtained by each algorithm in seconds. Columns $|\widehat{X}_{\{\cdot\}}|$ show the number of efficient solutions found by the corresponding algorithm. Finally operator $\Delta_{\{\cdot\}}^{\{\cdot\}}$ shows a value that is equivalent to $(\{\cdot\} - \{\cdot\})/\{\cdot\} \times 100$.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

- [1] D. Maravall and J. de Lope, "Multi-objective dynamic optimization with genetic algorithms for automatic parking," *Soft Computing*, vol. 11, no. 3, pp. 249–257, 2007.
- [2] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multiobjective optimization: Nsga II," in *Parallel Problem Solving from Nature PPSN VI*, M. Schoenauer, K. Deb, G. Rudolph et al., Eds., vol. 1917 of *Lecture Notes in Computer Science*, pp. 849–858, Springer, Berlin, Germany, 2000.
- [3] I. Borgulya, "An algorithm for the capacitated vehicle routing problem with route balancing," *Central European Journal of Operations Research*, vol. 16, no. 4, pp. 331–343, 2008.
- [4] P. J. Angeline, Z. Michalewicz, M. Schoenauer, X. Yao, and A. Zalzal, Eds., *The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Pareto Multiobjective Optimisation*, vol. 1, IEEE Press, 1999.
- [5] C. A. Coello Coello and M. Lechuga, "MOPSO: a proposal for multiple objective particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC '02)*, vol. 2, pp. 1051–1056, 2002.
- [6] W. K. Mashwani, "Comprehensive survey of the hybrid evolutionary algorithms," *International Journal of Applied Evolutionary Computation*, vol. 4, pp. 1–19, 2013.
- [7] R. Bhattacharya and S. Bandyopadhyay, "Solving conflicting bi-objective facility location problem by NSGA II evolutionary algorithm," *International Journal of Advanced Manufacturing Technology*, vol. 51, no. 1–4, pp. 397–414, 2010.
- [8] B. Crawford, C. Lagos, C. Castro, and F. Paredes, "A cultural algorithm for solving the set covering problem," in *Analysis and Design of Intelligent Systems using Soft Computing Techniques*, P. Melin, O. Castillo, E. Ramirez, J. Kacprzyk, and W. Pedrycz, Eds., vol. 41 of *Advances in Soft Computing*, pp. 408–415, Springer, Berlin, Germany, 2007.

- [9] Y. Guo, J. Cheng, Y. Cao, and Y. Lin, "A novel multi-population cultural algorithm adopting knowledge migration," *Soft Computing*, vol. 15, no. 5, pp. 897–905, 2011.
- [10] R. G. Reynolds, "An introduction to cultural algorithms," in *Proceedings of the 3rd Annual Conference on Evolutionary Programming*, pp. 131–139, World Scientific, 1994.
- [11] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*, Springer, Secaucus, NJ, USA, 2006.
- [12] C. A. Coello, C. Dhaenens, and L. Jourdan, *Advances in Multi-Objective Nature Inspired Computing*, Springer, 1st edition, 2010.
- [13] C. A. Coello and R. Landa, "Evolutionary multiobjective optimization using a cultural algorithm," in *Proceedings of the IEEE Swarm Intelligence Symposium*, pp. 6–13, IEEE Service Center, Piscataway, NJ, USA, 2003.
- [14] R. Zhang, J. Zhou, L. Mo, S. Ouyang, and X. Liao, "Economic environmental dispatch using an enhanced multi-objective cultural algorithm," *Electric Power Systems Research*, vol. 99, pp. 18–29, 2013.
- [15] S. Srinivasan and S. Ramakrishnan, "A social intelligent system for multi-objective optimization of classification rules using cultural algorithms," *Computing*, vol. 95, no. 4, pp. 327–350, 2013.
- [16] G. G. Cabrera, C. Vasconcellos, R. Soto, J. M. Rubio, F. Paredes, and B. Crawford, "An evolutionary multi-objective optimization algorithm for portfolio selection problem," *International Journal of Physical Sciences*, vol. 6, no. 22, pp. 5316–5327, 2011.
- [17] R. Reynolds and D. Liu, "Multi-objective cultural algorithms," in *Proceedings of the IEEE Congress of Evolutionary Computation (CEC '11)*, pp. 1233–1241, June 2011.
- [18] G. Cabrera, J. M. Rubio, D. Díaz, B. Fernández, C. Cubillos, and R. Soto, "A cultural algorithm applied in a BiObjective uncapacitated facility location problem," in *Evolutionary Multi-Criterion Optimization*, R. Takahashi, K. Deb, E. Wanner, and S. Greco, Eds., vol. 6576 of *Lecture Notes in Computer Science*, pp. 477–491, Springer, Berlin, Germany, 2011.
- [19] J. Knowles and D. Corne, "On metrics for comparing non-dominated sets," in *Proceedings of the Congress on Evolutionary Computation (CEC '02)*, vol. 1, pp. 711–716, Honolulu, Hawaii, USA, May 2002.
- [20] I. Kaliszewski, *Soft Computing for Complex Multiple Criteria Decision Making*, vol. 85 of *International Series in Operations Research & Management Science*, Springer, 2006.
- [21] M. Ehrgott, *Multicriteria Optimization*, Springer, Berlin, Germany, 2nd edition, 2005.
- [22] A. Farhang-mehr and S. Azarm, "Minimal sets of quality metrics," in *Proceedings of the 2nd International Conference on Evolutionary Multi-Criterion Optimization (EMO '03)*, Lecture Notes in Computer Science, pp. 405–417, Springer, 2003.
- [23] M. P. Hansen and A. Jaszkiewicz, "Evaluating the quality of approximations to the non-dominated set," Tech. Rep. IMM-REP-1998-7, Institute of Mathematical Modelling, Technical University of Denmark, 1998.
- [24] E. Zitzler, *Evolutionary algorithms for multiobjective optimization: methods and applications [Ph.D. thesis]*, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, 1999.
- [25] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.
- [26] J. G. Villegas, F. Palacios, and A. L. Medaglia, "Solution methods for the bi-objective (cost-coverage) unconstrained facility location problem with an illustrative example," *Annals of Operations Research*, vol. 147, pp. 109–141, 2006.
- [27] M. Ehrgott and X. Gandibleux, "Hybrid metaheuristics for multi-objective combinatorial optimization," in *Hybrid Metaheuristics*, C. Blum, M. J. B. Aguilera, A. Roli, and M. Sampels, Eds., vol. 114 of *Studies in Computational Intelligence*, pp. 221–259, Springer, Berlin, Germany, 2008.
- [28] J. Bramel and D. Simchi-Levi, *The Logic of Logistics: Theory, Algorithms, and Applications for Logistics Management*, Springer, New York, NY, USA, 1997.
- [29] M. S. Daskin, *Network and Discrete Location: Models, Algorithms, and Applications*, Wiley-Interscience, New York, NY, USA, 1st edition, 1995.
- [30] Z. Drezner and H. Hamacher, *Facility Location: Applications and Theory*, Springer, Berlin, Germany, 2002.
- [31] R. Z. Farahani, M. SteadieSeifi, and N. Asgari, "Multiple criteria facility location problems: a survey," *Applied Mathematical Modelling*, vol. 34, no. 7, pp. 1689–1709, 2010.
- [32] C. S. Revelle and G. Laporte, "The plant location problem: new models and research prospects," *Operations Research*, vol. 44, no. 6, pp. 864–874, 1996.
- [33] R. G. Reynolds, *New Ideas in Optimization*, McGraw-Hill, Maidenhead, UK, 1999.
- [34] R. Landa Becerra and C. A. Coello Coello, "A cultural algorithm with differential evolution to solve constrained optimization problems," in *Advances in Artificial Intelligence (IBERAMIA '04)*, C. Lemaitre, C. Reyes, and J. A. González, Eds., vol. 3315 of *Lecture Notes in Computer Science*, pp. 881–890, Springer, Berlin, Germany, 2004.
- [35] C. Soza, R. Landa, M. Riff, and C. Coello, "A cultural algorithm with operator parameters control for solving timetabling problems," in *Foundations of Fuzzy Logic and Soft Computing*, P. Melin, O. Castillo, L. Aguilar, J. Kacprzyk, and W. Pedrycz, Eds., vol. 4529 of *Lecture Notes in Computer Science*, pp. 810–819, Springer, Berlin, Germany, 2007.
- [36] M. Hofer, "UflLib, Benchmark Instances for the Uncapacitated Facility Location Problem," 2014.

Research Article

Towards Enhancement of Performance of K-Means Clustering Using Nature-Inspired Optimization Algorithms

Simon Fong,¹ Suash Deb,² Xin-She Yang,³ and Yan Zhuang¹

¹ Department of Computer and Information Science, University of Macau, Macau

² Department of Computer Science and Engineering, Cambridge Institute of Technology, Ranchi 835103, India

³ School of Science and Technology, Middlesex University, The Burroughs, London NW4 4BT, UK

Correspondence should be addressed to Simon Fong; ccfong@umac.mo

Received 4 May 2014; Accepted 9 June 2014; Published 18 August 2014

Academic Editor: T. O. Ting

Copyright © 2014 Simon Fong et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Traditional K-means clustering algorithms have the drawback of getting stuck at local optima that depend on the random values of initial centroids. Optimization algorithms have their advantages in guiding iterative computation to search for global optima while avoiding local optima. The algorithms help speed up the clustering process by converging into a global optimum early with multiple search agents in action. Inspired by nature, some contemporary optimization algorithms which include Ant, Bat, Cuckoo, Firefly, and Wolf search algorithms mimic the swarming behavior allowing them to cooperatively steer towards an optimal objective within a reasonable time. It is known that these so-called nature-inspired optimization algorithms have their own characteristics as well as pros and cons in different applications. When these algorithms are combined with K-means clustering mechanism for the sake of enhancing its clustering quality by avoiding local optima and finding global optima, the new hybrids are anticipated to produce unprecedented performance. In this paper, we report the results of our evaluation experiments on the integration of nature-inspired optimization methods into K-means algorithms. In addition to the standard evaluation metrics in evaluating clustering quality, the extended K-means algorithms that are empowered by nature-inspired optimization methods are applied on image segmentation as a case study of application scenario.

1. Introduction

Based on a partitioning strategy, K-means clustering algorithm [1] assigns membership to data points by measuring the distance between each pair of data point and centroid of a designated cluster. The membership assignment will be progressively refined until the best possible assignment is yielded—that is when the total intradistances of the data points within a cluster are minimized and the total interdistances of the data points across different clusters are maximized. The final quality of the clustering results, however, depends largely on the values of the initial centroids at the beginning of the partitioning process. These initial centroid values are randomly generated each time the clustering kick-starts which are different from time to time. By such random chance, K-means can probably plunge into local optima whereby the final quality of the clusters falls short from the globally best. An example that is depicted in Figure 1

demonstrates some possible outcomes of K-means. The top two snapshots represent good clustering results where the cluster distributions are even; the bottom two snapshots show otherwise, the clustering results in uneven distribution. All these depend on the starting positions of the centroids which are randomly generated.

Technically it is possible though not feasible in achieving a globally optimum clustering result, via a brute-force approach in trying out exhaustively all partitioning possibilities. As the number of clusters and the number of data points increase, the combinatorial number of possible grouping arrangements escalates, leading to computationally prohibitive. Therefore, heuristic approach is desired for seeking global optima stochastically, improving the quality of the final clustering results iteration by iteration. Metaheuristics which enable incremental optimization by design are ideal candidates for such computation.

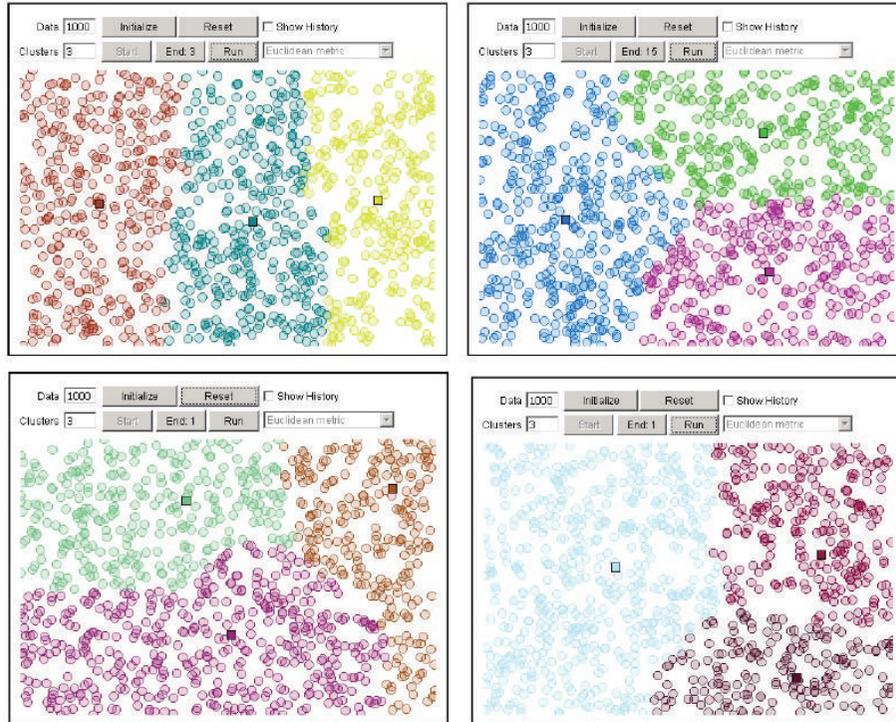


FIGURE 1: Examples of clustering results by random centroids of K-means where $k = 3$.

A substantial collection of nature-inspired optimization methods aka metaheuristics have emerged recently with designs mimicking swarm behavior exhibited by living creatures. Each of the search agents represents a particular combination of centroid positions; they move and search for optimality in their own ways, they sometimes communicate with each other and are collectively guided towards a global optimization goal. To date, the proposed nature-inspired optimization algorithms have gained much attention among data mining researchers. The computational merits have been verified mathematically and their feasibilities have been applied in various practical applications. However, validating the efficacy of hybrids combining such nature-inspired algorithms with classical data mining algorithms is still at an infant stage [2].

By the merits of design, nature-inspired optimization algorithms are believed to be able to overcome the shortcomings of K-means clustering algorithms on the issue of getting stuck in local optima. The objective of this study is to validate the efficacy of the hybrids and to quantitatively measure the quality of clustering results produced by each of the hybrids. In our experiments, we used four popular nature-inspired optimization methods to combine with K-means clustering algorithm. The integration is paramount, because it enables enhancement over data mining algorithms which have many wide applications.

A preliminary experiment reported in [3] shows their feasibility as a successful pioneer exemplar. This paper reports further experiment including a case study of image segmentation using these hybrid algorithms. In the past, some researchers started to integrate nature-inspired optimization methods

into K-means algorithms [4]; their research efforts are limited to almost the same form of swarming maneuvers—that is, there is constantly a leader in the swarm which the fellow agents follow. Some examples are nature-inspired optimization algorithms such as the Artificial Bee Colony [5], Firefly [6], and Particle Swarm Optimization [7]. For the sake of intellectual curiosity, in our experiments as well as whose models described in this paper, two nature-inspired algorithms which take on a slightly different course of swarming are included. They are the Bat Algorithm [8] which swarm with varying speeds and the Cuckoo Algorithm [9] which do not swarm but iterate with fitness selection improvement. They represent another two main groups of algorithms and their variants, which are adaptive to the environment by their sensing abilities, and utilize a special method to improve the solution by evolving the old solutions from one generation into better ones in new generations. Specifically, the performance indicators in terms of speed and time consumption for clustering by these two bioinspired algorithms integrated with K-means are observed. The technical details of the aforementioned nature-inspired algorithms and the K-means clustering are not duplicated here. Readers are referred to the respective references for the background information of the algorithms involved in this paper.

2. Enhancing K-Means Clustering by Nature-Inspired Optimization Algorithms

A major reason in obtaining quality K-means clustering results is having the right combination of centroid positions.

The resultant centroids ideally should be dispersed in such a way that the clusters formed upon them yield the maximum quality, which we call a global optimum. It is characterized by having the properties of maximum intrasimilarities and minimum intersimilarities in clustering. K-means is known to produce clustered data with nonoverlapping convex clusters, and they always converge quickly. One drawback is that clusters in K-means do not always converge into the global optimum. Just like any other partition-based clustering algorithm, initial partition is made by some randomly generated centroids which do not guarantee the subsequent convergence will lead to the best possible clustering result. K-means, for example, is reputed to have the final clusters stuck at local optima which prevent further exploration for the best results [10]. In reality, to achieve the best clustering results, it would require running many rounds with each round taken different random initiations of centroid values. This practice certainly would have to be done at the very high cost of computation and model training time. In [3], the authors first proposed the integration of nature-inspired optimization algorithms into K-means. We take a step further in the experiments and applying it for image segmentation. To start with the integration, the formation of centroids, which are computed stochastically from start to convergence, is directed by the searching agents of the nature-inspired optimization algorithms. The evolution of the new generation is based on the principle that the centroids which are being relocated in each iteration are inclined to enable the new clusters that are being formed with better results. Hence, in order to achieve the optimal configuration of centroids as an objective, we let $cen_{j,v}$ be the centroid as the center point of the j th cluster in the multidimensional search space by the v th attribute. $W_{i,j}$ is the membership of data point x_i whether it exists in cluster j . The centroid location can be calculated by (2) for each attribute v and for each cluster j ; the clustering objective function is defined as (3):

$$w_{i,j} = \begin{cases} 1, & x_i \in \text{cluster } j, \\ 0, & x_i \notin \text{cluster } j, \end{cases} \quad (1)$$

$$cen_{j,v} = \frac{\sum_{i=1}^S w_{i,j} x_{i,v}}{\sum_{i=1}^S w_{i,j}}, \quad j = 1, \dots, K, \quad v = 1, \dots, K * D, \quad (2)$$

where S is the number of search agents in the whole population, K is the maximum number of clusters, and j is the current cluster being processed. The highest dimension of attributes is D for the dataset; a centroid is hence located by a tuple of size D . In the design of our computational model, cen is a 2D matrix of size $K \times D$ holding all the values of the centroids (the cluster centers) indicated by $cen_{j,v}$:

$$F(cen) = \sum_{j=1}^K \sum_{i=1}^S w_{i,j} \sum_{v=1}^{K*D} (x_{i,v} - cen_{j,v})^2. \quad (3)$$

The computation process scans through the cen matrix up to $K \times D$ times to check the values of all the attributes of the data point x for measuring the distance or similarity

between each pair of x and the centroid. This process repeats for each cluster v . For the optimization algorithm to work, each searching agent represents a particular combination of centroids for all the clusters, as an optimization solution in the $K \times D$ dimensional search space. The best search agent being found in each iteration is supposed to produce the best clustering result in that particular iteration. For instance, in a simple dual-cluster clustering task, there are three variables for the objective function to work with. In this case, there are three dimensions in the search space. In the three dimensional search space, the i th search agent may take the form of $x_i = (i, [x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4}, x_{i,5}, x_{i,6}])$. Due to the fact that there are 2×3 attributes for a search agent, the centroid can be coded in the same format for the coordinate of the second dimension. A search agent may have a best fitness value as $cen = (x_{i,1}, x_{i,2}, x_{i,3}, x_{i,4}, x_{i,5}, x_{i,6})$. According to the given definitions, the clustering strategy can be constructed as a minimization function, as follows, that aims at shortening the distances among the data points in a cluster:

$$clmat_{i,j} = \min_{k \in K} \{ \|x_i - cen_k\| \}. \quad (4)$$

The ranges of the parameters are as follows: $i = 1, \dots, N$, $j = 1, \dots, S$, and $k = 1, \dots, K$. The double line notation in (4) means it is a function of Euclidean distance. The interpretation of (4) is that the i th search agent that is now handling the k th cluster takes a value by measuring the minimized distance between the i th search agent and centroid of the k th center. The equation is an objective function in which the smaller the value the better. As long as the value of $clmat$ is minimized by this metaheuristic optimization approach, every data point within a cluster would be drawn as close as possible to the centroid. The metaheuristic will guide the search agents to find the appropriate centroids for the clusters.

Nature-inspired optimization algorithms require certain functional parameters to be initiated with values to run. The function parameters are defined as follows. They allow the users to set with user-specific values for customizing the operations of the algorithms. Some of the parameters are common across different bioinspired optimization algorithms. In this paper, we have four hybrids, which resulted from combining four bioinspired optimization algorithms into K-means. With the capital letter C denoting "clustering," the four hybrid algorithms are called C-ACO, C-Bat, C-Cuckoo, and C-Firefly, respectively. The original mathematical models for the four bioinspired optimization algorithms can be found in [6, 8, 9, 11], respectively.

Table 1 consists of the parameters for the C-Firefly algorithm. X is a composite matrix of size $[N, (K \otimes D)]$, where $x \in X$ since X has a maximum of K centroids and each centroid is represented by a maximum of D dimensions by the attributes.

The C-Bat algorithm has more parameters than the others because it includes the velocity and location of each bat (Table 3). Velocity is determined by frequency, loudness, and pulse rate. However, only two of the four bioinspired clustering algorithms (C-Cuckoo and C-Bat) are described in this paper due to space limitations. Nonetheless, C-Firefly

TABLE 1: Parameters used for C-Firefly.

N	Population of fireflies
M	Number of pseudo time steps
K	Number of groups or clusters
S	Number of samples
D	Number of attributes
	The element in matrix $[s, k]$ where $s \in S, k \in K$
$w(s, k)$	$w_{s,k} = \begin{cases} 1, & x_s \in \text{cluster}_k \\ 0, & x_s \notin \text{cluster}_k \end{cases}$
$x(i, j)$	The best solution in matrix $[i, j]$ where $i \in N, j \in K \otimes D$
$\text{Cen}(k, d)$	The centroid in matrix $[k, d]$ where $k \in K, d \in K \otimes D$
$\text{Clmat}(n, s)$	The classification matrix $[n, s]$ where $n \in N, s \in S$

Table 1 consists of the parameters for the C-Firefly algorithm. X is a composite matrix of size $[N, (K \otimes D)]$, where $x \in X$ since X has a maximum of K centroids and each centroid is represented by a maximum of D dimensions by the attributes.

and C-Bat have recently been reported in [1]. Readers may refer to [1] for the detailed integration of a K-Means clustering algorithm with the firefly and bat algorithms.

2.1. Cuckoo Clustering Algorithm (C-Cuckoo). In the original cuckoo algorithm, Yang and Deb used an analogy whereby each egg in a nest represents a solution, and a cuckoo egg represents a new solution. The goal is to use the new and better solution to replace a relatively poor solution by chasing off an old egg with a new cuckoo egg. We adopt the same analogy in constructing our C-Cuckoo algorithm. The solution x represents the host nest. In the clustering algorithm, the solution is composed of a set of real numbers representing the cluster center. As defined earlier, x takes the form of a $(N, K \otimes D)$ matrix where N is the population, K is the number of clusters, and D is the number of attributes associated with each data point. The second index of the matrix represents the center of all K clusters, and the whole x represents the current locations of all the cuckoos. We now give a simple example. If there is a given data set and two dimensions and we need to create two clusters ($K = 2, D = 2$), then the value of x is four. The i th x in the middle of the clustering process may look something like this:

Cluster 1: 1 3
Cluster 2: 8 9

In the initialization phase, the population of n host nests x_i , where $i = 1, 2, \dots, n$, is generated. The cluster centers are represented by the means of the attributes. Each cuckoo has the same parameters (Table 2): Tol (tolerance) and pa (alien egg discovery rate). In this phase, the most important action is that a cluster ID is randomly assigned to each cuckoo as the initial clustering result.

Because the cuckoo has characteristics typical of Levy flight, when it comes to generating a solution $x^{(t+1)}$ for cuckoo i , we use the equation $x^{(t+1)} = x^{(t)} + \delta \oplus \text{Levy}(\lambda)$, where δ is

TABLE 2: Parameters used for C-Cuckoo.

N	Population of cuckoos
M	Number of pseudo time steps
Pa	Discovery rate of alien eggs
Tol	Tolerance
K	Number of groups (clusters)
S	Number of samples
D	Number of attributes
	The element in matrix $[s, k]$ where $s \in S, k \in K$
$w(s, k)$	$w_{s,k} = \begin{cases} 1, & x_s \in \text{cluster}_k \\ 0, & x_s \notin \text{cluster}_k \end{cases}$
$x(i, j)$	The best solution in matrix $[i, j]$ where $i \in N, j \in K \otimes D$
$\text{Cen}(k, d)$	The centroid in matrix $[k, d]$ where $k \in K, d \in K \otimes D$
$\text{Clmat}(n, s)$	The classification matrix $[n, s]$ where $n \in N, s \in S$

Pa and Tol are conditional variables used for controlling execution of the cuckoo optimization algorithm [9].

TABLE 3: Parameters used for C-Bat.

Q	Frequency
V	Velocity
R	Pulse rate
A	Loudness
...	All other parameters are the same as those in Table 1

the step size scalar used to control the resolution of the step length. In our algorithm, we use $\delta = 1$, which satisfies most cases. The above formula means the cuckoo takes a random walk. In this case, the random walk is implemented as a Levy flight, which is based on Mantegna's algorithm [12]. The algorithm takes the following steps. First, the number of centroids k is initialized, as are the other variables. By going through a random walk, the nest, which is regarded as the central point of a cluster, is updated. The step length s is calculated by $s = u/|v|^{1/\beta}$ and $\beta \in [1, 2]$, where u and v are drawn from normal distributions. That is, $u \sim N(0, \sigma_u^2)$ and $v \sim N(0, \sigma_v^2)$, where $\sigma_u = [\Gamma(1 + \beta) \sin(\pi\beta/2) / \Gamma((1 + \beta)/2) \beta 2^{(\beta-1)/2}]^{1/\beta}$. This distribution follows the Levy distribution.

The goal of this clustering algorithm is to search for the best center to minimize the distance between the center of the cluster and its points. Our objective function is, thus, the same as (3) and its result is the degree of fitness. After calculating the degree of fitness, we use (4) to assign each point to a suitable cluster. A better degree of fitness represents a good quality cuckoo solution.

The best solution derived from the above equations is then nominated, and the new solution replaces the old. If no new solution is better than the old one, the current solution is retained as the best. The host bird cleans out the nest according to a given probability. The next iteration of the computation process then occurs to look for the next best solution. In each iteration, the best and hence the optimal solution to date is set as the clustering centroid.

The centroid is represented as a paired tuple, $cen(i, :)$, where i is the central point of the cluster. The tuple has the format of (k, d) , where k is the k th cluster and d is the coordinates in $(x, y, z, \dots \text{etc.})$ or higher dimensions. For example, the locations of three clusters can be represented as $[1, (3, 4, 5), 2, (8, 8, 9), 3, (5, 6, 4)]$. Each cuckoo represents a cluster with coordinates (k, d) , and d is the central coordinate of the cuckoo. Each cuckoo is initially assigned a random d value. Subsequently, d is updated by iterative optimization. The progressive search for the best solution helps to avoid local optima in a manner similar to chromosome mutations in genetic algorithms. When the locations of the cuckoos are set in each round, the distances between the data points and the centers are measured by their Euclidean distance. The data points are reassigned to their nearest cluster by

$$\text{MIN} \{x - cen\}. \tag{5}$$

The clusters are then reformed on the basis of the newly assigned data points. At the beginning, the averages of the data points are used as starting centroids by

$$cen = \frac{1}{N} \sum_{x_i \in \text{cluster}_i} x_i, \quad i = 1, \dots, K. \tag{6}$$

In this way, the algorithm achieves better partitioning of clusters at the start to avoid the center points being too near or too far from each other, as would occur if they were assigned purely by random chance. As the algorithm runs, the clustering distribution is refined and changes to quality centroids are avoided by averaging the data. This is why (6) is only needed at the beginning to initialize the starting centroids. According to survival of the fittest, the partitioning process reaches a final optimum. The logic of the C-Cuckoo algorithm is shown as a flow chart in Figure 2.

2.2. Bat Clustering Algorithm (C-Bat). Each bat has its own unique ID, position, and fitness attributes. However, in the bat algorithm, each bat is assigned the same loudness, pulse frequency, and wave length. The position of a bat is represented by a solution x . Its location is determined by the values of D dimensions. As for the C-Cuckoo algorithm, the solution uses a $(N, K \otimes D)$ matrix, the second term identifying the location of the bat.

The initiation step is similar to that employed for the C-Cuckoo algorithm. However, the bats have an additional feature: each bat has a velocity v_i , which is similar to particle swarm optimization. The bat's position is partly determined by its velocity. At first, the bats are randomly distributed. After initialization, the bats move to a better place according to (8). A random number is then produced: if it is larger than the current bat rate, the algorithm selects a solution from those calculated and generates a local solution. The centroids are the averages of the nearby data points. The distances are then minimized according to the direction of the optimization goal. The objective functions are identical to (5) and (6) above. The convergence process then starts to iterate based on the following formula:

$$\beta = Q \times \text{rand}(), \tag{7}$$

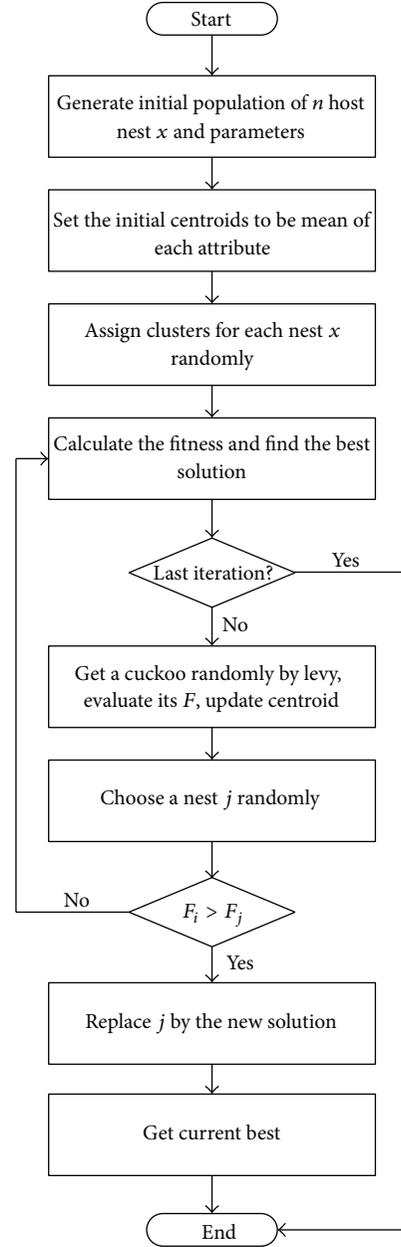


FIGURE 2: Workflow of C-Cuckoo algorithm.

where $\text{rand}()$ is a random value generator and the random values are distributed over $[0, 1]$. Consider

$$\begin{aligned} f_i &= f_{\min} + (f_{\max} - f_{\min}) \times \beta, \\ v_i^t &= v_i^{t-1} + (x_i^{t-1} - x_*) \times f_i, \\ x_i^t &= x_i^{t-1} + v_i^t. \end{aligned} \tag{8}$$

The positions of the bats are then updated. f represents the frequency of echolocation. When the frequency equals the sum of the minimum frequency and the difference between the maximum and minimum frequencies, the speed of the bat is updated. The new speed is set to the previous

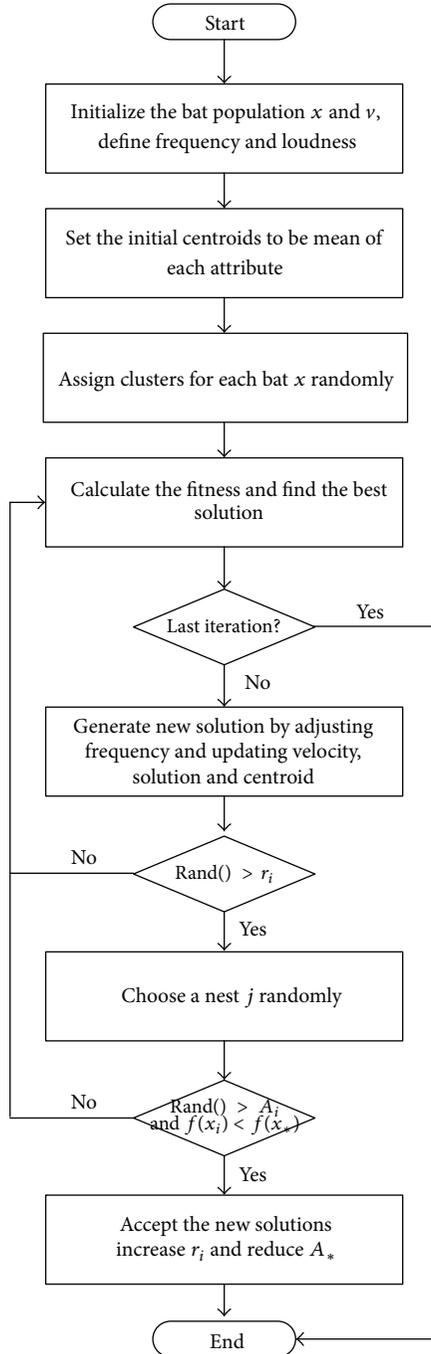


FIGURE 3: Workflow of C-Bat algorithm.

speed plus the product of the previous frequency and the difference between the current position and the previous position. A variable called the pulse rate is also used in the algorithm. When the pulse rate is exceeded, the following formula is updated:

$$x_i^t = x_* + 0.01 \times \text{rand}(). \quad (9)$$

Equation (9) serves as the updating function, where x_* is taken as the best solution. It is also used to represent the best position for the bat to move towards. If the loudness value is

TABLE 4: Testing dataset information.

Dataset	Instances	Attributes	Clusters
Iris	150	4	3
Wine	178	13	3
Haberman's survival	306	3	2
Libras	360	91	15
Synthetic	600	60	6

not high enough and the new solution is better than the old one, the better one becomes the solution. A fitness function the same as that employed for the C-Cuckoo algorithm is then applied by checking whether echolocation is loud enough. The logic of the C-Bat algorithm is shown as a flow chart in Figure 3.

3. Experiments

There are two sets of experiments; one is focused in evaluating the performance of algorithms using a series of multivariate real-life datasets. The other is to test out the efficacy of the algorithms in image segmentation. The purpose is to validate the new algorithms with respect to their clustering quality. The first test is about how the new algorithms work with general-purpose datasets with different number of attributes and instances. Their performances in particular are evaluated in details. The latter test is to observe how well these algorithms will work in the domain of machine vision. The setups of the experiments and their results are discussed as follow.

3.1. Experiment Set-Up. The new nature-inspired clustering algorithms (C-ACO, C-Bat, C-Cuckoo, and C-Firefly) proposed here are experimented over six datasets which are available for download from UCI data repository (<http://archive.ics.uci.edu/ml>). The clustering results of the new hybrid clustering algorithms are compared with those of original K-means which serves as a benchmarking reference. The computer simulation environment is implemented in MATLAB software and the algorithms are coded in MATLAB programs. The hardware platform is a MacBook Pro computer configured with a 2.3 GHz CPU and 4 GB RAM. In each trail run, each of the testing datasets is run repeatedly ten times for measuring up the average CPU time consumption, as well as obtaining the average values of the top objective fitness. The datasets used are collected from real-life applications, namely, Iris, Wine, Haberman's survival, Libras, and Synthetic. Synthetic is a dataset of control chart time-series which are artificially generated clustering data point values in random. The information regarding the testing datasets is shown in Table 4.

The full length of dataset is used for training—in clustering, building clusters are referred to until perfection is attained using the full set of data. Performance of the clustering is evaluated in terms of cluster integrity which is reflected by the intra- and intersimilarities of data points within and across different clusters, the average sum of

CPU time consumption per iteration during the clustering operation, and the number of loops taken for all the clusters to get converged. The criterion for convergence which decides when the looping of evolution stops is the fraction of the minimum distance between the initial cluster centers that takes on a numeric value between [0, 1]. For example, if the criterion is initialized with 0.01, the looping halts when a complete iteration does not move any of the cluster centers by a distance of more than 1% of the smallest distance between any of the initial cluster centers.

The quality of the final outcome of clustering is measured by the integrity of each of the clusters, which in turn is represented by the final fitness value of the objective function. The resultant fitness value of the objective function is driven by how much each variable contributes towards the final goal which is being optimized in the process. From the perspective of clustering the goal is finding a suitable set of centroids as guided by the metaheuristic of the nature-inspired algorithm. The metaheuristic will always insist that the relocation of centroids in each step is progressive aiming at exploring for the optimum grouping. The end result of the ideal group should lead to having the data points within each cluster closest to their centroid. Iteratively, the search for the optimum grouping proceeds. During the search the k centroids relocate in the search space step-by-step according to the swarming pattern of the nature-inspired optimization algorithm until no more improvement is observed. It stops when there is no further relocation that will offer a better result. To be precise, no other new relocation of centroids seems to provide better integrity of the clusters. The algorithm is geared at minimizing the intrasimilarity of each cluster by an objective function. In this case, it is a squared error function so any slight difference will be enlarged by the square function. Equation (10) defines such objective function. Consider

$$V = \sum_{j=1}^K \sum_{i=1}^N \|x_{i,j} - cen_j\|^2. \tag{10}$$

In the experiments, each dataset is run ten times to test and obtain the average CPU time and is also run ten times to test the objective function values/best fitness value. The parameters are set as reported in Tables 5 and 6.

3.2. Testing the New Clustering Algorithms with General-Purpose Multivariate Datasets. The five diagrams in Figure 4 below present snapshots of the experimental run for the Iris dataset. The original data points are shown in the topmost plot Figure 4(a), and the data points in different colors obtained by the new clustering algorithms are shown in Figures 4(b), 4(c), 4(d), and 4(e) by C-Firefly, C-Cuckoo, C-ACO, and C-Bat, respectively.

The quantitative experimental results are shown in Tables 7, 8, 9, 10, and 11. To make observation easier, the best result across the four algorithms under test (in each column) is highlighted with a double asterisks. It is apparent to observe that the C-Cuckoo and C-Bat algorithms achieve a lot better of objective fitness value than do the C-ACO and C-Firefly algorithms. Overall, the four nature-inspired

TABLE 5: Parameters set for C-Firefly and C-Cuckoo.

C-Firefly	C-Cuckoo
α (randomness): 0.2	Pa (discovery rate): 0.25
γ (absorption): 1.0	Tol (tolerance): $1.0e^{-5}$
Number of clusters: 4	β (Levy): $\frac{3}{2}$
Max number of iterations: 20	ϵ (Levy): $\sigma_u = \left[\frac{\Gamma(1 + \beta) \sin(\pi\beta/2)}{\Gamma[(1 + \beta)/2] \beta 2^{(\beta-1)/2}} \right]^{1/\beta}$
Population: 400	Number of clusters: 4
$\beta = 0$	Max number of iterations: 20
$\beta_0 = 0$	Population: 400

TABLE 6: Parameters set for C-Bat and C-ACO.

C-Bat	C-ACO
A (loudness): 0.5	Population: 400
R (pulse rate): 0.5	q (threshold): 0.9
Number of clusters: 4	Number of clusters: 4
Max number of iterations: 20	ρ (evaporation rate): 0.1
Q_{\min} (frequency min): 0	Number of clusters: 4
Q_{\max} (frequency max): 0.2	Max number of iterations: 20
Population: 400	

TABLE 7: Testing objective function fitness and CPU time consumption on IRIS.

	Objective function value		
	Best	Worst	Average
K-means	78.9451	152.3687	127.8941
C-ACO	139.3081	150.9815	144.79
C-Firefly	78.9408**	109.4036	89.88241
C-Cuckoo	78.9408**	78.9408**	78.9408**
C-Bat	78.9408**	81.2655	79.46626
	CPU time (second)		
	Best	Worst	Average
K-means	51.291414	51.673431	51.444755
C-ACO	440.7980	443.9105	443.0668
C-Firefly	142.899517	177.743484	167.2426438
C-Cuckoo	15.499098	15.798136	15.7319299
C-Bat	8.6212**	8.8354**	8.75495**

clustering algorithms execute in less time and succeed in achieving higher accuracy in clustering than the plain K-means. High accuracy is referred to high cluster integrity where the data points in a cluster are close to their centroid. This observation tallies with our proposition that K-means enhanced by nature-inspired optimization algorithms speeds up the searching time for the good centroids for good clustering integrity. This enhancement is important because for all the partitioning-based clustering methods potentially they can be enhanced by nature-inspired algorithms in the similar way—the end result is expected in search process acceleration and local optima avoidance.

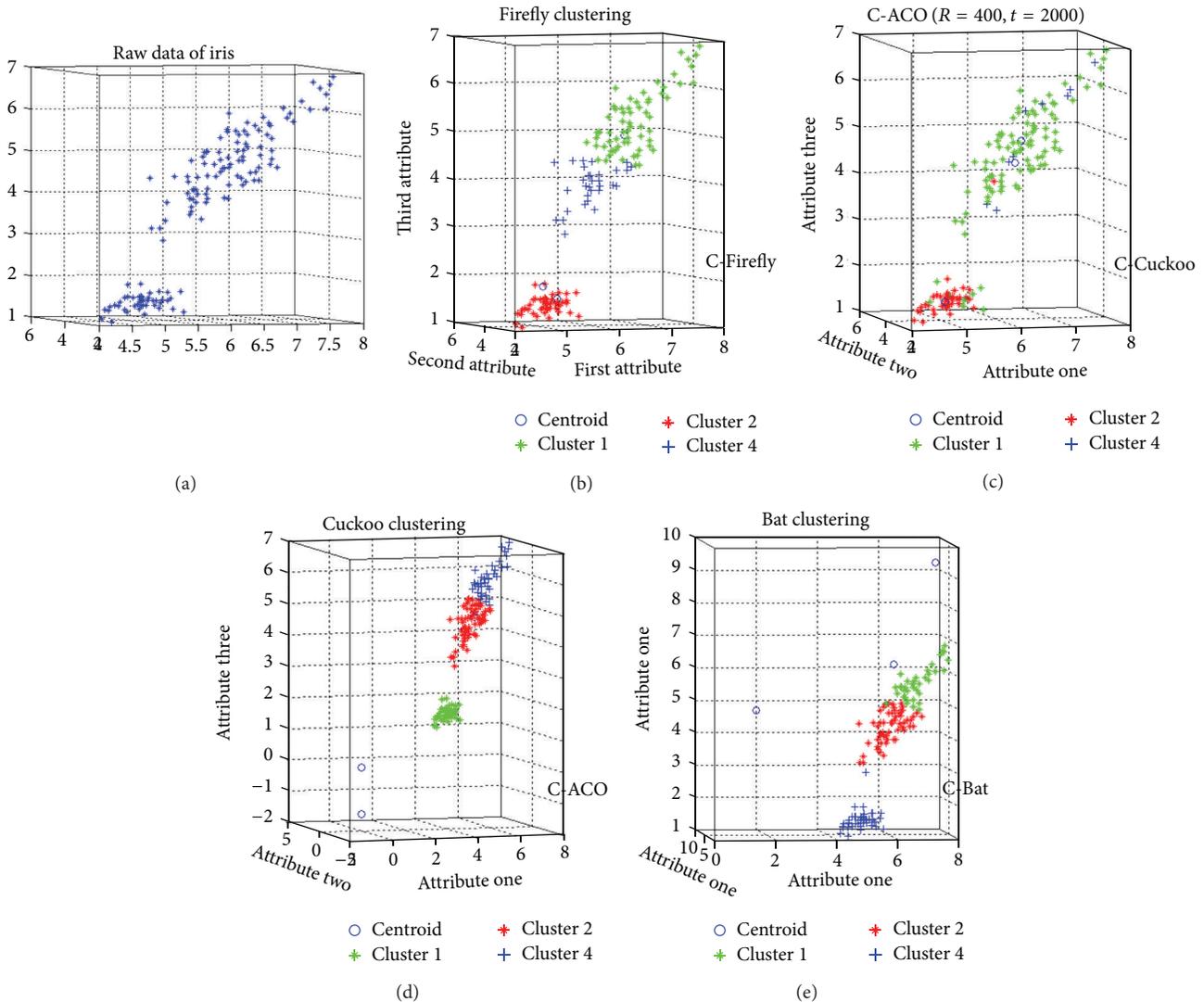


FIGURE 4: Snapshots of clustering operations.

Our detailed performance evaluation tests include computing the final objective function fitness values and the CPU time consumption for clustering the data in the general-purpose UCI datasets in Table 4. The objective function fitness value is computed using (10) which represents the overall cluster integrity, and CPU time consumption is timed as how long it is necessary for the clustering algorithm to converge from beginning to end. Given the datasets are of fixed volume, CPU time consumption is related directly to the speed of the clustering operation.

Tables 7 to 11 clearly show that the C-Cuckoo and C-Bat algorithms both yield better objective values than the C-ACO and C-Firefly algorithms. The study reported in [1] has already shown that the C-ACO and C-Firefly algorithms perform more quickly and accurately than a traditional K-means specification. Our evaluation results provide further evidence confirming this phenomenon: nature-inspired algorithms do indeed accelerate the process of finding globally optimum centroids in clustering, and partitioning clustering methods

can be combined with nature-inspired algorithms to speed up the clustering process and avoid local optima. Furthermore, our results show that two new hybrid clustering algorithms, the C-Cuckoo and C-Bat specifications, are more efficient and accurate than the others we test.

The next experiment is undertaken to measure the average computation time required per iteration in the clustering process. Only the Iris dataset is used here, as it is one of the datasets in the UCI repository most commonly used for testing time spent per iteration for clustering with nature-inspired algorithms.

From Figures 5 and 6, it can be seen that all four algorithms scale quite well—as the number of iterations increases, the computation time taken remains flat. In particular, C-ACO is very fast. It takes only a fraction of a second to execute each iteration of the clustering process. C-Firefly takes about 8 to 10 seconds. C-Cuckoo and C-Bat are relatively fast, taking less than a second per iteration for code execution. The CPU times taken for each algorithm are reported in Table 12. The

TABLE 8: Testing objective function fitness and CPU time consumption on wine.

	Objective function value		
	Best	Worst	Average
K-means	2.3707e + 006	2.3707e + 006	2.3707e + 006
C-ACO	2.3707e + 006	2.3707e + 006	2.3707e + 006
C-Firefly	2.3707e + 006	2.3707e + 006	2.3707e + 006
C-Cuckoo	2.3707e + 006	2.3707e + 006	2.3707e + 006
C-Bat	2.3707e + 006	2.3707e + 006	2.3707e + 006
	CPU time (second)		
	Best	Worst	Average
K-means	73.793875	77.226464	75.7386
C-ACO	623.3123	695.2312	653.2154
C-Firefly	260.725002	278.206271	267.8236183
C-Cuckoo	18.582329	19.137426	18.7511367
C-Bat	10.3032**	10.8225**	10.42148**

TABLE 9: Testing objective function fitness and CPU time consumption on Libras.

	Objective function value		
	Best	Worst	Average
K-means	822.8381	899.2441	842.3452
C-ACO	1.1361e + 003	1.6345e + 003	1.3981e + 003
C-Firefly	743.3432	892.0506	777.1993
C-Cuckoo	707.5916**	819.1392**	763.1102**
C-Bat	745.8008	918.2488	841.71931
	CPU time (second)		
	Best	Worst	Average
K-means	1332.059811	1392.3113	1362.0344
C-ACO	1.0142e + 004	1.3245e + 004	1.1195e + 004
C-Firefly	260.725002	278.206271	267.8236183
C-Cuckoo	168.502947	169.849548	169.17942
C-Bat	10.3032**	10.8225**	10.42148**

TABLE 10: Testing objective function fitness and CPU time consumption on Haberman.

	Objective function value		
	Best	Worst	Average
K-means	30507.0207	31321.2134	30912.2141
C-ACO	2888.4833**	3051.1412**	2933.1641**
C-Firefly	30507.2735	31567.7231	30822.3426
C-Cuckoo	30507.0207	30574.2412	30591.1234
C-Bat	30507.8928	31455.1241	30712.8321
	CPU time (second)		
	Best	Worst	Average
K-means	170.0535	174.123152	172.743
C-ACO	825.2858	856.3724	831.223
C-Firefly	1257.6772	1289.3124	1266.4123
C-Cuckoo	25.1611	26.2312	25.4431
C-Bat	14.5597**	14.9087**	14.6722**

TABLE 11: Testing objective function fitness and CPU time consumption on Synthetic.

	Objective function value		
	Best	Worst	Average
K-means	9.8221e + 005	1.0451e + 006	9.9632e + 005
C-ACO	43847.7264**	5.4264.3234**	47321.9682**
C-Firefly	9.4509e + 005	9.8221e + 005	9.6622e + 005
C-Cuckoo	9.4499e + 005	9.7800e + 005	9.5721e + 005
C-Bat	9.5232e + 005	9.8525e + 005	9.6150e + 005
	CPU time (second)		
	Best	Worst	Average
K-means	1579.2311	1663.2234	1617.305490
C-ACO	4354.9328	4725.4525	4556.2073
C-Firefly	51092.57336	51404.66323	51232.55324
C-Cuckoo	138.204069	151.547294	141.9006882
C-Bat	77.8049**	80.8928**	79.63274**

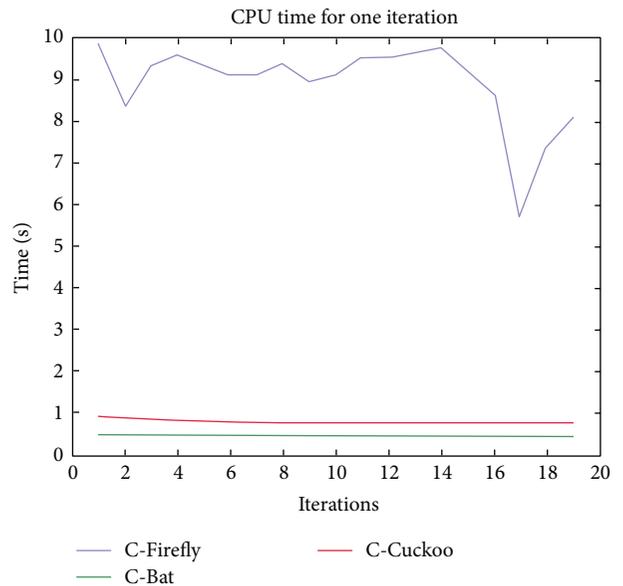


FIGURE 5: Computation time (in secs.) for C-Firefly, C-Cuckoo, and C-Bat algorithms.

figures are averaged, and the table shows the net CPU time taken per iteration.

The following graphs in Figures 7 and 8 show the number of iterations required for the clustering algorithms to converge according to the given threshold criterion.

As shown in Figure 7, the C-Firefly, C-Cuckoo, and C-Bat algorithms take about two or three iterations to achieve convergence, which is extremely fast. In contrast, C-ACO in Figure 8 takes many rounds to converge, 4681 iterations to be exact. For the other three algorithms, the best objective function value is reached at 78.94. C-ACO goes no lower than 101 and remains there even if the number of iterations increases to a large value.

Therefore, we may conclude that the C-Firefly, C-Cuckoo, and C-Bat algorithms are suitable for static data. C-Firefly

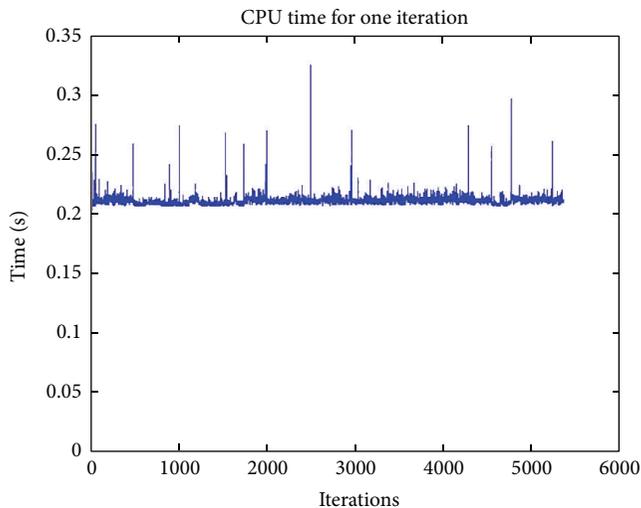


FIGURE 6: Computation time (in secs.) for C-ACO algorithm.

TABLE 12: Comparison of algorithms with respect to CPU time taken per iteration.

CPU time per iteration			
Iris			
	Best	Worst	Average
C-ACO	0.2148**	0.2625**	0.2260**
C-Firefly	8.1591	9.8726	8.5261
C-Cuckoo	0.7824	0.8812	0.8035
C-Bat	0.4340	0.5066	0.4448
Wine			
	Best	Worst	Average
C-ACO	0.3362*	0.5362*	0.4271*
C-Firefly	10.6828	16.4298	12.5449
C-Cuckoo	0.937	1.0074	0.95102
C-Bat	0.5231	0.5576	0.52809
Haberman			
	Best	Worst	Average
C-ACO	0.4156*	0.4551*	0.4243*
C-Firefly	64.3562	67.9928	66.1953
C-Cuckoo	1.2376	1.3037	1.2581
C-Bat	0.4340	0.5066	0.4448

compares data $O(n^2)$ times in each iteration, so it takes a lot of time to converge. The traditional K-means algorithm converges easily to a local optimum, so the result of the objective function is worse than for the others. C-Bat has the ability to adjust itself in every iteration, and because it only changes location once, at the end of an iteration, it is very fast. Because C-Cuckoo retains better solutions and discards worse solutions, working like a PSO-clustering algorithm, it also performs well in providing objective function values.

Although C-Bat, C-Cuckoo, and C-Firefly may need more time for each iteration, they are good optimization algorithms. They can find the optimal solution relatively quickly overall (because they converge very fast). However, the ants

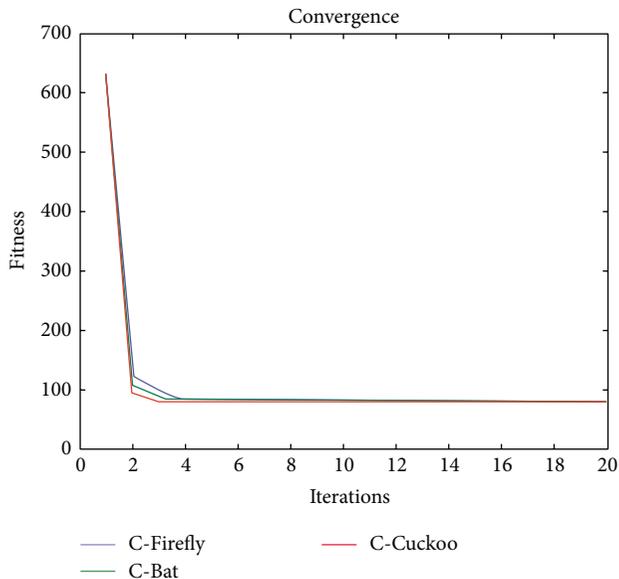


FIGURE 7: Number of iterations required for C-Firefly, C-Cuckoo, and C-Bat algorithms to converge.

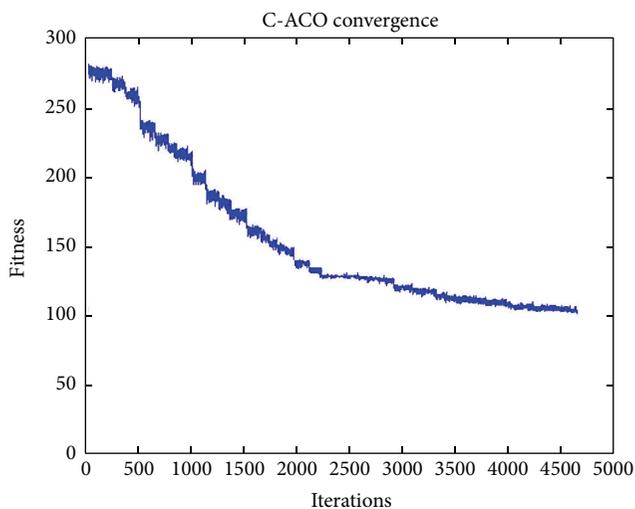


FIGURE 8: Number of iterations required for C-ACO to converge.

acting as the searching agents in the C-ACO algorithm make only a small move in each iteration. Many comparisons are thus required to find the best solution. In sum, C-ACO may be suitable for applications in which incremental optimization is desired and very little time is needed for each step, but it may take many steps to reach the optimal goal.

The next set of experiments tests the quality of clustering in terms of accuracy (measured as 100% minus the percentage of instances overlapping in wrong clusters) and standard deviation. Standard deviation is related to how much variation from the average (mean) is caused by clustered data. The misclustered data derived in the experiments can be seen in Figure 3. Most of the results are satisfactory. The standard

TABLE 13: Accuracy of clustering in the three datasets using different clustering algorithms.

Name	Accuracy of clustering algorithm		
	Iris	Wine	Haberman
K-means	0.8666	0.7225**	0.522
C-Firefly	0.7866	0.7225**	0.529
C-ACO	0.68	0.5505	0.562**
C-Cuckoo	0.89	0.7225**	0.526
C-Bat	0.92**	0.6966	0.473

TABLE 14: Standard deviation measures for different algorithms in different datasets.

The standard deviation of each algorithm			
Iris			
Algorithm	Cluster1	Cluster2	Cluster3
K-means	1.8476**	1.7277	1.9847
C-ACO	1.9185	1.809	1.9742
C-Firefly	1.8476**	1.6269**	1.9136**
C-Cuckoo	1.8476**	1.7345	1.982
C-Bat	1.8476**	1.7311	1.9741
Wine			
Algorithm	Cluster1	Cluster2	Cluster3
K-means	319.8828	194.4313	123.3868
C-ACO	296.271**	155.1661**	154.7441
C-Firefly	319.8828	194.4315	123.3868
C-Cuckoo	319.8828	194.4315	123.3836**
C-Bat	318.5658	200.5721	130.7622
Haberman			
Algorithm	Cluster1	Cluster2	
K-means	28.1977	25.1797	
C-ACO	28.0195**	25.4671	
C-Firefly	28.0842	25.1372**	
C-Cuckoo	28.1977	25.1797	
C-Bat	28.1627	25.193	

deviations indicate that the data points tend to be very close to the mean. The mathematical definition is simply

$$\sigma = \sqrt{\frac{1}{N} [(x_1 - u)^2 + (x_2 - u)^2 + \dots + (x_N - u)^2]}, \quad (11)$$

where $u = (1/N)(x_1 + \dots + x_N)$. Again, the most widely employed dataset, the iris dataset, is used for this set of experiments.

Table 13 shows that the results obtained using the C-Bat algorithm are the best in the iris dataset, whereas those derived with the C-Cuckoo algorithm are the best in the wine dataset. In the Haberman data, all five algorithms are almost equally accurate, though the C-ACO algorithm is slightly more precise.

Table 14 shows that the C-Firefly algorithm has the minimum deviation within clusters, whereas the original K-means algorithm deviates to the greatest extent.

TABLE 15: Performance of clustering results of image segmentation using different clustering Algorithms.

Algorithms	Intercluster distance	Intracluster distance	CPU time (s)
Tower Bridge, $k = 3$			
K-means	6.5932e + 4	1.1154e + 8	24.5465***
C-ACO	6.7741e + 4***	1.0561e + 8	351.9310
C-Bat	6.5932e + 4	1.0561e + 8	39.3933
C-Cuckoo	6.7412e + 4	1.0561e + 8	39.7490
C-Firefly	3.5565e + 4	1.0407e + 8***	150.1532
Cambridge University, $k = 3$			
K-means	1.7154e + 5	4.8854e + 8	166.7963***
C-ACO	1.9773e + 5	4.8926e + 8	343.1299
C-Bat	2.0971e + 5***	4.9134e + 8	386.0401
C-Cuckoo	1.9967e + 5	4.8915e + 8	380.7575
C-Firefly	0.0046e + 5	3.4682e + 8***	1477.0564
Le Mont-Saint-Michel, $k = 4$			
K-means	1.8142e + 5	2.5541e + 8	160.2645***
C-ACO	1.7534e + 5	2.5273e + 8	223.5515
C-Bat	1.8125e + 5	2.5793e + 8	250.0147
C-Cuckoo	2.0102e + 5***	2.4998e + 8***	250.7541
C-Firefly	0.4564e + 5	2.5273e + 8	941.8844
Château de Chenonceau, $k = 4$			
K-means	1.7651e + 5	2.8993e + 8	148.6572***
C-ACO	2.0091e + 5	2.5037e + 8	226.8645
C-Bat	1.8708e + 5	2.8424e + 8	251.7009
C-Cuckoo	2.0756e + 5***	2.4884e + 8	251.8786
C-Firefly	0.4310e + 5	1.5130e + 8***	982.2208

3.3. Testing the New Clustering Algorithms in Image Segmentation. In this set of experiment, the new hybrid clustering algorithms are put under test of image segmentation task. Pixel color oriented image segmentation is the core of image analysis which finds its applications in many areas of image interpretation, pattern identification/recognition, and robotic vision. Some popular applications [13] include but are not limited to geographical information remote sensing, medical microscopy, content-based audio/visual media retrieval, factory automation, and unmanned vehicle navigation, just to name a few.

In practical scientific and industrial applications, the quality and accuracy of image segmentation are very important which depend on the underlying data clustering algorithms. A common choice of unsupervised clustering algorithm is K-means in image segmentation based on color. The regions of the image depending on the color features are grouped into a certain set of segments by measuring the intercluster distance and intracluster distance between each image pixel and the centroid within the cluster. The clustering process is exactly the same as that used in the previous experiment on UCI datasets, except that the images under test in this experiment are larger in amount. An 8 MB high-resolution photo like those used in the experiment



FIGURE 9: Results of image segmentation by using different nature-inspired clustering algorithms, on a photo called “Tower Bridge.”

here has typically 5184×3456 pixels. In addition to spatial information, x - and y -axes of the pixel position in the image, each pixel is triplet of red, green, and blue information, ranging from 0 in darkness to 255 being the strongest in intensity. It is well-known that every pixel of an image is made up by mixing the 256 independent intensity levels of red, green, and blue light. Each data point of the 17,915,904 pixels is a five-dimensional matrix comprised of the pixel location information and RGB information, $[x, y, R, G, B]$ where $0 \leq x \leq 5184$, $0 \leq y \leq 3456$, and $0 \leq R, G, B \leq 255$. The hybrid clustering algorithms are extended from K-means in the same way as described in Section 2. The required image data can be technically extracted by using MATLAB functions *imread(filename)* that creates a three-dimensional matrix and *impixel()* that returns the values of the RGB triplet for the pixel.

The experiment is run over four images whose pixels are to be clustered using different preset numbers of $k = 3$ and $k = 4$. The four images are shots of sceneries, namely, Tower Bridge (TB), Cambridge University (CU), Le Mont-Saint-Michel (MSM), and Château de Chenonceau (CDC).

They have similar image composition and identical size. Some particular features that are subtly contained in the images are used for testing the efficacy of the clustering algorithms like those as follows.

- (i) TB, thickness of clouds in the sky, the shaded part of the tower bridge.
- (ii) CU, depths of perspectives along the cupula and college building, details on the lawn.
- (iii) MSM, details of the fortress wall and small windows on the chapel.
- (iv) CDC, reflection of the Château over the water.

The performance is again measured by intersimilarity and intrasimilarity across and within the same cluster, as well as time taken in seconds in processing a whole image. The performance results are tabulated in Table 15. The winning performance result by one of the four hybrid clustering algorithms or K-means is marked with a triple asterisks as distinction. The original images under test and the segmented images by the various clustering algorithms are shown in



FIGURE 10: Results of image segmentation by using different nature-inspired clustering algorithms, on a photo called “Cambridge University.”

Figures 9, 10, 11, and 12, respectively for TB, CU, MSM, and CDC.

The performance is again measured by intersimilarity and intrasimilarity across and within the same cluster, as well as time taken in seconds in processing a whole image. The performance results are tabulated in Table 15. The winning performance result by one of the four hybrid clustering algorithms or K-means is marked with a triple asterisks as distinction.

In all the tests, K-means seems to take the shortest time, probably because it stops early in local optima. This is evident by the fact that none of the results by K-means score the best in either intercluster distance or intracluster distance. In the experiment of TB, C-ACO scores the widest intercluster length, and C-Firefly has the tightest intracluster distance. As a result, visually C-ACO produces slightly more details on the cloud at the top right corner. C-Firefly seems to produce the most details on the sun-facing side of the tower as well as the shaded side of the tower. C-Firefly again scores the best in intracluster distance in CU and CDC. Again, in CU, C-Firefly offers the most details on the lawn; the copula likewise

has most details and reproduces seemingly most accurately on the college façade by C-Firefly. Interestingly, C-Cuckoo has the longest inter,cluster distance in MSM and CDC. In MSM C-Cuckoo gives the most structural outline of shades and colors on the wall of the chapel, while C-Firefly produces most details on the fortress wall. In CDC, C-Cuckoo and C-Firefly manage to produce the relatively best reflection images over the water, by visual inspect.

The overall results of the experiments described in this paper show that two of the new clustering algorithms observed here, the C-Cuckoo and C-Bat algorithms, which have never been tested by other researchers, are more efficient and accurate than the C-ACO and C-Firefly specifications. This represents a significant contribution to existing knowledge, because it sheds light on the encouraging possibility that optimization techniques derived from nature can be used to improve K-means clustering; we hope that this lays the foundation for more sophisticated bio-inspired optimization methods to be integrated with existing clustering algorithms.

The characteristics of each of the four bioinspired clustering algorithms are listed in the Appendix by way of

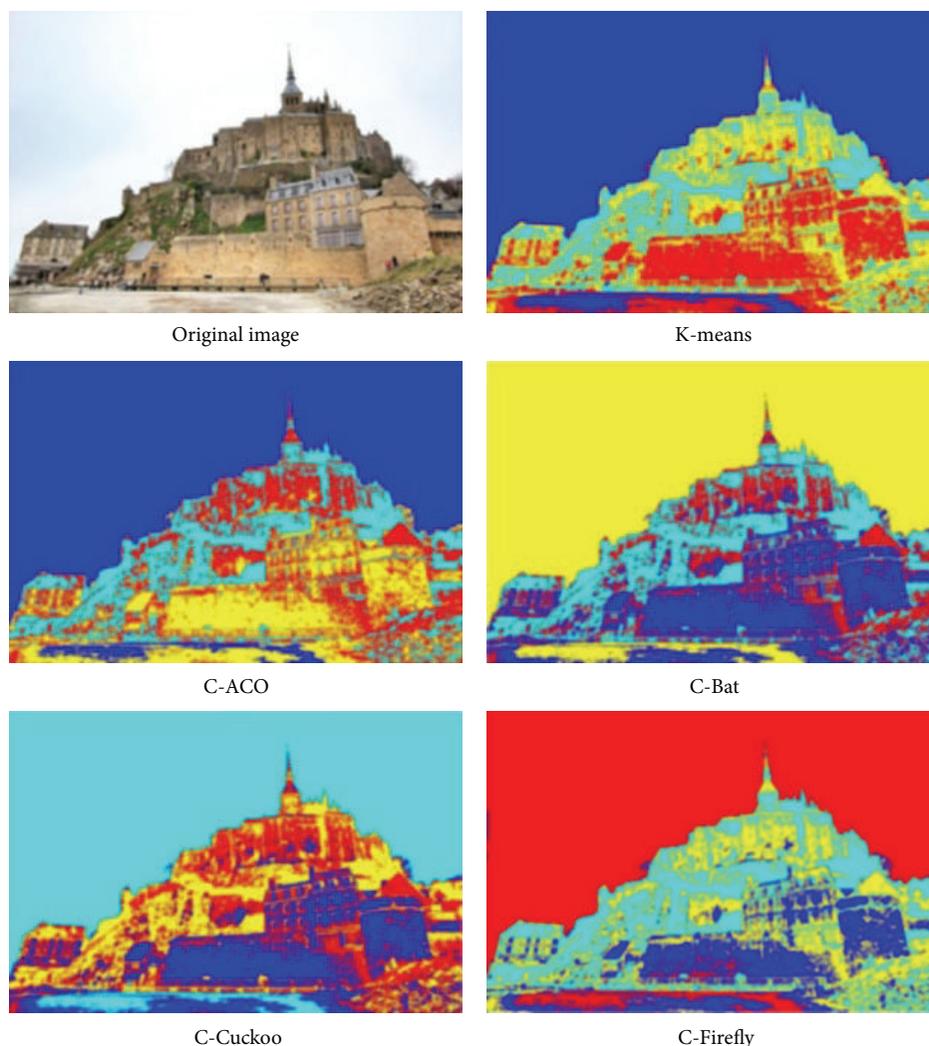


FIGURE 11: Results of image segmentation by using different nature-inspired clustering algorithms, on a photo called “Le Mont-Saint-Michel.”

summary. It is hoped that researchers will find them useful as a source of inspiration for developing better algorithms in future. As the phrase “meta-heuristics” suggests, these bioinspired optimization heuristics come in abstract and general forms. There is ample potential to extend, modify, and even build hybrids of them with other heuristic functions to suit different applications.

4. Conclusion

K-Means clustering algorithms, a classical class of partition-based algorithms used for merging similar data into clusters, are known to have the limitation of getting stuck in local optima. As a matter of intellectual curiosity in computer science, how best to cluster data such that the integrity of the clusters is maximized, has always been a challenging research question. The ideal solution is to find an optimal clustering arrangement which is globally best—so that no other possible combinations of data clustering exist that are better than the global one. One way of achieving this is

to try all the possible combinations by brute-force which could be computational intractable. Alternatively, nature-inspired optimization algorithms, which recently rise as a popular research topic, are extended to work with K-means in guiding the convergence of disparate data points and to steer them towards global optima, stochastically instead of deterministically. These two research directions of metaheuristic optimization and data mining do fit like hand and glove. Constrained by the inherent limitation of K-means design and the merits of nature-inspired optimization algorithms, it is feasible to combine them letting them complement and function together. This paper evaluates four hybrid types of clustering algorithms developed by integrating nature-inspired optimization algorithms into K-means. The results produced from the experiments clearly validate the new algorithms possess a performance enhancement, apparently for the C-Bat and C-Cuckoo. The extended versions of clustering algorithms enhanced by nature-inspired optimization methods perform better than their original versions, in two sets of experimental datasets—general purpose and image

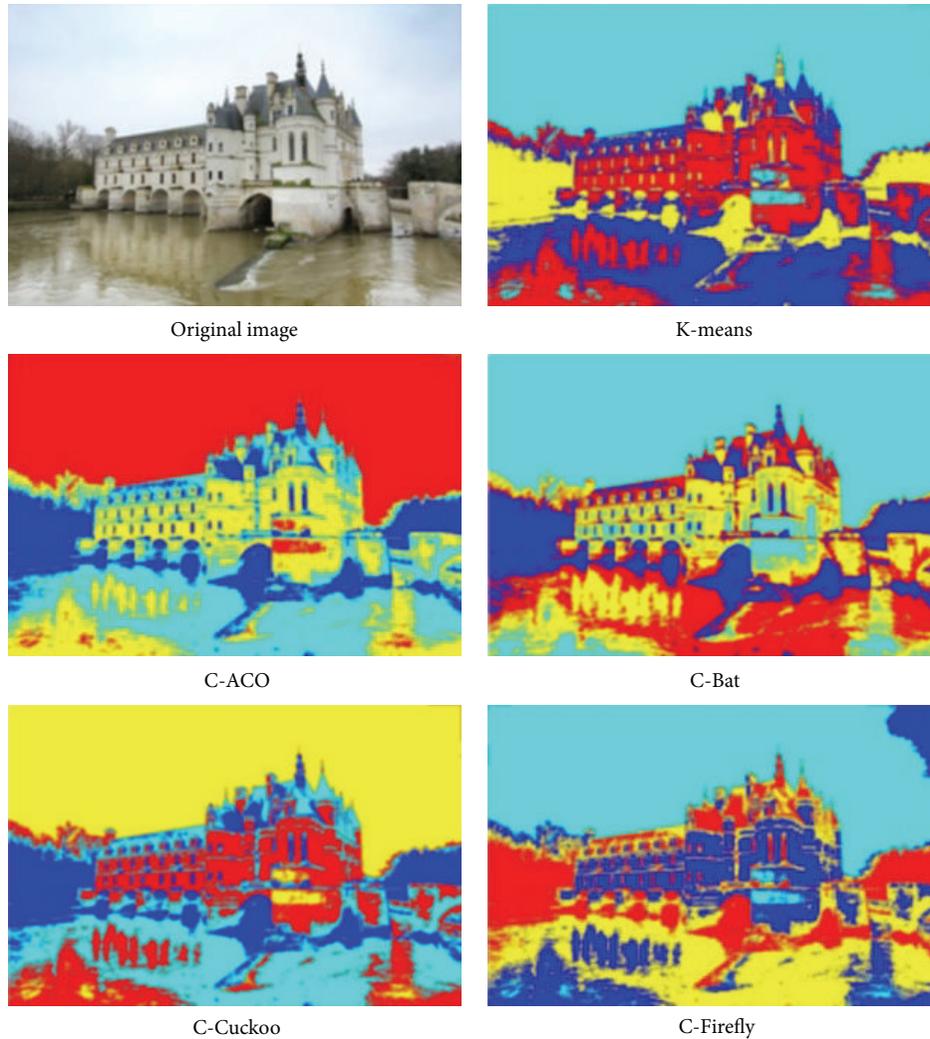


FIGURE 12: Results of image segmentation by using different nature-inspired clustering algorithms, on a photo called “Château de Chenonceau.”

segmentation. Experiments are conducted to validate the benefits of the proposed approach.

Appendix

C-Firefly. (1) In this algorithm, using light intensity as the objective function value, each firefly x is represented by its location.

(2) Every firefly needs $O(n^2)$ comparisons, and each one will move towards the brighter one. Therefore, it takes more time for each iteration, but convergence can be reached in very few iterations.

(3) The attractiveness varies according to the inverse square law $(r) = \beta_0 e^{-\gamma r^2}$, the γ is the absorb coefficient.

C-ACO. (1) This algorithm uses a pheromone matrix as the communication channel.

(2) The pheromone matrix leads the ant to choose the path.

(3) Ants construct the solution step-by-step, and the decision variables of associated mathematical objective function are discrete.

C-Cuckoo. (1) The nests represent the population.

(2) The cuckoo moves using Levy flight, which is a Markov chain in which the next location depends solely on the current location.

(3) This algorithm has the ability to discover foreign eggs and abandon them. This means that the algorithm can avoid local optima.

(4) In each iteration, the worst solution will be replaced by the better one. If there is no worse solution, the better one will be retained for the next iteration.

(5) Every cuckoo only cares about its nest. It is not necessary to communicate.

C-Bat. (1) Each bat has data on velocity and location. This is similar to PSO-clustering.

(2) Velocity is determined by frequency, loudness, and pulse rate.

(3) The solution can be adjusted by frequency, loudness, and pulse rate.

(4) In each iteration, the bat only updates its location once. Therefore, the algorithm runs very quickly.

(5) The bat has the ability to adjust itself. It can sense the surrounding environment.

Similarities among these algorithms.

(1) They all use attribute means to determine the initial centroid.

(2) Optimization algorithms are the key factors in accelerating clustering and avoiding local optima.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper. The authors of this paper do not have a direct financial relationship with the commercial identities mentioned in this paper that might lead to a conflict of interests.

Acknowledgments

The authors are thankful for the financial support from the research Grant no. MYRG152(Y3-L2)-FST11-ZY, offered by the University of Macau, RDAO. Special thanks go to Mr. Rui Tang, who was a research assistant and MSc software engineering graduate of University of Macau, for programming the Matlab codes and conducting the data mining experiments.

References

- [1] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, University of California Press, 1967.
- [2] S. Fong, "Opportunities and challenges of integrating bio-inspired optimization and data mining algorithms," in *Swarm Intelligence and Bioinspired Computation*, pp. 385–401, Elsevier, 2013.
- [3] R. Tang, S. Fong, X.-S. Yang, and S. Deb, "Integrating nature-inspired optimization algorithms to K-means clustering," in *Proceedings of the 7th International Conference on Digital Information Management (ICDIM '12)*, pp. 116–123, Macau, China, August 2012.
- [4] J. Senthilnath, S. N. Omkar, and V. Mani, "Clustering using firefly algorithm: performance study," *Swarm and Evolutionary Computation*, vol. 1, no. 3, pp. 164–171, 2011.
- [5] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Tech. Rep. TR06, Computer Engineering Department, Engineering Faculty, Erciyes University, 2005.
- [6] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications*, vol. 5792 of *Lecture Notes in Computer Sciences*, pp. 169–178, Saga, 2009.
- [7] X.-S. Yang, S. Deb, and S. Fong, "Accelerated particle swarm optimization and support vector machine for business optimization and applications," in *Proceedings of the 3rd International Conference on Networked Digital Technologies (NDT '11)*, vol. 136, pp. 53–66, Springer CCIS, Macau, 2011.
- [8] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, J. R. Gonzalez, D. A. Pelta, C. Cruz, G. Terrazas, and N. Krasnogor, Eds., vol. 284 of *Studies in Computational Intelligence*, pp. 65–74, Springer, Berlin, Germany, 2010.
- [9] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, IEEE, Coimbatore, India, December 2009.
- [10] S. Z. Selim and M. A. Ismail, "K-means-type algorithms: a generalized convergence theorem and characterization of local optimality," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 1, pp. 81–87, 1984.
- [11] M. Dorigo, V. Maniezzo, and A. Coloni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 26, no. 1, pp. 29–41, 1996.
- [12] B. B. Mandelbrot, *The Fractal Geometry of Nature*, W.H. Freeman, New York, NY, USA, 1982.
- [13] Z. Anil and S. K. K. Chitade, "Colour based image segmentation using K-means clustering," *International Journal of Engineering Science and Technology*, vol. 2, no. 10, pp. 5319–5325, 2010.

Research Article

Congestion Control for a Fair Packet Delivery in WSN: From a Complex System Perspective

**Daniela Aguirre-Guerrero, Ricardo Marcelín-Jiménez,
Enrique Rodríguez-Colina, and Michael Pascoe-Chalke**

Department of Electrical Engineering, Metropolitan Autonomous University (UAM), 09340 Mexico, DF, Mexico

Correspondence should be addressed to Michael Pascoe-Chalke; mpascoe@xanum.uam.mx

Received 25 April 2014; Accepted 9 July 2014; Published 10 August 2014

Academic Editor: Su Fong Chien

Copyright © 2014 Daniela Aguirre-Guerrero et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In this work, we propose that packets travelling across a wireless sensor network (WSN) can be seen as the active agents that make up a complex system, just like a bird flock or a fish school, for instance. From this perspective, the tools and models that have been developed to study this kind of systems have been applied. This is in order to create a distributed congestion control based on a set of simple rules programmed at the nodes of the WSN. Our results show that it is possible to adapt the carried traffic to the network capacity, even under stressing conditions. Also, the network performance shows a smooth degradation when the traffic goes beyond a threshold which is settled by the proposed self-organized control. In contrast, without any control, the network collapses before this threshold. The use of the proposed solution provides an effective strategy to address some of the common problems found in WSN deployment by providing a fair packet delivery. In addition, the network congestion is mitigated using adaptive traffic mechanisms based on a satisfaction parameter assessed by each packet which has impact on the global satisfaction of the traffic carried by the WSN.

1. Introduction

A wireless sensor network (WSN) consists of a large number of nodes, which have sensing, processing, and communicating capabilities. Nodes in a WSN do not only monitor their environment, but they also forward and route data packets to one or more appointed *sink* nodes. It is known that there are open issues that limit the practical adoption of WSNs [1]. These limitations come from the fact that each individual wireless node in a WSN has reduced processing capabilities as well as limited energy budget.

Network congestion occurs when the system is close to its carrying capacity and an increase on the incoming traffic, or even a burst, may overload node buffers with a potential domino effect that may turn into a disaster. On these conditions, and unless a congestion control mechanism is used, the number of packets arriving to their final end falls abruptly. In contrast, a network under congestion control keeps close to an ideal response. That is, the outgoing traffic

equals the network carrying capacity. Additionally, for WSN applications, congestion control mechanisms should also consider the side effects of wireless transmissions, such as interference and power loss.

In recent years, a set of models and tools initially developed by the physics community have shown their pertinence to address problems from other domains, such as biology and economics, among many others. The objects studied from this emerging body of knowledge are generally called complex systems. Mitchell [2] describes complex systems as “. . . *made up from a massive network of interacting components, without a central control, that follow a simple set of operational rules and are able to achieve an elaborated collective behavior, thanks to sophisticated information processing techniques and adaptation, based on learning or evolution.*” We can immediately identify many systems that fit into this definition, for example, ant colonies, bird flocks, fish schools, the international economy, the city vehicle traffic, and so forth. We know that none of them have a central control. We say that the interactions

of components occur in a microscopic level. Nevertheless, in a macroscopic level, all of them show an emergent behavior which adapts to changing conditions over the time.

In this work, we agree with other authors that a WSN can be described as a complex system. The novelty of our approach is based on the fact that packets are regarded as the key entities that shape the overall system behavior. From our perspective, there are two types of interactions, among packets and packets with nodes. At node level, packets compete for network resources (buffers and links) on their way to the sink node(s). Based on these local interactions, each node is programmed to take decisions about routing, medium contention, and packet dropping. As an emergent global behavior, it is shown that the network achieves a self-adaptive mechanism that matches the traffic with the network capacity. An underlying assumption of our work is that, as it was stated by the economist Adam Smith, there exists an "invisible hand" that guides individual packets to benefit the system through the pursuit of their private interests.

Our experimental results show that the proposed solution may provide an effective strategy to address some of the common problems found in WSN deployment, such as fair packet delivery, adaptive routing, energy consumption, and congestion control.

The rest of this work is comprised of the following sections. Section 2 summarizes relevant work found in the literature related to a variety of control mechanisms for wireless sensor networks. Section 3 describes the theoretical framework on which the proposed method is based on. Section 4 presents the experimental platform and a diversity of methods used to evaluate the performance of the proposed scheme. We describe and analyze in Section 5 the results obtained by simulations and we also discuss some relevant aspects that should be taken into consideration during the scheme implementation. Finally, Section 6 provides some concluding remarks.

2. Related Work

In this work, we introduce a model that comprises the idea of self-organized agents that can modulate a global behavior which, in our case of study, turns into the congestion control of a wireless sensor network. Our approach is strongly inspired by agent-directed simulations [3]. To the best of our knowledge, this approach is barely found in the literature of WSN, with a few related alternatives, such as [4], where authors propose the use of the complex system theory to deal with WSN. In our work we proposed a similar analogy but additionally we show results for specific applications and scenarios as well as a novel mechanism for the traffic control based on the packet satisfaction where the packet is regarded as the agent of the complex system.

In [5], a methodology for designing a self-organized WSN is presented. The authors suggest that cooperation between nodes is needed in order to accomplish more complex tasks for WSNs. In addition, they mentioned that a promising approach of how to cope with this is the use of the emergent self-organization. The proposed emergent

self-organizing system aims to achieve improved scalability by the simplicity of its elements and their interactions. We agreed with the authors in [5] with the macroscopic and microscopic approach; however, we proposed a subtle difference from that work in the distributed coordination mechanism, because they use an explicit feedback loop mechanism but we proposed the packet interactions with the nodes as main distributed coordination mechanism without a specific control mechanism based on positive and negative feedback loops. In our proposed mechanism the packet satisfaction is the main parameter to adjust the traffic flow without the need of an explicit feedback loop between nodes.

A more comprehensive approach is presented in [6], which introduces an agent-based simulation framework. This work focuses on modeling of sensed variables of the environment rather than a WSN itself. The work presented in [7, 8] also considers packets as active entities of a complex system. From an agent-based approach, an improvement on directed diffusion routing protocol is presented in [7], whereas in [8] a routing protocol inspired on bird flocking is proposed.

In [9], a congestion control is presented for sensor networks with an algorithm independent of the topology and any addressing scheme. Then, the authors emphasize that the absence of state information per node in the network becomes a critical issue if a congestion control algorithm is required. Thus, a common approach is to provide a solution based on biological mechanisms for self-organization and complex systems [9–11]. In our proposed model shown in this work, an important viewpoint introduced is that data packets circulating over the network can be described as independent entities that are shaping an emergent global process. The interaction rules that we introduce are settled on a local and short-termed level. Nevertheless, the resulting emerging properties are studied from a global perspective.

Several publications have paid attention to the fact that the WSNs are comprised of resource-constrained devices [11–13]. Therefore, their design is oriented to minimize efforts without compromising the task's integrity and the gathering of the vast information required for a huge and unknown deployed area.

In recent years, many protocols and implementations have been developed for congestion control in wireless sensor networks. The most popular approach is based on traditional methodologies and protocols recognized from the Internet. For instance, several of them have studied the congestion problem from the transport and link layers perspective. However, the possibility of developing congestion control of a network during the packet trip from source to destination has not been explored in depth.

The authors in [14] propose a multiagent system with the ability of its agents to find alternative paths for energy efficient data dissemination. This approach is mainly used for routing. In [15] the network is able to gather information using the multiagent approach, in order to take further decisions. In contrast, we propose a simple mechanism to detect changes in the network conditions without considering anything but the degree of satisfaction of the individual packets. Satisfaction is a measure that estimates whether

a packet has the possibility to reach its final destination within a bounded time.

3. Theoretical Framework

According to [16], a complex system is made up from a set of entities, called agents, which are deployed and act over a given environment. Agents have goals and show a particular behavior oriented to achieve such goals. From this perspective, it is important to recognize not only the acting elements of a system, but also the interactions of the agents with their environment (agent-to-environment), as well as the interactions among themselves (agent-to-agent).

In this work, we state that the traffic on a WSN, under stressing conditions, can be studied from a complex system perspective. Data packets are regarded as agents, whose goal is to reach a given sink, with minimum delay. In this case, the interactions between the packets and their environment can be described in terms of the network resources used by packets travelling to the sink, that is, links and buffers. Notice, for instance, that the conditions of a wireless link may induce undesired effects, such as noise and power loss, which have impact on the packet reception. Also, the size of the buffer limits the number of packets that can be temporarily stored in a given node. The interactions among packets arise from the fact that they compete for the network resources and this condition may cause network congestion, whose effects can be detected either at the link level or at the node level. In the former, it produces packets collisions. In the latter, it increases the delivery time, and under very stressing conditions it produces packet losses.

3.1. Taking the Pulse of the System. In order to evaluate the impact of the aforementioned interactions, we propose two measures that allow us to study the network traffic from a microscopic view. The first one estimates the fulfillment of a packet goal. The second one estimates the interference produced by a packet while interacting with other packets. Besides, we developed a macroscopic measure to estimate the overall network performance.

We can assume that each node of the network takes a snapshot of its current state, according to some schedule, including its buffer occupancy. Based on this state, each node calculates the following measures.

3.1.1. The Individual Packet Satisfaction. Let p be a packet travelling over a WSN, from a source S to a target sink T . Let us also assume that p has been temporarily stored in node N , which is the k -th stage on its route, after being forwarded from node M (see Figure 1). The packet satisfaction of p is assessed by the variable $\sigma_{p,N} \in [0, 1]$ and is calculated at each node it traverses (visits), including S . Accordingly, the satisfaction of p at node N is determined by the following expression:

$$\sigma_{p,N} = \sigma_{p,M} + \frac{(1 - \Delta t / \Delta t_{\max}) - \sigma_{p,M}}{k}, \quad (1)$$

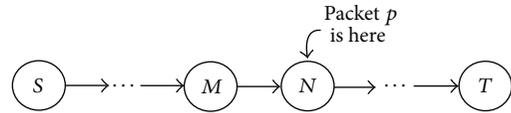


FIGURE 1: Packet p travelling from source node (S) to sink node (T).

where Δt is the time that p has been stored in the buffer at node N , while Δt_{\max} is the maximum time that p is allowed to remain in a buffer before it has to be discarded; then $0 < \Delta t \leq \Delta t_{\max}$. As we just mentioned, (1) applies to each node on the route of p , including node S . For the initial case at the departing node, $k = 1$ and $\sigma_{p,M} = 0$.

The first step for N to calculate $\sigma_{p,N}$ is to determine the normalized delay experienced by p while buffered at N . This delay is given as follows:

$$|\Delta t| = \left(1 - \frac{\Delta t}{\Delta t_{\max}}\right). \quad (2)$$

The second step consists in evaluating the change on the packet satisfaction ($\Delta\sigma_{p,N}$), which is given by the contribution of N to the normalized delay of p ; that is,

$$\Delta\sigma_{p,N} = \frac{|\Delta t| - \sigma_{p,M}}{k}. \quad (3)$$

Therefore, (1) can be written in terms of $\Delta\sigma_{p,N}$, as in the following:

$$\sigma_{p,N} = \sigma_{p,M} + \Delta\sigma_{p,N}. \quad (4)$$

Notice that $\sigma_{p,N}$ can be understood as a function that updates the accumulated delay of p on its journey from source to sink.

Each node on the route traveled by packet p must be able to label the packet with its corresponding time of arrival in order to estimate Δt . As for Δt_{\max} , this is a parameter that features each node, and it is oriented to shape desired properties of the traffic that will be carried by the network. For instance, Δt_{\max} can be settled according to the distance from a given node to the sink, in such a way that a node which is near to the sink will have a value greater than those nodes that are rather away from the sink. Thus, we foresee that packets travelling long routes do not accumulate an excessive delay at the beginning of their journey, while a longer delay is expected at nodes that are near the sink.

3.1.2. Interference among Packets. Let N be a node with a buffer that is temporarily occupied by a group of F packets, called *flock*, which includes packet p . The interference (ϕ_p) assesses the way that p affects the satisfaction of the flock it belongs to according to the following expression:

$$\phi_p = -\Delta\sigma_{p,N} - \frac{1}{F-1} \sum_{i \neq p} \Delta\sigma_{i,N}. \quad (5)$$

Equation (5) shows an algebraic addition with two parts. The first part corresponds to the satisfaction change experienced by packet p at the current node ($-\Delta\sigma_{p,N}$). The second

part is the average change experienced by the rest of the packets that share the current buffer with p , that is, $(-1/(F-1)) \sum_{i \neq p} \Delta \sigma_{i,N}$.

If the current change of the normalized packet delay is greater than its accumulated delay, then it results in a negative change, which means that the packet satisfaction has been reduced at the current node. Let us recall that the satisfaction is a number between 0 and 1, where 1 represents the maximum satisfaction and 0 a null satisfaction.

Now, let us assume that (5) produces a positive interference, called *friction*, which is an undesirable condition. This implies the following.

- (1) Both of its parts are negative. In this case, we say that both sides have lost satisfaction.
- (2) The change in the satisfaction of p is negative, while the average change in the rest of the packets is positive. Nevertheless, the absolute value of the first part is greater than the second. In this case we say that the loss of satisfaction on one side is not compensated by the increase on the other side.
- (3) The change in the satisfaction of p is positive, while the average change in the rest of the packets is negative. Nevertheless, the value of the second part is greater than the first. We also say that the loss of satisfaction on one side is not compensated by the increase on the other side.

Now, let us assume that (5) produces a negative interference, called *synergy*, which is a desirable condition. This implies the following.

- (1) Both of its parts are positive. In this case, we say that both sides have gained satisfaction.
- (2) The change in the satisfaction of p is positive, while the average change in the rest of the packets is negative. Nevertheless, the absolute value of the first part is greater than the second. In this case, we say that the gain of satisfaction on one side does not strongly affect the loss on the other side.
- (3) The change in the satisfaction of p is negative, while the average change in the rest of the packets is positive. Nevertheless, the value of the second part is greater than the first. We also say that a gain of satisfaction on one side does not strongly affect the other side loss.

Therefore, there are three possible interactions between packet p and the flock it belongs to.

If $\phi_p > 0$, then the packet p introduces *friction* on the flock, which is an undesirable condition.

If $\phi_p < 0$, then packet p fosters *synergy*, which is a desirable condition.

If $\phi_p = 0$, then the change on packet satisfaction cancels the change on the satisfaction of the others. In this case we say that p has a *neutral interaction* with the flock.

3.1.3. Overall System Performance or a Macroscopic View of the Network. Let us suppose now that we are able to synchronize,

at time t , the snapshots taken by the overall set of nodes that make up the WSN and gather these pictures to elaborate a global diagnostic of the system. We assume that, at this point in time, there are n packets stored at different places. For a packet, with identifier $i = 1, \dots, n$, we say that this packet is stored at a given node $N(i)$ and, at time t , this node takes its local snapshot. The network satisfaction is estimated according to the following expression:

$$\sigma_{\text{WSN}}(t) = \frac{1}{n} \sum_{i=1}^n \sigma_{i,N(i)}. \quad (6)$$

This expression means that, at the time of the snapshot, we evaluate the average satisfaction of the packets traveling across the network.

3.2. Congestion Control for WSN under Heavy Traffic Load.

Now, we present a new congestion control based on the aforementioned framework. Our proposal is targeted to maximize the network satisfaction $\sigma_{\text{WSN}}(t)$.

The solution consists of four stages: (1) initialization, (2) processing, (3) routing, and (4) transmission. Initialization is executed only once in order to settle down the working parameters of each active node. Processing takes place each time a new packet is received at the node in order to estimate its microscopic measurements. Based on these estimates, and during the routing stage, the node decides whether to forward the packet immediately, store it at its local buffer, or drop it. Finally, if transmission applies, the node invokes a CSMA MAC protocol, which includes a modified version of an adaptive back-off interval, to forward the packet to the next node.

3.2.1. Initialization. During initialization, the sink node grows a spanning tree rooted at this node [17]. We say that each node lying on the tree has a corresponding level (l). This level is related to the minimum number of hops from a given node to the sink. The root has a level 0; the children of the root have a level 1, and each node has the level of its father plus one. Each node knows its level as well as the level of its neighbors.

We consider that the workload of a given node depends on its distance to the sink. This means that the nodes in the vicinity of the sink concentrate on a higher load and therefore, they are expected to show a higher buffer occupancy compared to those nodes in the borders of the network. As a consequence, we also consider that those packets arriving at the last stages of their trip will experience longer delay. From this perspective, we assume that the processing of a packet must take into account the relative position of the node that is traversing, on its way to the sink. It is important to recall that the "goal" of an individual packet is to arrive to its final destination with the higher satisfaction, that is, the smallest accumulated delay. In contrast, the goal of the network, from a macroscopic view, is to keep congestion under control. Also, it is important to realize that the accumulated delay of a packet is strongly related to the flock size defined at the node where it is temporarily stored.

Assuming that all nodes have a uniform buffer size, given by b , (measured in bits), and in order to address the aforementioned considerations, the flock size must be related to the position of the node which allocates this set of packets. In general terms, a small flock introduces smaller delays than a bigger flock. If we accept that a packet should accumulate the least possible delays at the beginning of its journey and that it should be prepared to tolerate longer delays at the end, we conclude that the flock size must be a fraction of the buffer size, reflecting the distance of the node to the final target. To estimate this distance we may use the level of the node, according to the spanning tree that the sink has built over the underlying graph. Nevertheless we simplified this approach and decided to classify the network in three zones only: distant, intermediate, and nearby, all measured with respect to the sink node. For this purpose, we can think of the network as divided by three concentric “circles” drawn around the sink. These circles define the boundaries of such zones.

Let L be the highest level that can be achieved by a node in the network, that is, the maximum hop distance to the sink node. Then, we divide the spanned region into three zones according to L . A node having level l is considered to lay on zone 1, 2, or 3, according to the following expression:

$$\text{Zone} = \begin{cases} 1, & 0 < l < \left\lfloor \frac{L}{3} \right\rfloor, \\ 2, & \left\lfloor \frac{L}{3} \right\rfloor \leq l < \left\lfloor \frac{2L}{3} \right\rfloor, \\ 3, & \left\lfloor \frac{2L}{3} \right\rfloor \leq l. \end{cases} \quad (7)$$

As we stated before, packets are gathered in groups, called *flocks*. The size of a flock (f), measured in bits, depends on the level of the node where this flock is currently stored. Notice that F represents the number of packets that make up a flock, while f represents the overall “weight” of this group. Consider

$$f = \begin{cases} b, & \forall l \in \text{Zone } 1, \\ \left\lfloor \frac{2b}{3} \right\rfloor, & \forall l \in \text{Zone } 2, \\ \left\lfloor \frac{b}{3} \right\rfloor, & \forall l \in \text{Zone } 3, \end{cases} \quad (8)$$

where b is the buffer size, in bits. Let us recall that we are assuming that all nodes have the same buffer size.

Equation (8) implies that, depending on the zone where a given flock is travelling, the flock occupies one-third, two-thirds, or the overall buffer size, corresponding to zone 3, 2, or 1, respectively. In turn, this means that as the flocks get closer to the sink they can grow and, as a consequence, they may experience longer delays.

Each node configures its local time-to-live (TTL) parameter, which indicates the maximum number of hops that a packet may travel to reach the sink. A node on the role of source will use this parameter to stamp each issued packet.

As it can be deduced, the particular value of the TTL depends on the zone where the corresponding node lies. Consider

$$\text{TTL} = \left\lfloor \frac{4l}{3} \right\rfloor. \quad (9)$$

Let us notice that a long TTL value may foster the existence of packets that travel around the sink without finishing their trip but interfere with the rest of the packets and reinforce congestion. Therefore, TTL should be settled in order to give a packet a rather small chance to evade a troubled area before it is discarded.

Notice that the minimum number of hops that packets need to reach the sink is equal to level l of its source node. However, packets are settled with extra hops to dodge congestion zones. In order to finish the initialization stage, each node settles a value Δt_{\max} that corresponds to the maximum time a packet may be stored in its buffer. The value of Δt_{\max} depends on the zone where the node belongs to, according to the following expression:

$$\Delta t_{\max} = \frac{f - w}{v}, \quad (10)$$

where w is the packet length, in bits, such that $w < f$ and v is the local transmission rate, in bps.

Let us imagine that a packet arrives to a given node and it happens to be the last that completes a whole flock. In other words, regarded as a queue, this is the missing client that starts the forwarding service. How long will it wait to continue its trip? Well, it has to wait for the rest of its flock to be dispatched before it takes its turn.

From the above expressions, we observe that most of the working conditions of a node strongly depend on its distance to the sink, which is roughly estimated by its corresponding zone. In this way, we not only foresee that the nodes on the vicinity of the sink are able to tolerate high buffer occupancy, during harsh conditions, but also foster that packets coming from the farthest places do not accumulate long delays at the beginning of their trip.

3.2.2. Processing. When a source node S issues a new packet p , it labels such packet with an initial value $\sigma_{p,S}$. Then, each node N visited by p updates this value according to the following steps.

Before storing p in its buffer, N records the value $\sigma_{p,M}$ that p is carrying in its header and reduces the TTL field of p by 1. If TTL reaches 0, it simply drops the packet, otherwise it stores p .

When N has gathered a flock, it retrieves each packet p from its buffer and updates the header of p with a new value $\sigma_{p,N}$. Also, N calculates the interference caused by p on the flock it belongs to, according to the expressions (4) and (5), respectively.

The interaction of p with the rest of the flock may produce friction or synergy or may be neutral depending on the following cases.

Case A. Friction ($\phi_p > 0$).

Case B. Synergy ($\phi_p < 0$).

Case C. Neutral ($\phi_p = 0$).

Concurrently with these steps, each node is regularly scanning its buffer load. If it finds a stored packet which is about to reach its maximum storage time (Δt_{\max}), it retrieves this packet immediately and places it on the routing stage. Otherwise, the node waits to gather a whole flock to enter into this stage.

3.2.3. Routing. Based on the packet interference, this phase applies an adaptive mechanism that may produce either a vertical or a horizontal routing decision. The vertical and horizontal routing decisions depend on the distance from the sink to the node distributing the traffic. The routing decision is horizontal when a node forwards traffic to other nodes with equal level or hop distance. The routing decision is vertical when nodes forward packets to other nodes that are closer to the sink; it means nodes with lower level.

Once the node has updated the value $\sigma_{p,N}$, it checks whether there is a packet with a value $\Delta\sigma_{p,N} < 0$. If there is not a single packet in this condition, the flock goes to the transmission stage to be forwarded in a vertical routing. Otherwise, for each packet with $\Delta\sigma_{p,N} > 0$, it verifies if packet p introduces friction; that is, $\Delta\sigma_{p,N} > |\sum_{i \neq p} (\Delta\sigma_{i,N} / (F - 1))|$. The packets that do not meet this condition, along with those packets such that $\Delta\sigma_{p,N} < 0$, go to the transmission stage to be forwarded in a vertical routing. The remaining packets are forwarded in a horizontal routing.

3.2.4. Transmission. In this stage we assume the existence of a medium access control based on CSMA. Nevertheless, this scheme includes an adaptive back-off interval that depends on the zone where the corresponding node has been deployed. The node in charge forwards a packet according to the following rules.

- (1) The node senses the medium to find out whether it is busy or idle.
 - (a) If the medium is busy, it programs a back-off timer and goes back to step (1), when the timer expires.
 - (b) Otherwise, the node sends a “request-to-send” packet (RTS) to the immediate target node(s); this packet includes the number of data packets which is about to forward.
- (2) A node that receives the RTS packet answers with a “clear-to-send” (CTS) packet including the number of data packets which is able to store in its buffer.
- (3) Upon receiving the answers, the issuing node forwards the rest of the flock to as many nodes as possible. Transmission starts with those packets having the minimum $\Delta\sigma_{p,N}$.

(4) When the node has an empty buffer, it measures its remaining battery, and in case it has less than 10% of its initial charge, it broadcasts a “farewell” message to all of its neighbors to let them know that it is about to “die.”

(5) A node that receives this message eliminates the issuer from its routing tables.

4. The Experimental Platform and the Methods

Simulation tools are the best suited resources when analytic methods are unable to provide accurate solutions, or when direct experiments are not feasible. Both conditions can be met in WSN. Due to the amount of interacting entities, analytic tools can hardly assess the parameters that feature the system's performance. In addition, deploying an experimental settlement can be expensive and the obtained results may not be considered for a more general framework. Therefore, many specialists on WSN consider simulation tools as the departing point for experiments of network operations.

We developed a simulation model oriented to test and evaluate various congestion control schemes and routing mechanisms. The model is based on the description of the network traffic from the agent-based perspective. It is important to underline that we present a model that can be implemented using any tool supporting an agent-based specification, as well as discrete events. Our particular implementation is based on NetLogo [18], because it is a well-known tool which supports the development of a model where agents interact at discrete points in time and space.

4.1. An Agent-Based Model. Agent-directed simulation [3] is used to emulate a complex system behavior, such as the system analyzed in this work. The key premise that guides this analysis is that friction reduction between agents benefits the system satisfaction, thus increasing performance of the WSN. Maximizing $\sigma_{\text{WSN}}(t)$ can be accomplished by limiting the action of those agents that reduce the value $\sigma_{p,N}$ of others, while preserving functionalities and fostering synergy among them.

As agents may present conflicting goals, their behavior is limited or regulated by means of mediators. These entities are in charge of minimizing conflicts, interferences, and frictions in order to maximize cooperation and synergy. In our particular case, nodes play the role of mediators. Table 1 shows the correspondence between concepts used in WSN and complex systems. These equivalent concepts are the key to the construction of the proposed experimental platform.

4.2. A WSN Simulation Tool. The tool that we have devised comprises the following set of modules:

- (i) network deployment and properties,
- (ii) packet generation,
- (iii) routing,
- (iv) congestion control,

TABLE 1: Complex systems concepts and their equivalences for the proposed WSN model.

Concept	WSN equivalent
Agent	Data packet
Goal	Arrive to the sink with minimum delay
Satisfaction ¹	0 if data packet does not arrive to the sink, 1 if it arrives to the sink within minimum time
Behavior	Packets travel across the network to the sink
Friction	An increase on the incoming traffic (throughput), reduces the outgoing traffic (goodput)
System's satisfaction	Can be evaluated from the outgoing traffic
Mediators	Nodes
Mediator's role	To execute a (distributed) algorithm that prevents congestion and maximizes the outgoing traffic

¹Notice that satisfaction and friction are user-defined measures and they can change if required.

- (v) wireless effects,
- (vi) energy consumption,
- (vii) performance evaluation.

4.2.1. Network Deployment. This module fixes the space and time properties of the supported WSN model. It describes the extension of the area, spanned by the WSN, as well as its shape, for example, square, circle, or irregular. Also, it settles the total simulation time as well as the length of the window that measures the value of $\sigma_{WSN}(t)$. It is known that many control mechanisms strongly depend on the time scale chosen to introduce adaptive decisions, but also the effectiveness of these mechanisms becomes evident depending on the time scale chosen to evaluate the system's performance: "an apparently bad decision in the short term may turn into a good decision in the long term."

In our model we consider 3 types of nodes.

- (i) *Sink*: it is the final destination of the packets carried by the network. Therefore, the outgoing traffic, that is, throughput, is measured considering the number of packets delivered at this point.
- (ii) *Source*: these nodes generate the incoming traffic; that is, the packets that are carried by the network towards the sink. If necessary, these nodes may also develop forwarding operations.
- (iii) *Forwarding*: these nodes are in charge of routing the packets on their way to the sink.

The number of deployed nodes is defined by the user, who also defines the number of source nodes. The sink has a random location, as the rest of the deployed nodes.

The average node degree defines the average number of connections each node has. In WSN, nodes' degree can be featured as a random variable that follows a probability distribution function (PDF). In this particular case, we

generate a random number using a normal distribution and then we round it to the closest integer. According to this characteristic, each node is connected with those nodes within its propagation radio. The global effect of these local connections is the resulting graph that models the underlying communications network.

Due to the fact that coverage directly depends on the transmission power, coverage also affects the nodes' lifetime. A wider coverage area means that the corresponding node is running out of its energy more rapidly. Coverage is also considered a random variable that follows a normal distribution.

Other parameters that are settled within this module are buffer size, transmission rate, packet size, error rate, end-to-end delay, and energy consumption per packet which is related to coverage.

4.2.2. Packet Generation. The traffic scenarios are simulated by traffic generated at the source nodes. This has the possibility of issuing a given number of packets per unit time, according to any of the following options:

- (i) generating a random number using a normal PDF and then it is rounded to the closest integer,
- (ii) generating a random number using a Poisson PDF,
- (iii) a continuous generation rate,
- (iv) an augmenting rate that grows " x " packets per unit time, every fixed time step.

4.2.3. Routing. The routes from all sources to the sink node are established once at the beginning of each simulation, using a well-known distributed algorithm [17], called "Propagation of Information" (PI). This initialization step produces a spanning tree with root at the sink. As we have already explained, each node lying on the spanning tree has a level, which is given by the level of its father plus one. In the case of the root, we say that it has a level equal to zero. The routing strategy that a given node follows is very simple; when the node decides to forward a packet to its father node, that is, along the spanning tree, we say that it uses a vertical routing; when it forwards a packet to any node of a level less or equal to that of its own, we say that it uses horizontal routing. It is very important to remark that, due to our modular design, a different settlement can be introduced to test alternative strategies.

4.2.4. Congestion Control. The module in charge of congestion control is the core of this work. This module provides a self-organized and self-configurable congestion control distributed algorithm that works like a traffic light that controls the transit of packets on their way to the sink. Either let the packets to go forward or stop the packets at the buffers. The "green" and "red" lights depend on a function that intends to maximize the overall system satisfaction. The algorithm that rules each traffic light has been explained in the previous section.

4.2.5. Wireless Effects. Our design considers the free space loss model, the effects of thermal noise, and interferences. We calculate the normalized signal-to-noise ratio and, for the binary phase-shift keying modulation (BPSK), we assess the bit error rate (BER). We determine the error probability per packet according to a fixed error tolerance. With this probability we are able to simulate packet losses due to wireless effects.

The simulations of wireless effects are considered and in the final statistics, it can be distinguished between packet losses due to wireless effects from those due to congestion.

The equation to calculate the signal power is

$$P_r = \frac{P_t \lambda^2}{(4\pi d)^2}, \quad (11)$$

where P_r is the received signal power, P_t is the power of transmitted signal, λ is the wavelength, and d is the distance between transmitter and receiver.

Wireless simulation also includes the effects of thermal noise and interference in transmission of each node. We can calculate the maximum capacity of the wireless channel (C) using the Shannon capacity for a given bandwidth (B).

Using (12) we found the normalized signal-to-noise ratio (E_b/N_o). Consider

$$\frac{E_b}{N_o} = \frac{B}{C} (2^{C/B} - 1). \quad (12)$$

We have chosen BPSK because it requires less E_b/N_o than other traditional modulation techniques. From a table [19] of BER for BPSK, BER is obtained, and considering the application requirements, specifically the bit error tolerance, the packet error rate (PER) is estimated as follows.

$$\text{PER} = 1 - \sum_{i=0}^e \binom{w}{i} [\text{BER}^i (1 - \text{BER})^{w-i}], \quad (13)$$

where e is the number of bit errors tolerated by the application and w is the packet size in bits. The packet losses due to wireless effects are estimated by the PER.

4.2.6. Energy Consumption. The WSN topology changes are represented using the power consumption simulation, due to the loss of connectivity caused by nodes that have depleted their battery energy. This effect is simulated by setting a percentage of consumed power in the data transmission. The consumption of energy due to processing capabilities is not considered in the simulation.

Packet losses resulting from the effects described in the previous paragraph are considered losses due to node level congestion. It is clear that as the network topology changes, an adaptive and self-configurable routing protocol is required to reduce node level congestion.

4.2.7. Performance Evaluation. This module performs the recording of different parameters of the network, such as packet losses, offered traffic, throughput, and goodput. Different performance metrics are also calculated with

TABLE 2: Simulation settings.

Parameter	Value
Number of nodes	250
Number of source nodes	20
Packet size	2048 bits
Transmission rate	180 kb/s
Buffer size	16384 bits
Average degree of connectivity	6
Average coverage range	110 m
Percentage of consumed battery per transmission	0.01
Transmission frequency	2048 MHz
Transmission model	Free space loss
Modulation scheme	BPSK
Bandwidth	4 kHz
Generation packet rate	30 packets/s

the information obtained, for example, buffering time and lifetime of the network. Additionally, the user can propose other performance metrics, which can be drawn from the measurement of ϕ_p and $\sigma_{\text{WSN}}(t)$.

5. Results and Discussion

5.1. Simulation Results. In order to evaluate the benefits of using our congestion control scheme, four evaluation scenarios are proposed:

- (1) a scenario without congestion control,
- (2) an adaptive-routing scenario based on packet interference,
- (3) a medium access control based on CSMA and applying an adaptive-adjustment of back-off intervals,
- (4) a congestion control scenario.

The fourth scenario corresponds to our congestion control scheme, which also considers the use of mechanisms defined in 2 and 3, as previously explained in Section 3. Simulation settings for the abovementioned scenarios are shown in Table 2.

Figure 2 shows the performance of our congestion control scheme for the four proposed scenarios. From this figure, it can be observed that the WSN performance considerably decreases once generated traffic exceeds 65% of the WSN capacity. On one hand, if no congestion control mechanism is applied in this case, the WSN performance may be reduced up to a point where the network collapses. On the other hand, when our proposed congestion control is used, the amount of delivered packets approaches 80% of the WSN capacity. Figure 2 also illustrates the benefits of applying, combined and separated, both mechanisms considered in the proposed congestion control. It is worth pointing out that, due to the cross-layer approach used in the proposed congestion control mechanism, there is a synergistic effect while using both mechanisms.

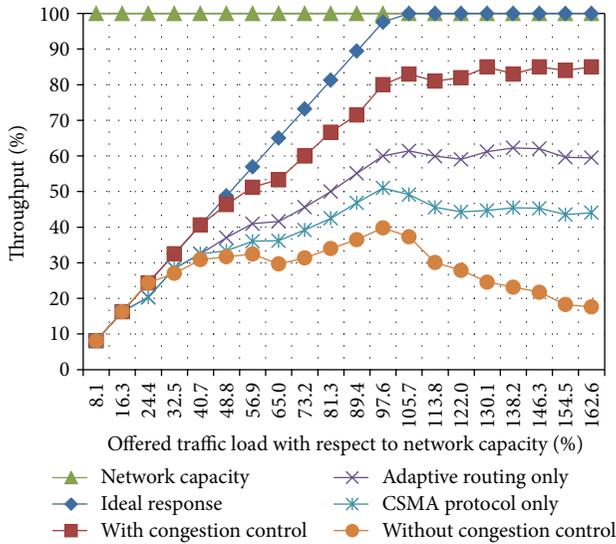


FIGURE 2: Throughput versus offered traffic load.

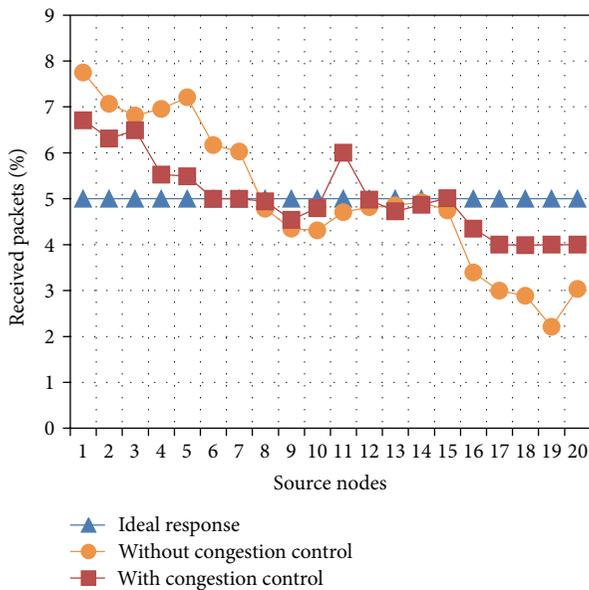


FIGURE 3: Percentage of received packets from each source node.

Another relevant advantage produced by the use of a congestion control mechanism is the equality of packet delivery. This goal is achieved upon adjusting specific congestion control parameters, which depend on node positions, in order to deliver information from all sources to sink node. Figure 3 shows the performance of the proposed congestion control for a simulation scenario which consists of 1 to 20 source nodes. An *ID.number* is assigned to each source node; a small *ID.number* indicates that the corresponding source node is found closer to the sink node and vice versa. In this figure, it can be observed that the congestion control improves a fair packet delivery. Ideally, if S is the number of source nodes, for a fair packet delivery, the percentage of received packets would be given by $P = 100/S$. Figure 3 shows

the ideal percentage of received packets when the number of source nodes is $S = 20$ nodes; that is, $P = 5\%$. As it can be observed from Figure 3, the use of the proposed congestion control increases the fairness in packet delivery.

In order to evaluate the fairness in packet delivery, we use Jain's Fairness Index [20]. In general terms, this metric provides a numerical estimate of how fair the resource assignment among network users is done by the system. In our case of study, we want to evaluate how fair the packet delivery from several sources to the sink in a WSN is performed. In order to use Jain's Fairness Index (JFI) to measure the average number of packets received by the sink from the source nodes, we use the following expression:

$$JFI = \frac{(\sum_{j=1}^m \hat{x}_j)^2}{m * \sum_{j=1}^m (\hat{x}_j)^2}, \tag{14}$$

where m corresponds to the number of source nodes in the network and \hat{x}_j corresponds to the percentage of received packets from source j ; (x_j) is measured with respect to the percentage of expected packets (\bar{x}_j) from source j ; that is,

$$\hat{x}_j = \frac{x_j}{\bar{x}_j}. \tag{15}$$

By using (14) with data presented in Figure 3, we obtained the following indexes:

$$JFI = 88.81\% \text{ (without using congestion control),}$$

$$JFI = 96.75\% \text{ (using congestion control).}$$

As it can be observed, the proposed control improves the fairness in packet delivery.

As mentioned above, the number of packets in a flock depends on the buffer size. Hence, the buffer size is an important parameter for the proposed congestion control. In order to evaluate the impact of this parameter on the congestion control behavior, a series of simulations were conducted while changing the buffer size. Figures 4 and 5 illustrate the percentage of lost packets versus the buffer size while using the proposed congestion control and by individually using CSMA and adaptive routing as well as without using a congestion control mechanism.

In these experiments, we consider two case scenarios which depend on the WSN application, that is, delay-tolerant and delay-sensitive applications. For delay-tolerant applications, if the buffer size is increased, the packet drop rate decreases up to reaching a certain level when the full network capacity is achieved, as it can be observed in Figure 4. For delay-sensitive applications, generally a packet expiration time is established. This parameter limits the number of packets in a flock, in addition to the buffer size. As a consequence of such expiration time, even if the buffer size is increased, the packet loss may increase due to the fact that all packets arriving after this time will be discarded by the application, as it can be seen in Figure 5. For this case scenario, the percentage of lost packets comprises the percentage of dropped packets plus the percentage of packets arriving out of time to the sink.

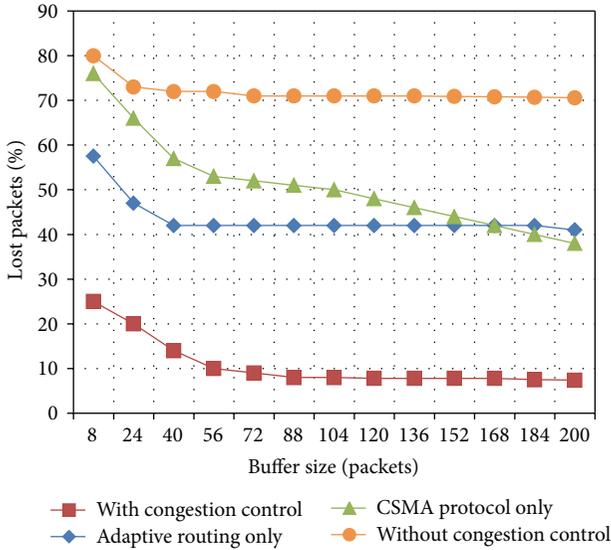


FIGURE 4: Lost packets versus buffer size for delay-tolerant applications.

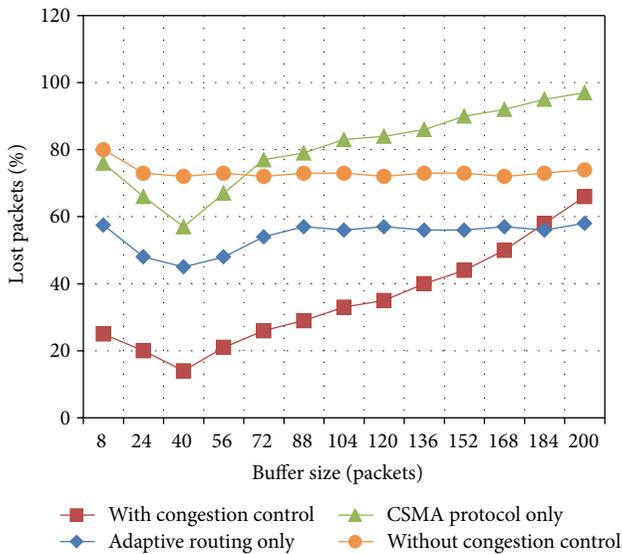


FIGURE 5: Lost packets versus buffer size for delay-sensitive applications.

Figure 6 shows the overall network satisfaction for different values of offered load. For each possible value on the horizontal axis, we took a snapshot after the sink has started receiving the first packets, and therefore we can assume that the control mechanism is already acting at every node. The network satisfaction only considers those packets having an individual satisfaction above zero. Otherwise, packets with null or negative satisfaction will be dropped in the next possible moment. This figure also shows that our proposed control should be regarded as a reactive mechanism that is triggered when the offered load is greater than the 50% of the network capacity. Nevertheless, we get the best out of

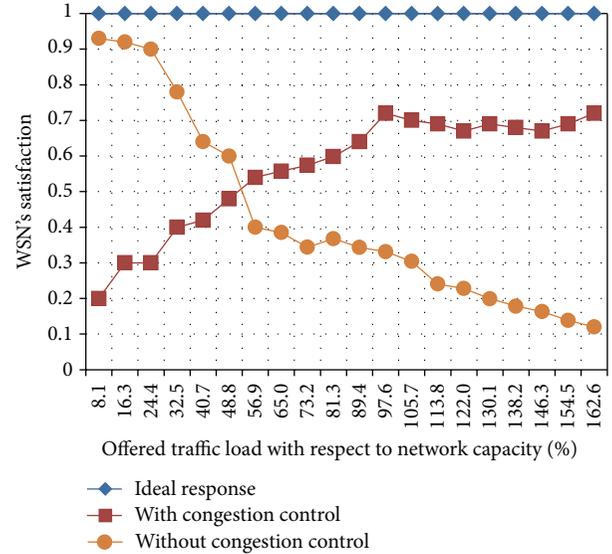


FIGURE 6: Network satisfaction as a function of the incoming traffic.

this control when the offered load is greater than the 100% of the network capacity.

The system requires some time to converge, and this time can be estimated as follows. Under these extreme conditions, it is considered that the offered load is over the full capacity of the network and flocks are assembled instantaneously. Then, the proposed adaptive mechanism reacts after the first packets arrive to the sink. Therefore, the time that the system takes to converge, that is, the time that the system takes to be in steady state, depends on the packet trip time, from the source(s) to the sink. If flocks are assembled with a certain delay, then the convergence will depend on the time it takes to assemble them, in addition to the packet trip time.

5.2. Discussion. The proposed congestion control provides a distributed solution. In this way, each node makes routing, contention, and selective packet dropping decisions based on its local traffic conditions. The main parameters of this solution are defined according to the zone where nodes are located, which can be translated on the hop distance between each node and the sink. Due to the fact that traffic conditions at each node of the WSN strongly depend on their distance to the sink node, traffic density is higher at zones closer to the sink node, and, therefore, it is expected that energy supply of these nodes is depleted more rapidly. The parameters of the congestion control, such as, packet lifetime, flock size, and maximum waiting time in the buffer node, are set up at each node according to its distance to the sink.

An important parameter of the proposed congestion control is given by the flock size, which must be adjusted with respect to buffer size. Flock size must be large enough to guarantee an effective operation of this solution. At this point, we can use as an analogy the vehicular traffic, where groups of vehicles are controlled by traffic lights. In this sense, a reduce-sized flock can be seen as a traffic light when the red light lasts only a short period of time, which is more

suitable for a low traffic density. However, for a WSN with a high density of data packets, the size of the flock should be large; otherwise, it may endeavor collision avoidance and self-organization capabilities of the WSN. In contrast, extremely large-sized flocks may produce long transmission delays, because packets must remain in a buffer waiting for the arrival of the amount of packets settled by the flock size. Considering previous conditions, the flock size can be specified at node level according to the following two factors: (a) the packet density found in each zone of the WSN and (b) the packet arrival rate.

Due to the fact that the proposed method has been designed to work on scenarios with high traffic densities, we have decided to define the flock size according to the packet density, as observed in (8), where the flock size depends on node distance with respect to the sink node.

We consider that, for variants of WSN, such as query-based WSN, where traffic conditions fluctuate more frequently, it would be useful to establish other congestion control parameters taking into account, for example, the arrival packet rate.

The proposed solution is based on *packet overhearing*, because it uses a modified version of CSMA as the medium access control, and transmission requests are used among neighboring nodes. *Packet overhearing* may represent a disadvantage for applications that require a long-life WSN, beyond keeping a continuous packet transmission. However, for a high traffic density WSN, which is caused by the detection of an event of interest, the main purpose of the network is to inform about the evolution of this event, although this may imply that active nodes may deplete their energy supply soon.

We mention at the beginning of this work that an underlying assumption was that, as in economic systems (which are also complex systems), there is an “invisible hand” that guides individual packets to benefit the system through the pursuit of their private interests. Nevertheless, modern economy admits the government intervention to settle regulations that help agents to achieve synergy. We consider that this is the role of nodes within our theoretical framework. Thus, in our solution, each node decides the status of a packet, depending on its microscopic measures. These local decisions enforce congestion control as an emergent behavior at the system level.

5.3. Implementation Issues. About the overhead required to deploy the control mechanism that has been proposed, we focus the analysis on two aspects: (i) the amount of additional information that a packet should carry and (ii) the additional processing that a node must perform.

The packet overhead is minimum, because it is only the satisfaction value ($\sigma_{p,M}$) achieved at the last node where the packet comes from.

As for the processing steps, each node has to calculate for each packet of a flock the following:

- (1) the local change of its satisfaction ($\Delta\sigma_{p,N}$, see (2) and (3)),
- (2) the accumulated satisfaction ($\sigma_{p,N}$, see (4)),

- (3) the interference each packet induces on the rest of its flock (ϕ_p , see (5)).

The first and second steps imply two subtractions, two divisions, and one addition, which means five elementary arithmetic operations. The third calculation seems to be expensive because (5) is invoked for each packet. Nevertheless, there is a simple shortcut that simplifies this step. Before the calculation of (ϕ_p) we add all the changes calculated at Step (1). Let us call X to this result, which implies $F - 1$ additions. Then we enter into a loop to calculate each individual interference but this time we subtract $\Delta\sigma_{p,N}$ from X . We divide this result by $(F-1)$ and we have the second part of ϕ_p . Finally, we again subtract this result from $-\Delta\sigma_{p,N}$. This step has taken three subtractions and one division, without considering the calculation of X (which is done only once). Besides this reduction in complexity we must recall that the calculation of ϕ_p is not always required. According to the forwarding procedure, ϕ_p is not necessary when none of the packets has experienced a reduction on its satisfaction. Otherwise, the node forwards immediately those packets whose satisfaction has been reduced. The node calculates the interference of each packet for the rest of the flock.

The number of packets that makes up a flock (F) plays a key role on the complexity of the processing steps that each node must perform. We also know that a flock occupies 1/3 of the buffer size for those nodes at zone 3 (the most distant places from the sink) and 2/3 of the buffers size for those nodes at zone 2, and it takes the overall buffer size for nodes in zone 1, which is in the vicinity of the sink. Therefore, the heaviest costs of this control are in charge of the nodes in zone 1. It is clear that these nodes define the carrying capacity of the network and therefore they are the busiest components of the system. This implies that these nodes may be the first to exhaust their energy budget. We consider that the deployed control contributes to extend the lifetime of these nodes, because there is a packet dropping mechanism acting at every node. This mechanism regulates the number of packets that finally arrive to the outskirts of the sink. Therefore, although the processing overhead mainly affects the nodes in zone 1, in compensation, the nodes in the other zones control the number of packets that arrive to zone 1. Also it is important to consider that, in terms of energy consumption, bit processing is less expensive than the corresponding transmission.

6. Conclusions

A conceptual framework for analyzing the traffic in wireless sensor networks (WSNs) from the “complex systems” perspective was developed. By means of this conceptual framework, it is possible to analyze the overall behavior of WSN traffic from local interactions of data packets. Although there is related work that analyzes the WSN as a “complex system,” the approach shown in our congestion control is different because the packets are considered as the main self-organized agents. In addition, the network congestion is mitigated using adaptive traffic mechanisms based on a satisfaction parameter assessed by each packet. These mechanisms have impact on the global satisfaction of

the traffic carried by the WSN. Then the proposed congestion control is able to preserve the traffic stream close to the WSN capacity, even in case scenarios with extreme congestion.

For the developed control, packets are sent in groups, called flocks, assessing whether a packet is causing friction to the flock. The main idea of the proposed solution is to reduce friction among data packets by means of a synergic encouragement, which increases the overall performance. According to this idea, each node makes decision about routing, contention, and packet dropping in order to improve global performance. For instance, in specific scenarios, flocks are able to avoid congested zones of the WSN. The CSMA protocol is used for the medium access control, which is modified to consider the remaining space in the buffer node. As a consequence, nodes can decide if they are able to carry the packets and how many bits they can receive avoiding congestion not only at the link level, but also at the node level. In addition, nodes far from the sink keep their packets in buffer for shorter time than those packets near to the sink. As a result, the congestion control also improves fairness in packet delivery. Furthermore, our solution provides a novel and efficient way to improve the performance of WSN.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported in part by research funds from CONACYT and PROMEP.

References

- [1] G. H. Raghunandan and B. N. Lakshmi, "A comparative analysis of routing techniques for Wireless Sensor Networks," in *Proceedings of the National Conference on Innovations in Emerging Technology (NCOIET '11)*, pp. 17–22, Tamilnadu, India, February 2011.
- [2] M. Mitchell, "What is complexity?" in *Complexity: A Guided Tour*, Oxford University Press, New York, NY, USA, 1st edition, 2009.
- [3] L. Yilmaz and T. I. Ören, "Agent-directed simulation systems engineering," in *Proceedings of the Summer Computer Simulation Conference (SCSC '07)*, pp. 897–904, San Diego, Calif, USA, July 2007.
- [4] Y. Chen and J. Qiao, "Application analysis of complex adaptive systems for WSN," in *International Conference on Computer Application and System Modeling (ICCASM '10)*, vol. 7, pp. 7328–7331, October 2010.
- [5] D. Orfanus, T. Heimfarth, and P. Janacik, "An approach for systematic design of emergent self-organization in wireless sensor networks," in *Proceedings of the Computation World: Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns (COMPUTATIONWORLD '09)*, pp. 92–98, Athens, Greece, November 2009.
- [6] M. A. Niazi and A. Hussain, "A novel agent-based simulation framework for sensing in complex adaptive environments," *IEEE Sensors Journal*, vol. 11, no. 2, pp. 404–412, 2011.
- [7] E. Shakshuki, H. Malik, and M. K. Denko, "Software agent-based directed diffusion in wireless sensor network," *Telecommunication Systems*, vol. 38, no. 3-4, pp. 161–174, 2008.
- [8] P. Antoniou, A. Pitsillides, A. Engelbrecht, and T. Blackwell, "Mimicking the bird flocking behavior for controlling congestion in sensor networks," in *Proceedings of the 3rd International Symposium on Applied Sciences in Biomedical and Communication Technologies (ISABEL '10)*, pp. 1–7, Rome, Italy, November 2010.
- [9] F. Dressler, "Locality driven congestion control in self-organizing wireless sensor networks," in *Proceedings of the 3rd International Conference on Pervasive Computing (Pervasive '05): International Workshop on Software Architectures for Self-Organization, and Software Techniques for Embedded and Pervasive Systems (SASO+STEPS '05)*, 2005.
- [10] A. A. Minai and Y. Bar-Yam, Eds., *Unifying Themes in Complex Systems, Volume IIIB: New Research*, Proceedings from the Third International Conference on Complex Systems, Springer, NECSI, 2006.
- [11] S. S. Doumit and D. P. Agrawal, "Self-organized criticality & stochastic learning based intrusion detection system for wireless sensor networks," in *Proceedings of the IEEE Military Communications Conference (MILCOM '03)*, vol. 1, pp. 609–614, October 2003.
- [12] S. Kawai and T. Asaka, "Event-driven Wireless Sensor Networks using energy-saving data collection," in *Proceedings of the 18th Asia-Pacific Conference on Communications (APCC '12)*, pp. 300–305, Jeju Island, Republic of Korea, October 2012.
- [13] H. Rangarajan and J. J. Garcia-Luna-Aceves, "Reliable data delivery in event-driven wireless sensor networks," in *Proceedings of the 9th International Symposium on Computers and Communications (ISCC '04)*, vol. 1, pp. 232–237, July 2004.
- [14] E. Shakshuki, H. Malik, and X. Xing, "Agent-based routing for wireless sensor network," in *Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues*, vol. 4681 of *Lecture Notes in Computer Science*, pp. 68–79, Springer, 2007.
- [15] E. M. Shakshuki, T. R. Sheltami, and M. Akhlaq, "Sandstorm monitoring system architecture using agents and sensor networks," in *Proceedings of the 10th International Conference on Advances in Mobile Computing & Multimedia (MoMM '12)*, I. Khalil, Ed., pp. 253–256, ACM, New York, NY, USA, 2012.
- [16] C. Gershenson, *Design and Control of Self-Organizing Systems*, 1st edition, 2007, <http://arxiv.org/abs/nlin/0505009>.
- [17] A. Segall, "Distributed network protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 1, pp. 23–35, 1983.
- [18] U. Wilensky, "NetLogo," 1999, <http://ccl.northwestern.edu/netlogo/>.
- [19] W. Stallings, "Antennas and propagation," in *Wireless Communications and Network*, pp. 95–126, Pearson Prentice Hall, 2005.
- [20] R. Jain, D. Chiu, and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer systems," Tech. Rep. TR-301, Eastern Research Lab, Digital Equipment Corporation, Hudson, Mass, USA, 1984.

Research Article

A Variable Neighborhood Walksat-Based Algorithm for MAX-SAT Problems

Noureddine Bouhmala

Department of Maritime Technology and Innovation, Buskerud and Vestfold University, Norway

Correspondence should be addressed to Noureddine Bouhmala; noureddine.bouhmala@hbv.no

Received 23 April 2014; Accepted 10 June 2014; Published 6 August 2014

Academic Editor: Su Fong Chien

Copyright © 2014 Noureddine Bouhmala. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The simplicity of the maximum satisfiability problem (MAX-SAT) combined with its applicability in many areas of artificial intelligence and computing science made it one of the fundamental optimization problems. This NP-complete problem refers to the task of finding a variable assignment that satisfies the maximum number of clauses (or the sum of weights of satisfied clauses) in a Boolean formula. The Walksat algorithm is considered to be the main skeleton underlying almost all local search algorithms for MAX-SAT. Most local search algorithms including Walksat rely on the 1-flip neighborhood structure. This paper introduces a variable neighborhood walksat-based algorithm. The neighborhood structure can be combined easily using any local search algorithm. Its effectiveness is compared with existing algorithms using 1-flip neighborhood structure and solvers such as CCLS and Optimax from the eighth MAX-SAT evaluation.

1. Introduction

Many optimization algorithms have been developed for successfully solving a wide range of optimization problems. Although these techniques have demonstrated excellent search capabilities for solving small or medium sized optimization problems, they still encounter serious challenges when applied to solving large scale optimization problems, that is, problems with several hundreds to thousands of variables. How well optimization algorithms handle this sort of real world large scale optimization problems still remains an open question for various optimization problems including MAX-SAT. MAX-SAT is a widely used modeling framework for solving various combinatorial problems. Many important applications can be naturally expressed as MAX-SAT [1]. Examples include routing [2], scheduling [3], model-checking [4] of finite state systems, design debugging [5], AI planning [6], and electronic markets [7]. Interested readers may refer to [8–10] for more details. Efficient methods that can solve large and hard instances of MAX-SAT are eagerly sought. Due to their combinatorial explosion nature, large and complex MAX-SAT problems are hard to solve using systematic algorithms based on branch and bound techniques

[11]. One way to overcome the combinatorial explosion is to give up completeness. Stochastic local search algorithms (SLS) are techniques which use this strategy and gained popularity in both worlds whether it is discrete or continuous due to their conceptual simplicity and good performance. The Walksat algorithm [12] is considered to be the main skeleton underlying almost all SLS algorithms for MAX-SAT. It works by assigning all the variables a random truth assignment and then tries to refine the assignment according to a selected heuristic until the CNF formula evaluates to true. The heuristic used for varying the truth assignment defines the variant of Walksat. All variants share the common behavior of exploiting the standard 1-flip neighborhood structure for which two truth value assignments are neighbors if they differ in the truth value of exactly one variable. The critical issue in the design of a neighborhood search strategy is the choice of the neighborhood structure, that is, the manner in which the neighborhood is defined. Larger neighborhood yields better local optima but the computational effort spent to search the neighborhood increases exponentially $O(n^k)$ where k is the cardinality of neighborhood (i.e., the number of variables to be flipped in order to move from the current solution s_i to a neighboring solution s_j) and n is the number of variables [13].

In this paper a variable neighborhood Walksat-based algorithm is introduced for MAX-SAT. The key feature of this algorithm aims at identifying improved neighbor solutions without explicitly enumerating and evaluating all the neighbors in the neighborhood. The strategy involves looking at the search as a process evolving from a k -flip neighborhood to the standard 1-flip neighborhood-based structure in order to achieve a tactical interplay between diversification (i.e., the ability to explore many different regions of the search space) and intensification (i.e., the ability to obtain high quality solutions within those regions). The authors in [14] discuss the latest design of hybrid approaches in order to find an adequate balance between diversification and intensification.

The rest of the paper is organized as follows. A definition of MAX-SAT is given in Section 2. Section 3 provides a short survey of methods used to solve MAX-SAT. Section 4 explains the Walksat algorithm. Section 5 introduces the variable neighborhood Walksat-Based algorithm and the experimental results. Finally, conclusions are drawn in Section 6.

2. The Maximum Satisfiability Problem

Generally, the satisfiability problem (SAT) which is known to be NP-complete [15] is defined as follows. Given is a propositional formula Φ consisting of a set of N variables usually represented in CNF (conjunctive normal form). In CNF, the formula is represented as a conjunction of clauses written as $\Phi = C_1 \wedge C_2 \wedge C_3 \wedge \dots \wedge C_M$, with M being the number of clauses. A clause $C_i(x)$ is a disjunction of literals and a literal is a variable or its negation. As a simple example, let $\Phi(x)$ be the following formula containing 4 variables and 3 clauses:

$$\Phi(x) = (x1 \vee \neg x4) \wedge (\neg x1 \vee x3) \wedge (\neg x1 \vee x4 \vee x2). \quad (1)$$

The task is to determine whether there exists an assignment of values to the variables under which $\Phi(x)$ evaluates to true. Such an assignment, if it exists, is called a satisfying assignment for Φ , and Φ is called satisfiable. Otherwise, Φ is said to be unsatisfiable. There exist two important variations of the MAX-SAT problem. The weighted MAX-SAT problem is the MAX-SAT problem in which each clause is assigned a positive weight. The goal of the problem is to maximize the sum of weights of satisfied clauses. The unweighted MAX-SAT problem is the MAX-SAT problem in which all the weights are equal to 1 and the goal is to maximize the number of satisfied clauses. In this paper, the focus is restricted to formulas in which all the weights are equal to 1 (i.e., unweighted MAX-SAT).

3. Short Survey of SLS for MAX-SAT

Stochastic local search algorithms [16] are based on what is perhaps the oldest optimization method, trial and error. Typically, they start with an initial assignment of values to variables randomly or heuristically generated. During each iteration, a new solution is selected from the neighborhood

of the current one by performing a move. Choosing a good neighborhood and a method for searching is usually guided by intuition, because very little theory is available as a guide. All the methods usually differ from each other in the criteria used to flip the chosen variable. One of the earliest local searches for solving SAT is GSAT [17]. The GSAT algorithm operates by changing a complete assignment of variables into one in which the maximum possible number of clauses is satisfied by changing the value of a single variable. Another widely used variant of GSAT is the Walksat based on a two-stage selection mechanism which is originally introduced in [12]. Several state-of-the-art local search algorithms are enhanced versions of GSAT and Walksat algorithms [18–20]. As the quality of the solution improves when larger neighborhood is used, the work proposed in [13] uses restricted 2- and 3-flip neighborhoods and better performance has been achieved compared to the 1-flip neighborhood for structured problems. Clause weighting based SLS algorithms [21, 22] have been proposed to solve SAT and MAX-SAT problems. The key idea is to associate the clauses of the given CNF formula with weights. Although these clause weighting SLS algorithms differ in the manner clause weights should be updated (probabilistic or deterministic), they all choose to increase the weights of all the unsatisfied clauses as soon as a local minimum is encountered. Numerous other methods such as Simulated Annealing [23], Evolutionary Algorithms [24, 25], Scatter Search [26], Greedy Randomized Adaptive Search Procedures [27], and guided local search [28] have also been developed. Lacking the theoretical guidelines while being stochastic in nature, the deployment of several SLS involves extensive experiments to find the optimal noise or walk probability settings. To avoid manual parameter tuning, new methods have been designed to automatically adapt parameter settings during the search [29, 30] and results have shown their effectiveness for a wide range of problems. The work conducted in [31] introduced Learning Automata (LA) as a mechanism for enhancing SLS based SAT solvers, thus laying the foundation for novel LA-based SAT solvers. A new strategy based on an automatic procedure for integrating selected components from various existing solvers has been devised in order to build new efficient algorithms that draw the strengths of multiple algorithms [32, 33]. The work conducted in [34] proposed an adaptive memory based local search algorithm that exploits various strategies in order to guide the search to achieve a suitable tradeoff between intensification and diversification. The computational results show that it competes favorably with some state-of-the-art MAX-SAT solvers. Finally, new solvers have emerged based on a new diversification scheme to prevent cycling [35–37].

4. Walksat/SKC Algorithm

In this section, the Walksat/SKC (WS) algorithm originally introduced in [12] which constitutes the chosen local search that will be combined with systematic changes of neighborhood is shown in Algorithm 1.

The algorithm starts with a random assignment (line 3). Thereafter, a random unsatisfied clause is selected (line 5).

```

input: Problem in CNF format
output: Number of satisfied clauses
(1) begin
(2)   for  $i \leftarrow 1$  to  $MAX\text{-TRIES}$  do
(3)      $T \leftarrow \text{Random-Assignment}()$ ;
(4)     for  $i \leftarrow 1$  to  $MAX\text{-FLIPS}$  do
(5)        $C_k \leftarrow \text{Random-Unsatisfied-Clause}()$ ;
(6)       if  $(\exists \text{variable } v \in C_k \text{ with } \text{breakcount} = 0)$  then  $\text{Chosen-Variable} \leftarrow v$ ;
(7)       else if  $(\text{random}(0,1) \leq p_{\text{noise}})$  then
(8)          $\text{Chosen-Variable} \leftarrow \text{Random-Variable}(C_k)$ ;
(9)       else
(10)         $\text{Chosen-Variable} \leftarrow \text{Random-Lowest-Break count}(C_k)$ ;
(11)      end
(12)    end
(13)  end
(14) end

```

ALGORITHM 1: Walksat algorithm.

If there exists a variable belonging to the selected clause with break count equal to zero (line 6), this variable is flipped; otherwise a random variable (line 8) or the variable with minimal break count (line 10) is selected with a certain probability (noise probability: line 7). The break count of a variable is defined as the number of clauses that would be unsatisfied by flipping the chosen variable. It turns out that the choice of unsatisfied clauses, combined with the randomness in the selection of variables, can enable Walksat to avoid local minima and to better explore the search space. The flips are repeated until a preset value of the maximum number of flips is reached (MAX-FLIPS) and this phase is repeated as needed up to MAX-TRIES times.

5. The Algorithm

The main difference between metaheuristics relies in the way neighborhood structures are defined and explored. Some metaheuristics work only with a single neighborhood structure. Others, such as numerous variants of variable neighborhood search, operate on a set of different neighborhood structures. Variable neighborhood search (VNS for short) [38–40] aims at finding a tactical interplay between diversification and intensification [16] to overcome local optimality using a combination of a local search and systematic changes of neighborhood. Diversification refers to the ability to explore many different regions of the search space, whereas intensification refers to the ability to obtain high quality solutions within those regions. The basic VNS starts by selecting a finite set of predefined neighborhood structures that will be used during the search. Let N_k ($k = 1, 2, \dots, k_{\max}$) denote the selected set and let $N_k(x)$ denote the set of solutions in the k th neighborhood of x . Let S_{start} denote the initial solution. VNS starts by generating a random solution S_{rand} from the neighborhood $N_1(S_{\text{rand}}) \in N_1(S_{\text{start}})$. Let S_{new} denote the reached local optimum when a local search is used with S_{rand} as input. If S_{new} is better compared to S_{rand} , the solution is updated and a new round of local

search with a random solution from $N_1(S_{\text{new}})$ is performed. If the test fails, VNS moves to the next neighborhood. The effectiveness of VNS is strongly affected by the ordering in which a given type of neighborhood is considered [41]. Bearing this concept in mind, it is obvious that the application order of the neighborhood structures is crucial for the performance of VNS. Most of the work published earlier on VNS starts from the first neighborhood and moves on to higher neighborhoods without controlling and adapting the ordering of neighborhood structures. Few research articles have begun to search for strategies to dynamically move from one neighborhood to another based on some benefit metrics. Algorithm 2 shows the details of the variable neighborhood Walksat-based Algorithm which consists of two phases.

(i) *Phase 1.* Let P denote the set of variables of the problem to be solved. The first phase of the algorithm consists in constructing a set of neighborhoods satisfying the following property: $N_1(x) \subset N_2(x) \subset \dots \subset N_{k_{\max}}(x)$. The starting neighborhood with $k = 0$ consists of a move based on the flip of a single variable. A flip means assigning the opposite state to a variable (i.e., change True \rightarrow False or False \rightarrow True). The first neighborhood N_1 is constructed from P by merging variables. The merging procedure is computed using a randomized algorithm. The variables are visited in a random order. If a variable l_i has not been matched yet, then a randomly unmatched variable l_j is selected and a new variable l_k (a cluster) consisting of the two variables l_i and l_j is created. The set N_1 consists of the move based on flipping predefined clusters each having 2^1 variables. The new formed clusters are used to define a new and larger neighborhood N_2 and recursively iterate the process until the desired number of neighborhoods (k_{\max}) is reached (lines 3, 4, and 5 of Algorithm 1). Thereafter, a random solution is generated from the largest neighborhood ($N_{k_{\max}}$) (line 2 of Algorithm 2). The random solution consists in assigning True or False to each cluster and all the literals within that cluster will get the same state.

```

input: Problem in CNF Format
output: Number of unsatisfied clauses
(1) /* Determine the set of neighborhood structures  $N_k$  ( $k = 1, 2, \dots, k_{\max}$ ) /*;
(2)  $k := 0$ ;
(3) while (Not reached the desired set of neighborhood) do
(4)    $N_{k+1} := \text{Construct}(P_k)$ ;
(5)    $k := k + 1$ ;
(6) Generate a random solution  $S_{\text{current}}$  from  $k_{\max}^{\text{th}}$  neighborhood;
(7)  $k \leftarrow k_{\max}$ ;
(8) while ( $k \geq 0$ ) do
(9)    $S_{\text{new}} \leftarrow \text{Apply Walksat with } S_{\text{current}} \text{ as input solution}$ ;
(10)   $S_{\text{current}} \leftarrow \text{Project}(S_{\text{new}})$ ;
(11)   $k \leftarrow k - 1$ ;
(12)  $S_{\text{new}} \leftarrow \text{Apply Walksat with } (S_{\text{current}}) \text{ as input solution}$ ;
(13) return ( $S_{\text{new}}$ );

```

ALGORITHM 2: VNS-WS.

(ii) *Phase 2.* The second phase which is the most crucial aims at selecting the different neighborhoods according to some strategy for the effectiveness of the search process. The strategy adopted in this work is to let VNS start the search process from the largest neighborhood $N_{k_{\max}}$ and continue to move towards smaller neighborhood structures (lines 7, 8, 9, 10, and 11 of Algorithm 2). The motivation behind this strategy is that the order in which the neighborhood structures have been selected offers a better mechanism for performing diversification and intensification. The largest neighborhood N_{\max} allows WS to view any cluster of variables as a single entity leading the search to become guided in faraway regions of the solution space and restricted to only those configurations in the solution space in which the variables grouped within a cluster are assigned the same value. As the switch from one neighborhood to another implies a decrease in the size of the neighborhood, the search is intensified around solutions from previous neighborhoods in order to reach better ones. Once the search has reached the convergence criterion with respect to neighborhood N_i , the assignment reached on that neighborhood must be projected on its parent neighborhood N_{i-1} . The projection algorithm (line 10 of Algorithm 2) is simple; if a cluster $c_i \in N_m$ is assigned the value of true, then the merged pair of clusters that it represents, $c_j, c_k \in N_{m-1}$, are also assigned the true value. Finally, the algorithm Walksat is applied at the default neighborhood (line 12 of Algorithm 2). This process is graphically illustrated in Figure 1 using an example with 12 variables. During the first phase, a random merging procedure is used to merge randomly the variables in pairs leading to the first neighborhood N_1 consisting of six clusters each of which is composed of 2 variables. The second neighborhood N_2 is constructed in the same manner. The clusters formed at neighborhood N_1 are merged randomly in pairs leading to a new neighborhood N_2 consisting of three clusters each of which is composed of 2 different clusters each having 2 variables. When the construction of the different neighborhoods comes to its end, a random solution is computed at the neighborhood N_2 . Each cluster will be

assigned a random value (True or False). Thereafter, the heuristic WS is applied at N_2 . When WS flips a cluster from True to False at N_2 , all the variables within that cluster (2^2) will get the same value. When WS reaches the convergence criterion, WS is applied to a smaller neighborhood (N_1), where a move made by WS will consist in flipping a cluster which is having 2 variables. The last step consists in applying WS at N_0 where a move made by WS will consist in flipping a single variable. At this neighborhood, one expects that WS has reached the maximum amount of unsatisfied clauses.

The performance of VNS-WS is evaluated against WS using a set of real industrial problems. This set is taken from the eighth MAX-SAT 2013 organized as an affiliated event of the 16th International Conference on Theory and Applications of Satisfiability Testing (SAT-2013). Due to the randomization nature of both algorithms, each problem instance was run 50 times with a cutoff parameter (max-time) set to 30 minutes. The tests were carried out on a DELL machine with 800 MHz CPU and 2 GB of memory. The code was written in C++ and compiled with the GNU C compiler version 4.6. The following parameters have been fixed experimentally and are listed below:

- (i) k_{\max} : the cardinality of the neighborhood is set such that the number of the formed clusters is 10% of the size of the problem instance (i.e., a problem with 100 literals will lead to k_{\max} equal to 3).
- (ii) WS spends equal amount of time (max-time/ k_{\max}) between the different neighborhoods.
- (iii) Noise probability: the performance of WS depends highly on the walking probability setting which in turns depends on the class of problems to be solved. The plots in Figures 1 and 2 show four selected tests that reflect the general trend observed on almost all the industrial instances tested. Peak performance with respect to the lowest number of unsatisfied clauses is achieved when the walking probability was set to 10.

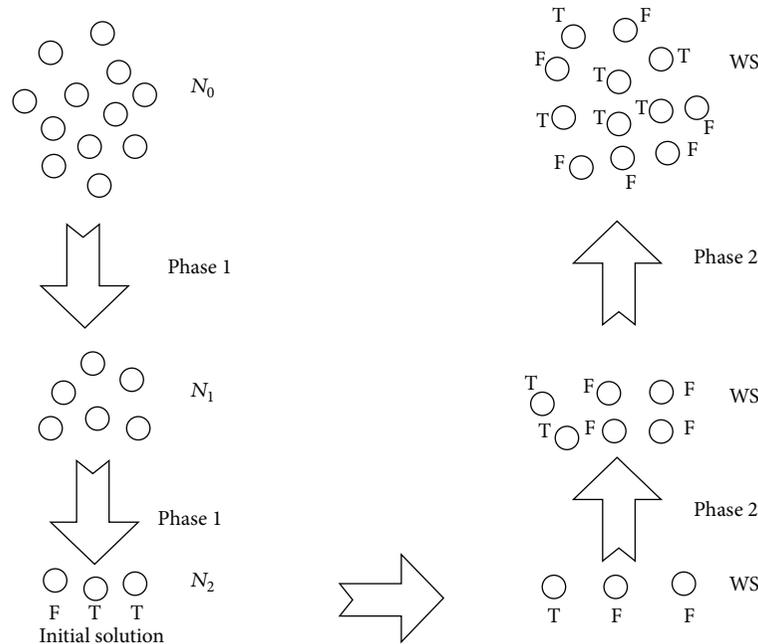


FIGURE 1: Example illustrating the different phases of the algorithm.

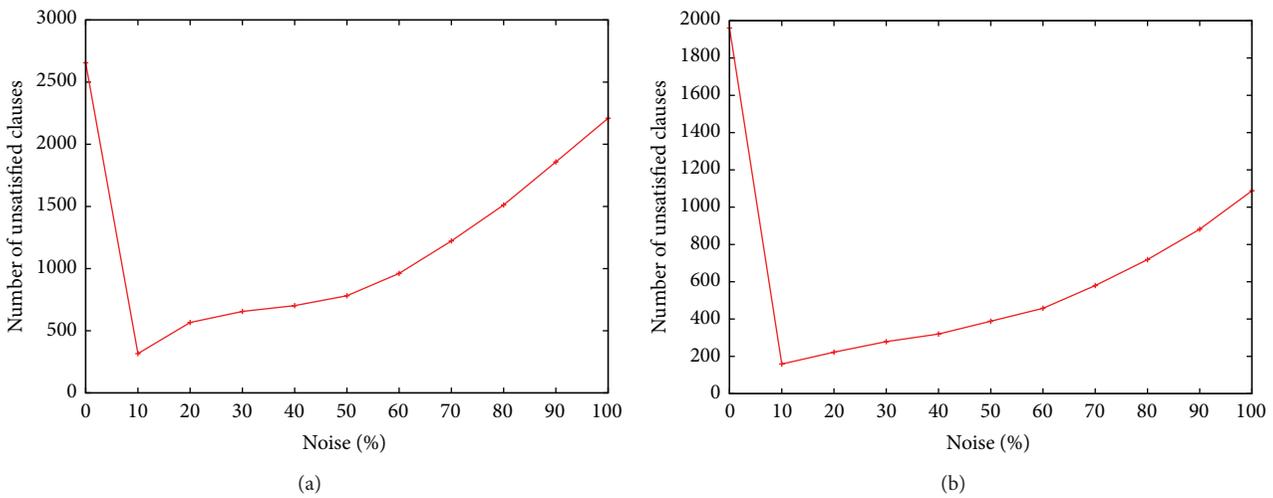


FIGURE 2: Noise probability versus number of unsatisfied clauses: (a) i2c-master1.dimacs.filtered.cnf: variables = 82429, clauses = 285987, (b) mem-ctrl-debug.dimacs.cnf: variables = 381721, clauses = 505547.

5.1. *Observed Search Trend.* Figures 3 and 4 show the evolution of the mean of unsatisfied clauses of both algorithms as a function of time. Both algorithms provide an initial solution of the same quality while showing a crossover in their corresponding curves. During the early phase of the search, the solution quality provided by WS is better compared to VNS-WS. The superiority of WS lasts for a short while before VNS-WS catches up and surpasses WS. Both algorithms were able to decrease the mean number of unsatisfied clauses at a high rate before entering the so-called plateau region where WS typically encounters a sequence of states that leave the number of unsatisfied clauses unchanged. While WS shows a premature stagnation behavior of the search, VNS-WS was

capable of finding neighboring states with fewer unsatisfied clauses, thereby exiting the plateau. VNS-WS shows equal or marginally better asymptotic convergence for small problems compared to WS as the two curves overlay each other closely, while the convergence behavior becomes more distinctive for larger problems. The key behind the efficiency of VNS-WS relies on the variable neighborhood structure. VNS-WS draws its strength from coupling WS across different neighborhoods. This paradigm offers two main advantages which enables WS to become much more powerful. During the improvement phase (i.e., each time WS is called with a different neighborhood), WS applies a local transformation (i.e., a move) within the neighborhood (i.e., the set of

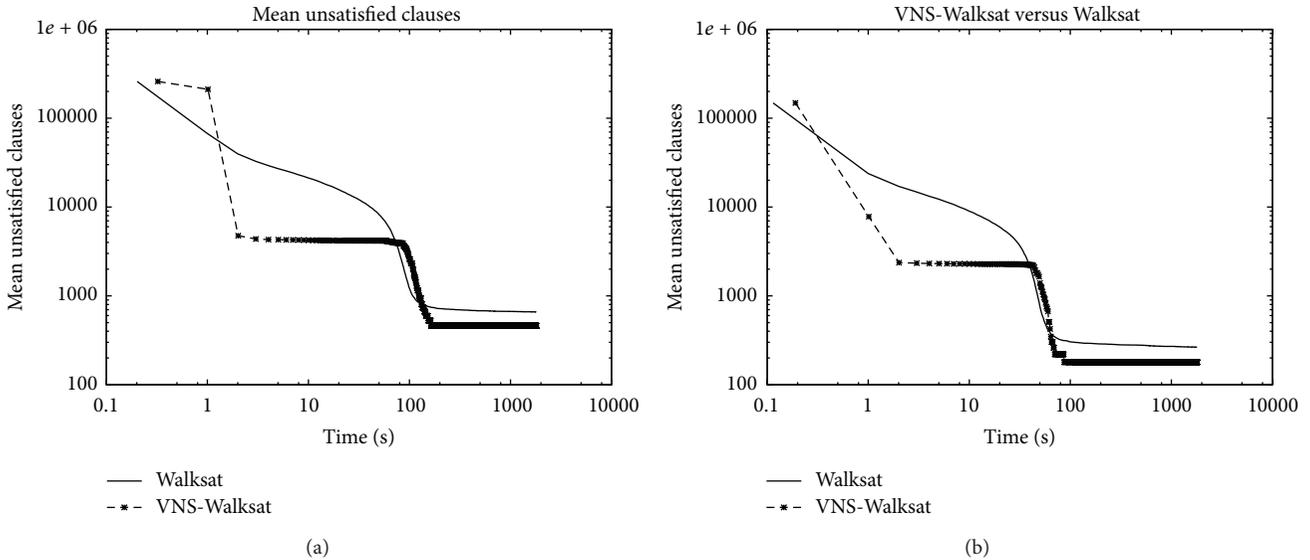


FIGURE 3: Log-Log Plot: (a) rsdecoder2.dimacs.filtered.cnf: variables = 415480, clauses = 1632526, (b) rsdecoder-fsm1.dimacs.filtered.cnf: variables = 238290, clauses = 936006. Time development for 100 runs in 15 minutes.

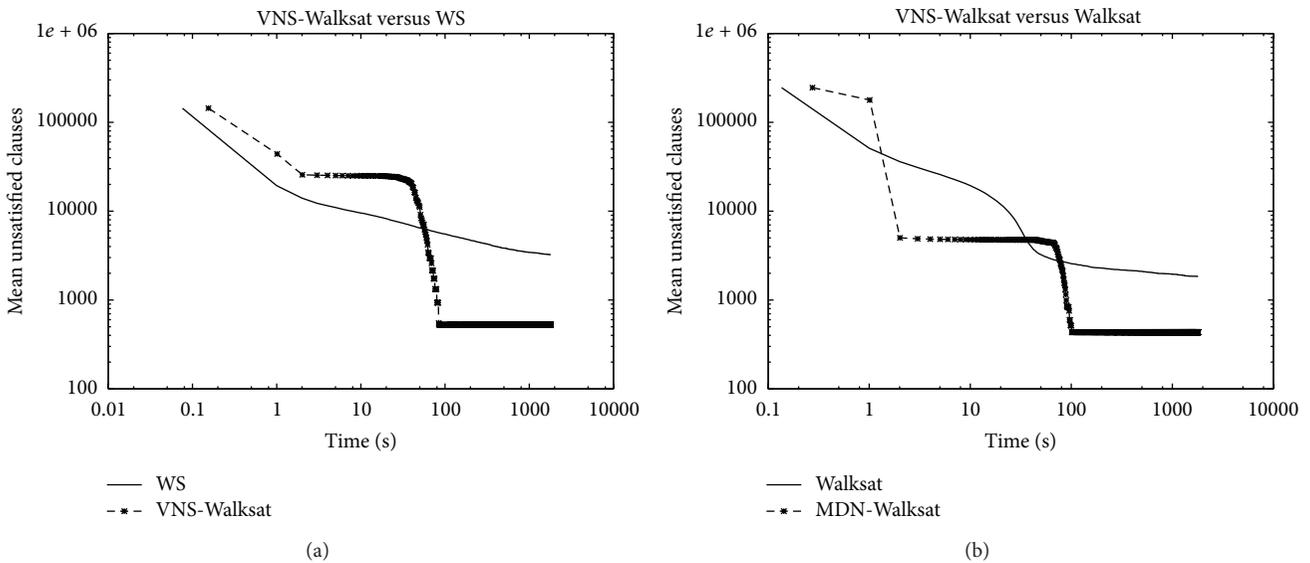


FIGURE 4: Log-Log Plot: (a) rsdecoder1-blackbox-CSEEBlock-problem.dimacs-32.filtered: variables = 277950, clauses = 806460, (b) rsdecoder-multivecl.dimacs.filtered: variables = 394446, clauses = 1626312. Time development for 100 runs in 15 minutes.

solutions that can be reached from the current one) of the current solution to generate a new one. The selected variable neighborhood structure offers a better mechanism for performing diversification and intensification. By allowing WS to view a cluster of variables as a single entity, the search becomes guided and restricted to only those configurations in the solution space in which the variables grouped within a cluster are assigned the same value. The switch from one neighborhood to another implies a decrease in the size of the neighborhood leading the search to explore different regions in the search space, while intensifying the search by exploiting the solutions from previous neighborhoods in order to reach better ones.

5.2. *Convergence Speed.* Figures 5 and 6 show the convergence speed behavior expressed as the ratio between the mean of unsatisfied clauses of the two algorithms as a function of time. A negative value demonstrates the superiority of WS while a positive value confirms the opposite. For some instances, WS exhibits a better convergence speed during the early stage of the search before the ratio turns in favor of VNS-WS which starts demonstrating its dominance as the search continues. The asymptotic performance offered by VNS-WS is impressive and dramatically improves on WS. In some cases, the difference in the convergence speed reaches 20% during the first seconds and maintains this level during the whole search process as expressed in the right plot of Figure 5.

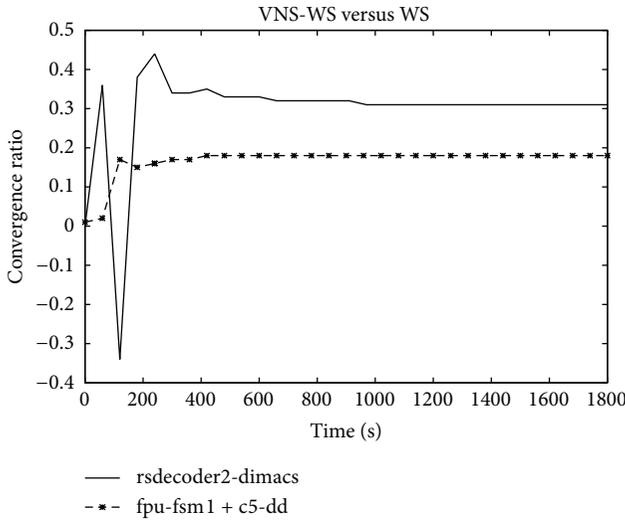


FIGURE 5: Convergence speed.

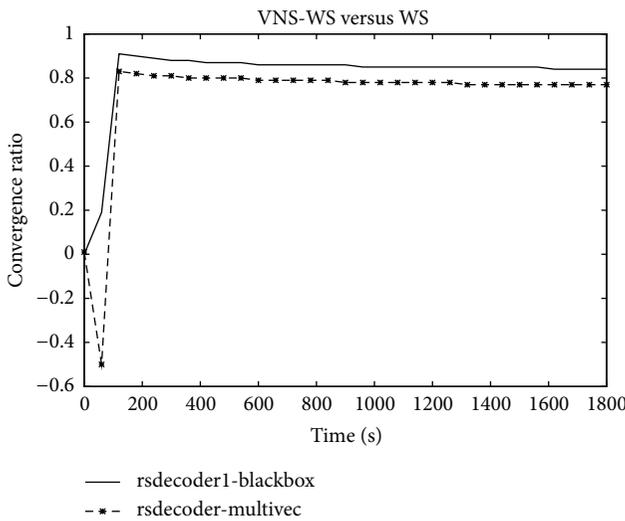


FIGURE 6: Convergence speed.

However, on other cases, the difference continues to increase as the search progresses and gets as high as 93% as shown in Figure 6. The plot depicted in Figure 7 shows the number of unsatisfied clauses as a function of the clause to variable ratio. The first thing to notice is that as the ratio of clauses to variables increases, the number of unsatisfied clauses produced by VNS-WS remains lower while not showing a substantial variation compared to WS. The second thing is the existence of a crossover point at which the difference in the solution quality between the two algorithms is the highest. This turning point occurs at 4.5 and might represent the set of instances that are harder to solve.

5.3. Comparison of VNS-WS with Other Algorithms. Tables 1, 2, and 3 compare VNS-WS with three state-of-art algorithms (WS, Walksat with weights W-w, and variable weighting scheme VW2) using the package UCBSAT [42]. W-w and VW2 have proven to be very effective giving the best known

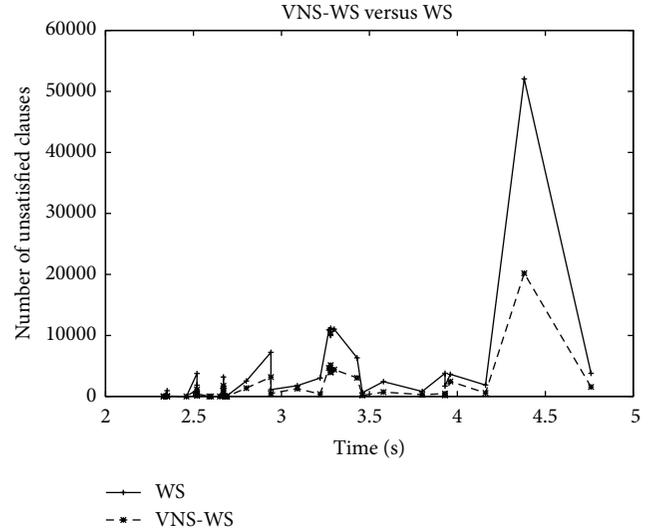


FIGURE 7: WS versus VNS-WS: clause to variable ratio.

results on some industrial benchmarks [43]. The first and second columns show the number of variables and clauses for the instance input. The last four columns show the number of unsatisfied clauses produced by each method. VNS-WS gave the better results than W-w and VW2 in 38 cases out of 44. When compared to VW2, the improvement ranges from 64% to 99% and from 29% to 99% when compared to W-w. Similar results were observed in 6 cases and beaten in one case by W-w. The comparison against WS shows that VNS-WS outperforms WS in 39 cases with an improvement ranging from 28% to 92% while similar results were observed in the remaining 5 cases. Table 4 compares VNS-WS with highly efficient solvers CCLS [35] and Optimax which is a modified version of glucose SAT solver [44] ranked 1st at the 2011 SAT competition. CCLS won four categories of the incomplete algorithms track of Max-SAT Evaluation 2013. The instances used in the benchmark belong to random and crafted categories used at SAT2013 competition. VNS-WS gave similar quality results in 20 cases out of 27. However the time of CCLS ranges from 10% to 96% of the time of VNS-WS except in one case (s3v80-900-2) where VNS-WS was 39% faster compared to CCLS. In the remaining cases where VNS-WS was beaten, the difference in quality ranges from 2% to 11%. Another interesting remark to mention is that the time required by VNS-WS does vary significantly depending on the problem instance while the variations observed with CCLS remain very low. The comparison between Optimax and VNS-WS shows that Optimax converges very fast at the expense of delivering solutions of poor quality compared to VNS-WS. VNS-WS was capable of delivering solutions of better quality than Optimax in all the cases and the improvement ranges from 13% to 66%.

6. Conclusions and Future Research

In this work, a variable neighborhood search combined with Walksat (VNS-WS) for the maximum satisfiability problem

TABLE 1: SAT2013 industrial benchmarks: comparison among VNS-WS, WS, W-w, and VW2.

Instances	Instance input		Unsatisfied clauses			
	Var	Cls	WS	W-w	VW2	VNS-WS
diverders11.dimacs.filtered	45552	162982	2434	2101	4169	715
fpu8-problem.dimacs24.filtered	160232	548848	6328	6306	11290	3043
fpu-fsm1-problem.dimacs15.filtered	160200	548843	6125	6213	11855	3055
i2c-master2.dimacs.filtered	63816	221320	615	590	2037	161
b14-opt-bug2-vec1-gate-0.dimacs	130328	402707	1763	1749	5985	1279
b15-bug-fourvec-gate-0.dimacs	581064	1712690	7241	7596	22082	3184
b15-bug-onevec-gate-0.dimacs	121836	359040	1122	1244	4094	493
c1-DD-s3-f1-e2-v1-bug-fourvec-gate-0.dimacs	391897	989885	955	1317	4966	60
c1-DD-s3-f1-e2-v1-bug-onevec-gate-0.dimacs	102234	258294	62	191	592	2
c2-DD-s3-f1-e2-v1-bug-onevec-gate-0.dimacs	84525	236942	2537	2634	3699	1362
c3-DD-s3-f1-e1-v1-bug-fourvec-gate-0.dimacs	33540	86944	4	4	4	4
c3-DD-s3-f1-e1-v1-bug-onevec-gate-0.dimacs	8358	21736	1	1	1	1
c4-DD-s3-f1-e1-v1-bug-gate-0.dimacs	797728	2011216	3761	5714	15798	1129
c4-DD-s3-f1-e2-v1-bug-fourvec-gate-0.dimacs	448465	1130672	1834	1993	7772	516
c4-DD-s3-f1-e2-v1-bug-onevec-gate-0.dimacs	131548	331754	439	482	1899	105

TABLE 2: SAT2013 industrial benchmarks: comparison among VNS-WS, WS, W-w, and VW2.

Instances	Instance input		Unsatisfied clauses			
	Var	Cls	WS	W-w	VW2	VNS-WS
c5315-bug-gate-0.dimacs.seq.filtered	1880	5049	1	1	1	1
c5-DD-s3-f1-e1-v1-bug-fourvec-gate-0.dimacs	100472	270492	25	25	943	4
c5-DD-s3-f1-e1-v1-bug-gate-0.dimacs	200944	540984	97	153	2026	8
c5-DD-s3-f1-e1-v2-bug-gate-0.dimacs	200944	540984	59	101	1873	8
c5-DD-s3-f1-e1-v1-bug-onevec-gate-0.dimacs	25118	67623	11	11	65	1
c6288-bug-gate-0.dimacs.seq.filtered	3462	9285	14	16	63	1
c6-DD-s3-f1-e1-v1-bug-fourvec-gate-0.dimacs	170019	454050	1668	1644	4900	921
c6-DD-s3-f1-e1-v1-bug-gate-0.dimacs	298058	795900	3188	3332	9340	1752
c6-DD-s3-f1-e1-v1-bug-onevec-gate-0.dimacs	44079	117720	277	302	986	175
c6-DD-s3-f1-e2-v1-bug-fourvec-gate-0.dimacs	170019	454050	1645	1705	5066	919
c7552-bug-gate-0.dimacs.seq.filtered	2640	7008	1	1	1	1
divider-problem.dimacs-11.filtered	215964	709377	9992	10090	15889	3978
divider-problem.dimacs12.filtered	229482	751921	10914	10844	17297	4667
divider-problem.dimacs1.filtered	215676	708801	10181	10308	16102	4295
divider-problem.dimacs2.filtered	228874	750705	10688	11180	17319	4392

is introduced. VNS-WS follows a simple principle that is based on systematic changes of neighborhood within the search. The set of neighborhoods proposed in this paper can easily be incorporated into any local search used for MAX-SAT. Starting the search from the largest neighborhood and moving systematically towards the smallest neighborhood is a better strategy to get a better heuristic. Thus, in order to get a comprehensive picture of the new algorithms performance, a set of large industrial instances is used. The results indicate that the proposed variable neighborhood strategy can enhance the convergence behavior of the Walksat algorithm. It appears clearly from the results that the performance of both WS and VNS-WS is fairly close with a slight edge in favor of VNS-WS for small problems. However, for larger problems, VNS-WS can find excellent solutions compared to

those of WS at a faster convergence rate. The difference lies between 30% and 93%. The larger the problem, the larger the size of the neighborhood needed, and consequently the more efficient the WS at different neighborhoods. The results have shown that VNS-WS consistently delivers better solutions than Optimax while requiring the least amount of time. When compared to CCLS, VNS-WS was capable of providing similar results in 74% of the studied cases; however, the time invested is several orders of magnitude slower than CCLS. The author aims at submitting this solver for the next MAX-SAT competition after having improved its performance. For the time being, further work is mainly conducted on improving the solution quality of VNS-WS. In particular, during the construction of the different neighborhoods, the random merging scheme does not exploit the information

TABLE 3: SAT2013 industrial benchmarks: comparison among VNS-WS, WS, W-w, and VW2.

Instances	Instance input		Unsatisfied clauses			VNS-WS
	Var	Cls	WS	W-w	VW2	
divider-problem.dimacs3.filtered	216900	711249	10054	10727	16249	3972
divider-problem.dimacs4.filtered	225340	743637	11030	11536	17240	4406
divider-problem.dimacs5.filtered	228874	750705	11194	11929	17459	5139
mot-comb1-red-gate-0.dimacs.seq.filtered	2159	5326	1	1	1	1
mrisc-mem2wire1.dimacs.filtered	168960	641598	823	998	8191	293
rsdecoder2.dimacs.filtered	415480	1632526	3799	4811	24221	461
rsdecoder-fsm1.dimacs.filtered	238290	936006	1703	1906	12689	179
rsdecoder-problem.dimacs-34.filtered	226040	728516	3016	3880	11408	386
s15850-bug-fourvec-gate-0.dimacs.seq.filtered	88544	206252	62	84	176	21
s15850-bug-onevec-gate-0.dimacs.seq.filtered	22136	51563	3	1	5	2
SM-AS-TOP-buggy1.dimacs.filtered	145900	694438	3791	4023	11888	1549
SM-MAIN-MEM-buggy1.dimacs.filtered	870975	3812147	45360	52045	97604	20196
SM-RX-TOP.dimacs.filtered	235456	934091	3645	3528	13487	2456
spi2.dimacs.filtered	124260	515813	1872	1985	9993	581

TABLE 4: Comparing VNS-WS with CCLS and Optimax.

Instance	CCLS		Optimax		VNS-WS	
	Quality	Time	Quality	Time	Quality	Time
brock400-1	255	0.38	340	0.08	255	15.02
brock400-2	252	0.84	310	0.08	252	24.02
brock400-3	238	1.27	278	0.08	238	9.05
brock400-4	249	0.73	374	0.08	249	4.05
brock800-1	205	0.95	273	0.09	205	1.05
brock800-2	207	0.97	270	0.09	207	3.08
brock800-3	203	0.47	315	0.07	203	10.01
brock800-4	200	0.32	310	0.13	200	4.07
hamming10-2	400	0.09	532	0.08	400	1.04
hamming10-4	319	0.42	341	0.09	319	64.03
hamming6-2	832	1.18	1100	0.13	843	55.01
hamming6-4	192	1.00	312	0.13	192	1.06
hamming8-2	441	0.12	551	0.13	441	1.05
s2v120c1600-8	240	2.55	289	0.09	251	118
s2v140c1200-3	155	2.97	195	0.08	165	107
s2v140c1300-4	164	2.34	208	0.10	181	74
s2v140c1400-3	193	3.38	239	0.12	206	121
s2v140c1500-5	205	2.111	248	0.13	218	145
s3v80c600-5	12	1.02	16	32.61	12	1.06
s3v80c700-6	18	1.56	26	4.28	18	1.06
s3v80c800-2	32	1.24	59	0.12	32	48.02
s3v80c900-1	35	1.76	73	0.12	35	25.02
s3v80c900-10	35	1.60	59	0.11	35	33.07
s3v80c900-2	37	1.71	64	0.08	37	1.06
t5pm3-7777.spn	78	1.46	120	0.10	78	23.05
t6pm3-8888.spn	136	3.30	222	0.09	144	36.02
t7pm3-9999.spn	209	4.36	343	0.11	233	87

structure of the problem. The author believes that VNS-WS might benefit from further research into merging strategies

used to construct the neighborhoods. A better strategy would be to construct the different neighborhoods based on merging variables by exploiting the number of clauses they have in common rather randomly.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper. In addition the author does not have a direct financial relation that might lead to a conflict of interests.

References

- [1] H. Hoos and T. Stützle, *Stochastic Local Search: Foundations and Applications*, Morgan Kaufmann, 2004.
- [2] H. Xu, R. A. Rutenbar, and K. Sakallah, “sub-SAT: a formulation for relaxed boolean satisfiability with applications in routing,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 6, pp. 814–820, 2003.
- [3] M. Vasquez and J. Hao, “A logic-constrained knapsack formulation and a tabu algorithm for the daily photograph scheduling of an earth observation satellite,” *Journal of Computational Optimization and Applications*, vol. 20, no. 2, pp. 137–157, 2001.
- [4] A. Biere, A. Cimatti, E. Clarke, and Y. Zhu, “Symbolic model checking without BDDs,” in *Tools and Algorithms for the Construction and Analysis of Systems*, pp. 193–207, 1999.
- [5] A. Smith, A. G. Veneris, M. F. Ali, and A. Viglas, “Fault diagnosis and logic debugging using Boolean satisfiability,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 10, pp. 1606–1621, 2005.
- [6] J. Rintanen, K. Heljanko, and I. Niemelä, “Planning as satisfiability: parallel plans and algorithms for plan search,” *Artificial Intelligence*, vol. 170, no. 12–13, pp. 1031–1080, 2006.
- [7] T. Sandholm, “An algorithm for optimal winner determination in combinatorial auctions,” in *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI ’99)*, pp. 542–547, Stockholm, Sweden, August 1999.

- [8] A. Biere, M. Helm, H. van Maaren, and T. Walsh, *Handbook of Satisfiability*, 2009.
- [9] A. Biere, "Picosat essentials," *Journal on Satisfiability, Boolean Modeling and Computation*, vol. 4, no. 2-4, pp. 75-97, 2008.
- [10] J. Marques-Silva, "Practical applications of boolean satisfiability," in *Proceedings of the 9th International Workshop on Discrete Event Systems (WODES '08)*, pp. 74-80, Gothenburg Sweden, May 2008.
- [11] R. J. Wallace and E. C. Freuder, "Comparative study of constraint satisfaction and Davisputnam algorithms for maximum satisfiability problems," in *Cliques, Coloring, and Satisfiability*, D. Johnson and M. Trick, Eds., pp. 587-615, 1996.
- [12] B. Selman, H. A. Kautz, and B. Cohen, "Noise strategies for improving local search," in *Proceedings of the 12th National Conference on Artificial Intelligence (AAAI '94)*, pp. 337-343, MIT Press, 1994.
- [13] M. Yagiura and T. Ibaraki, "Efficient 2 and 3-flip neighborhood search algorithms for the MAX SAT: experimental evaluation," *Journal of Heuristics*, vol. 7, no. 5, pp. 423-442, 2001.
- [14] M. Lozano and C. García-Martínez, "Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: overview and progress report," *Computers & Operations Research*, vol. 37, no. 3, pp. 481-497, 2010.
- [15] S. A. Cook, "The complexity of theorem-proving procedures," in *Proceedings of the 3rd ACM Symposium on Theory of Computing*, pp. 151-158, 1971.
- [16] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: overview and conceptual comparison," *ACM Computing Surveys*, vol. 35, no. 3, pp. 268-308, 2003.
- [17] B. Selman, H. Levesque, and D. Mitchell, "A new method for solving hard satisfiability problems," in *Proceedings of the 10th National Conference on Artificial Intelligence (AAAI '92)*, pp. 440-446, MIT Press, 1992.
- [18] H. H. Hoos, "An adaptive noise mechanism for walkSAT," in *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI '02) and 14th Innovative Applications of Artificial Intelligence Conference (IAAI '02)*, pp. 655-660, August 2002.
- [19] H. H. Hoos, "On the run-time behaviour of stochastic local search algorithms for SAT," in *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI '99)*, pp. 661-666, July 1999.
- [20] D. McAllester, B. Selman, and H. Kautz, "Evidence for invariants in local search," in *Proceedings of the 14th National Conference on Artificial Intelligence (AAAI 97)*, pp. 321-326, July 1997.
- [21] B. Cha and K. Iwama, "Performance test of local search algorithms using new types of random CNF formulas," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI '95)*, pp. 304-309, Morgan Kaufmann, Montreal, Canada, 1995.
- [22] J. Frank, "Learning short-term clause weights for GSAT," in *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI '97)*, pp. 384-389, Morgan Kaufmann Publishers, 1997.
- [23] W. Spears, "Adapting crossover in evolutionary algorithms," in *Proceedings of the 4th Annual Conference on Evolutionary Programming*, pp. 367-384, MIT Press, Boston, Mass, USA, 1995.
- [24] N. Buhmala, "A multilevel memetic algorithm for large SAT-encoded problems," *Evolutionary Computation*, vol. 20, no. 4, pp. 641-664, 2012.
- [25] F. Lardeux, F. Saubion, and J.-K. Hao, "GASAT: a genetic local search algorithm for the satisfiability problem," *Evolutionary Computation*, vol. 14, no. 2, pp. 223-253, 2006.
- [26] D. Boughaci, B. Benhamou, and H. Drias, "Scatter search and genetic algorithms for MAXSAT problems," *Journal of Mathematical Modelling and Algorithms*, vol. 7, no. 2, pp. 101-124, 2008.
- [27] D. S. Johnson and M. A. Trick, Eds., *Cliques, Coloring, and Satisfiability*, vol. 26 of *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, American Mathematical Society, 1996.
- [28] P. Mills and E. Tsang, "Guided local search for solving SAT and weighted MAX-SAT problems," *Journal of Automated Reasoning*, vol. 24, no. 1-2, pp. 205-223, 2000.
- [29] C. M. Li, W. Wei, and H. Zhang, "Combining adaptive noise and look-ahead in local search for SAT," in *Theory and Applications of Satisfiability Testing (SAT-07)*, J. Marques-Silva and K. A. Sakallah, Eds., vol. 4501 of *Lecture Notes in Computer Science*, pp. 121-133, Springer, Berlin, Germany, 2007.
- [30] D. J. Patterson and H. Kautz, "Auto-Walksat: a self-tuning implementation of Walksat," *Electronic Notes in Discrete Mathematics*, vol. 9, pp. 360-368, 2001.
- [31] O. C. Granmo and N. Buhmala, "Solving the satisfiability problem using finite learning automata," *International Journal of Computer Science and Applications*, vol. 4, no. 3, pp. 15-29, 2007.
- [32] A. R. KhudaBukhsh, L. Xu, H. H. Hoos, and K. Leyton-Brown, "SATenstein: automatically building local search SAT solvers from components," in *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI '09)*, pp. 517-524, Pasadena, Calif, USA, July 2009.
- [33] L. Xu, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "SATzilla: Portfolio-based algorithm selection for SAT," *Journal of Artificial Intelligence Research*, vol. 32, pp. 565-606, 2008.
- [34] Z. Lü and J. Hao, "Adaptive memory-based local search for MAX-SAT," *Applied Soft Computing*, vol. 12, no. 8, pp. 2063-2071, 2012.
- [35] S. Cai, C. Luo, and K. Su, "CCASat: solver description," in *Proceedings of the SAT Challenge 2012: Solver and Benchmark Descriptions*, pp. 13-14, 2012.
- [36] S. Cai and K. Su, "Configuration checking, with aspiration in local search for SAT," in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI '12)*, pp. 434-440, 2012.
- [37] S. Cai, K. Su, and A. Sattar, "Local search with edge weighting and configuration checking heuristics for minimum vertex cover," *Artificial Intelligence*, vol. 175, no. 9-10, pp. 1672-1696, 2011.
- [38] P. Hansen, B. Jaumard, N. Mladenovic, and A. D. Parreira, "Variable neighborhood search for maximum weighted satisfiability problem," Tech. Rep. G-2000-62, Les Cahiers du GERAD, Group for Research in Decision Analysis, 2000.
- [39] P. Hansen and N. Mladenovic, "An introduction to variable neighborhood search," in *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, S. Voss, S. Martello, I. H. Osman, and C. Roucairol, Eds., pp. 433-458, Kluwer, Boston, Mass, USA, 1999.
- [40] N. Mladenović and P. Hansen, "Variable neighborhood search," *Computers & Operations Research*, vol. 24, no. 11, pp. 1097-1100, 1997.
- [41] B. Hu and R. Raidl, "Variable neighborhood descent with self-adaptive neighborhood ordering," in *Proceedings of the 7th EU/Meeting on Adaptive, Self-Adaptive, and Multi-Level*

Metaheuristics, C. Cotta, A. J. Fernandez, and J. E. Gallardo, Eds., Malaga, Spain, 2006.

- [42] A. D. Tompkins and H. Hoos, "UBCSAT: an implementation and experimentation environment for SLS algorithms for SAT and MAX-SAT," 2004.
- [43] S. Prestwich, "Random walk with continuously smoothed variable weights," in *Theory and Applications of Satisfiability Testing*, vol. 3569 of *Lecture Notes in Computer Science*, pp. 203–215, 2005.
- [44] G. Audemard and L. Simon, in *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI'09)*, July 2009.

Research Article

Tuning of Kalman Filter Parameters via Genetic Algorithm for State-of-Charge Estimation in Battery Management System

T. O. Ting,¹ Ka Lok Man,^{2,3} Eng Gee Lim,¹ and Mark Leach¹

¹ Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, No. 111, Ren'ai Road, HET, SIP, Suzhou, Jiangsu 215123, China

² Department of Computer Science and Software Engineering, Xi'an Jiaotong-Liverpool University, No. 111, Ren'ai Road, HET, SIP, Suzhou, Jiangsu 215123, China

³ Department of Computer Science, Yonsei University, 50 Yonsei-ro, Seodaemun-gu, Seoul 120-749, Republic of Korea

Correspondence should be addressed to T. O. Ting; toting@xjtlu.edu.cn

Received 22 April 2014; Revised 3 June 2014; Accepted 4 June 2014; Published 5 August 2014

Academic Editor: Su Fong Chien

Copyright © 2014 T. O. Ting et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In this work, a state-space battery model is derived mathematically to estimate the state-of-charge (SoC) of a battery system. Subsequently, Kalman filter (KF) is applied to predict the dynamical behavior of the battery model. Results show an accurate prediction as the accumulated error, in terms of root-mean-square (RMS), is a very small value. From this work, it is found that different sets of \mathbf{Q} and \mathbf{R} values (KF's parameters) can be applied for better performance and hence lower RMS error. This is the motivation for the application of a metaheuristic algorithm. Hence, the result is further improved by applying a genetic algorithm (GA) to tune \mathbf{Q} and \mathbf{R} parameters of the KF. In an online application, a GA can be applied to obtain the optimal parameters of the KF before its application to a real plant (system). This simply means that the instantaneous response of the KF is not affected by the time consuming GA as this approach is applied only once to obtain the optimal parameters. The relevant workable MATLAB source codes are given in the appendix to ease future work and analysis in this area.

1. Introduction

Battery Management System (BMS) [1–3] comprises of mechanism that monitors and controls the normal operation of a battery system so as to ensure its safety while maintaining its State-of-Health (SoH). The BMS, in essence, measures the voltage, current, and temperature of each cell in a battery pack. These data are then analyzed by a management system that guarantees safe and reliable operations. A common example of an independent battery pack is portrayed in Figure 1. The battery is an essential component and should be accurately modeled in order to design an efficient management system [4]. Hence, a generic tool to describe the battery performance under a wide variety of conditions and applications is highly desirable [4]. As such, electrical modeling is necessary to provide such a tool that enables visualization of the processes occurring inside rechargeable batteries. These generic models are necessary for the development of battery management algorithms. These algorithms

control the operation and maintain the performance of battery packs. In short, the ultimate aim of BMS is to prolong battery life, while ensuring reliable operation alongside many applications, especially in photovoltaic systems [5–7].

Battery modeling is performed in many ways depending on the types of battery. In general, the resulting battery model is a mathematical model comprising numerous mathematical descriptions [8]. Ultimately, battery models aim to determine the state-of-charge (SoC) of the battery system. However, the complexity of the nonlinear electrochemical processes has been a great barrier to modeling this dynamic process accurately. The accurate determination of the SoC will enable utilization of the battery for optimal performance and long-life and prevent irreversible physical damage to the battery [9]. The solution to the SoC via neural networks [10] and fuzzy logic [11] has been difficult and costly for online implementation due to the large computation required, causing the battery pack controller to be heavily loaded. This may



FIGURE 1: Typical Lithium-ion battery packs for electric vehicle.

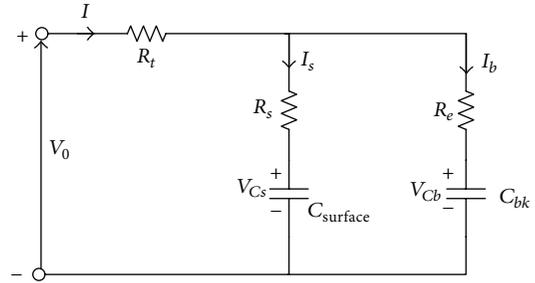


FIGURE 2: Schematic of RC battery model.

however be a good alternative in the near future as the computational power of processing chips increase alongside their declining cost.

The state-estimation process usually leads to some state variables in a dynamical system. The SoC is a measure of a battery’s available power and thus it is important to calculate this value accurately from BMS by the cell voltage, temperature, and polarization effect caused by the electrolyte concentration gradient during high rate charging/discharging cycle [12]. Recently, the battery state-of-charge (SoC) is one of the significant elements attracting significant attention [13, 14]. By definition, SoC is the ratio of remaining capacity to the nominal capacity of the battery. Here, the remaining capacity is the number of ampere-hours (Ah) that can be extracted at normal operating temperature. The mathematical expression for the SoC is given in [13, 14], which is

$$Z(t) = Z(0) + \int_0^t \frac{I_b(\tau)}{C_n} d\tau, \quad (1)$$

where t is time, $z(t)$ is battery SoC, in ampere-hours (Ah), I_b is current flowing through the battery (passing through C_{bk}), illustrated in Figure 2, and C_n is nominal battery capacity. For charging, $I_b > 0$ and for discharging, $I_b < 0$.

From this mathematical expression, it is noted that the SoC cannot be explicitly measured. In the literature, there is a myriad of methods dealing with predicting and estimating of SoC. The most popular of these methods are described in the following paragraphs.

At present, Coulomb-counting [15], also known as charge counting, or current integration is the most commonly used technique; it requires dynamic measurement of battery current. This is an open-loop method; however, it suffers from a reliance on the mathematical integration, and errors (noise, resolution, and rounding) are cumulative, which can lead to large SoC errors at the end of the integration process in (1). On the positive side, if an accurate current sensor is incorporated, the implementation will be much easier.

Another prominent SoC estimator is the well-known Kalman filter (KF), invented by Kalman in 1960. Although

his popular work was published almost 54 years ago in [16], it remains as an important citation source in the literature. Readers who are new to this method can refer to an excellent KF tutorial by Faragher in [17]. The KF method is a well-known technique for dynamic state estimation of systems such as target tracking, navigation, and also battery systems [18, 19]. The state-of-the-art method provides recursive solution to linear filtering for state observation and prediction problems. The key advantage of the KF is that it accurately estimates states affected by external disturbances such as noises governed by Gaussian distribution. On the contrary, the disadvantage of KF is that it requires highly complex mathematical calculations. This can be realized by a state-space model, as shown in previous work by the author in [20, 21]. The modeling is a heavy duty task and is also presented in this work to ease the understanding of readers. As such, there exist some possibilities of divergence due to an inaccurate model and complex calculation. In the case of a slow processor, the calculation results may be delayed and exceed the sampling interval, thereby result in an inaccurate tracking.

Various artificial intelligence (AI) methods, mainly the neural networks and fuzzy logic, are being applied in the estimation of battery’s SoC [10, 22]. Neural networks are computationally expensive, which can overload the BMS and thus this approach, though theoretically feasible, may not be suitable for online implementation that requires instantaneous feedback and action. Also, neural networks require huge datasets in the time-consuming training process. Other techniques for SoC, include the sliding mode observer, are reported in [12].

In this work, a mathematical derivation leading to a state-space model is presented. The basic schematic model is adopted from [18, 20]. A thorough analysis in the form of state variables with the application of the Kalman filter is presented. The rest of the paper is organized as follows. The mathematical model is derived in Section 2, resulting in a state-space model. Further, in Section 3, the KF is applied to estimate the SoC of a battery system. This is followed by the tuning of KF’s parameter by adopting a metaheuristic approach, namely, a genetic algorithm in Section 4. Relevant results are presented in Section 5, and finally the conclusions are derived in Section 6.

2. Battery Modeling

Many model-based state-estimations have been proposed in [18, 20, 21, 23]. In [18, 21], the well-known Kalman filter [16] had been applied successfully for both state observation and prediction of the SoC. Work in [24] utilized manufacturers' data in modeling the dynamic behavior of the battery. Several battery models exist from many works over the past few years. Each of these models varies in terms of its complexity and applications. In this work, a dynamical battery model is adopted, consisting of state variable equations, from [20, 21]. The schematic representation of this model is shown in Figure 2. In this model, there exists a bulk capacitor C_{bk} that acts as an energy storage component in the form of the charge, a capacitor that models the surface capacitance and diffusion effects within the cell $C_{surface}$, a terminal resistance R_t , a surface resistance R_s , and an end resistance R_e . The voltages across both capacitors are denoted as V_{Cb} and V_{Cs} , respectively.

2.1. Mathematical Derivations of Battery Model. In this derivation, we aim to form a state-space model consisting of the state variables V_{Cb} , V_{Cs} , and V_0 . State variables are mathematical descriptions of the "state" of a dynamic system. In practice, the state of a system is used to determine its future behavior. Models that consist of a paired first-order differential equations are in the state-variable form.

Following the voltages and currents illustrated in Figure 2, the terminal voltage V_0 can be expressed as

$$V_0 = IR_t + I_b R_e + V_{Cb}, \quad (2)$$

which is similar to

$$V_0 = IR_t + I_s R_s + V_{Cs}. \quad (3)$$

By (2) and (3), and following straightforward algebraic manipulation, V_0 can be written as

$$I_b R_e = I_s R_s + V_{Cs} - V_{Cb}. \quad (4)$$

From Kirchoff's current law, $I = I_b + I_s$,

$$I_s = I - I_b. \quad (5)$$

Thus, substituting (5) into (4) yields

$$I_b (R_e + R_s) = IR_s + V_{Cs} - V_{Cb}. \quad (6)$$

By assuming a slow varying C_{bk} , that is, $I_b = C_{bk} \dot{V}_{Cb}$ (from basic formula of $i = C(\partial V/\partial t)$), and then substituting it into (6), and after rearranging results in

$$\dot{V}_{Cb} = \frac{IR_s}{C_{bk}(R_e + R_s)} + \frac{V_{Cs}}{C_{bk}(R_e + R_s)} - \frac{V_{Cb}}{C_{bk}(R_e + R_s)}. \quad (7)$$

By applying a similar derivation, the rate of change of the surface capacitor voltage \dot{V}_{Cs} , derived also from (2) and (3), gives

$$\dot{V}_{Cs} = \frac{IR_e}{C_{surface}(R_e + R_s)} - \frac{V_{Cs}}{C_{surface}(R_e + R_s)} + \frac{V_{Cb}}{C_{surface}(R_e + R_s)}. \quad (8)$$

Further, by assuming $A = 1/C_{bk}(R_e + R_s)$ and $B = 1/C_{surface}(R_e + R_s)$, (7) and (8) can be written as

$$\begin{aligned} \dot{V}_{Cb} &= A \cdot I \cdot R_s + A \cdot V_{Cs} - A \cdot V_{Cb}, \\ \dot{V}_{Cs} &= B \cdot I \cdot R_e - B \cdot V_{Cs} + B \cdot V_{Cb}, \end{aligned} \quad (9)$$

respectively. Further, (9) and (10) can be combined to form a state variable relating voltages V_{Cs} and V_{Cb} and current flow I , which is

$$\begin{bmatrix} \dot{V}_{Cb} \\ \dot{V}_{Cs} \end{bmatrix} = \begin{bmatrix} -A & A \\ B & -B \end{bmatrix} \begin{bmatrix} V_{Cb} \\ V_{Cs} \end{bmatrix} + \begin{bmatrix} A \cdot R_s \\ B \cdot R_e \end{bmatrix} I. \quad (10)$$

Next, the output voltage is derived from (2) and (3). By adding both equations, we obtain

$$2V_0 = 2IR_t + I_b R_e + I_s R_s + V_{Cb} + V_{Cs}. \quad (11)$$

Then by substituting $I_b = R_s/(R_s + R_e) \cdot I$ and $I_s = R_e/(R_s + R_e) \cdot I$ into (11), it is further simplified as

$$V_0 = \frac{V_{Cb} + V_{Cs}}{2} + \left(R_t + \frac{R_e R_s}{R_e + R_s} \right) I. \quad (12)$$

Taking the time derivative of the output voltage and assuming $dI/dt \approx 0$ (this simply means that the change rate of terminal current can be ignored when implemented digitally), we obtain

$$\dot{V}_0 = \frac{\dot{V}_{Cb} + \dot{V}_{Cs}}{2}. \quad (13)$$

Substituting the formulae obtained in (9) and (10) into (13) results in

$$2\dot{V}_0 = (-A + B)V_{Cb} + (A - B)V_{Cs} + (AR_s + BR_e)I. \quad (14)$$

Then, solving for V_{Cs} from (12) we obtain

$$V_{Cs} = 2V_0 - 2 \left(R_t + \frac{R_e R_s}{R_e + R_s} \right) I - V_{Cb}, \quad (15)$$

and after substituting it into (14), it yields

$$\begin{aligned} \dot{V}_0 &= (-A + B)V_{Cb} + (A - B)V_0 \\ &+ [A(0.5R_s + R_t + D) + B(0.5R_e - R_t - D)]I. \end{aligned} \quad (16)$$

Finally, the complete state variable network is obtained by substituting (16) into (10), represented in matrix form as

$$\begin{aligned} \begin{bmatrix} \dot{V}_{Cb} \\ \dot{V}_{Cs} \\ \dot{V}_0 \end{bmatrix} &= \begin{bmatrix} -A & A & 0 \\ B & -B & 0 \\ (-A + B) & 0 & (A - B) \end{bmatrix} \cdot \begin{bmatrix} V_{Cb} \\ V_{Cs} \\ V_0 \end{bmatrix} \\ &+ \begin{bmatrix} A \cdot R_s \\ B \cdot R_e \\ A(0.5R_s - R_t - D) + B(0.5R_e + R_t + D) \end{bmatrix} I, \end{aligned} \quad (17)$$

TABLE 1: Parameters for cell model [18, 20].

C_{bk}	C_{surface}	R_e	R_s	R_t
88372.83 F	82.11 F	0.00375 Ω	0.00375 Ω	0.002745 Ω

whereby constants A , B , and D were derived previously but are hereby restated as

$$\begin{bmatrix} A \\ B \\ D \end{bmatrix} = \begin{bmatrix} \frac{1}{C_{\text{bk}}(R_e + R_s)} \\ \frac{1}{C_{\text{surface}}(R_e + R_s)} \\ \frac{R_e R_s}{R_e + R_s} \end{bmatrix}. \quad (18)$$

This completes the initial derivation of the battery model.

2.2. Numerical Example. By substituting all capacitor and resistor values from Table 1 into (18), the following values are obtained:

$$\begin{bmatrix} A \\ B \\ D \end{bmatrix} = \begin{bmatrix} 0.001508759347566 \\ 1.623837940973491 \\ 0.001875000000000 \end{bmatrix}. \quad (19)$$

By defining matrix \mathbf{A} ,

$$\mathbf{A} = \begin{bmatrix} -A & A & 0 \\ B & -B & 0 \\ (-A + B) & 0 & (A - B) \end{bmatrix}, \quad (20)$$

$$\mathbf{B} = \begin{bmatrix} A \cdot R_s \\ B \cdot R_e \\ A(0.5R_s - R_t - D) + B(0.5R_e + R_t + D) \end{bmatrix}. \quad (21)$$

Again by substituting the values from Table 1 to calculate A , B , and D , we obtain the value of \mathbf{A} as

$$\mathbf{A} = \begin{bmatrix} -1.51 \times 10^{-3} & 1.51 \times 10^{-3} & 0 \\ 1.6238 & -1.6238 & 0 \\ 1.6223 & 0 & -1.6223 \end{bmatrix} \quad (22)$$

and \mathbf{B} as

$$\mathbf{B} = \begin{bmatrix} 5.66 \times 10^{-6} \\ 6.08 \times 10^{-3} \\ 1.05 \times 10^{-2} \end{bmatrix}. \quad (23)$$

As such (17) can be rewritten as

$$\begin{bmatrix} \dot{V}_{Cb} \\ \dot{V}_{Cs} \\ \dot{V}_0 \end{bmatrix} = \mathbf{A} \cdot \begin{bmatrix} V_{Cb} \\ V_{Cs} \\ V_0 \end{bmatrix} + \mathbf{B} \cdot I, \quad (24)$$

or numerically as

$$\begin{bmatrix} \dot{V}_{Cb} \\ \dot{V}_{Cs} \\ \dot{V}_0 \end{bmatrix} = \begin{bmatrix} -0.0015 & 0.0015 & 0 \\ 1.6238 & -1.6238 & 0 \\ 1.6223 & 0 & -1.6223 \end{bmatrix} \cdot \begin{bmatrix} V_{Cb} \\ V_{Cs} \\ V_0 \end{bmatrix} + \begin{bmatrix} 5.66 \times 10^{-6} \\ 6.08 \times 10^{-3} \\ 1.05 \times 10^{-2} \end{bmatrix} \cdot I. \quad (25)$$

2.3. State-Space Modeling. Based on control theories, a lumped linear network can be written in the form

$$\begin{aligned} \dot{x}(t) &= \mathbf{A}x(t) + \mathbf{B}u(t), \\ y(t) &= \mathbf{C}x(t) + \mathbf{D}u(t), \end{aligned} \quad (26)$$

where in this work, the state variable $\dot{x}(t)$ is

$$\dot{x}(t) = \begin{bmatrix} \dot{V}_{Cb} \\ \dot{V}_{Cs} \\ \dot{V}_0 \end{bmatrix}. \quad (27)$$

Obviously,

$$x(t) = \begin{bmatrix} V_{Cb} \\ V_{Cs} \\ V_0 \end{bmatrix}, \quad (28)$$

with

$$u(t) = I. \quad (29)$$

Therefore, by restating the previous calculation values in (21) and (23), we should note that the values of \mathbf{A} , \mathbf{B} , \mathbf{C} , and \mathbf{D} are as follows:

$$\mathbf{A} = \begin{bmatrix} -1.51 \times 10^{-3} & 1.51 \times 10^{-3} & 0 \\ 1.6238 & -1.6238 & 0 \\ 1.6223 & 0 & -1.6223 \end{bmatrix}, \quad (30)$$

$$\mathbf{B} = \begin{bmatrix} 5.66 \times 10^{-6} \\ 6.08 \times 10^{-3} \\ 1.05 \times 10^{-2} \end{bmatrix}, \quad (31)$$

$$\mathbf{C} = [0 \ 0 \ 1], \quad (32)$$

$$\mathbf{D} = [0], \quad (33)$$

respectively. As such, with (32), the output $y(t)$ is in fact

$$y(t) = V_0. \quad (34)$$

This means that the output of the system is the open terminal voltage V_0 , as expected. Note that this is an important observation from this state-space modeling.

Further, the above state-space variables are transformed to a transfer function, $G(s)$. This is done by using *ss2tf* function in MATLAB, which after factorization yields

$$G(s) = \frac{0.01054s^2 + 0.0171s + 2.981 \times 10^{-5}}{s^3 + 3.248s^2 + 2.637s - 1.144 \times 10^{-18}}. \quad (35)$$

The plot of the unit step response for the gain in (35) is given in Figure 3. Basically, it shows that the open circuit terminal voltage V_0 in Figure 2 increases linearly during the charging operation in a very slow manner after transient behaviour for few seconds.

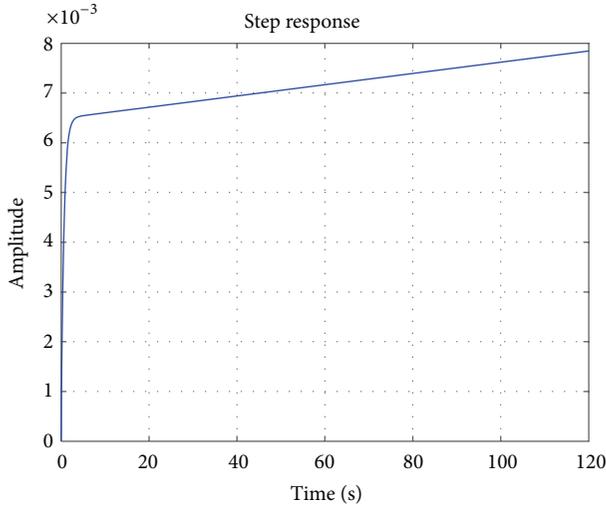


FIGURE 3: Output response of RC model due to constant input.

2.4. *Observability of the RC Battery Model.* In control theory, observability is the degree to which the internal states of a system can be predicted via its external outputs. As such, for an observable system, the behavior of the entire system can be predicted via the system’s outputs. On the other hand, if a system is not observable, the current values of some of its states cannot be estimated through the output signal. This means the controller does not know the states’ values. In theory, the observability of a system can be determined by constructing an observability matrix O_b .

$$O_b = \begin{bmatrix} C \\ CA \\ CA^2 \\ \vdots \\ CA^{n-1} \end{bmatrix}, \quad (36)$$

and a system is observable if the row rank of O_b is equal to n (this is also known as a full rank matrix). The ultimate rationale of such a test is that if n rows are linearly independent, then each of the n states is viewable through linear combinations of the output $y(t)$.

Further, substituting A and C values from (30) and (32) into (36) yields

$$O_b = \begin{bmatrix} 0 & 0 & 1 \\ 1.6223 & 0 & -1.6223 \\ -2.6344 & 0.0024 & 2.6320 \end{bmatrix}. \quad (37)$$

Clearly, in this case O_b is a full rank matrix, which concludes that this system is observable.

3. Kalman Filter for SoC Estimation

A continuous time-invariant linear system can be described in the state variable form as

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t), \end{aligned} \quad (38)$$

where $u(t)$ is the input vector, $x(t)$ is the state vector, $y(t)$ is the output vector, A is the time invariant dynamic matrix, B is the time invariant input matrix, and C is the time invariant measurement matrix.

If we assume that the applied input u is constant during each sampling interval, a discrete-time equivalent model of the system will now be

$$\begin{aligned} x(n+1) &= A_d \cdot x(n) + B_d \cdot u(n), \\ y(n+1) &= C_d \cdot x(n+1), \end{aligned} \quad (39)$$

where

$$A_d \approx I + A \cdot T_c, \quad B_d = B \cdot T_c, \quad C_d = C, \quad (40)$$

whereby I is the identity matrix and T_c is the sampling period. As for this system, two independent process noises are present which are additive Gaussian noise, w vector representing system disturbances and model inaccuracies and v vector representing the effects of measurement noise.

Both w and v have a mean value of zero and the following covariance matrices:

$$\begin{aligned} E[w \cdot w^T] &= Q, \\ E[v \cdot v^T] &= R, \end{aligned} \quad (41)$$

where E denotes the expectation (or mean) operator and superscript T means the transpose of the respective vectors. In usual case, Q and R are normally set to a constant before simulation; in our case both are set to one (see Section 5). Further, a genetic algorithm (GA) is applied in order to obtain a better set of Q and R values resulting in lower RMS error from the KF’s output.

By inclusion of these noises, the resulting system can now be described by

$$\begin{aligned} x(n+1) &= A_d \cdot x(n) + B_d \cdot u(n) + w, \\ z(n+1) &= C_d \cdot x(n+1) + v, \end{aligned} \quad (42)$$

which is illustrated in Figure 4,

3.1. *Property of Kalman Filter.* An important property of the KF is that it minimizes the sum-of-squared errors between the actual value x and estimated states \hat{x} , given as

$$f_{\min}(x) = E([x - \hat{x}] \cdot [x - \hat{x}]^T). \quad (43)$$

To understand the operations of the KF, the meaning of the notation $\hat{x}(m | n)$ is crucial. Simply stated, it means that the estimate of x at event m takes into account all discrete events up to event n . As such, (43) can include such information, now expanded as

$$f_{\min}(x) = E([x(n) - \hat{x}(n | n)] \cdot [x(n) - \hat{x}(n | n)]^T). \quad (44)$$

In recursive implementation of the KF, the current estimate $\hat{x}(n | n)$, together with the input $u(n)$ and measurement

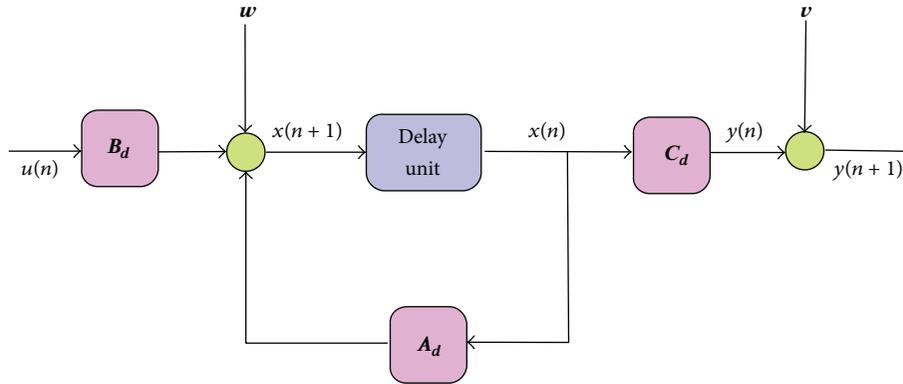


FIGURE 4: Discrete system model with noises w and v .

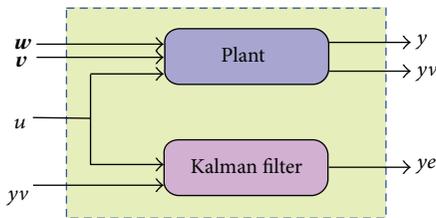


FIGURE 5: Parallel connection of plant and Kalman filter.

```

(1) BEGIN
(2) Initialize population  $P(X)$ 
(3) while  $Terminate = False$  do
(4)  $P(X) \leftarrow f(x)$ : Fitness evaluation,
(5) Selection,
(6) Crossover,
(7) Mutation,
(8) end while
(9) END
    
```

ALGORITHM 1: Pseudocode of GA.

signals z , is used for further estimating $\hat{x}(n+1 | n+1)$. This means that in this discrete system, the input for each sample step will be $u_1, u_2, u_3, \dots, u_{n+1}$ with respect to the output of $y_1, y_2, y_3, \dots, y_{n+1}$. The recursive KF algorithm is obtained with the predictor and corrector stages.

3.2. *KF Online Implementation.* For the case of a battery, it is well understood that only the terminal quantities can be measured (terminal voltage V_0 and current I). Assuming that battery parameters are time-invariant quantities, the recursive KF algorithm is applied. By applying (40) into (30)–(32), we obtain the following values of updated matrices, with $T_c = 1$:

$$\begin{aligned}
 \mathbf{A}_d &= \begin{bmatrix} 0.9984 & 1.51 \times 10^{-3} & 0 \\ 1.6238 & 0.6238 & 0 \\ 1.6223 & 0 & 0.6223 \end{bmatrix}, \\
 \mathbf{B}_d &= \begin{bmatrix} 5.66 \times 10^{-6} \\ 6.08 \times 10^{-3} \\ 1.05 \times 10^{-2} \end{bmatrix}, \\
 \mathbf{C}_d &= [0 \ 0 \ 1].
 \end{aligned}
 \tag{45}$$

Note that \mathbf{B}_d and \mathbf{C}_d remain similar to their previous values, as given in (31) and (32). By utilizing MATLAB’s control toolbox, the KF is placed in parallel to the state-space model, hereby represented by plant in Figure 5. The complete source code is given in the Appendix.

4. Genetic Algorithm

The genetic algorithm (GA), introduced by John Holland, is an approach based on biological evolution [25]. The algorithm is developed based on Charles Darwin’s theory of survival of the fittest. The GA has a very powerful encoding mechanism that enables the representation of a solution vector as either a real coded or binary string. Both encodings serve a different purpose in the context of different problem space. GAs are regarded as the global optimizer that often spot or locate the potential area or even accurately obtain the best solution, known as the global minimum [26–28]. Underneath this popular algorithm are the three operators that contribute to its success in performing optimization task.

- (i) *Selection.* In selection, offsprings with higher fitness have better chance for survival to the following generation in the evolutionary process. This basically is based on the theory of “survival of the fittest.”
- (ii) *Crossover.* The crossover increases and maintains the diversity of the entire population over the entire run. This is due to the fact that a population with higher diversity has the ability to explore a wider range of search space.
- (iii) *Mutation.* This enables chromosomes (potential solutions) to jump to a wider range than crossover. Again, mutation also increases the diversity of the entire population.

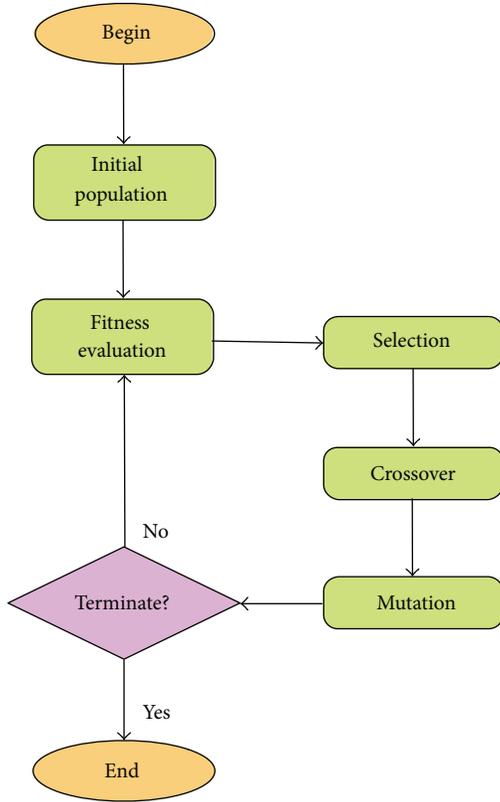


FIGURE 6: GA flowchart.

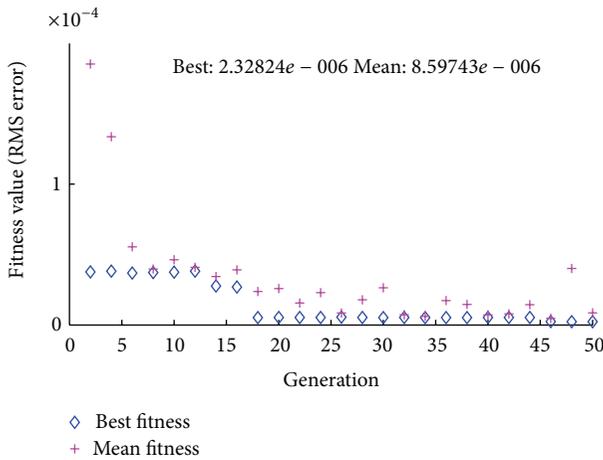


FIGURE 7: The convergence characteristic of GA.

The pseudocode of the GA is presented in Algorithm 1, and relevant figure depicting the algorithm flow is illustrated in Figure 6. To avoid extended discussions on GA, we include here a complete workable source code in the appendix. All parameter settings for the GA are available in the source code.

In the context of Kalman filter, GA is applied to tune the **Q** and **R** parameters, which was explained in detail in Section 3.

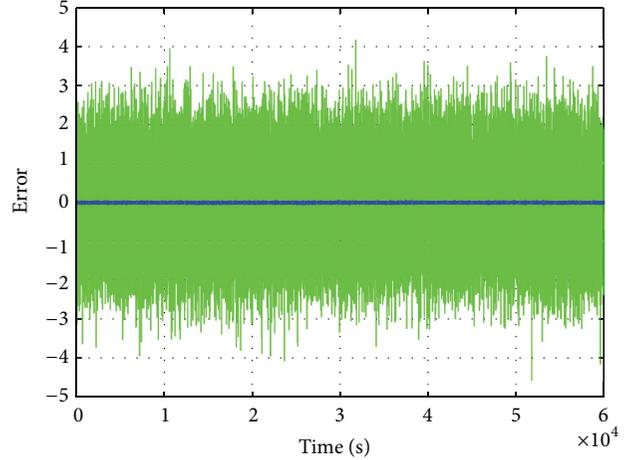


FIGURE 8: The voltage error recorded and depicting measured (green color) and estimated (blue color) errors.

TABLE 2: RMS error recorded during charging operation.

Output	Q, R	RMS error [V]
Measurement, $y - y_v$	1, 1	1.0013
Estimated (KF), $y - y_e$	1, 1	1.9185×10^{-4}
Estimated (KF), $y - y_e$ (After GA tuning)	0.012697316315642, 2.303940992875865	2.3282×10^{-6}

5. Results

The program, implemented in MATLAB, is given in the appendix to clarify the results obtained in this work. Take note that the **Q** and **R** mentioned in (41) are both set to a numerical value of one ($\mathbf{Q} = \mathbf{R} = 1$) in the first simulation. The results obtained are tabulated in Table 2. From these results, RMS of the estimated error, which is the error from KF, is far smaller in comparison to the measured error, with values of 1.0013 V and 1.92×10^{-4} V, respectively. This RMS error is further minimized by utilizing $\mathbf{Q} = 0.012697316315642$ and $\mathbf{R} = 2.303940992875865$, found using the GA approach. A graph depicting the convergence characteristic is shown in Figure 7.

The time plot of the estimated error from 0 s to 60000 s is shown in Figure 8, depicting a very small amplitude, in blue color ($\approx \pm 0.04$ V) along the timeline. This is observed through the zoomed display of the MATLAB graph. On the contrary, the measurement error, portrayed by green color noise in Figure 8, has an absolute magnitude of $\approx \pm 2$ V.

5.1. *Charging Behaviour.* The charging characteristic is illustrated in Figure 9 whereby the initial terminal voltage V_0 starts from 0 V and rises up to approximately 0.5 V within 60000 seconds (which is 100 minutes). Charging impulses of 1.53 A are applied in this case as shown in Figure 9. As expected, this is a time consuming process as in general case

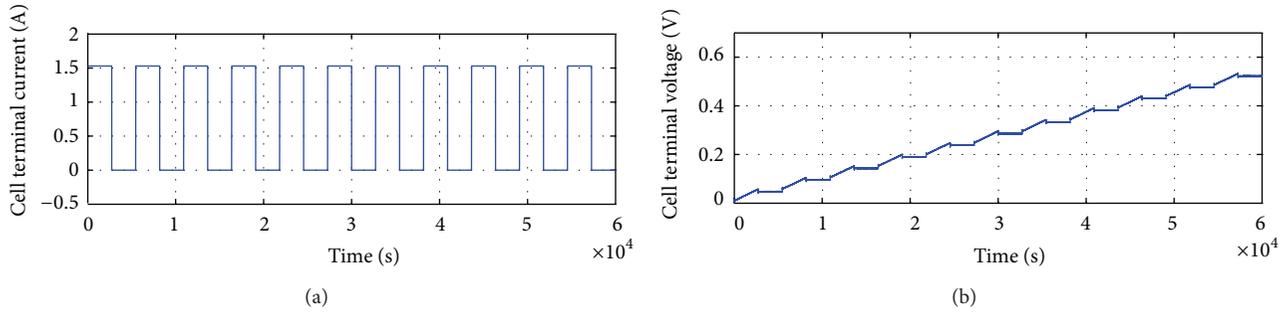


FIGURE 9: Response of RC battery model in terms of V_0 due to charging current $I = 1.53$ A pulses.

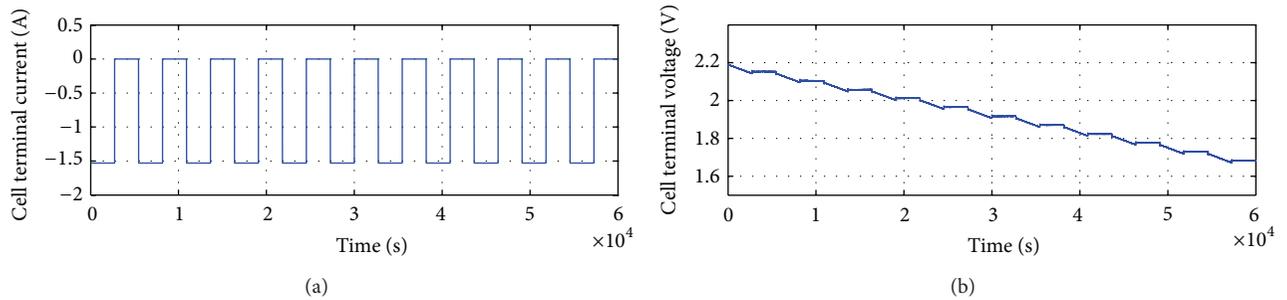


FIGURE 10: Response of RC battery model in terms of V_0 due to discharge current $I = -1.53$ A pulses.

it may take hours for a car battery (lead acid) to be completely charged.

5.2. *Discharging.* For the discharging process, the initial value of terminal voltage, $y_0 = V_0$, is set to 2.2 V in the MATLAB program. Again, in this case, impulses of 1.53 A are applied, but now in negative form. The dynamic behavior of the discharging characteristic is shown in Figure 10. From this figure, it is observed that the discharging process is similar to the charging process, but now with a linearly decreasing terminal voltage slope. The open terminal voltage V_0 drops from 2.2 V to 1.7 V in 60000 seconds (100 minutes); this is similar to the charging process as it takes 100 minutes to reach $V_0 = 0.5$ V from zero potential.

6. Conclusion

In this work, we successfully obtained the state variables of the RC model representing a battery in terms of mathematical derivations. The derivations lead to the conclusion that there exist three state variables relevant to a battery's state-space model. With this state-estimation model, a prominent technique known as the Kalman filter is applied in the aim of estimating state-of-charge for a Battery Management System. From numerical results, the KF is shown to be accurate in predicting the dynamic behavior of the system. This is shown by a very small RMS error of the estimation in comparison to its measurement. The estimated error is further reduced after incorporating the optimized values of \mathbf{Q} and \mathbf{R} through the

offline GA approach. As such, the efficacy of such an approach is, thus, validated.

Appendix

A. MATLAB Source Code, in a Script File

```
format long;
randn('seed',0) %Same sequence

%Value for resistors and capacitors
Csurface=82.11;
Cbk=88372.83;
Re=0.00375;
Rs=0.00375;
Rt=0.002745;

a=1/(Cbk*(Re+Rs));
b=1/(Csurface*(Re+Rs));
d=(Re*Rs)/(Re+Rs);

%State variable matrices
A=[-a a 0; b -b 0; (-a+b) 0 (a-b)];
B=[a*Re; b*Re;
    a*(0.5*Rs-Rt-d)+ b*(0.5*Re+Rt+d)];
C=[0 0 1];
D=[0];
```

```

%Transfer function
figure(1); %Figure 1
[num, den]=ss2tf(A,B,C,D,1);
G=tf(num,den)
step(G),grid;

% For Kalman filter:
% Identity matrix + diagonal element
A=[1-a a 0; b 1-b 0; (-a+b) 0 1+(a-b) ]
B=[a*Re; b*Re;
    a* (0.5*Rs-Rt-d)+ b*(0.5*Re+Rt+d)]
C=[0 0 1 ]

Tc=1;
A=A*Tc;
B=B*Tc;
C=C;

x=[2,2]
options = gaoptimset(...
'PopulationType', 'doubleVector',...
'PopInitRange', [0;1],...
'PopulationSize', 5,...
'EliteCount', 1,...
'CrossoverFraction', 0.8,...
'ParetoFraction', 0.35, ...
'MigrationDirection', 'both', ...
'MigrationInterval', 20,...
'MigrationFraction', 0.2,...
'Generations', 50,...
'TimeLimit', Inf,...
'FitnessLimit', -Inf,...
'StallGenLimit', 50,...
'StallTimeLimit', Inf,...
'TolFun', 0,...
'TolCon', 0,...%1e-6,...
'InitialPopulation', x,...
'InitialScores', [],...
'InitialPenalty', 10,...
'PenaltyFactor', 100,...
'CreationFcn', @gacreationuniform,...
'FitnessScalingFcn', @fitscalingrank,...
'SelectionFcn', @selectionroulette,...
'CrossoverFcn', @crossoverarithmetic,...
'MutationFcn', @mutationadaptfeasible,...
'DistanceMeasureFcn', @distancecrowding,...
'HybridFcn', [],...
'Display', 'final',...
'OutputFcns', [],...
'PlotFcns', @gaplotbestf, ...
'PlotInterval', 2, ...
'Vectorized', 'off', ...
'UseParallel', 'always');

Fmin =@(x)fobj(x, A, B, C);
lb=[0.001,0.001]; %Set lower bounds
v[X,fval,exitflag ]=...

ga(Fmin, 2, [],[],[],...
    [], lb, [],[], options );

% After simulation,
% repeat simulation all plot all graphs
if fval<0.0001
Q=X(1); R=X(2);

% Sample time=-1 for discrete model
Plant = ss(A,[B B],C,0,-1,...
'inputname',{'u' 'w'},...
'outputname', 'y');

[kalmf,L,P,M] = kalman(Plant,Q,R);
kalmf = kalmf(1,:);
Kalmf

a = A;
b = [B B 0*B];
c = [C;C];
d = [0 0 0;0 0 1];
P = ss(a,b,c,d,-1,...
'inputname',{'u' 'w' 'v'},...
'outputname',{'y' 'yv'});

sys = parallel(P,kalmf,1,1,[],[])

% Close loop for input #4 and output #2
SimModel = feedback(sys,1,4,2,1)

% Delete yv from I/O list
SimModel = SimModel([1 3],[1 2 3])
SimModel.inputname

%This part generates rectangular waveform
A=1.53; %Amplitude of negative pulses
t = 0:1:60000;
T=30000/5.5;
% I here represents current
I = (A/2)*square(2*pi*(1/T)*t);
I = I+(A/2);
figure(1); %Figure 2
subplot(211),plot(t,I),
axis([0 60000 -0.5 2.0]);
grid on; %charging
xlabel('Time (sec)');
ylabel('Cell terminal current (A)');

u = I';
n = length(t);
%randn('seed',0) %Same sequence
w = sqrt(Q)*randn(n,1);
v = sqrt(R)*randn(n,1);

```

```
[out,x] = lsim(SimModel,[w,v,u]);
y0=0; %This is initial terminal voltage
y = out(:,1)+y0; % true response
ye = out(:,2)+y0; % filtered response
yv = y + v; % measured response

figure(2);
plot(t,y, 'g--',t,ye, 'b-'), grid on;

figure(1);
subplot(212),plot(ye, 'b-'),
axis([0 60000 0 0.7]); grid on; %charging
xlabel('Time (s)'),
ylabel('Cell terminal voltage (V)')

%Kalman filter response
figure(3); %Figure 3
plot(t,y-yv, 'g-', t, y-ye, 'b-'), grid on;
xlabel('Time (s)'), ylabel('Error')

%Calculate Errors
MeasErr = y-yv; %Measurement error
MeasErrCov=...
sum(MeasErr.*MeasErr)/length(MeasErr);
EstErr = y-ye; %Estimated error
EstErrCov =...
sum(EstErr.*EstErr)/length(EstErr);

%Display onto screen
MeasErrCov %Measurement error
EstErrCov %Estimated error
End
```

B. MATLAB Source Code, in a Function File

```
function f = fobj(x, A, B, C)

Q=x(1);
R=x(2);

% Sample time=-1 for discrete model
Plant = ss(A,[B B],C,0,-1,...
'inputname',{'u' 'w'},...
'outputname','y');

[kalmf,L,P,M] = kalman(Plant,Q,R);
kalmf = kalmf(1,:);
Kalmf

a = A;
b = [B B 0*B];
c = [C;C];
d = [0 0 0;0 0 1];
```

```
P = ss(a,b,c,d,-1,...
'inputname',{'u' 'w' 'v'},...
'outputname',{'y' 'yv'});
sys = parallel(P,kalmf,1,1,[],[])

% Close loop around input #4 and output #2
SimModel = feedback(sys,1,4,2,1)

% Delete yv from I/O list
SimModel = SimModel([1 3],[1 2 3])
SimModel.inputname

%This part generates rectangular waveform
A=1.53; %Amplitude of negative pulses
t = 0:1:60000;
T=30000/5.5;
% I here represents current
I = (A/2)*square(2*pi*(1/T)*t);
I = I+(A/2);

u = I';
n = length(t);
%randn('seed',0) %Same sequence
w = sqrt(Q)*randn(n,1);
v = sqrt(R)*randn(n,1);

[out,x] = lsim(SimModel,[w,v,u]);
y0=0; %This is initial terminal voltage
y = out(:,1)+y0; % true response,
ye = out(:,2)+y0; % filtered response
yv = y + v; % measured response

%Calculate Errors
MeasErr = y-yv; %Measurement error
MeasErrCov= ...
sum(MeasErr.*MeasErr)/length(MeasErr);
EstErr = y-ye; %Estimated error
EstErrCov = ...
sum(EstErr.*EstErr)/length(EstErr);

f=EstErrCov;
end
```

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors would like to acknowledge the support from Xi'an Jiaotong-Liverpool University under RDF-13-01-13. The authors would also like to thank their colleague, Sanghyuk Lee, for the sharing of his expertise, contributing to the success of this project.

References

- [1] C. Chen, K. L. Man, T. O. Ting et al., "Design and realization of a smart battery management system," in *Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS '12)*, vol. 2, pp. 1173–1176, March 2012.
- [2] K. L. Man, K. Wan, T. O. Ting et al., "Towards a hybrid approach to SoC estimation for a smart Battery Management System (BMS) and battery supported Cyber-Physical Systems (CPS)," in *Proceedings of the 2nd Baltic Congress on Future Internet Communications (BCFIC '12)*, pp. 113–116, April 2012.
- [3] K. L. Man, C. Chen, T. O. Ting, T. Krilavičius, J. Chang, and S. H. Poon, *Artificial Intelligence Approach to Socestimation for Smart Battery Management System*, 2012.
- [4] P. H. L. Notten and D. Danilov, "From battery modeling to battery management," in *Proceedings of the 33rd International Telecommunications Energy Conference (INTELEC '11)*, pp. 1–8, October 2011.
- [5] D. Benchetrite, F. Mattera, M. Perrin et al., "Optimization of charge parameters for lead acid batteries used in photovoltaic systems," in *Proceedings of the 3rd World Conference on Photovoltaic Energy Conversion*, vol. 2, pp. 2408–2410, Osaka, Japan, May 2003.
- [6] T. O. Ting, K. L. Man, S. Guan, J. K. Seon, T. T. Jeong, and P. W. H. Wong, "Maximum power point tracking (MPPT) via weightless swarm algorithm (WSA) on cloudy days," in *Proceedings of the IEEE Asia Pacific Conference on Circuits and Systems (APCCAS '12)*, pp. 336–339, Kaohsiung, Taiwan, December 2012.
- [7] J. Ma, T. O. Ting, K. L. Man, N. Zhang, S. U. Guan, and P. W. H. Wong, "Parameter estimation of photovoltaic models via cuckoo search," *Journal of Applied Mathematics*, vol. 2013, Article ID 362619, 8 pages, 2013.
- [8] H. Gu, "Mathematical modeling in lead-acid battery development," in *Proceedings of the 6th Annual Battery Conference on Applications and Advances*, pp. 47–56, 1991.
- [9] C. S. Moo, K. S. Ng, Y. P. Chen, and Y. C. Hsieh, "State-of-charge estimation with open-circuit-voltage for lead-acid batteries," in *Proceedings of the 4th Power Conversion Conference (PCC '07)*, pp. 758–762, Nagoya, Japan, April 2007.
- [10] C. C. Chan, E. W. C. Lo, and S. Weixiang, "Available capacity computation model based on artificial neural network for lead-acid batteries in electric vehicles," *Journal of Power Sources*, vol. 87, no. 1, pp. 201–204, 2000.
- [11] P. Singh, C. Fennie Jr., and D. Reisner, "Fuzzy logic modelling of state-of-charge and available capacity of nickel/metal hydride batteries," *Journal of Power Sources*, vol. 136, no. 2, pp. 322–333, 2004.
- [12] I.-S. Kim, "Nonlinear state of charge estimator for hybrid electric vehicle battery," *IEEE Transactions on Power Electronics*, vol. 23, no. 4, pp. 2027–2034, 2008.
- [13] R. Restaino and W. Zamboni, "Comparing particle filter and extended Kalman Filter for battery state-of-charge estimation," in *Proceedings of the 38th Annual Conference on IEEE Industrial Electronics Society (IECON '12)*, pp. 4018–4023, Montreal, Canada, October 2012.
- [14] L. Y. Wang, M. P. Polis, G. G. Yin, W. Chen, Y. Fu, and C. C. Mi, "Battery cell identification and soc estimation using string terminal voltage measurements," *IEEE Transactions on Vehicular Technology*, vol. 61, no. 7, pp. 2925–2935, 2012.
- [15] H. Blanke, O. Bohlen, S. Buller et al., "Impedance measurements on lead-acid batteries for state-of-charge, state-of-health and cranking capability prognosis in electric and hybrid electric vehicles," *Journal of Power Sources*, vol. 144, no. 2, pp. 418–425, 2005.
- [16] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Fluids Engineering*, vol. 82, no. 1, pp. 35–45, 1960.
- [17] R. Faragher, "Understanding the basis of the kalman filter via a simple and intuitive derivation," *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 128–132, 2012.
- [18] B. S. Bhangu, P. Bentley, D. A. Stone, and C. M. Bingham, "Nonlinear observers for predicting state-of-charge and state-of-health of lead-acid batteries for hybrid-electric vehicles," *IEEE Transactions on Vehicular Technology*, vol. 54, no. 3, pp. 783–794, 2005.
- [19] G. L. Plett, "Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs—part 1. Background," *Journal of Power Sources*, vol. 134, no. 2, pp. 252–261, 2004.
- [20] T. O. Ting, K. L. Man, N. Zhang, C.-U. Lei, and C. Lu, "State-space battery modeling for smart battery management system," in *Proceedings of The International MultiConference of Engineers and Computer Scientists (IMECS '14)*, Lecture Notes in Engineering and Computer Science, pp. 866–869, Kowloon, Hong Kong, March 2014.
- [21] T. O. Ting, K. L. Man, C.-U. Lei, and C. Lu, "State-of-charge for battery management system via kalman filter," *IAENG Engineering Letters*, vol. 22, no. 2, 2014.
- [22] A. J. Salkind, C. Fennie, P. Singh, T. Atwater, and D. E. Reisner, "Determination of state-of-charge and state-of-health of batteries by fuzzy logic methodology," *Journal of Power Sources*, vol. 80, no. 1, pp. 293–300, 1999.
- [23] H. Saberi and F. R. Salmasi, "Genetic optimization of charging current for lead-acid batteries in hybrid electric vehicles," in *Proceedings of the International Conference on Electrical Machines and Systems (ICEMS '07)*, pp. 2028–2032, October 2007.
- [24] N. K. Medora and A. Kusko, "Dynamic battery modeling of lead-acid batteries using manufacturers' data," in *Proceedings of the 27th International Telecommunication Energy Conference (INTELEC '05)*, pp. 227–232, Berlin, Germany, September 2005.
- [25] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, vol. 412, Addison-Wesley Reading, Menlo Park, Calif, USA, 1989.
- [26] T. O. Ting, S. F. Chien, X.-S. Yang, and N. Ramli, "Resource allocation schemes in energy efficient OFDMA system via genetic algorithm," in *Proceedings of the 19th Asia-Pacific Conference on Communications (APCC '13)*, pp. 723–727, 2013.
- [27] S. F. Chien, T. O. Ting, X.-S. Yang, A. K. N. Ting, and D. W. Holtby, "Optimizing energy efficiency in multi-user ofdma systems with genetic algorithm," in *Proceedings of the International Conference on Advances in Computing, Communications and Informatics (ICACCI '13)*, pp. 1330–1334, Greater Noida, India, September 2013.
- [28] T. O. Ting, K. Wan, K. L. Man, and S. Lee, "Space exploration of multi-agent robotics via genetic algorithm," in *Network and Parallel Computing*, vol. 7513 of *Lecture Notes in Computer Science*, pp. 500–507, Springer, 2012.

Research Article

Improved Bat Algorithm Applied to Multilevel Image Thresholding

Adis Alihodzic¹ and Milan Tuba²

¹ Faculty of Mathematics, University of Sarajevo, 71000 Sarajevo, Bosnia And Herzegovina

² Faculty of Computer Science, Megatrend University Belgrade, 11070 Belgrade, Serbia

Correspondence should be addressed to Milan Tuba; tuba@ieee.org

Received 25 April 2014; Accepted 28 June 2014; Published 3 August 2014

Academic Editor: Xin-She Yang

Copyright © 2014 A. Alihodzic and M. Tuba. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Multilevel image thresholding is a very important image processing technique that is used as a basis for image segmentation and further higher level processing. However, the required computational time for exhaustive search grows exponentially with the number of desired thresholds. Swarm intelligence metaheuristics are well known as successful and efficient optimization methods for intractable problems. In this paper, we adjusted one of the latest swarm intelligence algorithms, the bat algorithm, for the multilevel image thresholding problem. The results of testing on standard benchmark images show that the bat algorithm is comparable with other state-of-the-art algorithms. We improved standard bat algorithm, where our modifications add some elements from the differential evolution and from the artificial bee colony algorithm. Our new proposed improved bat algorithm proved to be better than five other state-of-the-art algorithms, improving quality of results in all cases and significantly improving convergence speed.

1. Introduction

Image segmentation is process of subdivision of an image into homogeneous and disjoint sets sharing similar properties such as intensity, color, and contours. Homogeneous sets are introduced with respect to a certain criterion of homogeneity. Image segmentation usually represents the first step in image understanding and representation and the results obtained by segmentation are used for further high-level methods such as feature extraction, semantic interpretation, image recognition, and classification of objects. In general, image segmentation simplifies the process of dividing an image into regions that are used for further specific applications. Several practical applications cover character recognition [1], detection of video changes [2], medical imaging [3, 4], automatic target recognition [5], and so forth. Over the last few decades a lot of algorithms for image segmentation, either for gray level or color images, were presented in the literature. Good review of these algorithms can be found in [6]. In general, image segmentation algorithms can be grouped into thresholding, edge-based, region-grow, and clustering.

Image thresholding is one of the most widespread segmentation techniques that performs image segmentation based on the information contained in the global gray value of the image histogram. Thresholding is called bilevel thresholding in the case that an image is separated into two classes, one including those pixels with gray levels above a specified threshold and the other including the rest. Unlike bilevel thresholding, multilevel thresholding performs subdivision of an image into several classes. In this case the pixels belonging to the same class take gray levels from the intervals defined by successive thresholds. Multiple pixels belonging to the same class are not always homogeneous and may be represented by different feature values. Selection or computing of the multilevel thresholds is crucial in image segmentation since proper segmentation depends on adequately computed thresholds.

There are many different methods for computing the thresholds for an image such as maximizing the gray level variance [7], entropy [8], similarity [9], and measure of fuzziness [10]. In general, thresholding methods can be divided

into parametric and nonparametric methods. Using parametric methods, such as a novel image thresholding method based on Parzen window estimate [11], nonsupervised image segmentation based on multiobjective optimization [12], a multilevel thresholding approach using a hybrid optimal estimation algorithm [13], and optimal multithresholding using a hybrid optimization approach [14], may involve the solution of nonlinear equations which increases of the computational complexity. Therefore, the nonparametric methods [15] are introduced for finding the thresholds by optimizing some discriminating criteria. Among the mentioned different thresholding criteria, the entropy is the most popular optimization method. Using the entropy of the histogram, Pun was the first to introduce a new method for gray level image thresholding [8]. Later, this method was corrected and improved by Kapur et al. [16], since Kapur found some artifacts in Pun's method. Sahoo used Shannon's concept of entropy, considering two probability distributions for background and foreground objects. He has proposed a thresholding technique based on Renyi's entropy [17]. Information about the gray value of each pixel and the average value of its immediate neighborhood are obtained by two-dimensional entropy which is calculated by two-dimensional histogram.

Another important group of methods based on discriminant analysis is the clustering-based methods [18]. In these methods, gray values are clustered into several classes, so that there is a similarity of gray values within the class and dissimilarity between classes. To perform the separation of classes, Otsu has developed a thresholding method for computing the optimal thresholds by maximizing the between-class variance using an exhaustive search [7]. It has been shown that this method gives acceptable results when the number of pixels in each class is close to each other. For bilevel image thresholding, the above-mentioned methods are effective. However, for the optimal multilevel thresholding, the existing conventional methods are being hindered by an exhaustive search when the number of thresholds is increased. To overcome this problem, powerful metaheuristics are used to search for the optimal thresholds in order to achieve a fast convergence and reduce the computational time.

Metaheuristics are optimization methods that orchestrate an interaction between local improvement procedures and higher level strategies to create a process capable of escaping from local optima and performing a robust search of a solution space [19, 20]. Several metaheuristic algorithms derived from the behavior of biological and physical systems in the nature have been proposed as powerful methods for searching the multilevel image thresholds. Since magic algorithm that works for all problems does not exist [21], different approaches have been developed for different classes of problems such as combinatorial or continuous, with additions for constrained optimization problems [22]. Original versions of metaheuristic algorithms are often modified or hybridized in order to improve performance on some classes of problems. The most popular nature-inspired algorithms

for optimization, with improvements, adjustments, and hybridizations, include particle swarm optimization (PSO) [23], differential evolution (DE) [24], firefly algorithm (FA) [25, 26], cuckoo search (CS) [27–29], ant colony optimization [30–33], artificial bee colony algorithm [34–38], bat algorithm (BA) [39, 40], and human seeker optimization (HSO) [41–43].

DE algorithm has been adapted for searching the optimal multilevel thresholds [44]. PSO algorithm modified by Yin to search for the thresholds can be found in [45]. Akay presented a comprehensive comparative study of the ABC and PSO algorithms for finding multilevel thresholds using Kapur's and Otsu's criteria [46]. Maitra and Chatterjee proposed an improved variant of PSO algorithm for the task of image multilevel thresholding [47]. The results showed that the ABC algorithm with both the between-class variance and the entropy criterion can be efficiently used in multilevel thresholding. Hammouche focused on solving the image thresholding problem by combining between-class variance criterion with metaheuristic techniques such as GA, PSO, DE, ACO, SA, and TS [48].

In this paper, we adapted the bat algorithm for multilevel image thresholding. Bat algorithm is simple to implement and produces good results. However, based on our experiments, it is powerful in intensification, but at times it may get trapped into local optima when it is applied to some difficult problems. Therefore, we propose an improved version of bat algorithm adopted to search for multilevel thresholds using Kapur and Otsu criteria. Our proposed modification merges three approaches to produce a new improved bat-inspired (IBA) algorithm according to the principle of bat algorithm, differential evolution, and some scout technique taken from the ABC algorithm. We compared our proposed algorithm with state-of-the-art algorithms from [49]. The experimental results show that the proposed IBA algorithm always gives better results compared to PSO, DE, CS, FA, and BA algorithms, considering both accuracy and, especially, convergence speed.

The remainder of the paper is organized as follows. Section 2 describes the multilevel thresholding problem and presents Kapur's and Otsu's objective functions. Section 3 and Section 4 describe the original BA and IBA algorithms adopted to search for the optimal multilevel thresholds, respectively. Section 5 shows the experimental and comparative results of applying PSO, DE, CS, FA, BA, and IBA to multilevel segmentation to standard benchmark images. Finally, our conclusions are discussed in Section 6.

2. Multilevel Image Thresholding

Thresholding technique performs image segmentation based on the information contained in the image histogram. If we consider a gray-scale input image I as a set of pixels A , multilevel thresholding can be defined as a method of

dividing the set A into $n + 1$ disjoint subsets (A_0, A_1, \dots, A_n) by some numbers $(\alpha_0, \alpha_1, \dots, \alpha_{n-1})$ such that

$$\begin{aligned} A_0 &= \{x : 0 \leq f(x) < \alpha_0\}, \\ A_1 &= \{x : \alpha_0 \leq f(x) < \alpha_1\}, \\ &\vdots \\ A_n &= \{x : \alpha_{n-1} \leq f(x) \leq L - 1\}, \end{aligned} \tag{1}$$

where $x = (x_1, x_2)$ is a pixel defined by coordinates x_1 and x_2 in the Cartesian coordinate system, $f(x)$ presents a gray level value of pixel x , and the $f(x)$ takes values in the range $[0, 255]$. The aim of multilevel thresholding is to compute the optimal threshold values $(\alpha_0, \alpha_1, \dots, \alpha_{n-1})$. The sets (A_0, A_1, \dots, A_n) may represent different regions of the object. It is clear that $A_i \cap A_j = \emptyset$, and their union presents the whole input image I .

Optimal threshold selection for bilevel thresholding is not computationally expensive, while for multilevel thresholding, computing more than few optimal threshold values is an expensive and time consuming operation. The optimal threshold values can be determined by optimizing some criterion functions defined from the histogram of image. In this paper, we use two popular threshold criteria: Kapur's entropy criterion and Otsu's between-class variance criterion.

2.1. Kapur's Thresholding Method. Entropy is a measure of uncertainty proposed by Shannon [50], later widely used [51]. Let x be a discrete random variable taking values x_i with probabilities $p_i, i = 1, 2, \dots, n$, respectively. Then its entropy is defined by

$$H(x) = -\sum_{i=1}^n p_i \ln(p_i). \tag{2}$$

The Kapur's method [16] based on the entropy is used to perform multilevel thresholding. For this method, the threshold criteria can be formulated as follows. Assume that an image I contains n pixels with gray levels belonging to the set $\{0, 1, \dots, L - 1\}$. Let $h(i)$ present the number of pixels at gray level i , and $p_i = h(i)/n$ is the probability of occurrences of gray level i in the image I . The subdivision of an image into $k + 1$ classes can be considered as a k -dimensional optimization problem for the calculation of k optimal thresholds $(t_0, t_1, \dots, t_{k-1})$. The optimal thresholds are obtained by maximizing the objective function:

$$f(t_0, t_1, \dots, t_{k-1}) = \sum_{i=0}^k H_i, \tag{3}$$

where the entropies H_i are defined by

$$\begin{aligned} H_0 &= -\sum_{i=0}^{t_0-1} \frac{p_i}{w_0} \ln \frac{p_i}{w_0}, & w_0 &= \sum_{i=0}^{t_0-1} p_i, \\ H_1 &= -\sum_{i=t_0}^{t_1-1} \frac{p_i}{w_1} \ln \frac{p_i}{w_1}, & w_1 &= \sum_{i=t_0}^{t_1-1} p_i, \\ &\vdots \\ H_k &= -\sum_{i=t_{k-1}}^{L-1} \frac{p_i}{w_k} \ln \frac{p_i}{w_k}, & w_k &= \sum_{i=t_{k-1}}^{L-1} p_i. \end{aligned} \tag{4}$$

2.2. Otsu's Thresholding Method. Otsu's method [7] based on the maximization of the between-class variance is one of the most popular methods proposed for image thresholding. The algorithm for this method can be described as follows. Assume that an image I can be represented by L gray levels. The probabilities of pixels at level i are denoted by p_i so $p_i \geq 0$ and $p_0 + p_1 + \dots + p_{L-1} = 1$. Cumulative probabilities for classes $A_i, i = 0, 1, \dots, k$, can be defined as

$$w_0 = \sum_{i=0}^{t_0-1} p_i, \quad w_1 = \sum_{i=t_0}^{t_1-1} p_i, \dots, w_k = \sum_{i=t_{k-1}}^{L-1} p_i, \tag{5}$$

where t_j are the thresholds separating these classes. For $k + 1$ classes $A_i, (i = 0, 1, \dots, k)$, the goal is to maximize the objective function:

$$f(t_0, t_1, \dots, t_{k-1}) = \sum_{i=0}^k \sigma_i, \tag{6}$$

where the sigma functions are defined by

$$\begin{aligned} \sigma_0 &= w_0 \left(\sum_{i=0}^{t_0-1} \frac{i p_i}{w_0} - \sum_{i=0}^{t_0-1} i p_i \right)^2, \\ \sigma_1 &= w_1 \left(\sum_{i=t_0}^{t_1-1} \frac{i p_i}{w_1} - \sum_{i=t_0}^{t_1-1} i p_i \right)^2, \\ &\vdots \\ \sigma_k &= w_k \left(\sum_{i=t_{k-1}}^{L-1} \frac{i p_i}{w_k} - \sum_{i=t_{k-1}}^{L-1} i p_i \right)^2. \end{aligned} \tag{7}$$

3. Bat Algorithm Adapted for Multilevel Image Thresholding

Bat algorithm is a recent metaheuristic introduced by Yang [39], based on so-called echolocation of the bats. In this algorithm, bats detect prey and avoid the obstacles by using the echolocation. Bat algorithm was successfully applied to a number of very different problems like large-scale optimization problems [52], global engineering optimization

[53], fuzzy clustering [54], parameter estimation in dynamic biological systems [55], multiobjective optimization [56], image matching [57], economic load and emission dispatch problems [58], data mining [59], scheduling problems [60], neural networks [61], and phishing website detection [62].

In the bat algorithm, bats navigate by using time delay from emission to the reflection. The pulse rate can be simply determined in the range from 0 to 1, where 0 means that there is no emission and 1 means that the bat's emitting is at maximum. Apart from the control parameters, such as the population size and maximum iteration number which are common control parameters for all nature inspired algorithms, the BA has few important parameters such as frequency tuning parameter similar to the key feature used in the PSO and HS, parameter for automatically zooming into a region where the promising solutions have been found, and the control parameter for automatically switching from exploration to exploitation. This gives advantage to the BA over other metaheuristic algorithms in the literature.

In order to implement the bat algorithm, the following three idealized rules are used [39]:

- (i) all bats use echolocation to sense distance, and they also "know" the surroundings in some magical way;
- (ii) bats fly randomly with velocity v_i at position x_i with a fixed frequency f_{\min} , varying wavelength λ , and loudness A_0 to search for prey. They can automatically adjust the wavelength of their emitted pulses and adjust the rate of pulse emission r from $[0, 1]$, depending on the proximity of their target;
- (iii) although the loudness can vary in many ways, it is assumed that the loudness varies from a positive large value A_0 to a minimum constant value A_{\min} .

The proposed bat algorithm tries to select k threshold values which maximize the fitness functions which are described by (3) and (6), respectively. The details of the developed BA approach for multilevel image thresholding are given as follows.

Step 1 (generate initial population of solutions). The bat algorithm generates a randomly distributed initial population of N solutions (bats) ($i = 1, 2, \dots, N$), where each solution has k dimensions. All solutions can be presented by matrix X :

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & x_{1,3} & \cdots & x_{1,k} \\ x_{2,1} & x_{2,2} & x_{2,3} & \cdots & x_{2,k} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{N,1} & x_{N,2} & x_{N,3} & \cdots & x_{N,k} \end{bmatrix}, \quad (8)$$

where $x_{i,j}$ is the j th component value that is restricted to $\{0, \dots, L-1\}$ and $x_{i,j} < x_{i,j+1}$ for all j . The fitness values for all solutions are evaluated and variable *cycle* is set to one. The bat

algorithm detects the most successful solution as x_{best} before starting iterative search process.

Step 2 (calculation of new solutions). Calculation of a new solution x_i^t is performed by moving virtual bats x_i^{t-1} according to equation

$$x_i^t = x_i^{t-1} + v_i^t, \quad (9)$$

where v_i^t denotes the bat velocity of movement, and it is calculated by formula

$$v_i^t = v_i^{t-1} + (x_i^t - x_{\text{best}}) * f_i. \quad (10)$$

In (10), f_i denotes the frequency and x_{best} denotes the current global best solution. The frequency f_i can be calculated as

$$f_i^t = f_{\min} + (f_{\max} - f_{\min}) * \beta, \quad (11)$$

where β is a random vector generated by a uniform distribution belonging to the closed interval $[0, 1]$. For min and max frequency, the recommended values $f_{\min} = 0$ and $f_{\max} = 2$ are used. In this computation step, the bat algorithm controls the boundary conditions of the calculated new solution x_i^t . In the case that the value of a variable overflows the allowed search space limits, then the value of the related variable is updated with the value of the closer limit value to the related variable.

Step 3 (improving the current best solution). For each solution x_i^t apply the next operator which is defined by

$$x_{\text{new}} = \begin{cases} x_{\text{best}} + \epsilon A_t, & \text{if } \text{rand}_1 > r_i^t, \\ x_i^t, & \text{otherwise,} \end{cases} \quad (12)$$

where rand_1 is a uniform random number in range $[0, 1]$, ϵ is a scaling factor drawn from uniform distribution in the range $[-1, 1]$, $A_t = \langle A_i^t \rangle$ is the average loudness of all bats at the computation step t , and r_i^t is the pulse rate function. The pulse rate function is defined by

$$r_i^t = r_i^0 (1 - e^{-\beta t}), \quad (13)$$

where β is a constant and r_i^0 are initial pulse rates in the range $[0, 1]$. It can be seen from (12) that this function controls the intensive local search depending on the value of uniform variable rand_1 and the rate r_i^t . Also, at this step, the BA controls the boundary conditions at each iteration.

Step 4 (acceptation of a new solution by flying randomly). In this step, the solution x_{new} obtained in Step 3 is accepted as a new solution and $f(x_{\text{new}})$ as a new objective function value by using

$$(x_i^t, \text{fit}(x_i^t)) = \begin{cases} (x_{\text{new}}, f(x_{\text{new}})), & \\ \text{if } (\text{rand}_2 < A_i^t \text{ and } f(x_{\text{new}}) > f(x_i^{t-1})), & \\ (x_i^{t-1}, f(x_i^{t-1})), & \text{otherwise,} \end{cases} \quad (14)$$

where rand_2 is a uniform random number in range $[0, 1]$ and A_i^t is the loudness function defined by

$$A_i^t = \alpha A_i^{t-1}, \quad (15)$$

where α is a constant and plays a similar role as the cooling factor of a cooling schedule in the simulated annealing. Therefore, if the solution x_{new} has the higher objective function value compared to the old solution x_i^{t-1} and the loudness A_i^t is more than rand_2 , then the new solution is accepted, the old fitness value is updated, and functions defined by (13) and (15) are updated, too. Otherwise, the new solution x_{new} is abandoned, and the old best solution is kept.

Step 5 (memorize the best current solution). Record the best solution so far (x_{best}), that is, the solution with the highest objective function value.

Step 6 (check the stopping criteria). If the termination criterion is met or the variable *cycle* is equal to the maximum number of iterations, then the algorithm is finished. Otherwise, increase the variable *cycle* by one and go to Step 2.

4. Our Proposed Improved Bat Algorithm: IBA

As described in the previous section, we selected the BA for multilevel image thresholding. BA is simple to implement and it produces good results when the number of thresholds is small. However, based on our experiments, the BA algorithm often fails when the number of thresholds is larger, especially for the Kapur's objective function. Therefore, an adjustment of the bat algorithm was required. In this paper, the improved hybridized bat algorithm (IBA) is proposed to overcome the mentioned drawback of the pure bat algorithm. It combines two different solution search equations of the bat algorithm and DE algorithm [24]. The IBA algorithm includes differential operators mutation and crossover from DE algorithm, with the aim of speeding up convergence and to achieve a good balance between intensification and diversification. Mutation and crossover operators are used to improve the original BA generation of a new solution for each bat so that the IBA can more efficiently explore and exploit the new search space and avoid being trapped into local optima.

In the pure BA, exploration and exploitation are controlled by pulse rate function (13). Analyzing this function we noticed that the algorithm loses exploration capability as iterations progress. The form of this function makes switching from the exploration to exploitation and vice versa possible. In this way, the exploration capability of BA can be modified by inserting differential operators for crossover and mutation [24] instead of (12) and for the exploitation capability (12) continues to be used for a good intensification. Therefore, a good balance is established between intensification and diversification.

Although the above modification can improve many solutions, some solutions will still remain stuck in some local optimum. In order to fix this lack of the former modification, we introduced the second modification which is inspired by launch of the scouts in the scout phase of the ABC algorithm. When some solution gets trapped in a local optimum after a certain number of iterations, it will eventually exceed the predetermined number of allowed trials called "limit." When a solution exceeds the "limit" trials unchanged, it is redirected to search new space by using the random walk.

In the proposed IBA algorithm, bats form a population of threshold values. The threshold values produced by the bat i are noted as $x_i = (x_{i,1}, x_{i,2}, \dots, x_{i,k})$, $i = 1, \dots, N$. All bats perform searching in the solution search space with the aim to optimize the objective functions described by (3) or (6). The details of the proposed IBA approach for multilevel thresholding are given as follows.

Step 1 (generate the initial population of solutions). The IBA begins by randomly generating population with k dimensions as in the case of the proposed BA approach for multilevel thresholding. Each threshold value $x_{i,j}$ ($i = 1, \dots, n$; $j = 1, \dots, k$) of the matrix X generated by the bat i is restricted to set $\{0, 1, \dots, L - 1\}$ and for all j holds $x_{i,j} < x_{i,j+1}$. Also, at this step initialization is done for the parameter *limit* which presents the number of allowed attempts to improve a bat, the initial loudness A_i and pulse rate r_i^0 , as well as the initial values of the parameters in the DE algorithm such as the differential weight F and crossover probability C_r . After generation of the initial population, the fitness value for each solution x_i is evaluated. Then the IBA algorithm detects the most successful solution as x_{best} , before starting iterative search process. After that it sets the variable *cycle* to one.

Step 2 (calculate the new population). Calculation of a new threshold x_i^t is performed by moving virtual bats x_i^{t-1} according to (9). The velocity v_i^t and frequency f_i are calculated by (10) and (11), respectively. At this computation step, the IBA controls the boundary conditions of the calculated new solution x_i^t . In the case that the value of the x_i^t is less than 0 or is more than $L - 1$, then the value of the x_i^t is updated with the value of the closer limit value to the variable x_i^t .

Step 3 (improving the current best solution by differential operators). For each solution x_i^t apply the next operator which is defined by

$$x_{\text{new}} = \begin{cases} x_{\text{dif}}^t & \text{if } \text{rand}_1 > r_i^t, \\ x_{\text{loc}}^t & \text{otherwise,} \end{cases} \quad (16)$$

where rand_1 is randomization term in the range $[0, 1]$, r_i^t is the pulse rate function defined by (13), x_{dif}^t is the differential operator for mutation and crossover, and x_{loc}^t is the operator based on the local search in the BA. The differential mutation and crossover operations are performed by

$$x_{\text{dif},j}^t = \begin{cases} x_{c,j}^t + F(x_{a,j}^t - x_{b,j}^t), & \text{if } (\text{rand}_2 < C_r \text{ or } j = j_r), \\ x_{i,j}^t, & \text{otherwise,} \end{cases} \quad (17)$$

where x_a , x_b , and x_c are three randomly chosen different vectors in the range $[0, N - 1]$ at the cycle t , F is the differential weight that scales the rate of modification, C_r is the crossover probability in the interval $[0, 1]$, j_r is randomly selected in the range $[0, k]$, and rand_2 is a uniform variable in the range $[0, 1]$. Inside the implementation of the differential operator x_{dif} , the boundary conditions for all j ($j = 1, \dots, k$) are controlled. As an important improvement of the proposed method, the binomial "DE/rand/1/bin" scheme is used in

order to increase the diversity of the bats and achieve both the precision and search efficiency. The local search is performed by

$$x_{loc,j}^t = \begin{cases} x_{lbest,j}^t, & \text{if } (f(x_{lbest,j}^t) > f(x_{i,j}^t)), \\ x_{i,j}^t, & \text{otherwise,} \end{cases} \quad (18)$$

where $x_{lbest,j}^t$ is defined by

$$x_{lbest,j}^t = x_{best,j}^{t-1} + \epsilon A_{i,j}^{t-1}. \quad (19)$$

As in the ordinary BA, parameters ϵ and $A_{i,j}$ denote the scaling factor and the loudness function, respectively. Also, inside the local search operator x_{loc} , the boundary conditions for all j ($j = 1, \dots, k$) are checked. In our proposed approach, we found that it is beneficial to replace (13) by $r_i^t = r_i^0(1 - \beta^t)$. It will be shown in experimental study that the best results are obtained for initial pulse rates $r_i^0 = 0.5$, initial loudness $A_0 = 0.95$, and $\beta = 0.9$.

Step 4 (acceptation of a new solution by flying randomly). In this step, the solution x_{new} obtained in Step 3 is accepted as a new solution and $f(x_{new})$ as a new objective function value by using

$$(x_i^t, \text{fit}(x_i^t)) = \begin{cases} (x_{new}^t, f(x_{new}^t)), & \text{if } (rand_3 < A_i^t \text{ and } f(x_{new}^t) > f(x_i^{t-1})), \\ (x_i^{t-1}, f(x_i^{t-1})) & \text{tr}_i = \text{tr}_i + 1, \\ \text{otherwise,} & \end{cases} \quad (20)$$

where $rand_3$ is a random number in the range $[0, 1]$, tr_i is a vector recording the number of attempts through which solution x_i^t could not be improved at cycle t , and A_i^t is defined by (15). In the above equation, if the solution x_i^{t-1} cannot be improved, then the new solution x_{new} is abandoned and the i th element of the trial vector tr is increased by one. Also, after certain number of cycles determined by the variable limit, if the solution x_i^t cannot be further improved, it is abandoned and replaced by randomly generated solution. In this case, the i th element of the trial vector is set to 0. This modification can improve the exploration process and it will help to avoid trapping into some local optima. Also, it will improve the solution quality and speed convergence.

Step 5 (memorize the best current solution). Record the best solution so far (x_{best}), that is, the solution with the highest objective function value.

Step 6 (check the stopping criteria). If the termination criterion is met or the variable *cycle* is equal to the maximum number of iterations, then the algorithm is finished. Otherwise, increase the variable *cycle* by one and go to Step 2.

5. Experimental Results

The multilevel image thresholding problem deals with finding optimal thresholds within the range $[0, L - 1]$ that maximize the functions defined by (3) and (6). The dimension of the

optimization problem is the number of thresholds k , and the search space is $[0, L - 1]^k$. In this study our proposed IBA algorithm was compared against four other standard population based metaheuristic techniques: PSO, DE, CS, and FA from [49] and pure BA.

The experiments were conducted on 6 standard images, the same as used in [49], in order to make comparison of the obtained results simpler. Images used in this paper, namely, Barbara, Living room, Boats, Goldhill, and Lake, are of size (512×512) and Aerial has size (256×256) . Barbara and Boats images are available at http://decsai.ugr.es/~javier/denoise/test_images/. The Living room and Lake images were chosen from http://www.imageprocessingplace.com/root_files_V3/image_databases.htm. The Goldhill image can be found at https://ece.uwaterloo.ca/~z70wang/research/quality_index/demo.html. The Aerial image was taken from the University of Southern California Signal and Image Processing Institute's image database at <http://sipi.usc.edu/database/database.php?volume=misc>. These original images and their gray level histograms are depicted in Figures 1 and 2, respectively.

For the Kapur's and Otsu's thresholding methods, the exhaustive search method was conducted first to derive the optimal solutions, the corresponding optimal objective function values, and the processing time for comparison with the results generated by the PSO, DE, CS, FA, BA, and IBA algorithms. These results generated by the exhaustive search for Kapur's and Otsu's criterion are presented in Tables 1 and 2, respectively. It is obvious that computational times increase exponentially and for more than 5 thresholds become unacceptable. We did not implement optimal use of multicore processor, but improvements would not be significant.

The number of thresholds k explored in the experiments were 2, 3, 4, and 5. Since metaheuristic algorithms have stochastic characteristics, each experiment was repeated 50 times for each image and for each k value. Each run of an algorithm was terminated when the fitness value of the best solution $f(x_{best})$ reached the known optimal value (from the exhaustive search) of the objective function f_{opt} , that is, $|f(x_{best}) - f_{opt}| < \epsilon$, where $\epsilon = 10^{-9}$ was a tolerance for the accuracy of the measurement. Hence, the stopping condition for all algorithms was the value of the fitness, unless optimum could not be reached within 2000 iterations.

The proposed IBA method has been implemented in C# programming language, as the rest of the algorithms. Results for CS and FA are from [49]. All tests were done on an Intel Core i7-3770K @3.5 GHz with 16 GB of RAM running under the Windows 8 x64 operating system. The PSO and DE algorithms have been implemented in their basic versions, while the BA and IBA have been implemented as it was described in the previous two sections.

5.1. Parameters Setup. To compare the proposed IBA algorithm with PSO, DE, CS, FA [49], and BA algorithms, the objective function evaluation was computed $N \times G$ times, where N is the population size and G is the maximum number of generations (unless optimum was reached earlier).



(a)



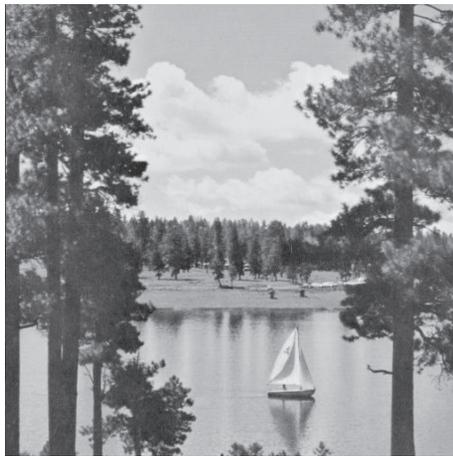
(b)



(c)



(d)



(e)



(f)

FIGURE 1: Test images: (a) Barbara, (b) Living room, (c) Boats, (d) Goldhill, (e) Lake, and (f) Aerial.

The population size in all algorithms was set to $N = 40$ and the number of generation is set to $G = 2000$ for all algorithms, as in [49]. Besides these common control parameters, each of mentioned algorithms has additional control parameters that directly improve their performance.

For both the proposed IBA and pure BA algorithms, the additional control parameters f_{\min} and f_{\max} were set to 0 and 2.0, respectively. The initial values for parameters r_i^0 and loudness A_i were set to 0.5 and 0.99, respectively. The constant β was set to 0.9. Instead of the average loudness $\langle A_i^t \rangle$

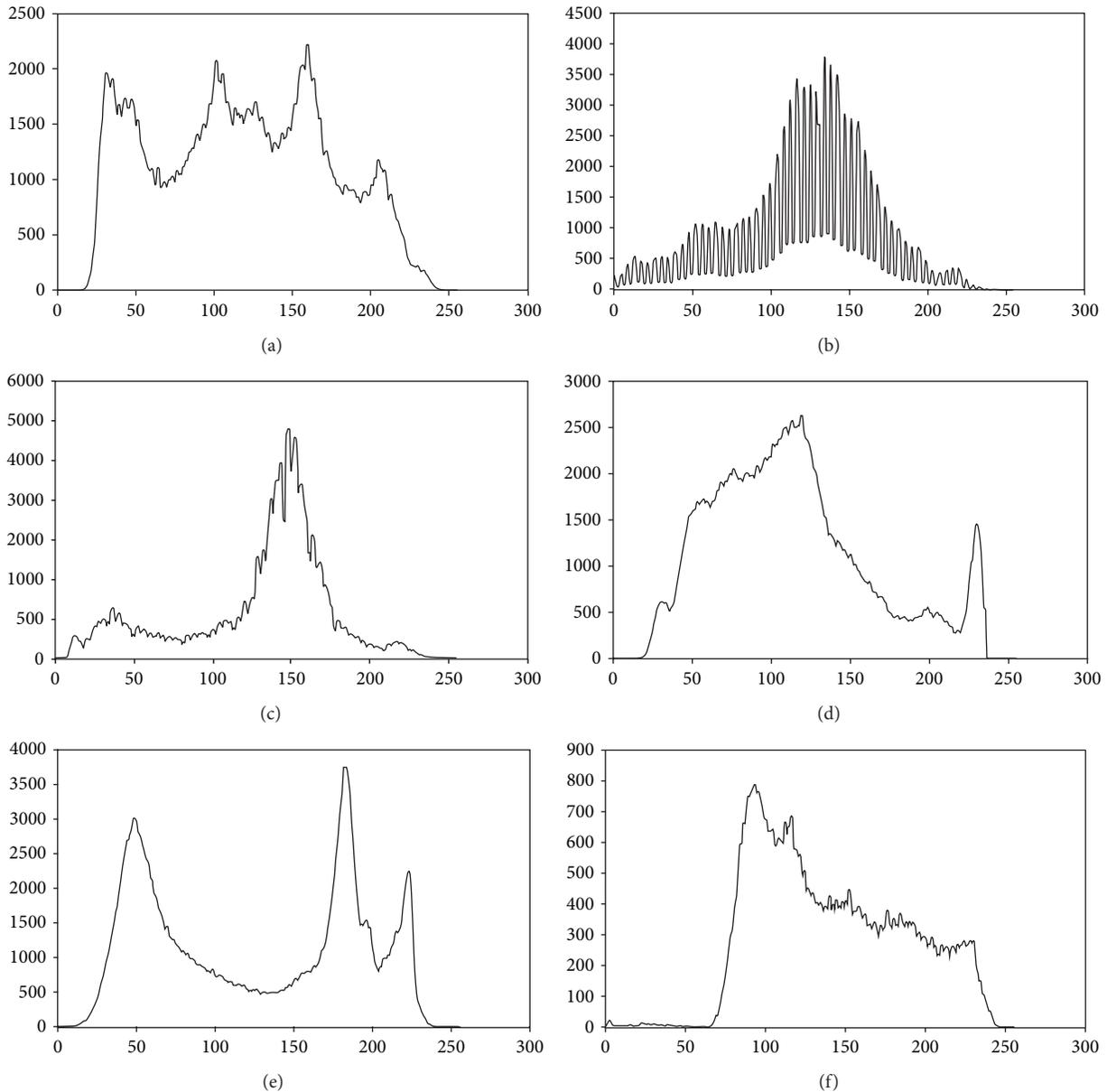


FIGURE 2: Gray-level histogram of test images: (a) Barbara, (b) Living room, (c) Boats, (d) Goldhill, (e) Lake, and (f) Aerial.

of all bats, we found that the value 1.66 was acceptable for all images. In the proposed IBA algorithm, control parameters introduced from DE algorithm, such as differential weight F and crossover probability C_r , were set to 0.75 and 0.95, respectively. Also, in the IBA method, the parameter limit was set to 150.

5.2. Quality and Computational Analysis of the Results. The mean and standard deviations for 50 runs for six tested metaheuristic algorithms have been calculated and are presented in Table 3 for the experiments based on Kapur's entropy and in Table 4 for the experiments based on Otsu's objective function. These mean values can be compared to the optimal

values of the corresponding objective functions found by an exhaustive search from Tables 1 and 2.

The first conclusion that can be drawn from the results in Tables 3 and 4 is that the cases when the number of desired thresholds is 2 or 3 are too easy and are not interesting for nondeterministic metaheuristics. Almost all algorithms in almost all cases reached optimal results (PSO and DE had few misses). We included these results in the tables for comparison with results in [49], but we will not discuss them further. All the remaining discussion is only about cases when the number of desired thresholds is 4 or 5.

From Tables 3, 4, 5, and 6 many details can be seen. We will here, in three additional tables, synthesize the most

TABLE 1: Thresholds, objective function values, and time processing provided by the exhaustive search for Kapur’s method.

Images	K	Threshold values	Objective function	Time (ms)
Barbara	2	96, 168	12.668336540	25
	3	76, 127, 178	15.747087798	341
	4	60, 99, 141, 185	18.556786861	11103
	5	58, 95, 133, 172, 210	21.245645310	666869
Living room	2	94, 175	12.405985592	31
	3	47, 103, 175	15.552622213	339
	4	47, 98, 149, 197	18.471055578	12612
	5	42, 85, 124, 162, 197	21.150302316	478114
Boats	2	107, 176	12.574798244	25
	3	64, 119, 176	15.820902860	342
	4	48, 88, 128, 181	18.655733570	11461
	5	48, 88, 128, 174, 202	21.401608305	469862
Goldhill	2	90, 157	12.546393623	24
	3	78, 131, 177	15.607747002	329
	4	65, 105, 147, 189	18.414213765	11958
	5	59, 95, 131, 165, 199	21.099138996	399458
Lake	2	91, 163	12.520359742	24
	3	72, 119, 169	15.566286745	336
	4	70, 111, 155, 194	18.365636309	12658
	5	64, 99, 133, 167, 199	21.024982760	410753
Aerial	2	68, 159	12.538208248	29
	3	68, 130, 186	15.751881495	347
	4	68, 117, 159, 200	18.615899102	11390
	5	68, 108, 141, 174, 207	21.210455499	599570

important conclusions concerning the quality of the results and the convergence speed.

Table 7, computed from Tables 3 and 4, shows for each tested algorithm in what percentage of cases it achieved the best result, considering all tested images and both optimization criteria. From Table 7, we can see that PSO and DE were very inferior compared to other tested algorithms. The results for the CS and FA [49] algorithms are quite acceptable, where FA had slightly better results.

For the BA we can notice that it gives rather poor results for the Kapur’s method, while it gives rather good results for the Otsu’s method. When the Kapur’s criterion is used, the BA gets trapped in local optima, so it consumes the maximum number of iterations without switching to another subspace

TABLE 2: Thresholds, objective function values, and time processing provided by the exhaustive search for Otsu’s method.

Images	K	Threshold values	Objective function	Time (ms)
Barbara	2	82, 147	2608.610778507	39
	3	75, 127, 176	2785.163280467	89
	4	66, 106, 142, 182	2856.262131671	3014
	5	57, 88, 118, 148, 184	2890.976609405	100079
Living room	2	87, 145	1627.909172752	39
	3	76, 123, 163	1760.103018395	88
	4	56, 97, 132, 168	1828.864376614	2945
	5	49, 88, 120, 146, 178	1871.990616316	130397
Boats	2	93, 155	1863.346730649	38
	3	73, 126, 167	1994.536306242	89
	4	65, 114, 147, 179	2059.866280428	2931
	5	51, 90, 126, 152, 183	2092.775965336	75879
Goldhill	2	94, 161	2069.510202452	38
	3	83, 126, 179	2220.372641501	88
	4	69, 102, 138, 186	2295.380469158	2775
	5	63, 91, 117, 147, 191	2331.156597921	74674
Lake	2	85, 154	3974.738214185	39
	3	78, 140, 194	4112.631097687	89
	4	67, 110, 158, 198	4180.886161109	2613
	5	57, 88, 127, 166, 200	4216.943583790	73019
Aerial	2	125, 178	1808.171050536	46
	3	109, 147, 190	1905.410606582	103
	4	104, 134, 167, 202	1957.017965982	2670
	5	99, 123, 148, 175, 205	1980.656737348	99880

which is more promising. That explains why it needed some modifications to be introduced to help it leave the local optimum space and continue to search new spaces.

Our proposed improved IBA algorithm, by taking some features of the DE and ABC algorithms, obtained the best results compared to the rest of algorithms. It actually achieved the best result for both mean value and variance, for all tested cases.

Tables 5 and 6 report the mean number of iterations and the average CPU time taken by each algorithm to satisfy the stopping condition for Kapur’s and Otsu’s criteria, respectively. Most significant conclusions concerning the convergence speed of the tested algorithms are shown in Tables 8 and 9.

In Table 8 (for Kapur’s criterion) in each column labeled by Thrs. k ($k = 2, 3, 4, 5$) we calculated for each of the tested algorithms: PSO, DE, CS, FA, BA, and IBA, the sum of mean number of required iterations for each test image. We can observe that in the case of the FA and especially the IBA

TABLE 3: Comparison of the mean values and standard deviations obtained for the PSO, DE, CS, FA, BA, and IBA based on Kapur's entropy criterion for six test images over 50 runs.

Alg.	K	Barbara			Living room			Boats			Goldhill			Lake			Aerial								
		Mean values	St. Dev.	St. Dev.	Mean values	St. Dev.	St. Dev.	Mean values	St. Dev.	St. Dev.	Mean values	St. Dev.	St. Dev.	Mean values	St. Dev.	St. Dev.	Mean values	St. Dev.	St. Dev.						
PSO	2	12.668336540	5.33E-15	12.405792709	1.64E-04	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15	12.668336540	5.33E-15	12.405792709	1.64E-04	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15
	3	15.747087798	1.42E-14	15.552015642	2.82E-03	15.820679619	8.84E-04	15.607747002	1.42E-14	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14	15.552015642	2.82E-03	15.820679619	8.84E-04	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14
	4	18.549612938	1.94E-02	18.467328310	6.64E-03	18.640100415	3.00E-02	18.414173744	2.07E-04	18.357505953	2.02E-02	18.615899102	1.78E-14	18.549612938	1.94E-02	18.467328310	6.64E-03	18.640100415	3.00E-02	18.414173744	2.07E-04	18.357505953	2.02E-02	18.615899102	1.78E-14
	5	21.241857967	6.71E-03	21.131564234	2.18E-02	21.392020144	4.12E-02	21.099092699	1.28E-04	21.015922726	4.40E-02	21.192396874	5.44E-02	21.241857967	6.71E-03	21.131564234	2.18E-02	21.392020144	4.12E-02	21.099092699	1.28E-04	21.015922726	4.40E-02	21.192396874	5.44E-02
	2	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15
DE	3	15.747087798	1.42E-14	15.552578874	3.03E-04	15.820902860	8.88E-15	15.607743578	2.40E-05	15.566286745	1.24E-14	15.751881495	5.33E-15	15.747087798	1.42E-14	15.552578874	3.03E-04	15.820902860	8.88E-15	15.607743578	2.40E-05	15.566286745	1.24E-14	15.751881495	5.33E-15
	4	18.556749938	1.51E-04	18.470970822	3.16E-04	18.655660844	1.64E-04	18.603017275	1.78E-03	18.365579671	2.79E-04	18.615899102	1.78E-14	18.556749938	1.51E-04	18.470970822	3.16E-04	18.655660844	1.64E-04	18.603017275	1.78E-03	18.365579671	2.79E-04	18.615899102	1.78E-14
	5	21.245566656	2.34E-04	21.149062508	1.78E-03	21.401458219	3.21E-04	21.409946039	3.55E-15	21.024780111	4.59E-04	21.210411012	1.12E-04	21.245566656	2.34E-04	21.149062508	1.78E-03	21.401458219	3.21E-04	21.409946039	3.55E-15	21.024780111	4.59E-04	21.210411012	1.12E-04
	2	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15
	3	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-14	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14
CS	4	18.556786861	2.49E-14	18.471055578	2.49E-14	18.655733570	1.07E-14	18.414197322	6.53E-05	18.365636309	1.78E-14	18.615899102	1.78E-14	18.556786861	2.49E-14	18.471055578	2.49E-14	18.655733570	1.07E-14	18.414197322	6.53E-05	18.365636309	1.78E-14	18.615899102	1.78E-14
	5	21.245645311	1.42E-14	21.149400604	1.64E-03	21.401608305	7.11E-15	21.099125539	6.59E-05	21.024962923	5.95E-05	21.210455499	1.78E-15	21.245645311	1.42E-14	21.149400604	1.64E-03	21.401608305	7.11E-15	21.099125539	6.59E-05	21.024962923	5.95E-05	21.210455499	1.78E-15
	2	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15
	3	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-14	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14
	4	18.556786861	2.49E-14	18.471014902	2.85E-04	18.655723798	4.79E-05	18.414213765	2.13E-14	18.365636309	1.78E-14	18.615899102	1.78E-14	18.556786861	2.49E-14	18.471014902	2.85E-04	18.655723798	4.79E-05	18.414213765	2.13E-14	18.365636309	1.78E-14	18.615899102	1.78E-14
FA	5	21.245645311	1.42E-14	21.149483979	1.46E-03	21.401583877	7.33E-05	21.099138996	0.00E-00	21.024982760	0.00E-00	21.210455499	1.78E-15	21.245645311	1.42E-14	21.149483979	1.46E-03	21.401583877	7.33E-05	21.099138996	0.00E-00	21.024982760	0.00E-00	21.210455499	1.78E-15
	2	12.668336540	5.33E-15	12.405885825	1.52E-04	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15	12.668336540	5.33E-15	12.405885825	1.52E-04	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15
	3	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-14	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14
	4	18.55593147	8.36E-03	18.469796068	3.52E-03	18.644796771	2.68E-02	18.414213765	2.13E-14	18.356346829	2.13E-02	18.615899102	1.78E-14	18.55593147	8.36E-03	18.469796068	3.52E-03	18.644796771	2.68E-02	18.414213765	2.13E-14	18.356346829	2.13E-02	18.615899102	1.78E-14
	5	21.245645311	1.42E-14	21.133082528	1.59E-02	21.399205241	7.12E-03	21.099138996	0.00E-00	21.024982760	0.00E-00	21.210452617	2.02E-05	21.245645311	1.42E-14	21.133082528	1.59E-02	21.399205241	7.12E-03	21.099138996	0.00E-00	21.024982760	0.00E-00	21.210452617	2.02E-05
BA	2	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15
	3	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-14	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14
	4	18.55593147	8.36E-03	18.469796068	3.52E-03	18.644796771	2.68E-02	18.414213765	2.13E-14	18.356346829	2.13E-02	18.615899102	1.78E-14	18.55593147	8.36E-03	18.469796068	3.52E-03	18.644796771	2.68E-02	18.414213765	2.13E-14	18.356346829	2.13E-02	18.615899102	1.78E-14
	5	21.245645311	1.42E-14	21.133082528	1.59E-02	21.399205241	7.12E-03	21.099138996	0.00E-00	21.024982760	0.00E-00	21.210452617	2.02E-05	21.245645311	1.42E-14	21.133082528	1.59E-02	21.399205241	7.12E-03	21.099138996	0.00E-00	21.024982760	0.00E-00	21.210452617	2.02E-05
	2	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15
IBA	3	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-14	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14
	4	18.556786861	2.49E-14	18.471055578	2.49E-14	18.655733570	1.07E-14	18.414213765	2.13E-14	18.365636309	1.78E-14	18.615899102	1.78E-14	18.556786861	2.49E-14	18.471055578	2.49E-14	18.655733570	1.07E-14	18.414213765	2.13E-14	18.365636309	1.78E-14	18.615899102	1.78E-14
	5	21.245645311	1.42E-14	21.150302316	1.78E-14	21.401608305	7.11E-15	21.099138996	0.00E-00	21.024982760	0.00E-00	21.210455499	1.78E-15	21.245645311	1.42E-14	21.150302316	1.78E-14	21.401608305	7.11E-15	21.099138996	0.00E-00	21.024982760	0.00E-00	21.210455499	1.78E-15
	2	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15	12.668336540	5.33E-15	12.405985592	5.33E-15	12.574798244	1.42E-14	12.546393623	7.11E-15	12.520359742	5.33E-15	12.538208248	1.78E-15
	3	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-14	15.607747002	1.42E-14	15.566286745	5.33E-15	15.747087798	1.42E-14	15.552622213	1.07E-14	15.820902860	8.88E-15	15.607747002	1.42E-1				

TABLE 4: Comparison of the mean values and standard deviations obtained for the PSO, DE, CS, FA, BA, and IBA based on Otsu's criterion for six test images over 50 runs.

Alg. K	Barbara			Living room			Boats			Goldhill			Lake			Aerial										
	Mean values	St. dev.	St. dev.	Mean values	St. dev.	St. dev.	Mean values	St. dev.	St. dev.	Mean values	St. dev.	St. dev.	Mean values	St. dev.	St. dev.	Mean values	St. dev.	St. dev.								
PSO	2	2608.610778507	1.82E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	3.64E-12	1808.171050536	2.27E-13	2	2608.610778507	1.82E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	3.64E-12	1808.171050536	2.27E-13
	3	2785.163280467	2.27E-12	1760.103018395	2.27E-13	1994.536306242	1.59E-12	2220.372641501	1.36E-12	4112.631097687	4.55E-12	1905.410606582	1.14E-12	3	2785.163280467	2.27E-12	1760.103018395	2.27E-13	1994.536306242	1.59E-12	2220.372641501	1.36E-12	4112.631097687	4.55E-12	1905.410606582	1.14E-12
	4	2856.260804034	6.66E-03	1828.864376614	1.59E-12	2059.866220175	4.22E-04	2295.380095430	1.48E-03	4180.883976390	7.41E-03	1955.085619462	7.65E+00	4	2856.260804034	6.66E-03	1828.864376614	1.59E-12	2059.866220175	4.22E-04	2295.380095430	1.48E-03	4180.883976390	7.41E-03	1955.085619462	7.65E+00
	5	2890.975549258	5.05E-02	1871.984827146	2.29E-02	2092.771150715	8.36E-03	2331.156479206	3.56E-04	4216.942888298	3.99E-03	1979.170306260	2.51E+00	5	2890.975549258	5.05E-02	1871.984827146	2.29E-02	2092.771150715	8.36E-03	2331.156479206	3.56E-04	4216.942888298	3.99E-03	1979.170306260	2.51E+00
	2	2608.610778507	1.82E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	3.64E-12	1808.171050536	2.27E-13	2	2608.610778507	1.82E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	3.64E-12	1808.171050536	2.27E-13
DE	3	2785.162093432	8.31E-03	1760.103018395	2.27E-13	1994.535269293	7.26E-03	2220.372641501	1.36E-12	4112.631097687	4.55E-12	1905.410606582	1.14E-12	3	2785.162093432	8.31E-03	1760.103018395	2.27E-13	1994.535269293	7.26E-03	2220.372641501	1.36E-12	4112.631097687	4.55E-12	1905.410606582	1.14E-12
	4	2856.261305066	2.80E-03	1828.860328016	1.30E-02	2059.865271461	6.85E-03	2295.380095430	1.48E-03	4180.883976390	7.41E-03	1955.085619462	7.65E+00	4	2856.261305066	2.80E-03	1828.860328016	1.30E-02	2059.865271461	6.85E-03	2295.380095430	1.48E-03	4180.883976390	7.41E-03	1955.085619462	7.65E+00
	5	2890.971346990	2.05E-02	1871.976701063	2.34E-02	2092.766907541	2.71E-02	2331.155240485	4.76E-03	4216.942888298	3.99E-03	1979.170306260	2.51E+00	5	2890.971346990	2.05E-02	1871.976701063	2.34E-02	2092.766907541	2.71E-02	2331.155240485	4.76E-03	4216.942888298	3.99E-03	1979.170306260	2.51E+00
	2	2608.610778507	1.82E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	3.64E-12	1808.171050536	2.27E-13	2	2608.610778507	1.82E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	3.64E-12	1808.171050536	2.27E-13
	3	2785.163280467	2.27E-12	1760.103018395	2.27E-13	1994.536306242	1.59E-12	2220.372641501	1.36E-12	4112.631097687	4.55E-12	1905.410606582	1.14E-12	3	2785.163280467	2.27E-12	1760.103018395	2.27E-13	1994.536306242	1.59E-12	2220.372641501	1.36E-12	4112.631097687	4.55E-12	1905.410606582	1.14E-12
CS	4	2856.261511717	2.45E-03	1828.864376614	1.59E-12	2059.866280428	1.36E-12	2295.380469158	2.27E-12	4180.886161109	0.00E-00	1957.017965982	0.00E-00	4	2856.261511717	2.45E-03	1828.864376614	1.59E-12	2059.866280428	1.36E-12	2295.380469158	2.27E-12	4180.886161109	0.00E-00	1957.017965982	0.00E-00
	5	2890.976540127	4.85E-04	1871.990230213	2.70E-03	2092.775817560	1.03E-03	2331.155240485	4.76E-03	4216.943583790	9.09E-13	1980.651043072	1.16E-02	5	2890.976540127	4.85E-04	1871.990230213	2.70E-03	2092.775817560	1.03E-03	2331.155240485	4.76E-03	4216.943583790	9.09E-13	1980.651043072	1.16E-02
	2	2608.610778507	1.82E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	3.64E-12	1808.171050536	2.27E-13	2	2608.610778507	1.82E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	3.64E-12	1808.171050536	2.27E-13
	3	2785.163280467	2.27E-12	1760.103018395	2.27E-13	1994.536306242	1.59E-12	2220.372641501	1.36E-12	4112.631097687	4.55E-12	1905.410606582	1.14E-12	3	2785.163280467	2.27E-12	1760.103018395	2.27E-13	1994.536306242	1.59E-12	2220.372641501	1.36E-12	4112.631097687	4.55E-12	1905.410606582	1.14E-12
	4	2856.262131671	4.55E-13	1828.864376614	1.59E-12	2059.866280428	1.36E-12	2295.380469158	2.27E-12	4180.886161109	0.00E-00	1957.017965982	0.00E-00	4	2856.262131671	4.55E-13	1828.864376614	1.59E-12	2059.866280428	1.36E-12	2295.380469158	2.27E-12	4180.886161109	0.00E-00	1957.017965982	0.00E-00
FA	5	2890.976609405	3.64E-12	1871.990616316	0.00E-00	2092.773515829	3.57E-03	2331.156597921	2.27E-12	4216.943583790	9.09E-13	1980.656737348	0.09E-13	5	2890.976609405	3.64E-12	1871.990616316	0.00E-00	2092.773515829	3.57E-03	2331.156597921	2.27E-12	4216.943583790	9.09E-13	1980.656737348	0.09E-13
	2	2608.610778507	1.82E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	3.64E-12	1808.171050536	2.27E-13	2	2608.610778507	1.82E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	3.64E-12	1808.171050536	2.27E-13
	3	2785.163280467	2.27E-12	1760.103018395	2.27E-13	1994.536306242	1.59E-12	2220.372641501	1.36E-12	4112.631097687	4.55E-12	1905.410606582	1.14E-12	3	2785.163280467	2.27E-12	1760.103018395	2.27E-13	1994.536306242	1.59E-12	2220.372641501	1.36E-12	4112.631097687	4.55E-12	1905.410606582	1.14E-12
	4	2856.262131671	4.55E-13	1828.864376614	1.59E-12	2059.866280428	1.36E-12	2295.380469158	2.27E-12	4180.886161109	0.00E-00	1957.017965982	0.00E-00	4	2856.262131671	4.55E-13	1828.864376614	1.59E-12	2059.866280428	1.36E-12	2295.380469158	2.27E-12	4180.886161109	0.00E-00	1957.017965982	0.00E-00
	5	2890.976609405	3.64E-12	1871.990616316	0.00E-00	2092.773515829	3.57E-03	2331.156597921	2.27E-12	4216.943583790	9.09E-13	1980.656737348	0.09E-13	5	2890.976609405	3.64E-12	1871.990616316	0.00E-00	2092.773515829	3.57E-03	2331.156597921	2.27E-12	4216.943583790	9.09E-13	1980.656737348	0.09E-13
BA	2	2608.610778507	1.36E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	4.09E-12	1808.171050536	2.27E-13	2	2608.610778507	1.36E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	4.09E-12	1808.171050536	2.27E-13
	3	2785.163280467	1.36E-12	1760.103018395	2.27E-13	1994.536306242	1.14E-12	2220.372641501	1.36E-12	4112.631097687	3.64E-12	1905.410606582	1.14E-12	3	2785.163280467	1.36E-12	1760.103018395	2.27E-13	1994.536306242	1.14E-12	2220.372641501	1.36E-12	4112.631097687	3.64E-12	1905.410606582	1.14E-12
	4	2856.262131671	4.55E-13	1828.864376614	2.27E-12	2059.866280428	9.09E-13	2295.380469158	2.27E-12	4180.886161109	0.00E-00	1957.017965982	2.27E-13	4	2856.262131671	4.55E-13	1828.864376614	2.27E-12	2059.866280428	9.09E-13	2295.380469158	2.27E-12	4180.886161109	0.00E-00	1957.017965982	2.27E-13
	5	2890.976609405	2.73E-12	1871.990616316	0.00E-00	2092.772750357	3.78E-03	2331.156597921	2.27E-12	4216.943583790	3.64E-12	1979.513584665	2.29E+00	5	2890.976609405	2.73E-12	1871.990616316	0.00E-00	2092.772750357	3.78E-03	2331.156597921	2.27E-12	4216.943583790	3.64E-12	1979.513584665	2.29E+00
	2	2608.610778507	1.36E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	4.09E-12	1808.171050536	2.27E-13	2	2608.610778507	1.36E-12	1627.909172752	0.00E-00	1863.346730649	0.00E-00	2069.510202452	4.55E-13	3974.738214185	4.09E-12	1808.171050536	2.27E-13
IBA	3	2785.163280467	1.36E-12	1760.103018395	2.27E-13	1994.536306242	1.14E-12	2220.372641501	1.36E-12	4112.631097687	3.64E-12	1905.410606582	1.14E-12	3	2785.163280467	1.36E-12	1760.103018395	2.27E-13	1994.536306242	1.14E-12	2220.372641501	1.36E-12	4112.631097687	3.64E-12	1905.410606582	1.14E-12
	4	2856.262131671	4.55E-13	1828.864376614	2.27E-12	2059.866280428	9.09E-13	2295.380469158	2.27E-12	4180.886161109	0.00E-00	1957.017965982	0.00E-00	4	2856.262131671	4.55E-13	1828.864376614	2.27E-12	2059.866280428	9.09E-13	2295.380469158	2.27E-12	4180.886161109	0.00E-00	1957.017965982	0.00E-00
	5	2890.976609405	2.73E-12	1871.990616316	0.00E-00	2092.775965336	1.36E-12	2331.156597921	2.27E-12	4216.943583790	3.64E-12	1980.656737348	0.09E-13	5	2890.976609405	2.73E-12	1871.990616316	0.00E-00								

TABLE 5: Mean of the CPU times (in milliseconds) and mean of the iteration numbers obtained for the PSO, DE, CS, FA, BA, and IBA based on Kapur's entropy criterion for six test images over 50 runs.

Alg.	K	Barbara			Living room			Boats			Goldhill			Lake			Aerial				
		Mean time (ms)	Mean iteration																		
PSO	2	2.18	9.22	102.82	1165.14	3.08	11.84	2.54	8.84	2.41	8.86	2.62	10.7	2.41	8.86	2.62	10.7	2.41	8.86	2.62	10.7
	3	3.30	14.28	21.96	218.56	16.23	136.58	3.18	13.54	3.65	14.58	3.24	13.9	3.65	14.58	3.24	13.9	3.65	14.58	3.24	13.9
	4	49.00	495.36	77.23	853.62	123.62	1367.86	10.22	97.86	26.99	295.76	4.03	19.76	26.99	295.76	4.03	19.76	26.99	295.76	4.03	19.76
	5	88.16	1050.8	153.53	1725.22	74.46	814.22	23.56	258.5	77.64	893.98	58.54	695.92	77.64	893.98	58.54	695.92	77.64	893.98	58.54	695.92
	2	3.56	14.59	4.34	18.92	3.38	16.92	2.15	16.44	5.0	15.74	3.81	17.01	5.0	15.74	3.81	17.01	5.0	15.74	3.81	17.01
DE	3	6.3	30.0	8.02	70.60	8.84	43.32	7.94	69.30	6.24	30.04	6.51	30.96	6.24	30.04	6.51	30.96	6.24	30.04	6.51	30.96
	4	24.46	240.68	33.62	322.91	46.92	477.16	48.36	125.48	22.12	165.08	14.12	124.56	22.12	165.08	14.12	124.56	22.12	165.08	14.12	124.56
	5	47.8	527.96	77.14	801.21	65.48	683.29	22.2	116.28	55.9	603.01	56.38	624.16	55.9	603.01	56.38	624.16	55.9	603.01	56.38	624.16
	2	71.04	194.54	53.30	129.84	53.84	135.58	82.29	209.48	70.48	183.36	56.32	150.88	70.48	183.36	56.32	150.88	70.48	183.36	56.32	150.88
	3	150.71	420.42	125.48	322.42	128.92	330.22	149.68	441.64	138.23	375.66	104.75	292.22	138.23	375.66	104.75	292.22	138.23	375.66	104.75	292.22
CS	4	189.31	518.58	222.48	570.6	170.28	436.02	180.31	500.11	220.44	604.58	174.84	482.26	220.44	604.58	174.84	482.26	220.44	604.58	174.84	482.26
	5	301.65	786.64	499.42	1303.8	247.15	631.36	280.55	720.34	336.29	915.92	193.21	532.76	336.29	915.92	193.21	532.76	336.29	915.92	193.21	532.76
	2	15.01	11.96	17.02	11.9	16.39	12.32	13.66	10.08	15.92	13.86	11.0	11.0	15.92	13.86	11.0	11.0	15.92	13.86	11.0	11.0
	3	34.07	29.82	37.24	29.7	36.24	29.24	32.08	28.66	29.18	32.83	29.18	29.18	32.08	28.66	29.18	32.83	29.18	32.08	28.66	29.18
	4	43.30	38.6	50.25	77.9	54.68	117.8	41.11	36.6	43.84	37.66	38.96	38.96	41.11	36.6	43.84	37.66	43.18	43.84	37.66	43.18
FA	5	50.15	44.04	104.74	515.06	76.22	241.94	48.95	42.01	50.31	43.6	45.46	45.46	50.31	43.6	45.46	45.46	50.31	43.6	45.46	45.46
	2	1.78	2.04	96.82	722.92	24.17	142.96	2	1.8	1.6	1.74	5.06	5.06	1.6	1.74	5.06	5.06	1.6	1.74	5.06	5.06
	3	2.52	6.74	89.78	734.8	64.96	421.48	2.84	6.94	2.32	8.82	14.4	14.4	2.32	8.82	14.4	14.4	2.32	8.82	14.4	14.4
	4	40.2	146.46	124.16	1098.72	115.34	969.66	6.76	37.18	56.88	397.8	134.52	134.52	56.88	397.8	134.52	134.52	56.88	397.8	134.52	134.52
	5	60.9	412.92	175.32	1718.82	98.42	785.04	30.92	157.92	64.48	463.78	463.78	463.78	30.92	157.92	64.48	463.78	64.48	463.78	64.48	463.78
IBA	2	2.12	9.14	6.34	25.14	5.36	10.02	2.88	8.92	2.28	9.18	11.18	11.18	2.28	9.18	11.18	11.18	2.28	9.18	11.18	11.18
	3	3.84	16.8	5.3	22.8	5.5	22.4	3.84	16.62	5	17.5	19.88	19.88	3.84	16.62	5	17.5	5	17.5	19.88	19.88
	4	7.16	26.26	9.92	35.48	10.3	43.3	6.28	28.82	7.14	26.98	30.98	30.98	6.28	28.82	7.14	26.98	7.14	26.98	7.48	7.48
	5	8.66	40.06	24.7	134.38	14.4	50.9	7.88	38.62	9.5	42.7	44.98	44.98	7.88	38.62	9.5	42.7	9.5	42.7	9.5	44.98
	2	1.78	2.04	96.82	722.92	24.17	142.96	2	1.8	1.6	1.74	5.06	5.06	1.6	1.74	5.06	5.06	1.6	1.74	5.06	5.06

TABLE 6: Mean of the CPU times (in milliseconds) and mean of the iteration numbers obtained for the PSO, DE, CS, FA, BA, and IBA based on Otsu's entropy criterion for six test images over 50 runs.

Alg.	K	Barbara			Living room			Boats			Goldhill			Lake			Aerial		
		Mean time (ms)	Mean iteration	Mean iteration	Mean time (ms)	Mean iteration	Mean iteration	Mean time (ms)	Mean iteration	Mean iteration	Mean time (ms)	Mean iteration	Mean iteration	Mean time (ms)	Mean iteration	Mean iteration	Mean time (ms)	Mean iteration	Mean iteration
PSO	2	0.84	9.4	9.6	0.90	0.94	8.8	0.94	8.98	10.12	0.81	8.98	10.12	0.81	9.04	10.12	0.81	9.04	10.12
	3	1.22	13.26	14.68	1.24	1.28	14.1	1.28	14.3	1.17	14.3	13.48	1.17	15.16	13.48	1.29	13.48	15.16	13.48
	4	4.84	138.52	19.18	1.60	3.12	56.56	3.12	136.32	6.55	175.58	5.18	175.58	5.18	138.44	175.58	5.18	138.44	175.58
	5	10.02	261.06	221.64	8.14	32.96	1012.3	7.52	222.92	7.01	179.38	20.21	179.38	20.21	622.38	179.38	20.21	622.38	179.38
	2	0.73	14.06	15.66	1.0	1.0	16.59	0.81	15.14	0.76	17.72	0.85	17.72	0.85	15.9	17.72	0.85	15.9	17.72
DE	3	3.01	66.3	30.3	2.25	2.81	70.01	1.51	27.88	1.90	29.19	3.20	29.19	70.1	29.19	3.20	70.1	29.19	70.1
	4	8.0	199.22	282.01	11.12	4.72	120.58	7.60	199.5	7.42	200.23	7.01	200.23	162.5	200.23	7.01	162.5	200.23	200.23
	5	26.6	796.88	683.11	24.3	31.90	950.33	23.9	686.32	21.65	604.76	18.52	604.76	523.22	604.76	18.52	523.22	604.76	604.76
	2	35.80	179.0	223.34	30.03	32.74	204.04	27.88	195.30	38.04	267.66	32.45	267.66	254.10	267.66	32.45	254.10	267.66	267.66
	3	51.66	370.88	371.50	56.67	53.74	378.40	59.98	424.62	49.68	381.36	52.78	381.36	395.44	381.36	52.78	395.44	381.36	381.36
CS	4	100.78	723.22	578.94	83.43	76.98	595.66	70.92	531.74	85.30	646.44	83.84	646.44	593.52	646.44	83.84	593.52	646.44	646.44
	5	114.63	802.80	836.32	122.78	102.84	677.04	159.69	1115.44	109.90	746.36	209.36	746.36	1487.06	746.36	209.36	1487.06	746.36	746.36
	2	8.16	12.02	12.54	8.72	7.17	10.64	8.61	12.68	7.94	11.66	7.94	11.66	12.18	11.66	7.94	12.18	11.66	11.66
	3	15.73	28.62	28.54	12.67	16.10	28.7	15.61	27.78	13.38	30.04	15.56	30.04	27.7	30.04	15.56	27.7	30.04	30.04
	4	17.84	37.7	38.78	19.16	18.43	38.2	17.12	37.66	18.49	38.86	18.05	38.86	38.58	38.86	18.05	38.58	38.86	38.86
FA	5	21.27	43.9	43.78	20.15	43.51	669.52	20.93	44.54	21.27	43.12	20.58	43.12	44.68	43.12	20.58	44.68	43.12	43.12
	2	0.4	1.86	1.66	0.4	0.42	1.58	0.42	1.75	0.38	1.7	0.38	1.7	1.64	1.7	0.38	1.64	1.7	1.7
	3	0.98	6.08	6.82	1.48	1.24	8.54	0.8	5.58	0.82	6.14	1.08	6.14	6.94	6.14	1.08	6.94	6.14	6.14
	4	3.68	30.16	28.72	3.56	4	33.56	3.72	30.36	4.48	31.32	5.1	31.32	42.14	31.32	5.1	42.14	31.32	31.32
	5	21.16	30.16	134.7	14.32	73.44	963.32	16.88	170.42	16.54	156.36	55.12	156.36	654.72	156.36	55.12	654.72	156.36	156.36
IBA	2	1.34	9.02	8.5	1.31	1.68	9.18	1.26	8.88	1.36	8.98	1.34	8.98	8.86	8.98	1.34	8.86	8.98	8.98
	3	2.38	16.36	16.44	2.32	2.24	16.34	2.44	16.6	2.36	16.54	2.36	16.54	16.38	16.54	2.36	16.38	16.54	16.54
	4	3.74	26.60	26.48	3.6	3.66	26.56	3.7	26.3	3.58	26.56	3.46	26.56	25.84	26.56	3.46	25.84	26.56	26.56
	5	6.88	38.62	39.2	5.3	7.16	52.48	6.38	40.14	5.36	37.48	5.66	37.48	41.08	37.48	5.66	41.08	37.48	37.48

TABLE 7: The percent of the best results for thresholds 4 and 5.

Alg.	Kapur's method	Otsu's method
PSO	8%	8%
DE	0%	0%
CS	67%	58%
FA	67%	92%
BA	42%	83%
IBA	100%	100%

TABLE 8: The number of evaluations for all test images and all threshold values for Kapur's method.

Alg.	Trsh. 2	Trsh. 3	Trsh. 4	Trsh. 5	Total
PSO	1214	411	3130	5439	10194
DE	96	186	1456	3356	5094
CS	1004	2183	3112	4891	11189
FA	70	176	347	932	1525
BA	876	1193	2784	3777	8631
IBA	74	116	192	352	734

TABLE 9: The number of evaluations for all test images and all threshold values for Otsu's method.

Alg.	Trsh. 2	Trsh. 3	Trsh. 4	Trsh. 5	Total
PSO	56	85	665	25206	3326
DE	95	294	1164	4245	5798
CS	1323	2322	3669	5665	12979
FA	72	171	230	889	1362
BA	10	40	196	2110	2356
IBA	53	99	158	249	559

method, the number of iterations does not grow rapidly with the increase of the number of thresholds as is the case with the rest of algorithms. From Table 8 we can also observe that the proposed IBA converges in considerably less iterations compared to the rest of algorithms.

From Table 9 (for the Otsu's criterion), it can be seen that the proposed IBA method in this case also converges in considerably less iterations compared to the other methods. It also maintains the feature of linearity with increasing the number of thresholds. Actually, in both cases, for Kapur's and Otsu's criteria, our proposed IBA algorithm improved the convergence speed by more than a factor of 2, compared to the next best algorithm.

6. Conclusion

In this paper, we considered an important optimization problem of multilevel image thresholding. It is an exponential problem and as such it is appropriate for swarm intelligence metaheuristics. We adapted new bat algorithm for this problem and compared it to other state-of-the-art algorithms from [49]. Pure version of the bat algorithm performed well, but the results were slightly below the average, especially when Kapur's criterion was used. We determined that the pure bat algorithm, when applied to this problem, may be easily

trapped into local optimum so we modified it by changing new solution equation by hybridized one with elements from DE. We also included limit parameter similar to the one used in the ABC algorithm.

Our proposed improved bat-inspired hybridized with DE (IBA) algorithm was tested on 6 standard benchmark images, the same as used in [49]. It proved to be superior to all other tested algorithms considering the quality of the solutions (it actually achieved the best result for both mean value and variance, for all tested cases), especially it significantly improved convergence speed (more than two times better than the next algorithm). This shows that our proposed algorithm is excellent choice for the multilevel image thresholding problem. Additional adjustments can be done in the future using larger set of synthetic images which will allow more precise modifications and parameter adjustment.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The research is supported by the Ministry of Science of Republic of Serbia, Grant no. III-44006.

References

- [1] J. Lázaro, J. L. Martín, J. Arias, A. Astarloa, and C. Cuadrado, "Neuro semantic thresholding using OCR software for high precision OCR applications," *Image and Vision Computing*, vol. 28, no. 4, pp. 571–578, 2010.
- [2] Y.-T. Hsiao, C.-L. Chuang, Y.-L. Lu, and J.A. Jiang, "Robust multiple objects tracking using image segmentation and trajectory estimation scheme in video frames," *Image and Vision Computing*, vol. 24, no. 10, pp. 1123–1136, 2006.
- [3] R. Adollah, M. Y. Mashor, H. Rosline, and N. H. Harun, "Multilevel thresholding as a simple segmentation technique in acute leukemia images," *Journal of Medical Imaging and Health Informatics*, vol. 2, no. 3, pp. 285–288, 2012.
- [4] A. Rojas Domínguez and A. K. Nandi, "Detection of masses in mammograms via statistically based enhancement, multilevel-thresholding segmentation, and region selection," *Computerized Medical Imaging and Graphics*, vol. 32, no. 4, pp. 304–315, 2008.
- [5] G. C. Anagnostopoulos, "SVM-based target recognition from synthetic aperture radar images using target region outline descriptors," *Nonlinear Analysis: Theory, Methods and Applications*, vol. 71, no. 12, pp. e2934–e2939, 2009.
- [6] N. R. Pal and S. K. Pal, "A review on image segmentation techniques," *Pattern Recognition*, vol. 26, no. 9, pp. 1277–1294, 1993.
- [7] N. Otsu, "A threshold selection method for grey level histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.
- [8] T. Pun, "A new method for grey-level picture thresholding using the entropy of the histogram," *Signal Processing*, vol. 2, no. 3, pp. 223–237, 1980.

- [9] T. Chaira and A. K. Ray, "Threshold selection using fuzzy set theory," *Pattern Recognition Letters*, vol. 25, no. 8, pp. 865–874, 2004.
- [10] T. Chaira and A. K. Ray, "Segmentation using fuzzy divergence," *Pattern Recognition Letters*, vol. 24, no. 12, pp. 1837–1844, 2003.
- [11] S. Wang, F. L. Chung, and F. Xiong, "A novel image thresholding method based on Parzen window estimate," *Pattern Recognition*, vol. 41, no. 1, pp. 117–129, 2008.
- [12] A. Nakib, H. Oulhadj, and P. Siarry, "Non-supervised image segmentation based on multiobjective optimization," *Pattern Recognition Letters*, vol. 29, no. 2, pp. 161–172, 2008.
- [13] S. S. Fan and Y. Lin, "A multi-level thresholding approach using a hybrid optimal estimation algorithm," *Pattern Recognition Letters*, vol. 28, no. 5, pp. 662–669, 2007.
- [14] E. Zahara, S. S. Fan, and D.-M. Tsai, "Optimal multi-thresholding using a hybrid optimization approach," *Pattern Recognition Letters*, vol. 26, no. 8, pp. 1082–1095, 2005.
- [15] M. Horng, "A multilevel image thresholding using the honey bee mating optimization," *Applied Mathematics and Computation*, vol. 215, no. 9, pp. 3302–3310, 2010.
- [16] J. N. Kapur, P. K. Sahoo, and A. K. C. Wong, "A new method for gray-level picture thresholding using the entropy of the histogram," *Computer Vision, Graphics, and Image Processing*, vol. 29, no. 3, pp. 273–285, 1985.
- [17] S. Zarezadeh and M. Asadi, "Results on residual Rényi entropy of order statistics and record values," *Information Sciences*, vol. 180, no. 21, pp. 4195–4206, 2010.
- [18] P. K. Sahoo, S. Soltani, and A. K. C. Wong, "A survey of thresholding techniques," *Computer Vision, Graphics and Image Processing*, vol. 41, no. 2, pp. 233–260, 1988.
- [19] X.-S. Yang, "Efficiency analysis of swarm intelligence and randomization techniques," *Journal of Computational and Theoretical Nanoscience*, vol. 9, no. 2, pp. 189–198, 2012.
- [20] X.-S. Yang, "Review of meta-heuristics and generalised evolutionary walk algorithm," *International Journal of Bio-Inspired Computation*, vol. 3, no. 2, pp. 77–84, 2011.
- [21] X.-S. Yang, "Free lunch or no free lunch: that is not just a question?" *International Journal on Artificial Intelligence Tools*, vol. 21, no. 3, Article ID 1240010, pp. 5360–5366, 2012.
- [22] A. H. Gandomi and X.-S. Yang, "Evolutionary boundary constraint handling scheme," *Neural Computing & Applications*, vol. 21, no. 6, pp. 1449–1462, 2012.
- [23] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks (ICNN '95)*, vol. 4, pp. 1942–1948, December 1995.
- [24] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
- [25] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications*, vol. 5792 of *Lecture Notes in Computer Science*, pp. 169–178, Springer, Berlin, Germany, 2009.
- [26] I. Fister, I. Fister Jr., X.-S. Yang, and J. Brest, "A comprehensive review of firefly algorithms," *Swarm and Evolutionary Computation*, vol. 13, no. 1, pp. 34–46, 2013.
- [27] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, Coimbatore, India, December 2009.
- [28] X.-S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343, 2010.
- [29] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," *Engineering with Computers*, vol. 29, no. 1, pp. 17–35, 2013.
- [30] M. Dorigo and L. M. Gambardella, "Ant colonies for the travelling salesman problem," *BioSystems*, vol. 43, no. 2, pp. 73–81, 1997.
- [31] M. Tuba and R. Jovanovic, "Improved ACO algorithm with pheromone correction strategy for the traveling salesman problem," *International Journal of Computers, Communications & Control*, vol. 8, no. 3, pp. 477–485, 2013.
- [32] R. Jovanovic and M. Tuba, "An ant colony optimization algorithm with improved pheromone correction strategy for the minimum weight vertex cover problem," *Applied Soft Computing Journal*, vol. 11, no. 8, pp. 5360–5366, 2011.
- [33] R. Jovanovic and M. Tuba, "Ant colony optimization algorithm with pheromone correction strategy for the minimum connected dominating set problem," *Computer Science and Information Systems*, vol. 10, no. 1, pp. 133–149, 2013.
- [34] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Tech. Rep. TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.
- [35] N. Bacanin and M. Tuba, "Artificial bee colony (ABC) algorithm for constrained optimization improved with genetic operators," *Studies in Informatics and Control*, vol. 21, no. 2, pp. 137–146, 2012.
- [36] I. Brajevic and M. Tuba, "An upgraded artificial bee colony (ABC) algorithm for constrained optimization problems," *Journal of Intelligent Manufacturing*, vol. 24, no. 4, pp. 729–740, 2013.
- [37] M. Subotic and M. Tuba, "Parallelized multiple swarm artificial bee colony algorithm (MS-ABC) for global optimization," *Studies in Informatics and Control*, vol. 23, no. 1, pp. 117–126, 2014.
- [38] M. Tuba and N. Bacanin, "Artificial bee colony algorithm hybridized with firefly metaheuristic for cardinality constrained mean-variance portfolio problem," *Applied Mathematics & Information Sciences*, vol. 8, no. 6, pp. 2831–2844, 2014.
- [39] X.-S. Yang, "A new metaheuristic bat-inspired Algorithm," *Studies in Computational Intelligence*, vol. 284, pp. 65–74, 2010.
- [40] A. Alihodzic and M. Tuba, "Improved hybridized bat algorithm for global numerical optimization," in *Proceedings of the 16th IEEE International Conference on Computer Modelling and Simulation (UKSim-AMSS '14)*, pp. 57–62, March 2014.
- [41] C. Dai, W. Chen, Y. Song, and Y. Zhu, "Seeker optimization algorithm: A novel stochastic search algorithm for global numerical optimization," *Journal of Systems Engineering and Electronics*, vol. 21, no. 2, pp. 300–311, 2010.
- [42] M. Tuba, I. Brajevic, and R. Jovanovic, "Hybrid seeker optimization algorithm for global optimization," *Applied Mathematics & Information Sciences*, vol. 7, no. 3, pp. 867–875, 2013.
- [43] M. Tuba and N. Bacanin, "Improved seeker optimization algorithm hybridized with firefly algorithm for constrained optimization problems," *Neurocomputing*, 2014.
- [44] S. Sarkar, G. R. Patra, and S. Das, "A differential evolution based approach for multilevel image segmentation using minimum cross entropy thresholding," in *Swarm, Evolutionary, and Memetic Computing*, vol. 7076 of *Lecture Notes in Computer Science*, pp. 51–58, 2011.

- [45] P. Yin, "Multilevel minimum cross entropy threshold selection based on particle swarm optimization," *Applied Mathematics and Computation*, vol. 184, no. 2, pp. 503–513, 2007.
- [46] B. Akay, "A study on particle swarm optimization and artificial bee colony algorithms for multilevel thresholding," *Applied Soft Computing Journal*, vol. 13, no. 6, pp. 3066–3091, 2013.
- [47] M. Maitra and A. Chatterjee, "A hybrid cooperative-comprehensive learning based PSO algorithm for image segmentation using multilevel thresholding," *Expert Systems with Applications*, vol. 34, no. 2, pp. 1341–1350, 2008.
- [48] K. Harnrnouche, M. Diaf, and P. Siarry, "A comparative study of various meta-heuristic techniques applied to the multilevel thresholding problem," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 5, pp. 676–688, 2010.
- [49] I. Brajevic and M. Tuba, "Cuckoo search and firefly algorithm applied to multilevel image thresholding," in *Cuckoo Search and Firefly Algorithm: Theory and Applications*, X.-S. Yang, Ed., vol. 516 of *Studies in Computational Intelligence*, pp. 115–139, Springer, Berlin, Germany, 2014.
- [50] D. Campos, "Real and spurious contributions for the Shannon, Rényi and Tsallis entropies," *Physica A*, vol. 389, no. 18, pp. 3761–3768, 2010.
- [51] M. Tuba, "Asymptotic behavior of the maximum entropy routing in computer networks," *Entropy*, vol. 15, no. 1, pp. 361–371, 2013.
- [52] G.-Q. Huang, W.-J. Zhao, and Q.-Q. Lu, "Bat algorithm with global convergence for solving large-scale optimization problem," *Application Research of Computers*, vol. 30, no. 5, pp. 1323–1328, 2013.
- [53] X.-S. Yang and A. H. Gandomi, "Bat algorithm: A novel approach for global engineering optimization," *Engineering Computations*, vol. 29, no. 5, pp. 464–483, 2012.
- [54] K. Khan, A. Nikov, and A. Sahai, "A fuzzy bat clustering method for ergonomic screening of office workplaces," in *Advances in Intelligent and Soft Computing*, vol. 101, pp. 59–66, Springer, 2011.
- [55] L. Jiann-Horng, C. Chao-Wei, Y. Chorng-Horng, and T. Hsien-Leing, "A chaotic levy ight bat algorithm for parameter estimation in nonlinear dynamic biological systems," *Journal of Computer and Information Technology*, vol. 2, no. 2, pp. 56–63, 2012.
- [56] X.-S. Yang, "Bat algorithm for multi-objective optimisation," *International Journal of Bio-Inspired Computation*, vol. 3, no. 5, pp. 267–274, 2011.
- [57] J. Zhang and G. Wang, "Image matching using a bat algorithm with mutation," *Applied Mechanics and Materials*, vol. 203, no. 1, pp. 88–93, 2012.
- [58] B. Ramesh, C. Jagan Mohan, and V. Reddy, "Application of bat algorithm for combined economic load and emission dispatch," *International Journal of Electrical and Electronics Engineering & Telecommunications*, vol. 2, no. 1, pp. 1–9, 2013.
- [59] F. A. Banu and C. Chandrasekar, "An optimized approach of modified BAT algorithm to record deduplication," *International Journal of Computer Applications*, vol. 62, no. 1, pp. 10–15, 2012.
- [60] M. K. Marichelvam and T. Prabaharam, "A bat algorithm for realistic hybrid flowshop scheduling problems to minimize makespan and mean flow time," *ICTACT International Journal on Soft Computing*, vol. 3, no. 1, pp. 428–433, 2012.
- [61] K. Khan and S. Ashok, "A comparison of ba, ga, pso, bp and lm for training feed forward neural networks in e-learning context," *International Journal of Intelligent Systems and Applications*, vol. 4, no. 7, pp. 23–29, 2011.
- [62] R. Damodaram and M. L. Va larmathi, "Phishing website detection and optimization using modified bat algorithm," *International Journal of Engineering Research and Applications*, vol. 2, no. 1, pp. 870–876, 2012.

Research Article

Focusing on the Golden Ball Metaheuristic: An Extended Study on a Wider Set of Problems

E. Osaba, F. Diaz, R. Carballedo, E. Onieva, and A. Perallos

Deusto Institute of Technology (DeustoTech), University of Deusto, Avenida Universidades 24, 48007 Bilbao, Spain

Correspondence should be addressed to E. Osaba; e.osaba@deusto.es

Received 16 April 2014; Revised 6 June 2014; Accepted 8 June 2014; Published 3 August 2014

Academic Editor: Xin-She Yang

Copyright © 2014 E. Osaba et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Nowadays, the development of new metaheuristics for solving optimization problems is a topic of interest in the scientific community. In the literature, a large number of techniques of this kind can be found. Anyway, there are many recently proposed techniques, such as the artificial bee colony and imperialist competitive algorithm. This paper is focused on one recently published technique, the one called Golden Ball (GB). The GB is a multiple-population metaheuristic based on soccer concepts. Although it was designed to solve combinatorial optimization problems, until now, it has only been tested with two simple routing problems: the traveling salesman problem and the capacitated vehicle routing problem. In this paper, the GB is applied to four different combinatorial optimization problems. Two of them are routing problems, which are more complex than the previously used ones: the asymmetric traveling salesman problem and the vehicle routing problem with backhauls. Additionally, one constraint satisfaction problem (the n -queen problem) and one combinatorial design problem (the one-dimensional bin packing problem) have also been used. The outcomes obtained by GB are compared with the ones got by two different genetic algorithms and two distributed genetic algorithms. Additionally, two statistical tests are conducted to compare these results.

1. Introduction

Today, optimization problems receive much attention in artificial intelligence. There are several types of optimization, such as numerical [1], linear [2], continuous [3], or combinatorial optimization [4]. Typically, problems arising in these fields are of high complexity. Additionally, many of the problems arising in optimization are applicable to the real world. For these reasons, in the literature, many different techniques designed to be applied to these problems can be found.

Some classical examples of these techniques are the simulated annealing [5], the tabu search [6], the genetic algorithm (GA) [7, 8], ant colony optimization [9], or the particle swarm optimization [10]. Since their proposal, all these metaheuristics have been widely applied in a large amount of fields. In fact, these techniques are the focus of many research studies nowadays [11–14].

Despite the existence of these conventional algorithms, the development of new metaheuristics for solving optimization problems is a topic of interest in the scientific community. On the one hand, optimization problems represent

a great challenge because they are hard to solve. For this reason, the development of new techniques that outperform the existing ones is a topic of interest for the researchers. On the other hand, optimization problems (such as routing problems) are very important from a social perspective. This is because their resolution directly affects the economy and sustainability in terms of cost reduction and energy saving.

In this way, there have been many recently proposed metaheuristics, which have been successfully applied to various fields and problems. One example is the imperialist competitive algorithm (ICA) [15]. This population based metaheuristic, proposed by Gargari and Lucas in 2007, is based on the concept of imperialisms. In ICA, individuals are called countries and they are divided into various empires, which evolve independently. Throughout the execution, different empires battle each other with the aim of conquering their colonies. When one empire conquers all the colonies, the algorithm converges into the final solution. Some examples of its application can be seen in recent papers [16, 17]. Another example is the artificial bee colony. This technique was proposed in 2005 by Karaboga [18, 19] for

multimodal and multidimensional numeric problems. Since then, it has been used frequently in the literature for solving different kinds of problems [20–22]. The artificial bee colony is a swarm based technique which emulates the foraging behaviour of honey bees. The population of this metaheuristic consists in a colony, with three kinds of bees: employed, onlooker, and scout bees, each with a different behaviour. The harmony search, presented by Geem et al. in 2001, is another example [23, 24]. This metaheuristic mimics the improvisation of music players. In this case, each musical instrument corresponds to a decision variable; a musical note is the value of a variable; and the harmony represents a solution vector. With the intention of imitating the musicians in a jam session, variables have random values or previously memorized good values in order to find the optimal solution. This algorithm is also used frequently in the literature [25–27].

Bat-inspired algorithm is a more recent technique [28, 29]. This metaheuristic, proposed by Yang in 2010, is based on the echolocation behaviour of microbats, which can find their prey and discriminate different kinds of insects even in complete darkness. Yang and Deb proposed the cuckoo search algorithm in 2009 [30, 31]. On this occasion, as authors claim in [30], this metaheuristic is based on the obligate brood parasitic behaviour of some cuckoo species in combination with the Levy flight behaviour of some birds and fruit flies. Another recently developed technique which is very popular today is the firefly algorithm [32, 33]. This nature-inspired algorithm is based on the flashing behaviour of fireflies, which act as a signal system to attract other fireflies. Like the aforementioned techniques, these metaheuristics have been the focus of several research [34–40] and review papers [41–43].

As can be seen, there are many metaheuristics in the literature to solve optimization problems. Although several techniques have been mentioned, many other recently developed ones could be cited, such as the spider monkey optimization [44] or seeker optimization algorithm [45]. This large amount of existing techniques demonstrates the growing interest in this field, on which several books, special issues in journals, and conferences proceedings are published annually. Moreover, combinatorial optimization is a widely studied field in artificial intelligence nowadays. Being NP-Hard [46], a lot of problems arising in this field are particularly interesting for the researchers. This kind of optimization is the subject of a large number of works every year [47–49]. This scientific interest is the reason why this study is focused on this sort of optimization.

This paper is focused on one recently proposed metaheuristic called Golden Ball (GB). This technique is a multiple-population based metaheuristic, and it is based on soccer concepts. A preliminary version of the GB and some basic results were firstly introduced in 2013 by Osaba et al. [50]. Furthermore, the final version of the GB and its practical use for solving complex problems have been presented this very year (2014) by the same authors [51]. In that paper, the GB is introduced, and it is compared with some similar metaheuristics of the literature. In addition, it is successfully applied to two different routing problems: the traveling salesman problem (TSP) [52] and the capacitated vehicle routing

problem (CVRP) [53]. Additionally, the results obtained by GB were compared with the ones obtained by two different GAs and two distributed genetic algorithms (DGA) [54, 55]. As a conclusion of that study, it can be said that the GB outperforms these four algorithms when it is applied to the TSP and CVRP.

The authors of that study claim that GB is a technique to solve combinatorial optimization problems. Even so, they only prove its success with two simple routing problems, the TSP and the CVRP. This is the reason that motivates the work presented in this paper. Thus, the objective of this paper is to verify if the GB is a promising metaheuristic to solve combinatorial optimization problems, performing a more comprehensive and rigorous experimentation than that presented to date. Thereby, in this research study, the GB is applied to four different combinatorial optimization problems. Two of them are routing problems, which are more complex than the ones used in [51]: the asymmetric traveling salesman problem (ATSP) [56] and the vehicle routing problem with backhauls (VRPB) [57]. Furthermore, in order to verify that the GB is also applicable to other types of problems apart from the routing ones, two additional problems have also been used in the experimentation, the n-queen problem (NQP) [58] and the one-dimensional bin packing problem (BPP) [59]. As in [51], the results obtained by GB are compared with the ones obtained by two different GAs and two DGAs. Besides, with the objective of performing a rigorous comparison, two statistical tests are conducted to compare these outcomes: the well-known normal distribution z -test and the Friedman test.

The rest of the paper is structured as follows. In Section 2, the GB is introduced. In Section 3, the problems used in the experimentation are described. Then, in Section 4, the experimentation conducted is described. In Section 5, the results obtained are shown and the statistical tests are performed. This work finishes with the conclusions and future work (Section 6).

2. Golden Ball Metaheuristic

In this section, the GB is described. As has been mentioned in Section 1, the GB is a multiple-population based metaheuristic which takes several concepts related to soccer. To begin with, the technique starts with the initialization phase (Section 2.1). In this first phase, the whole population of solutions (called players) is created. Then, these players are divided among the different subpopulations (called teams). Each team has its own training method (or coach). Once this initial phase has been completed, the competition phase begins (Section 2.2). This second phase is divided in seasons. Each season is composed of weeks, in which the teams train independently and face each other creating a league competition. At the end of every season, a transfer procedure happens, in which the players and coaches can switch teams. The competition phase is repeated until the termination criterion is met (Section 2.3). The entire procedure of the technique can be seen in Figure 1. Now, the different steps that form the proposed technique are explained in detail.

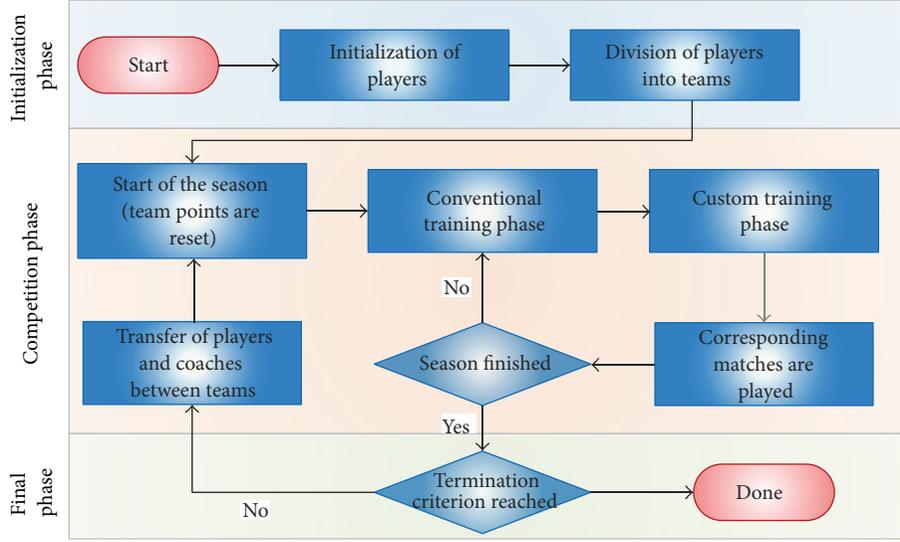


FIGURE 1: Flowchart of GB metaheuristic.

2.1. *Initialization Phase.* As has been said, the first step of the execution is the creation of the initial population P . The initial population is composed of $PT \cdot TN$ number of solutions p_i , called *players*. Note that PT is the number of players per team, and TN is the number of teams. Additionally, both parameters must have a value higher than 1.

After the whole population P is created, all the p_i are randomly divided in the TN different teams t_i . Once the players are divided between the different teams, they are represented by the variable p_{ij} , which means *the player number j of the team i* . The total set of teams T forms the league. All these concepts may be represented mathematically as follows:

$$\begin{aligned}
 P &: \{p_1, p_2, p_3, p_4, p_5, \dots, p_{PT \cdot TN}\} \\
 T &: \{t_1, t_2, t_3, t_4, \dots, t_{TN}\} \\
 \text{Team } t_1 &: \{p_{11}, p_{12}, p_{13}, \dots, p_{1PT}\} \\
 \text{Team } t_2 &: \{p_{21}, p_{22}, p_{23}, \dots, p_{2PT}\} \\
 &\vdots \\
 \text{Team } t_{TN} &: \{p_{TN1}, p_{TN2}, \dots, p_{TNPT}\} \\
 PT &= \text{Number of players per team,} \\
 TN &= \text{Total number of teams of the system.}
 \end{aligned} \tag{1}$$

Furthermore, every p_{ij} has its own quality, which is represented by the variable q_{ij} (quality of the player i of team j). This variable is represented by a real number, which is determined by a cost function $f(p_{ij})$. This function depends on the problem. For example, for some routing problems, this function is equivalent to the traveled distance. On the other hand, for the NQP, for instance, this function is the number of collisions. In addition, each t_i has a captain (p_{icap}), which

is the player with the best q_{ij} of their team. To state this in a formal way, consider

$$p_{icap} = p_{ik} \in t_i \iff \forall j \in \{1, \dots, PT\} : q_{ik} \geq q_{ij}. \tag{2}$$

It should be borne in mind that, depending on the problem characteristics, the objective is to minimize or maximize $f(p_{ij})$. In the problems used in this paper, for example, the lower the q_{ij} is, the better the player is.

Moreover, each team has a strength value associated with TQ_i . This value is crucial for the matches between teams (Section 2.2.2). It is logical to think that the better the players are, the stronger a team is. Thereby, if one team is strong, it can win more matches and it can be better positioned in the classification of the league. In this way, the strength value of a team t_i is equal to the average of the q_{ij} of the players of that team. TQ_i can be expressed by the following formula:

$$TQ_i = \frac{\sum_{j=1}^{PT} q_{ij}}{PT}. \tag{3}$$

Once the initialization phase is completed, the competition phase begins. This phase is repeated iteratively until the ending criterion is met.

2.2. *Competition Phase.* This is the central phase of the metaheuristic. In this stage, each team evolves independently and improves its TQ_i (Section 2.2.1). Additionally, in this phase, the teams face each other, creating a league competition (Section 2.2.2). This league helps to decide the player transfers between teams (Section 2.2.3). The competition stage is divided into seasons (S_i). Each S_i has two different periods of player transfers. In each season, every team face each other twice. In this way, each team plays $2NT-2$ matches in a season. Lastly, an amount of training sessions equal to the number of matches played is performed.

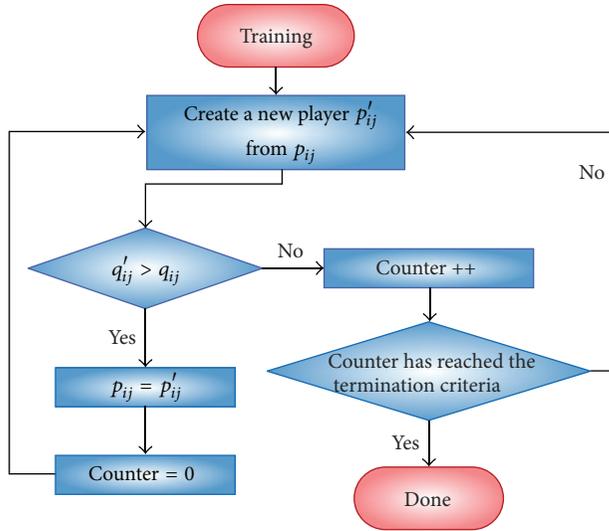


FIGURE 2: Workflow of a training process.

2.2.1. *Training of Players.* As in real life, trainings are the processes that make players improve their quality. In GB, each t_i has its own training method, and some of them are more effective than others. This fact makes some teams improve more than others. There are two kinds of training methods in GB: *conventional trainings* and *custom trainings*.

Conventional trainings are those that are performed regularly throughout the season. This type of training is applied individually for each player. A training method is a successor function, which works on a particular neighborhood structure in the solution space. Taking the TSP as example, one training function could be the 2-opt [60]. As has been said, each team has its own training function, which acts as the coach of the team. The training function is assigned randomly at the initialization phase. Thereby, each p_{ij} uses the method of its team. For each training session, the function is applied a certain number of times, and the p'_{ij} generated is accepted if $q'_{ij} > q_{ij}$. Besides, this process could make a change in the p_{icap} of a team, if one p_{ij} outperforms the quality of its captain.

It is worth mentioning that one training session has its own termination criterion. A training session ends when there is a number of successors without improvement in the quality of the p_{ij} trained. This number is proportional to the neighborhood of the team training function. For example, taking the 2-opt and a 30-noded TSP instance, the training ends when there are $n/4 + \sum_{k=1}^{n/4} k$ (the size of its neighborhood) successors without improvement, with n being the size of the problem (30). Figure 2 schematizes this process.

Furthermore, the fact that every t_i explores the space solution in a different way increases the exploration and exploitation capacity of the GB. This fact occurs because of the use of a different training method for each team. Besides, this is enhanced by the fact that players can change their teams.

On the other hand, the procedure of *custom trainings* is different. These trainings are performed when one p_{ij} receives a number of conventional training sessions without experiencing any improvement (in this study, this number

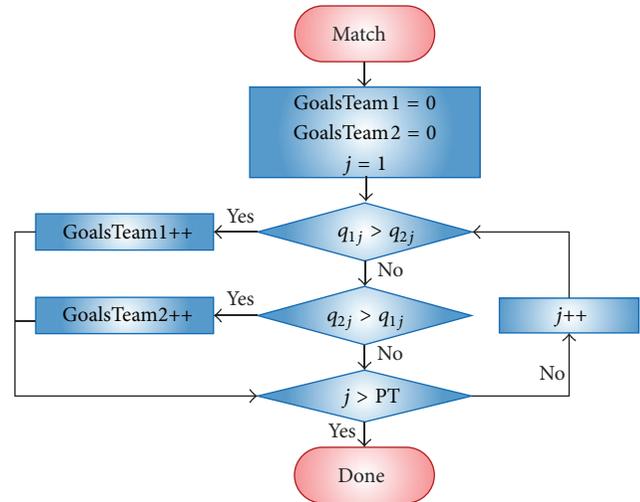


FIGURE 3: Flowchart of a match.

has been set in $PT/2$). When this fact happens, it can be considered that p_{ij} is trapped in a local optimum. A *custom training* is conducted by two players: the trapped p_{ij} and the p_{icap} of their team. The purpose of these operations is to help p_{ij} to escape from the local optimum and to redirect them to another promising region of the solution space. From a practical point of view, custom training combines two players (like the crossover operator of a GA), resulting in a new player p'_{ij} who replaces p_{ij} . Taking the TSP as example, a function that combines the characteristics of two players could be the order crossover (OX) [61] or the partially mapped crossover [62]. The custom training helps a thorough exploration of the solution space.

2.2.2. *Matches between Teams.* In GB, as in the real world, two teams (t_j, t_k) are involved in a match. Each match consists in PT goal chances, which are resolved in the following way: first, the players of both teams are sorted by quality (in descending order). Then, each p_{ij} faces p_{ik} . The player with the highest quality wins the goal chance, and their team scores a goal. As can be surmised, the team that achieves more goals wins the match. Furthermore, the team that wins the match obtains 3 points and the loser obtains 0 points. If both teams obtain the same number of goals, each one receives one point. These points are used to perform a team classification, sorting the teams by the points obtained in a whole season. Figure 3 shows the flowchart of a match.

2.2.3. *Player Transfers between Teams.* The transfers are the processes in which the teams exchange their players. There are two types of transfers in GB: *season transfers* and *special transfers*. The former are the conventional ones, and they take place twice in a S_i . In these transfers, the points obtained by each team and its position in the league are crucial factors. In this way, in the middle and the end of each S_i , the teams placed in the top half of the league classification “hire” the best players of the teams located on the lower half. Moreover, teams of the bottom half receive in return the worst players

of the top teams. In addition, the better the position of the team is, the better the player it receives is. In other words, the best t_i is reinforced with the best player of the worst team of the league. Furthermore, the second t_i obtains the second best p_{ij} of the penultimate team, and so on. Finally, if the league has an odd number of teams, the team placed in the middle position does not exchange any p_{ij} .

On the other hand, the *special transfers* are sporadic exchanges that can occur at any time during a season. If one player receives a certain number of conventional and custom trainings without experiencing any improvement, they changes their team (in this study, this number has been set in PT conventional training sessions without improvement). This change is made in order to obtain a different kind of training. Besides, it is no matter whether the destination team has less TQ_i than its current team. Additionally, with the aim of maintaining the PT per team, there is an exchange with a random player of the destination team.

As authors said in [51], these interchanges help the search process. This neighborhood changing improves the exploration capacity of the technique.

Lastly, it is noteworthy that another sort of transfer exists in GB. In this case, these transfers are not performed with players, but with team coaches. This process has been called *cessation of coaches*. In each period of *season transfers*, the teams positioned in the bottom half of the league classification change their training form. This change is made hoping to get another kind of training which improves the performance and the TQ_i of the team. This training exchange is performed randomly among all the training types existing in the system, allowing repetitions between different t_i . This random neighborhood change increases the exploration capacity of the metaheuristic.

2.3. Termination Criterion. The termination criterion is a critical factor in the development of a metaheuristic. It must be taken into account that this criterion has to allow the search to examine a wide area of the solution space. On the other hand, if it is not strict enough, it can lead to a considerable waste of time. In this way, the termination criterion of the GB is composed of three clauses:

$$\begin{aligned} \sum_{i=1}^{TN} q'_{icap} &\leq \sum_{i=1}^{TN} q_{icap}, \\ \sum_{i=1}^{TN} TQ'_i &\leq \sum_{i=1}^{TN} TQ_i, \\ BestSol' &\leq BestSol. \end{aligned} \tag{4}$$

In other words, the execution of the GB finishes when (1) the sum of the quality q'_{icap} of all the captains is not improved over the previous season, (2) the sum of the strengths TQ'_i of all the teams has not been improved compared to the previous season, and (3) there is no improvement in the best solution found ($BestSol'$) in relation to the previous season. When these three conditions are fulfilled, the p_{ij} with the best q_{ij} of the system is returned as the final solution.

3. Description of the Used Problems

As has been mentioned in the introduction of this study, four combinatorial optimization problems have been used in the experimentation conducted. In this section, these problems are described. The first two are routing problems: the ATSP (Section 3.1) and the VRPB (Section 3.2). Besides, with the aim of verifying whether the GB is also a promising technique with other kinds of problems apart from the routing ones, the NQP (Section 3.3) and the BPP (Section 3.4) have been used.

It is important to highlight that the objective of the present paper is not to find an optimal solution to these problems. In fact, in the literature, there are multiple efficient techniques with this objective. Instead, these four problems have been used as benchmarking problems. In this way, the objective of using them is to compare the performance of the GB with the one of the GAs and DGAs and to conclude which obtains better results using the same parameters and functions.

3.1. Asymmetric Traveling Salesman Problem. As the symmetric version of this problem (the TSP), the ATSP has a great scientific interest, and it has been used in many research studies since its formulation [63, 64]. This problem can be defined as a complete graph $G = (V, A)$, where $V = \{v_1, v_2, \dots, v_n\}$ is the set of vertexes which represents the nodes of the system and $A = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ is the set of arcs which represents the connection between nodes. Additionally, each arc has an associated distance cost d_{ij} . Unlike in the TSP, in the ATSP, the distance cost between two nodes is different depending on the direction of the flow; that is, $d_{ij} \neq d_{ji}$. Thereby, the objective of the ATSP is to find a route that, starting and finishing at the same node, visits every v_i once and minimizes the total distance traveled. In this way, the objective function is the total distance of the route.

In this study, the solutions for the ATSP are encoded using the permutation representation [65]. According to this encoding, each solution is represented by a permutation of numbers, which represents the order in which the nodes are visited. For example, for a possible 10-node instance, one feasible solution would be encoded as $X = (0, 5, 2, 4, 3, 1, 6, 8, 9, 7)$, and its fitness would be $f(X) = d_{05} + d_{52} + d_{24} + d_{43} + d_{31} + d_{16} + d_{68} + d_{89} + d_{97} + d_{70}$. This situation is depicted in Figure 4.

3.2. Vehicle Routing Problem with Backhauls. The VRPB is a variant of the basic VRP where customers can demand either a delivery or a pick-up of certain goods [57]. In this problem, all deliveries are done before the pick-ups. This is so because, otherwise, it could be a movement of material within the mobile unit that could be counterproductive, for example, putting collected materials on the front of the trunk, whereas at the bottom some goods remain undelivered. The VRPB is widely used in the literature thanks to its applicability to the real world and to its solving complexity [66–68].

The VRPB can be defined as a complete graph $G = (V, A)$, where $V = \{v_0, v_1, \dots, v_n\}$ is the set of vertexes and $A = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$ is the set of arcs.

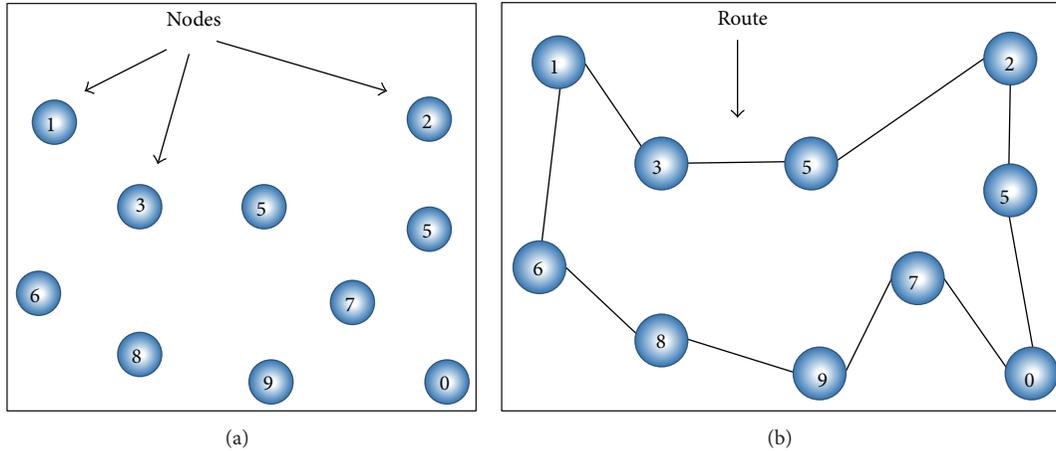


FIGURE 4: Example of a 10-node ATSP instance and a possible solution.

The vertex v_0 represents the depot, and the rest are the customers. Besides, in order to facilitate the formulation, the set of customers V can be separated into two subsets [69]. The first one, L , called *linehaul customers*, contains those users who demand the delivery of goods. On the other hand, the second subset, B , called *backhaul customers*, demand the pick-up of a certain amount of material. To express customer demand (q_i), positive values are used for linehaul customers and negative values for backhaul ones.

Additionally, a fleet of K vehicles is available with a limited capacity Q . The objective of the VRPB is to find a number of routes with a minimum cost such that (i) each route starts and ends at the depot, (ii) each client is visited exactly by one route, (iii) all deliveries are made before pick-ups, and (iv) the total demand of the customers visited by one route does not exceed the total capacity of the vehicle that performs it.

Finally, the permutation representation is also used for this problem [70], and the routes are also encoded as nodes permutation. In addition, to distinguish the different routes in a solution, they are separated by zeros. As an example, suppose a set of five linehaul customers $L = \{L1, L2, L3, L4, L5\}$ and seven backhaul customers $B = \{B1, B2, B3, B4, B5, B6, B7\}$. One possible solution with three vehicles would be $X = (L5, L3, B1, B6, 0, L4, L1, B3, B7, 0, L2, B4, B5, B2)$ and its fitness would be $f(X) = d_{0L5} + d_{L5L3} + d_{L3B1} + d_{B1B6} + d_{B60} + d_{0L4} + d_{L4L1} + d_{L1B3} + d_{B3B7} + d_{B70} + d_{0L2} + d_{L2B4} + d_{B4B5} + d_{B5B2} + d_{B20}$. In Figure 5(a), an example of a VRPB instance is depicted. Furthermore, in Figure 5(b), a possible solution for this instance is shown.

3.3. *n*-Queen Problem. The NQP is a generalization of the problem of putting eight nonattacking queens on a chessboard [71], which was introduced by Bezzel in 1848 [72]. The NQP consists in placing N queens on a $N \times N$ chessboard, in order that they cannot attack each other. This problem is a classical combinatorial design problem (constraint satisfaction problem), which can also be formulated as a combinatorial optimization problem [73]. In this paper, the NQP has been formulated as a combinatorial optimization problem,

where a solution X is coded as an N -tuple (q_1, q_2, \dots, q_n) , which is a permutation of the set $(1, 2, \dots, N)$. Each q_i represents the row occupied by the queen positioned in the i th column. Using this representation, vertical and horizontal collisions are avoided, and the complexity of the problem becomes $O(N!)$. Thereby, the fitness function is defined as the number of diagonal collisions along the board. Notice that i th and j th queens collide diagonally if

$$|i - q_i| = |j - q_j| \quad \forall i, j : \{1, 2, \dots, N\}; i \neq j. \quad (5)$$

In this way, the objective of NQP is to minimize the number of conflicts, the ideal fitness being zero. A possible solution for an 8-queen chessboard is depicted in Figure 6. According to the encoding explained, the solution represented in this figure would be encoded as $f(X) = (4, 3, 1, 6, 5, 8, 2, 7)$. Additionally, its fitness would be 3, since there are three diagonal collisions (4-3, 6-5, and 6-8). This same formulation has been used before in the literature [74, 75].

3.4. One-Dimensional Bin Packing Problem. The packing of items into boxes or bins is a daily task in distribution and production. Depending on the item characteristics, as well as the form and capacity of the bins, a wide amount of different packing problems can be formulated. In [59], an introduction to bin-packing problems can be found. The BPP is the simplest one, and it has been frequently used as a benchmarking problem [76–78]. The BPP consists of a set of items $I = (i_1, i_2, \dots, i_n)$, each with an associated size s_i and an unlimited supply of bins with the same capacity q . The objective of the BPP is to pack all the items into a minimum number of bins. In this way, the objective function is the number of bins, which has to be minimized.

In this study, the solutions are encoded as a permutation of items. To count the number of bins needed in one solution, the item sizes are accumulated in a variable (*sumSize*). When *sumSize* exceeds q , the number of bins is incremented in 1, and *sumSize* is reset to 0. Thereby, we suppose a simple instance of 9 items $I = \{i_1, i_2, \dots, i_9\}$, three different sizes $s_{1-3} = 20$, $s_{4-6} = 30$, and $s_{7-9} = 50$, and $q = 100$. One

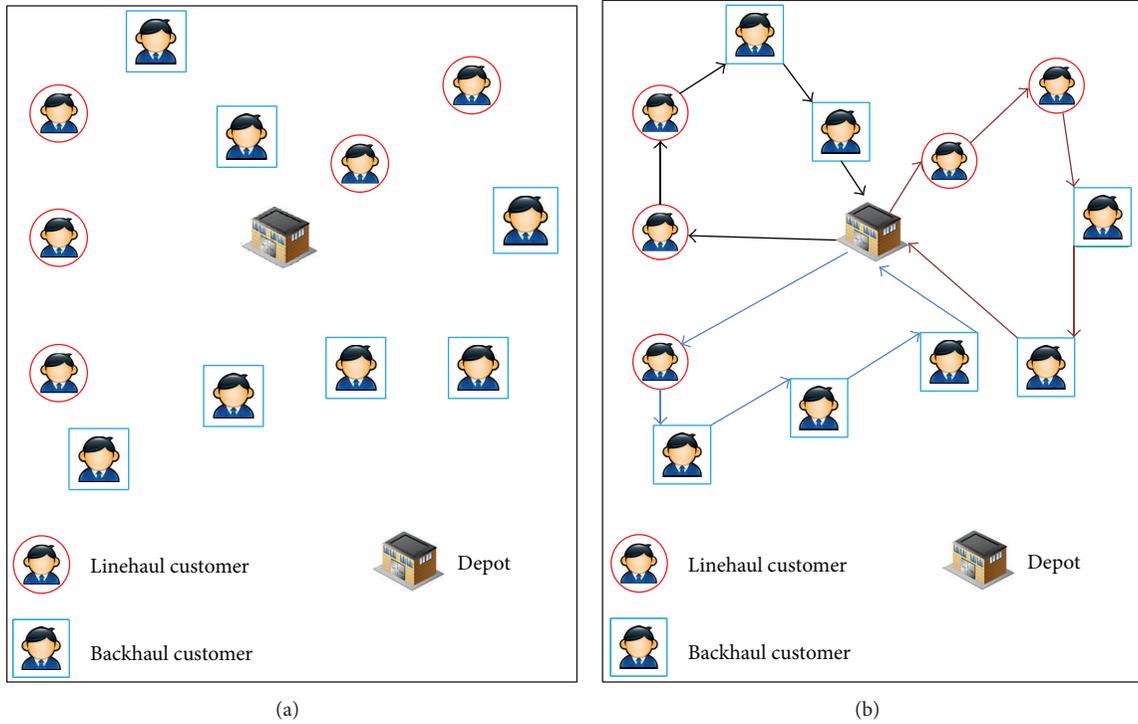


FIGURE 5: Example of a VRPB instance and a possible solution.

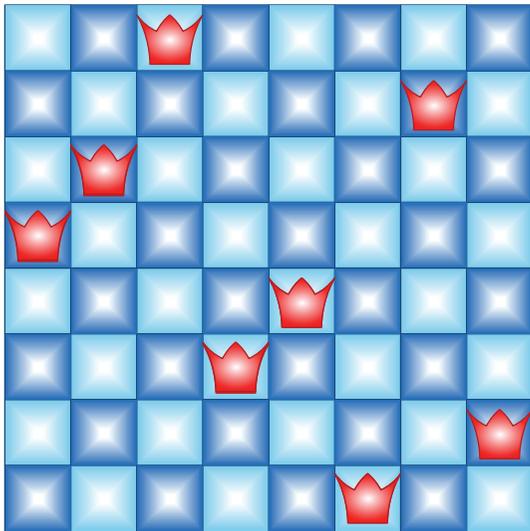


FIGURE 6: Example of an 8×8 instance for the NQP.

possible solution could be $X = (i_1, i_4, i_7, i_2, i_5, i_8, i_3, i_6, i_9)$, and its fitness would be 3 (the number of bins needed to hold all the items). This example is represented in Figure 7.

4. Experimentation Setup

In this section, the experimentation performed is described. According to the study carried out in [51], the GB metaheuristic provides some originality regarding the well-known

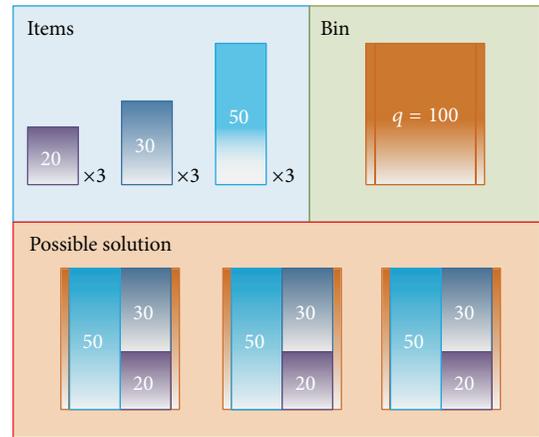


FIGURE 7: Example of a BPP instance and a possible solution.

techniques that can be found today in the literature. In line with this, analyzing the philosophy and the working way of GB, it can be concluded that the DGA is the technique which shares the most similarities with it. Among other similarities, in the evolution of their individuals, these metaheuristics rely on two operators, a local and a cooperative one, which are used for the exploitation and exploration. In addition, these techniques are easy to apply to combinatorial optimization problems.

For these reasons, to prove the quality of the GB, two single-population GAs and two DGAs are used for the experimentation. The general characteristics of these four

```

(1) Initialization of the population
(2) while termination criterion not reached do
(3)   Parents selection
(4)   Crossover phase
(5)   Mutation phase
(6)   Survivors selection
(7) end
(8) Return the best individual found

```

ALGORITHM 1: Pseudocode of GA_1 and GA_2 .

```

(1) Initialization of the subpopulations
(2) while termination criterion not reached do
(3)   for each subpopulation do
(4)     Parents selection
(5)     Crossover phase
(6)     Mutation phase
(7)     Survivors selection
(8)   end
(9)   Individual migration phase
(10) end
(11) Return the best individual found

```

ALGORITHM 2: Pseudocode of DGA_1 and DGA_2 .

techniques are explained in Section 4.1. In addition, the parametrization of all the algorithms is described in the same section. The details of the experimentation are introduced in Section 4.2.

4.1. General Description of Developed Techniques. As has been mentioned, the outcomes obtained by the GB are compared with the ones obtained by two basic single-population GAs (GA_1 and GA_2) and two different DGAs (DGA_1 and DGA_2). The structure used for both GAs is the one represented in Algorithm 1, and it is considered the conventional one. Furthermore, in Algorithm 2, the structure of both DGAs is depicted, which is also the conventional one.

On one hand, for GA_1 and DGA_1 conventional operators and parameters have been used, that is, a high crossover probability and a low mutation probability. These concepts are based on the concepts outlined in many previous studies [54, 79, 80]. On the other hand, for GA_2 and DGA_2 , parameters have been adjusted to be similar to those used for the GB. Thereby, the numbers of cooperative movements (crossovers and custom trainings) and individual movements (mutations and conventional trainings) performed are the same. In addition, the same functions have been used for GA_2 , DGA_2 , and GB. In this way, the only difference between them is the structure. Thereby, it can be deduced which algorithm is the one that obtains better results, using the same operators for the same number of times.

The population size used for each metaheuristic is 48. All the initial solutions have been generated randomly. For DGA_1 and DGA_2 , this population has been divided into

4 subpopulations of 12 individuals. For the GB, the whole population is also divided into 4 teams of 12 players each. The crossover probability (p_c) and mutation probability (p_m) of the GA_1 are 95% and 5%, respectively. On the other hand, different p_c and p_m have been used for every subpopulation of DGA_1 . For p_c , 95%, 90%, 80%, and 75% have been utilized, and for p_m , 5%, 10%, 20%, and 25% have been utilized. At last, for GA_2 and DGA_2 , $p_c = 0.003\%$ and $p_m = 100\%$ have been used, in order to fit with the GB parameters.

In relation to the parents selection criteria for the GAs and DGAs, first, each individual of the population is selected as parent with a probability equal to p_c . If one individual is selected for the crossover, the other parent is selected randomly. Regarding the survivor function, a 100% elitist function has been developed for GA_2 and DGA_2 , and a 50% elitist random (which means that the half of the survivor population is composed of the best individuals, and the remaining ones are selected randomly) has been developed for GA_1 and DGA_1 . In DGA_1 and DGA_2 , the classic best-replace-worst migration strategy has been used. In this strategy, every subpopulation i shares its best individual with the following $i + 1$ deme, in a ring topology. This communication happens every generation and the immigrant replaces the worst individual of deme $i + 1$. Ultimately, the execution of both GAs and DGAs finishes when there are $n + \sum_{k=1}^n k$ generations without improvements in the best solution, where n is the size of the problem. The problem size is the number of customers in the two routing problems, the number of queens in the NQP and the number of items in the BPP.

The successor functions employed by GB as conventional training functions for the ATSP, NQP, and BPP are the following.

- (i) 2-opt and 3-opt: these functions, proposed by Lin for the TSP [60], have been used widely in the literature [81–83]. These operators eliminate at random 2 (for the 2-opt) and 3 (for the 3-opt) arcs of the solution, and they create two or three new arcs, avoiding the generation of cycles.
- (ii) Insertion function: this operator selects and extracts one random node of a solution, and it inserts it in another random position. Because of its easy implementation and its good performance, this function is often used in the literature for any kind of permutation encoded problem [84, 85].
- (iii) Swapping function: this well-known function is also widely employed in lots of research studies [86]. In this case, two nodes of a solution are selected randomly, and they swap their position.

These successor functions have also been used as mutation functions for the DGA_2 (a different function for each subpopulation). On the other hand, for the GA_1 , GA_2 , and DGA_1 , the 2-opt has been utilized for this purpose, since it is the one that gets the best results.

For the same problems (ATSP, NQP, and BPP), GB uses the half crossover (HX) operator [81] as a custom training function. This operator is a particular case of the traditional

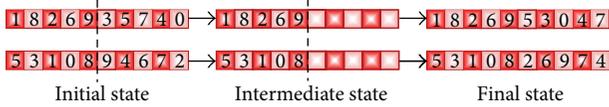


FIGURE 8: Example of HX with a 10-node instance.

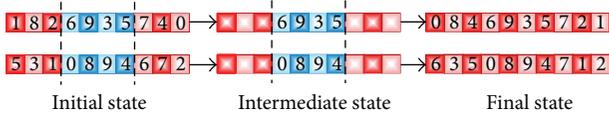


FIGURE 9: Example of OX with a 10-node instance.

one-point crossover, in which the cut-point is made always in the middle of the solution. Assuming a 10-node instance, in Figure 8, an example of this function can be seen. This function has been used as a crossover operator for the GA₂ and DGA₂. On the other hand, for the GA₁ and DGA₁, the well-known order crossover (OX) [61] has been implemented as a crossover function. In Figure 9, an example of the OX is shown. Finally, in Table 1, a summary of the characteristics of both GAs and DGAs is depicted.

Regarding the VRPB, *2-opt* and *Insertion* functions are also used as conventional training functions. These operators, as Savelsbergh called them [87], are intraroute functions, which means that they work within a specific route. Additionally, two interroute functions have been developed.

- (i) Insertion Routes: this function selects and extracts one random node from a random route. After that, this node is reinserted in a random position in another randomly selected route. This function could create new routes.
- (ii) Swapping Routes: this operator selects randomly two nodes of two randomly selected routes. These nodes are swapped.

It is noteworthy that all these functions take into account both the vehicles capacity and the class of the nodes' demands, never making infeasible solutions. As in the previous problems, these same operators have been also developed as mutation functions for the DGA₂ (a different operator for each subpopulation). Moreover, *Insertion Routes* operator has been used for the same purpose in GA₁, GA₂, and DGA₁.

For the VRPB, the half route crossover (HRX) has been used as custom training. This function has been used recently in several studies [49, 85], and it operates as follows: first, half of the routes of one parent are directly inserted into the child. Then, the nodes that remain to be inserted are added in the same order in which they appear in the other parent. As the above functions, the HRX takes into account the VRPB constraints, and it does not generate infeasible solutions. In Figure 10, an example of the HRX procedure in a 20-node instance is shown. Additionally, in Table 2, a summary of the characteristics of both GAs and DGAs for VRPB is shown. Finally, the features of the GB for the four problems are depicted in Table 3.

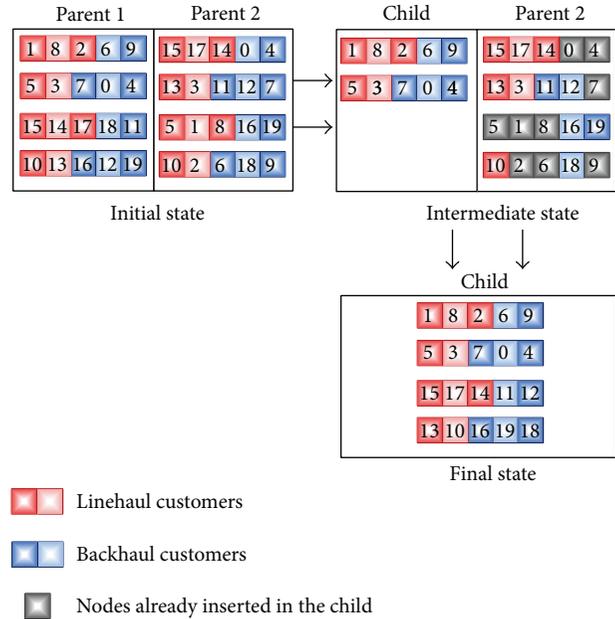


FIGURE 10: A possible example of HRX with a 20-node instance.

4.2. Description of the Experimentation. In this section, the basic aspects of the experimentation are detailed. First, all the tests have been run on an Intel Core i7 3930 computer, with 3.20 GHz and a RAM of 16 GB. Microsoft Windows 7 has been used as OS. All the techniques were coded in Java. For the ATSP, 19 instances have been employed, which have been obtained from the TSPLib Benchmark [88]. These 19 instances are all the available ones that can be found in the TSPLib webpage (<https://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/>). Additionally, 12 instances have been utilized for the VRPB. These instances have been created by the authors of this study. With the aim of allowing the replication of this experimentation, the benchmark used is available under request, and it can be obtained from the personal site of the corresponding author of this paper (<http://paginaspersonales.deusto.es/e.osaba>). The first 6 instances of the benchmark have been picked from the VRPTW Solomon Benchmark (<http://w.cba.neu.edu/~msolomon/problems.htm>). For these instances, the time constraints have been removed. Furthermore, the demands type has been modified in order to create backhaul and linehaul customers. The vehicles capacity and the amount of customer demands have been retained. On the other hand, the remaining instances have been taken from the VRPWeb, and they belong to the Christofides/Eilon (<http://neo.lcc.uma.es/vrp>) CVRP set. In this case, only the demand nature has been modified. These problem instances have been adapted for experimentation purpose and so, their optimum solutions are unknown.

In regard to the NQP, 15 different instances have been developed. The name of each of them describes the number of queens and the dimension of the chessboard. For example, the 20-queen instance consists in placing 20 queens on a 20x20 board. For this problem, the optimum is also not

TABLE 1: Summary of the characteristics of GA₁, GA₂, DGA₁, and DGA₂ for the ATSP, NQP, and BPP.

Alg.	Population	p_c and p_m	Surviv. funct.	Cross. oper.	Mut. operators
GA ₁	1 population, 48 individuals	80% & 20%	50% elitist—50% random	OX	2-opt
GA ₂	1 population, 48 individuals	0.003% & 100%	100% elitist	HX	2-opt
DGA ₁	4 subpopulations, each with 12 individuals	95% & 5%, 90% & 10%, 75% & 25%, 80% & 20%, respectively	50% elitist—50% random	OX	2-opt (the same for all subpopulations)
DGA ₂	4 subpopulations, each with 12 individuals	0.003% & 100%	100% elitist	HX	2-opt, 3-opt, Swapping, and Insertion (a different function for each population)

TABLE 2: Summary of the characteristics of GA₁, GA₂, DGA₁, and DGA₂ for the VRPB.

Alg.	Population	p_c and p_m	Surviv. funct.	Cross. oper.	Mut. operators
GA ₁	1 population, 48 individuals	80% & 20%	50% elitist—50% random	HRX	Insertion Routes
GA ₂	1 population, 48 individuals	0.003% & 100%	100% elitist	HRX	Insertion Routes
DGA ₁	4 subpopulations, each with 12 individuals	95% & 5%, 90% & 10%, 75% & 25%, 80% & 20%, respectively	50% elitist—50% random	HRX	Insertion Routes (the same for all subpopulations)
DGA ₂	4 subpopulations, each with 12 individuals	0.003% & 100%	100% elitist	HRX	2-opt, Insertion, Insertion Routes, and Swapping Routes (a different function for each population)

TABLE 3: Summary of the characteristics of GB.

Number of teams (TN)	4
Number of players per team (PT)	12
Number of trainings without improvement for a <i>custom training</i>	6
Number of trainings without improvement for a <i>special transfer</i>	12
Custom training function for the ATSP, NQP, and BPP	HX
Custom training function for the VRPB	HRX
Conventional training functions for the ATSP, NQP, and BPP	2-opt, 3-opt, Swapping, and Insertion
Conventional training functions for the VRPB	2-opt, Insertion, Swapping Routes, and Insertion Routes

shown, since it is 0 in every case. At last, regarding the BPP, 16 instances have been chosen from the well-known Scholl/Klein benchmark (<http://www.wiwi.uni-jena.de/entscheidung/binpp/index.htm>). These cases are named $NxCyWz_a$, where x is 1 (50 items), 2 (100 items), 3 (200 items), or 4 (500 items); y is 1 (capacity of 100), 2 (capacity of 120), and 3 (capacity of 150); z is 1 (items size between 1 and

100) and 2 (items size between 20 and 100); and a is A, B, or C as benchmark indexing parameter.

Each instance has been run 40 times. Besides, with the intention of conducting a fair and rigorous outcomes' comparison, two different statistical tests have been performed: the normal distribution z -test and the Friedman test. Thanks to these tests, it can be concluded whether the differences in the results are statistically significant or not. The details of these statistical tests are explained in the next section.

5. Experimentation Results

In this section, the results obtained by each technique for the chosen problems are shown and analysed. In addition, the statistical tests are also depicted in this section. First, the results and statistical tests are displayed (Section 5.1). Then, the analysis of the outcomes obtained is conducted (Section 5.2).

5.1. Results and Statistical Tests. In this section, the outcomes and statistical tests are shown. In Table 4, the results obtained for the ATSP are introduced. Furthermore, in Table 5, the outcomes got for the VRPB are presented. Besides, results obtained for the NQP and BPP are detailed in Tables 6 and 7, respectively. For each instance, the average result and

TABLE 4: Results of GB, GA₁, GA₂, DGA₁, and DGA₂ for the ATSP. For each instance results average, standard deviation, and time average are shown.

Instance		GB			GA ₁			GA ₂			DGA ₁			DGA ₂		
Name	Optimum	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time
br17	39	39.0	0.0	0.1	39.2	0.4	0.1	39.1	0.2	0.1	39.0	0.0	0.1	39.0	0.0	0.2
ftv33	1286	1329.2	33.7	0.2	1412.5	81.5	0.4	1540.3	83.1	0.2	1403.7	60.9	0.4	1416.8	90.4	0.4
ftv35	1473	1509.5	28.8	0.2	1609.1	76.9	0.4	1678.3	165.3	0.2	1606.8	74.7	0.4	1598.3	57.0	0.4
ftv38	1530	1580.4	37.3	0.3	1676.1	71.7	0.5	1709.1	145.8	0.3	1703.6	91.8	0.5	1699.4	74.5	0.4
p43	5620	5620.6	0.8	0.3	5627.7	5.5	0.9	5626.9	3.8	0.4	5625.9	3.7	0.8	5624.8	3.4	0.4
ftv44	1613	1695.1	42.7	0.4	1787.1	93.2	1.0	2071.5	147.7	0.4	1832.6	131.9	0.9	1835.0	108.0	0.6
ftv47	1776	1862.2	55.2	0.5	1961.4	86.7	1.4	2526.2	705.5	0.6	2020.2	139.1	1.0	2038.2	130.7	0.8
ry48p	14422	14614.2	164.5	0.6	15008.2	348.6	1.6	14976.5	259.7	0.8	15038.8	381.9	1.8	14945.2	178.8	0.7
ft53	6905	7335.0	204.7	0.8	8077.2	344.9	1.8	9401.1	632.6	0.9	8331.5	462.9	1.7	7997.4	232.2	0.9
ftv55	1608	1737.1	73.2	0.8	1879.3	110.7	1.4	2152.4	312.5	1.4	2021.2	153.4	1.7	1990.9	109.4	1.4
ftv64	1839	2023.5	93.4	1.6	2203.5	129.5	2.1	3032.9	226.8	1.8	2284.3	163.2	3.2	2321.8	141.3	1.7
ftv70	1950	2151.9	83.9	1.8	2313.7	145.2	2.7	3335.5	330.2	2.1	2390.0	127.0	2.5	2509.6	140.4	2.1
ft70	38673	40135.9	461.4	2.1	40416.0	623.4	3.2	47067.0	1647.2	2.1	40813.1	746.0	2.6	41129.9	823.5	2.3
kro124p	36230	38924.6	1157.4	7.4	42259.0	1813.8	9.4	44084.0	1932.5	8.8	43408.1	2020.3	11.4	41116.5	1044.9	7.8
ftv170	2755	3873.4	468.7	41.2	4214.8	361.8	49.8	4210.1	481.3	43.5	4367.0	470.7	51.7	4252.4	174.2	39.8
rbg323	1326	1494.2	35.7	120.3	1601.0	76.8	130.7	1596.1	77.3	124.9	1584.7	73.7	130.7	1614.7	194.4	124.9
rbg358	1163	1364.8	40.1	147.7	1781.9	62.5	158.1	1799.8	66.2	150.4	1720.8	175.0	164.8	1724.7	189.7	159.4
rbg403	2465	2510.4	29.6	222.0	3088.4	199.6	227.4	3298.8	378.1	224.2	2870.2	194.5	235.1	2766.2	138.4	220.4
rbg443	2720	2767.9	17.5	324.5	3142.5	219.3	335.9	3154.4	242.5	321.0	2992.2	125.6	335.9	2989.6	128.1	329.0

TABLE 5: Results of GB, GA₁, GA₂, DGA₁, and DGA₂ for the VRPB. For each instance results average, standard deviation, and time average are shown.

Instance		GB			GA ₁			GA ₂			DGA ₁			DGA ₂		
Name	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time	
C101	675.3	39.1	3.3	722.3	67.7	10.4	706.0	40.2	2.9	739.1	47.7	12.6	707.0	65.6	3.4	
C201	648.6	44.1	1.1	852.3	124.5	1.2	834.8	75.3	1.4	795.8	50.2	2.4	717.2	133.7	1.1	
R101	895.8	25.1	3.1	995.8	80.9	7.8	946.1	48.8	2.5	959.5	43.9	9.1	903.9	51.5	3.1	
R201	1047.6	22.6	7.0	1270.0	62.5	13.0	1137.2	65.4	6.5	1188.9	75.1	12.4	1085.7	39.5	6.8	
RC101	583.3	15.1	0.5	778.9	118.9	0.9	660.2	59.2	0.9	645.1	66.3	1.1	626.3	44.1	1.1	
RC201	1164.6	41.6	6.2	1304.5	76.5	13.2	1261.0	87.9	5.9	1337.2	60.1	12.4	1182.1	63.8	6.1	
En23k3	696.8	13.5	0.5	797.0	67.8	0.9	748.4	33.9	0.8	771.0	49.3	0.8	702.6	24.1	0.8	
En30k4	509.6	16.3	0.5	672.2	51.7	1.5	630.7	39.7	1.3	600.6	56.6	1.4	593.3	69.2	0.8	
En33k4	777.9	30.7	0.6	851.7	41.9	1.7	835.7	47.3	1.1	833.6	35.3	1.2	819.7	28.7	0.9	
En51k5	630.5	20.7	2.0	716.8	52.3	2.6	715.0	46.5	2.3	721.5	33.5	2.5	646.0	35.6	1.9	
En76k8	830.7	26.4	6.3	915.2	43.1	10.7	916.3	54.3	6.1	918.5	74.0	9.7	871.4	39.2	5.8	
En101k14	1088.0	24.2	22.0	1183.8	38.8	26.3	1164.8	56.2	20.8	1231.9	42.9	24.8	1191.4	59.1	19.8	

standard deviation are shown. Additionally, average runtimes are also displayed (in seconds).

As mentioned, two statistical tests have been performed according to these outcomes. The first one is the normal distribution z-test. By this test, the results obtained by the GB are compared with those obtained by the other techniques. Thanks to the normal distribution z-test, it can be concluded whether the differences between GB and the other techniques are statistically significant or not. The z statistic has the following form:

$$z = \frac{\overline{X}_{GB} - \overline{X}_i}{\sqrt{(\sigma_{GB}^2/n_{GB}) + (\sigma_i^2/n_i)}} \tag{6}$$

where \overline{X}_{GB} : average of the GB, σ_{GB} : standard deviation of the GB, n_{GB} : sample size for GB, \overline{X}_i : average of the technique i , σ_i : standard deviation of the technique i , and n_i : sample size for technique i .

It is noteworthy that the GB has been faced with the other four implemented metaheuristics. Thereby, the parameter i can be GA₁, GA₂, DGA₁, and DGA₂. The confidence interval has been stated at 95% ($z_{0.05} = 1.96$). In this way, the result of the test can be positive (+), if $z \geq 1.96$; negative (-), if $z \leq -1.96$; or neutral (*), if $-1.96 < z < 1.96$. A + indicates that GB is significantly better. In the opposite case, it obtains substantially worse solutions. If the result is (*), the difference is not significant. In this study, the numerical value of z is also displayed. Thereby, the difference in results may be seen more

TABLE 6: Results of GB, GA₁, GA₂, DGA₁, and DGA₂ for the NQP. For each instance, results average, standard deviation, and time average are shown. The optimum of each instance is 0.

Instance Name	GB			GA ₁			GA ₂			DGA ₁			DGA ₂		
	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time
8-queen	0.0	0.0	0.1	0.0	0.0	0.1	0.0	0.0	0.1	0.0	0.0	0.1	0.0	0.0	0.1
20-queen	0.1	0.2	0.1	1.4	0.6	0.1	0.1	0.3	0.1	1.5	1.1	0.2	0.8	0.7	0.1
50-queen	0.0	0.0	0.7	5.3	1.7	0.8	1.9	0.7	0.8	5.0	1.1	1.1	4.3	1.6	0.8
75-queen	0.1	0.2	4.1	8.1	1.6	4.1	4.6	1.8	4.6	9.1	1.7	5.4	6.1	1.7	4.8
100-queen	0.5	0.7	5.8	13.6	2.1	6.8	7.2	1.7	7.2	12.0	2.0	10.1	11.4	3.0	11.0
125-queen	0.3	0.4	13.4	16.4	3.2	15.8	12.6	2.4	14.8	16.2	2.5	18.4	14.3	2.4	14.8
150-queen	1.7	1.4	16.7	18.1	3.2	18.4	17.0	2.9	16.5	20.0	3.2	20.6	19.0	1.9	16.5
200-queen	3.3	1.9	23.1	26.0	3.9	26.1	24.5	3.5	26.1	32.8	4.8	31.1	23.4	3.1	26.1
225-queen	4.3	1.7	35.4	31.9	5.0	41.5	37.9	3.2	31.2	38.4	3.5	31.2	29.2	4.3	35.8
250-queen	3.5	1.6	72.4	44.3	3.9	83.1	32.7	6.7	78.1	41.2	5.3	78.1	32.0	3.1	78.1
275-queen	5.6	3.0	101.6	50.0	11.2	104.2	39.5	4.9	102.5	44.1	7.5	107.6	39.9	4.9	104.7
300-queen	6.4	2.6	131.0	61.9	5.2	132.9	44.4	5.3	130.9	52.8	5.9	134.5	44.4	5.9	128.4
325-queen	4.8	2.4	215.6	63.5	5.6	225.3	47.4	6.4	220.7	54.4	3.6	228.7	49.1	4.1	218.1
350-queen	5.1	3.0	275.3	71.4	5.6	286.7	51.0	4.7	281.2	65.5	5.7	289.6	49.9	5.8	278.5
400-queen	4.3	2.2	359.7	59.9	10.1	371.8	54.0	9.7	365.7	59.4	8.1	379.5	56.1	7.6	357.8

TABLE 7: Results of GB, GA₁, GA₂, DGA₁, and DGA₂ for the BPP. For each instance results average, standard deviation, and time average are shown.

Instance Name	Optimum	GB			GA ₁			GA ₂			DGA ₁			DGA ₂		
		Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time	Avg.	S. dev.	Time
N1C1W1.A	25	26.0	0.0	0.2	26.5	0.5	0.2	26.7	0.4	0.1	26.8	0.5	0.3	26.7	0.5	0.3
N1C1W1.B	31	31.0	0.0	0.2	31.9	0.4	0.2	31.5	0.5	0.2	31.5	0.5	0.4	31.6	0.6	0.3
N1C2W1.A	21	21.1	0.2	0.2	21.9	0.5	0.3	21.9	0.5	0.2	21.8	0.4	0.4	22.0	0.4	0.3
N1C2W1.B	26	26.1	0.2	0.3	27.6	0.5	0.3	27.1	0.4	0.3	26.8	0.4	0.3	26.8	0.5	0.3
N2C1W1.A	48	51.0	0.3	1.8	53.1	0.6	1.7	52.4	0.6	1.4	52.9	0.6	1.8	52.2	0.7	1.4
N2C1W1.B	49	51.4	0.5	1.8	52.6	0.6	1.9	53.0	0.8	1.5	53.3	0.8	1.8	52.8	0.6	1.4
N2C2W1.A	42	43.9	0.2	1.8	44.6	0.6	1.8	45.4	0.5	1.7	45.7	0.6	1.9	45.3	0.6	1.7
N2C2W1.B	50	51.4	0.5	2.1	52.4	0.6	1.9	53.1	0.7	1.8	53.4	0.6	1.9	53.2	0.6	1.5
N3C2W2.A	107	114.1	1.1	15.0	121.8	1.3	14.8	118.7	1.5	13.5	120.0	1.4	15.2	118.0	1.3	14.1
N3C2W2.B	105	109.6	0.5	17.1	119.8	1.5	16.5	113.4	1.1	16.1	115.3	1.8	15.4	111.9	0.7	14.9
N3C3W1.A	66	70.2	0.5	12.2	74.6	0.7	12.9	71.5	0.7	12.1	72.6	0.9	14.8	71.4	0.8	13.8
N3C3W1.B	71	76.1	0.5	12.1	78.4	0.6	13.1	77.4	0.9	12.7	78.6	1.0	15.7	77.6	1.0	14.5
N4C1W1.A	240	260.5	1.5	194.7	271.6	2.5	187.4	268.4	3.8	181.0	270.1	2.4	200.7	267.7	2.1	199.9
N4C2W1.A	210	231.2	1.2	195.8	239.1	1.6	188.5	233.3	5.2	186.4	241.0	1.9	203.2	235.4	1.3	200.1
N4C2W1.B	213	233.3	1.6	190.5	241.5	2.4	186.2	234.3	4.7	184.2	243.6	1.9	198.6	239.1	0.7	195.4
N4C2W1.C	213	234.5	1.6	199.8	241.7	1.8	194.2	239.7	6.8	191.4	241.3	2.0	201.5	238.1	1.9	198.3

easily. In Table 8, the tests performed for the chosen problems are shown.

The second statistical test conducted is the Friedman test. In Table 9, the results of overall ranking calculated using this test are summarized, where the smaller the score is, the better the ranking is. This ranking is conducted considering the average results of each technique and comparing them instance by instance. Furthermore, in order to check if there are statistical differences between the developed techniques, the value of X_r^2 is also depicted in Table 9. This value has been obtained using the following formula:

$$X_r^2 = \frac{12}{HK(K+1)} \sum (HRc)^2 - 3H(K+1). \quad (7)$$

H is the number of problem instances (e.g., for ATSP, $H = 19$), K is the number of techniques ($K = 5$), and Rc is the value of the Friedman test ranking score. The confidence interval has been stated at the 99% confidence level. The critical point in a χ^2 distribution with 4 degrees of freedom is 13.277. Thereby, if $X_r^2 > 13.277$, it can be concluded that there are significant differences between the five techniques. Otherwise, the differences are not substantial.

5.2. Analysis of the Results. Looking at the results presented in the previous section, one clear conclusion can be drawn: the GB outperforms the other techniques in terms of quality. Analyzing Tables 4–7, it can be seen how GB obtains

TABLE 8: Normal distribution z-test.

Instance	Versus GA ₁	Versus GA ₂	Versus DGA ₁	Versus DGA ₂
br17	+(3.16)	+(3.16)	*(0.00)	*(0.00)
ftv33	+(5.97)	+(14.88)	+(6.76)	+(5.74)
ftv35	+(7.67)	+(6.35)	+(7.68)	+(8.79)
ftv38	+(7.48)	+(5.40)	+(7.86)	+(9.03)
p43	+(8.07)	+(10.26)	+(8.85)	+(7.60)
ftv44	+(5.67)	+(15.48)	+(6.27)	+(7.61)
ftv47	+(6.10)	+(5.93)	+(6.67)	+(7.84)
ry48p	+(6.46)	+(7.45)	+(6.45)	+(8.61)
ft53	+(11.70)	+(19.65)	+(12.45)	+(13.53)
ftv55	+(6.77)	+(8.18)	+(10.57)	+(12.19)
ftv64	+(7.12)	+(26.02)	+(8.77)	+(11.13)
ftv70	+(6.10)	+(21.97)	+(9.89)	+(13.83)
ft70	+(2.28)	+(25.62)	+(4.88)	+(6.65)
kro124p	+(9.80)	+(14.48)	+(12.17)	+(8.89)
ftv170	+(3.64)	+(3.16)	+(4.69)	+(4.79)
rbg323	+(7.97)	+(7.56)	+(6.98)	+(3.85)
rbg358	+(35.52)	+(35.54)	+(12.54)	+(11.73)
rbg403	+(18.11)	+(13.14)	+(11.56)	+(11.43)
rbg443	+(10.76)	+(10.05)	+(10.96)	+(10.84)
C101	+(3.80)	+(3.46)	+(6.54)	+(2.62)
C201	+(9.75)	+(13.49)	+(13.93)	+(3.08)
R101	+(7.46)	+(5.79)	+(7.96)	*(0.89)
R201	+(21.16)	+(8.18)	+(11.39)	+(5.29)
RC101	+(10.32)	+(7.96)	+(5.74)	+(5.83)
RC201	+(10.16)	+(6.26)	+(14.93)	*(1.45)
En23k3	+(9.16)	+(8.94)	+(9.18)	*(1.32)
En30k4	+(18.97)	+(17.84)	+(9.77)	+(7.44)
En33k4	+(8.98)	+(6.48)	+(7.53)	+(6.29)
En51k5	+(9.70)	+(10.49)	+(14.61)	+(2.38)
En76k8	+(10.57)	+(8.96)	+(7.06)	+(5.44)
En101k14	+(13.24)	+(7.93)	+(18.47)	+(10.24)
8-queen	*(0.00)	*(0.00)	*(0.00)	*(0.00)
20-queen	+(13.00)	*(0.00)	+(7.91)	+(6.08)
50-queen	+(19.71)	+(17.16)	+(28.74)	+(16.99)
75-queen	+(31.37)	+(15.71)	+(33.25)	+(22.16)
100-queen	+(37.42)	+(23.04)	+(34.32)	+(22.37)
125-queen	+(31.57)	+(31.97)	+(39.71)	+(36.39)
150-queen	+(29.69)	+(30.04)	+(33.13)	+(46.36)
200-queen	+(33.09)	+(33.66)	+(36.14)	+(34.96)
225-queen	+(33.05)	+(58.64)	+(55.42)	+(34.05)
250-queen	+(61.21)	+(26.80)	+(43.06)	+(51.66)
275-queen	+(24.21)	+(37.31)	+(30.14)	+(37.75)
300-queen	+(60.37)	+(40.71)	+(45.51)	+(37.27)
325-queen	+(60.93)	+(39.41)	+(72.50)	+(58.97)
350-queen	+(66.00)	+(52.06)	+(59.30)	+(43.39)
400-queen	+(34.01)	+(31.60)	+(41.51)	+(41.40)
N1C1W1.A	+(6.32)	+(11.06)	+(10.11)	+(8.85)
N1C1W1.B	+(14.23)	+(6.32)	+(6.32)	+(6.32)
N1C2W1.A	+(9.39)	+(9.39)	+(9.89)	+(12.72)
N1C2W1.B	+(17.61)	+(14.14)	+(9.89)	+(8.22)

TABLE 8: Continued.

Instance	Versus GA ₁	Versus GA ₂	Versus DGA ₁	Versus DGA ₂
N2C1W1_A	+(19.79)	+(13.19)	+(17.91)	+(9.96)
N2C1W1_B	+(9.71)	+(10.72)	+(12.73)	+(11.33)
N2C2W1_A	+(7.00)	+(17.61)	+(18.00)	+(14.00)
N2C2W1_B	+(8.09)	+(12.49)	+(16.19)	+(14.57)
N3C2W2_A	+(28.59)	+(15.64)	+(20.95)	+(14.48)
N3C2W2_B	+(40.08)	+(19.89)	+(19.29)	+(16.90)
N3C3W1_A	+(32.34)	+(9.55)	+(14.74)	+(8.04)
N3C3W1_B	+(20.57)	+(7.98)	+(14.14)	+(8.48)
N4C1W1_A	+(24.07)	+(12.23)	+(21.45)	+(17.64)
N4C2W1_A	+(24.98)	+(2.48)	+(27.58)	+(15.01)
N4C2W1_B	+(17.97)	+(1.27)	+(26.22)	+(21.00)
N4C2W1_C	+(18.90)	+(4.70)	+(16.79)	+(9.16)

+GB is significantly better. -It is worse. *The difference is not significant (at 95% confidence level).

TABLE 9: Results of Friedman’s test (smaller is better). The last column depicts the X_r^2 value.

Problem	GB	GA ₁	GA ₂	DGA ₁	DGA ₂	X_r^2
ATSP	1.01	3.16	4.52	3.17	3.01	42.04
VRPB	1.00	4.33	3.25	4.00	2.25	28.00
NQP	1.04	4.21	2.28	4.21	2.64	15.49
BPP	1.00	3.96	3.03	4.06	2.81	34.00

better results than the other metaheuristics in 95.16% of the instances (59 out of 62). In the remaining 3 instances, GB obtains the same outcomes as one or more of the other techniques. Besides, as can be proved, GB has never obtained worse results. In addition, as Table 8 demonstrates, GB obtains significantly better results in 95.96% of the cases (238 out of 248), the differences being insignificant in the remaining 4.04%. The conclusions that can be extracted by performing a problem-by-problem analysis are the same. Regarding the ATSP, the GB gets better results in 94.73% of the cases (18 out of 19). In the remaining instance, GB obtains the same results as DGA₁ and DGA₂. Furthermore, according to Table 8, GB is substantially better in 97.36% of the confrontations (74 out of 76). In regard to VRPB and BPP, GB outperforms the other alternatives in 100% of the cases, and the differences are significant in 93.75% (45 out of 48) of the confrontations for the VRPB and in 100% (64 out of 64) for the BPP. Finally, in relation to NQP, GB proves to be better in 86.66% of the instances. In the remaining 2 cases, it obtains the same results as one or more of the other techniques. Besides, these differences are significantly better for the GB in 91% of the confrontations (55 out of 60).

At last, observing the results offered by the Friedman test (Table 9), it can be seen how GB is arguably the best technique for all the problems. In addition, all the values of X_r^2 are higher than the critical point, 13.277. For this reason, it can be concluded again that there are significant differences among the results obtained by the four techniques for all the problems.

The reasons why GB performs better than the other algorithms are the same that were presented in [51]. On the one hand, the GB combines local improvement phases (conventional trainings) with cooperative (custom trainings and player transfers) and competitive phases (matches). This technique gives greater importance to the autonomous improvement of the players, while the other four algorithms are more focused on the cooperation and competition of individuals. Furthermore, GB uses cooperation between players through custom trainings. Anyway, this resource is used to avoid local optima and to increase the exploration capacity of the technique. For this reason, this kind of trainings is used sporadically and only when it is beneficial for the search process. Besides, in the GB metaheuristic, players can explore different neighborhood structures. This feature is another alternative to avoid local optima, and it helps players to explore in different ways the solution space. On the other hand, GA₁, GA₂, DGA₁, and DGA₂ have also some mechanisms to avoid local optima, but optimization mechanisms are not as powerful as the GB.

Regarding runtimes, GB is faster than GA₁ and DGA₁, while GA₂ and DGA₂ need similar times to GB. This fact gives an advantage to GB, since it can obtain better results than the rest of techniques needing similar runtimes as GA₂ and DGA₂.

The reason why GB is faster than GA₁ and DGA₁ can be explained following the same concepts introduced in several recently published works [49, 81]. Comparing individual improvement operators (mutation and custom training) and cooperative operators (crossover and custom training), the first ones need less time. They operate with one solution, and they perform a simple modification which can be made in a minimum time. On the other hand, cooperative operators work with two different solutions, and their working ways are more complex, needing more runtime. GB makes less cooperative movements than GA₁ and DGA₁, and this fact is perfectly reflected in runtimes. Additionally, GB, GA₂, and DGA₂ obtain similar runtimes because they use their operators similarly.

Another noteworthy fact is the robustness of the GB. The standard deviation of the GB is lower than the one of the other techniques in 93.54% of the instances (58 out of 62). This means that the differences between the worst and the best result found for an instance are lower for the GB, in comparison with the four other algorithms. This fact provides robustness and reliability to the metaheuristic, something very important for a real-world problem.

As a final conclusion, it can be drawn that the GB has proved to be better than the other four metaheuristics for all the used problems. In this way, adding these outcomes to those presented in [51] for the TSP and CVRP, it can be confirmed that the GB is a promising technique for solving combinatorial optimization problems.

6. Conclusions and Further Work

The Golden Ball is a recently published multiple-population metaheuristic, which is based on soccer concepts. Until now, its performance has been tested with two simple routing problems, the TSP and the CVRP. In this paper, the quality of this technique is demonstrated applying it to four additional combinatorial problems. Two of them are routing problems, which are more complex than the previously used ones: the ATSP and the VRPB. Furthermore, one constraint satisfaction problem (NQP) and one combinatorial design problem (BPP) have also been used. In the paper presented, GB has been tested with 62 new problem instances. The outcomes obtained by the GB have been compared with the ones got by two different GAs and two DGAs. Additionally, two statistical tests have been conducted, in order to perform a rigorous and fair comparison. As a conclusion, adding the results obtained in this study with those obtained in [51], it can be concluded that the GB is a promising metaheuristic to solve combinatorial optimization problems.

As future work, it has been planned to apply the GB to other types of optimization problems. In addition, it is intended to compare the technique with other population metaheuristics, in terms of concepts and results.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

- [1] H. P. Schwefel, *Numerical Optimization of Computer Models*, John Wiley & Sons, 1981.
- [2] D. Bertsimas and J. N. Tsitsiklis, *Introduction to Linear Optimization*, Athena Scientific, Belmont, Mass, USA, 1997.
- [3] H. A. Eiselt, G. Pederzoli, and C. L. Sandblom, *Continuous Optimization Models*, Walter De Gruyter, Berlin, Germany, 1987.
- [4] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Dover Publications, 1998.
- [5] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
- [6] F. Glover, "Tabu search—part I," *ORSA Journal on Computing*, vol. 1, no. 3, pp. 190–206, 1989.
- [7] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Professional, 1989.
- [8] K. de Jong, *Analysis of the behavior of a class of genetic adaptive systems [Ph.D. thesis]*, University of Michigan, Michigan, Mich, USA, 1975.
- [9] M. Dorigo and C. Blum, "Ant colony optimization theory: a survey," *Theoretical Computer Science*, vol. 344, no. 2-3, pp. 243–278, 2005.
- [10] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, Perth, Wash, USA, December 1995.
- [11] H. Ahonen, A. G. de Alvarenga, and A. R. S. Amaral, "Simulated annealing and tabu search approaches for the Corridor Allocation Problem," *European Journal of Operational Research*, vol. 232, no. 1, pp. 221–233, 2014.
- [12] H. Dezani, R. D. Bassi, N. Marranghello, L. Gomes, F. Damiani, and I. Nunes da Silva, "Optimizing urban traffic flow using genetic algorithm with petri net analysis as fitness function," *Neurocomputing*, vol. 124, pp. 162–167, 2014.
- [13] T. Liao, T. Stützle, M. A. M. de Oca, and M. Dorigo, "A unified ant colony optimization algorithm for continuous optimization," *European Journal of Operational Research*, vol. 234, no. 3, pp. 597–609, 2014.
- [14] P. C. Pendharkar, "A misclassification cost risk bound based on hybrid particle swarm optimization heuristic," *Expert Systems with Applications*, vol. 41, no. 4, pp. 1483–1491, 2014.
- [15] E. Atashpaz-Gargari and C. Lucas, "Imperialist competitive algorithm: an algorithm for optimization inspired by imperialistic competition," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 4661–4667, September 2007.
- [16] S. H. Mirhoseini, S. M. Hosseini, M. Ghanbari, and M. Ahmadi, "A new improved adaptive imperialist competitive algorithm to solve the reconfiguration problem of distribution systems for loss reduction and voltage profile improvement," *International Journal of Electrical Power & Energy Systems*, vol. 55, pp. 128–143, 2014.
- [17] A. Yazdipour and M. R. Ghaderi, "Optimization of weld bead geometry in gtaw of cp titanium using imperialist competitive algorithm," *The International Journal of Advanced Manufacturing Technology*, vol. 72, no. 5–8, pp. 619–625, 2014.
- [18] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Tech. Rep. tr06, Computer Engineering Department, Engineering Faculty, Erciyes University, 2005.
- [19] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.
- [20] Q.-K. Pan, L. Wang, J.-Q. Li, and J.-H. Duan, "A novel discrete artificial bee colony algorithm for the hybrid flowshop scheduling problem with makespan minimisation," *Omega*, vol. 45, pp. 42–56, 2014.
- [21] M. K. Apalak, D. Karaboga, and B. Akay, "The artificial bee colony algorithm in layer optimization for the maximum fundamental frequency of symmetrical laminated composite plates," *Engineering Optimization*, vol. 46, no. 3, pp. 420–437, 2014.
- [22] B. Li, L. G. Gong, and W. I. Yang, "An improved artificial bee colony algorithm based on balance-evolution strategy for unmanned combat aerial vehicle path planning," *The Scientific World Journal*, vol. 2014, Article ID 232704, 10 pages, 2014.

- [23] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.
- [24] Z. W. Geem, C. Tseng, and J. C. Williams, "Harmony search algorithms for water and environmental systems," *Studies in Computational Intelligence*, vol. 191, pp. 113–127, 2009.
- [25] H. Ceylan, O. Baskan, C. Ozan, and G. Gulhan, "Determining on-street parking places in urban road networks using meta-heuristic harmony search algorithm," in *Computer-Based Modelling and Optimization in Transportation*, pp. 163–174, Springer, 2014.
- [26] J. Contreras, I. Amaya, and R. Correa, "An improved variant of the conventional harmony search algorithm," *Applied Mathematics and Computation*, vol. 227, pp. 821–830, 2014.
- [27] X. S. Yang, "Harmony search as a metaheuristic algorithm," in *Music-Inspired Harmony Search Algorithm*, pp. 1–14, Springer, New York, NY, USA, 2009.
- [28] X. S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, vol. 284 of *Studies in Computational Intelligence*, pp. 65–74, Springer, 2010.
- [29] X. Yang, "Bat algorithm for multi-objective optimisation," *International Journal of Bio-Inspired Computation*, vol. 3, no. 5, pp. 267–274, 2011.
- [30] X. S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, IEEE, December 2009.
- [31] X.-S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343, 2010.
- [32] X. S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications*, vol. 5792 of *Lecture Notes in Computer Science*, pp. 169–178, Springer, 2009.
- [33] X. Yang, "Firefly algorithm, stochastic test functions and design optimization," *International Journal of Bio-Inspired Computation*, vol. 2, no. 2, pp. 78–84, 2010.
- [34] D. Rodrigues, L. A. Pereira, R. Y. Nakamura et al., "A wrapper approach for feature selection based on bat algorithm and optimum-path forest," *Expert Systems with Applications*, vol. 41, no. 5, pp. 2250–2258, 2014.
- [35] B. Bahmani-Firouzi and R. Azizipناه-Abarghooee, "Optimal sizing of battery energy storage for micro-grid operation management using a new improved bat algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 56, pp. 42–54, 2014.
- [36] O. Hasançebi and S. Carbas, "Bat inspired algorithm for discrete size optimization of steel frames," *Advances in Engineering Software*, vol. 67, pp. 173–185, 2014.
- [37] I. Fister, S. Fong, J. Brest, and I. Fister, "A novel hybrid self-adaptive bat algorithm," *The Scientific World Journal*, vol. 2014, Article ID 709738, 12 pages, 2014.
- [38] X. Yang and S. Deb, "Cuckoo search: recent advances and applications," *Neural Computing and Applications*, vol. 24, no. 1, pp. 169–174, 2014.
- [39] M. Marichelvam, T. Prabaharan, and X. S. Yang, "Improved cuckoo search algorithm for hybrid flow shop scheduling problems to minimize makespan," *Applied Soft Computing*, vol. 19, pp. 93–101, 2014.
- [40] J. H. Yi, W. H. Xu, and Y. T. Chen, "Novel back propagation optimization by cuckoo search algorithm," *The Scientific World Journal*, vol. 2014, Article ID 878262, 8 pages, 2014.
- [41] I. Fister Jr., X. S. Yang, D. Fister, and I. Fister, "Cuckoo search: a brief literature review," in *Cuckoo Search and Firefly Algorithm*, pp. 49–62, Springer, New York, NY, USA, 2014.
- [42] I. Fister, X. S. Yang, D. Fister, and I. Fister Jr., "Firefly algorithm: a brief review of the expanding literature," in *Cuckoo Search and Firefly Algorithm*, *Studies in Computational Intelligence*, pp. 347–360, Springer, 2014.
- [43] I. Fister, I. Fister Jr., X. S. Yang, and J. Brest, "A comprehensive review of firefly algorithms," *Swarm and Evolutionary Computation*, vol. 13, pp. 34–46, 2013.
- [44] J. C. Bansal, H. Sharma, S. S. Jadon, and M. Clerc, "Spider monkey optimization algorithm for numerical optimization," *Memetic Computing*, vol. 6, no. 1, pp. 31–47, 2013.
- [45] C. Dai, Y. Zhu, and W. Chen, "Seeker optimization algorithm," in *Computational Intelligence and Security*, pp. 167–176, Springer, New York, NY, USA, 2007.
- [46] J. K. Lenstra and A. H. G. R. Kan, "Complexity of vehicle routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 221–227, 1981.
- [47] C. Papadimitriou, "The new faces of combinatorial optimization," in *Combinatorial Optimization*, vol. 7422 of *Lecture Notes in Computer Science*, pp. 19–23, Springer, New York, NY, USA, 2012.
- [48] G. Donets and I. Sergienko, "A method for modeling the structure of initial data and subclasses of solvable combinatorial optimization problems," *Cybernetics and Systems Analysis*, vol. 50, no. 1, pp. 1–7, 2014.
- [49] E. Osaba, E. Onieva, R. Carballedo, F. Diaz, A. Perallos, and X. Zhang, "A multi-crossover and adaptive island based population algorithm for solving routing problems," *Journal of Zhejiang University Science C*, vol. 14, no. 11, pp. 815–821, 2013.
- [50] E. Osaba, F. Diaz, and E. Onieva, "A novel meta-heuristic based on soccer concepts to solve routing problems," in *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation (GECCO '13)*, pp. 1743–1744, ACM, July 2013.
- [51] E. Osaba, F. Diaz, and E. Onieva, "Golden ball: a novel meta-heuristic to solve combinatorial optimization problems based on soccer concepts," *Applied Intelligence*, vol. 41, no. 1, pp. 145–166, 2014.
- [52] E. Lawler, J. Lenstra, A. Kan, and D. Shmoys, *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, vol. 3, John Wiley & Sons, New York, NY, USA, 1985.
- [53] G. Laporte, "The vehicle routing problem: an overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, no. 3, pp. 345–358, 1992.
- [54] E. Cantú-Paz, "A survey of parallel genetic algorithms," *Calculateurs Paralleles, Reseaux et Systems Repartis*, vol. 10, no. 2, pp. 141–171, 1998.
- [55] E. Alba and J. M. Troya, "A survey of parallel distributed genetic algorithms: a structured and extensive overview on an up-to-date search paradigm," *Complexity*, vol. 4, no. 4, pp. 31–52, 1999.
- [56] A. M. Frieze, G. Galbiati, and F. Maffioli, "On the worst-case performance of some algorithms for the asymmetric traveling salesman problem," *Networks*, vol. 12, no. 1, pp. 23–39, 1982.
- [57] B. Golden, E. Baker, J. Alfaro, and J. Schaffer, "The vehicle routing problem with backhauling: two approaches," in *Proceedings of the 21st Annual Meeting of SE TIMS*, pp. 90–92, South Carolina, SC, USA, 1985.
- [58] I. Rivin, I. Vardi, and P. Zimmermann, "The n -queens problem," *The American Mathematical Monthly*, vol. 101, no. 7, pp. 629–639, 1994.

- [59] S. Martello and P. Toth, *Knapsack Problems: Algorithms and Computer Implementations*, Wiley, New York, NY, USA, 1990.
- [60] S. Lin, "Computer solutions of the traveling salesman problem," *The Bell System Technical Journal*, vol. 44, pp. 2245–2269, 1965.
- [61] L. Davis, "Applying adaptive algorithms to epistatic domains," in *Proceedings of the International Joint Conference on Artificial Intelligence*, vol. 1, pp. 161–163, 1985.
- [62] D. E. Goldberg and R. Lingle, "Alleles, loci, and the traveling salesman problem," in *Proceedings of the 1st International Conference on Genetic Algorithms and Their Applications*, pp. 154–159, Lawrence Erlbaum Associates, 1985.
- [63] J. LaRusic and A. P. Punnen, "The asymmetric bottleneck traveling salesman problem: algorithms, complexity and empirical analysis," *Computers & Operations Research*, vol. 43, pp. 20–35, 2014.
- [64] J. Bai, G. K. Yang, Y. W. Chen, L. S. Hu, and C. C. Pan, "A model induced max-min ant colony optimization for asymmetric traveling salesman problem," *Applied Soft Computing Journal*, vol. 13, no. 3, pp. 1365–1375, 2013.
- [65] P. Larrañaga, C. M. H. Kuijpers, R. H. Murga, I. Inza, and S. Dizdarevic, "Genetic algorithms for the travelling salesman problem: a review of representations and operators," *Artificial Intelligence Review*, vol. 13, no. 2, pp. 129–170, 1999.
- [66] S. Salhi, N. Wassan, and M. Hajarat, "The fleet size and mix vehicle routing problem with backhauls: formulation and set partitioning-based heuristics," *Transportation Research E: Logistics and Transportation Review*, vol. 56, pp. 22–35, 2013.
- [67] S. P. Anbuudayasankar, K. Ganesh, S. C. Lenny Koh, and Y. Ducq, "Modified savings heuristics and genetic algorithm for bi-objective vehicle routing problem with forced backhauls," *Expert Systems with Applications*, vol. 39, no. 3, pp. 2296–2305, 2012.
- [68] A. A. Juan, J. Faulin, E. Pérez-Bernabeu, and N. Jozefowicz, "Horizontal cooperation in vehicle routing problems with backhauling and environmental criteria," *Procedia-Social and Behavioral Sciences*, vol. 111, pp. 1133–1141, 2014.
- [69] P. Toth and D. Vigo, "Vrp with backhauls," in *The Vehicle Routing Problem*, vol. 9 of *SIAM Monographs on Discrete Mathematics and Applications*, pp. 195–221, 2002.
- [70] *The vehicle routing problem*, vol. 9 of *SIAM Monographs on Discrete Mathematics and Applications*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2002.
- [71] J. Bell and B. Stevens, "A survey of known results and research areas for n -queens," *Discrete Mathematics*, vol. 309, no. 1, pp. 1–31, 2009.
- [72] M. Bezzel, "Proposal of 8-queens problem," *Berliner Schachzeitung*, vol. 3, p. 363, 1848.
- [73] X. Hu, R. C. Eberhart, and Y. Shi, "Swarm intelligence for permutation optimization: a case study of n -queens problem," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '03)*, pp. 243–246, April 2003.
- [74] E. Masehian and H. a. . Akbaripour, "Landscape analysis and efficient metaheuristics for solving the n -queens problem," *Computational Optimization and Applications*, vol. 56, no. 3, pp. 735–764, 2013.
- [75] I. Martinjak and M. Golub, "Comparison of heuristic algorithms for the n -queen problem," in *Proceedings of the 29th International Conference on Information Technology Interfaces (ITI '07)*, pp. 759–764, Cavtat, Croatia, June 2007.
- [76] K. Fleszar and C. Charalambous, "Average-weight-controlled bin-oriented heuristics for the one-dimensional bin-packing problem," *European Journal of Operational Research*, vol. 210, no. 2, pp. 176–184, 2011.
- [77] K. Sim, E. Hart, and B. Paechter, "A hyper-heuristic classifier for one dimensional bin packing problems: Improving classification accuracy by attribute evolution," in *Proceeding of the 12th Conference on Parallel Problem Solving from Nature*, pp. 348–357, Springer, 2012.
- [78] K. Sim and E. Hart, "Generating single and multiple cooperative heuristics for the one dimensional bin packing problem using a single node genetic programming island model," in *Proceedings of the 15th Genetic and Evolutionary Computation Conference (GECCO '13)*, pp. 1549–1556, ACM, July 2013.
- [79] M. Tomassini, "A survey of genetic algorithms," *Annual Reviews of Computational Physics*, vol. 3, no. 2, pp. 87–118, 1995.
- [80] D. B. Fogel, "Introduction to simulated evolutionary optimization," *IEEE Transactions on Neural Networks*, vol. 5, no. 1, pp. 3–14, 1994.
- [81] E. Osaba, R. Carballedo, F. Diaz, and A. Perallos, "Analysis of the suitability of using blind crossover operators in genetic algorithms for solving routing problems," in *Proceedings of the 8th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI '13)*, pp. 17–22, 2013.
- [82] K. Rocki and R. Suda, "Accelerating 2-opt and 3-opt local search using GPU in the travelling salesman problem," in *Proceedings of the 10th Annual International Conference on High Performance Computing and Simulation (HPCS '12)*, pp. 489–495, Madrid, Spain, July 2012.
- [83] H. Nagarajan, P. Wei, S. Rathinam, and D. Sun, "Heuristics for synthesizing robust networks with a diameter constraint," *Mathematical Problems in Engineering*, vol. 2014, Article ID 326963, 11 pages, 2014.
- [84] J.-F. Cordeau and G. Laporte, "A tabu search heuristic for the static multi-vehicle dial-a-ride problem," *Transportation Research B: Methodological*, vol. 37, no. 6, pp. 579–594, 2003.
- [85] E. Osaba, E. Onieva, R. Carballedo, F. Diaz, and A. Perallos, "An adaptive multi-crossover population algorithm for solving routing problems," in *Proceedings of the 6th International Workshop on Nature Inspired Cooperative Strategies for Optimization*, pp. 113–124, Springer, New York, NY, USA, 2014.
- [86] C. D. Tarantilis, "Solving the vehicle routing problem with adaptive memory programming methodology," *Computers and Operations Research*, vol. 32, no. 9, pp. 2309–2327, 2005.
- [87] M. Savelsbergh, "The vehicle routing problem with time windows: minimizing route duration," *ORSA Journal on Computing*, vol. 4, no. 2, pp. 146–154, 1992.
- [88] G. Reinelt, "TSPLIB: a traveling salesman problem library," *ORSA Journal on Computing*, vol. 3, no. 4, pp. 376–384, 1991.

Research Article

Multiobjective Memetic Estimation of Distribution Algorithm Based on an Incremental Tournament Local Searcher

Kaifeng Yang,¹ Li Mu,² Dongdong Yang,^{1,3} Feng Zou,¹ Lei Wang,¹ and Qiaoyong Jiang¹

¹ School of Computer Science and Engineering, Xi'an University of Technology, P.O. Box 666, No. 5 South Jinhua Road, Xi'an 710048, China

² School of Computer Science, Xi'an Polytechnic University, China

³ Shaanxi Huanghe Group Co., Ltd., Xi'an, China

Correspondence should be addressed to Dongdong Yang; ddyang@mail.xidian.edu.cn

Received 16 April 2014; Accepted 18 June 2014; Published 23 July 2014

Academic Editor: T. O. Ting

Copyright © 2014 Kaifeng Yang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

A novel hybrid multiobjective algorithm is presented in this paper, which combines a new multiobjective estimation of distribution algorithm, an efficient local searcher and ϵ -dominance. Besides, two multiobjective problems with variable linkages strictly based on manifold distribution are proposed. The Pareto set to the continuous multiobjective optimization problems, in the decision space, is a piecewise low-dimensional continuous manifold. The regularity by the manifold features just build probability distribution model by globally statistical information from the population, yet, the efficiency of promising individuals is not well exploited, which is not beneficial to search and optimization process. Hereby, an incremental tournament local searcher is designed to exploit local information efficiently and accelerate convergence to the true Pareto-optimal front. Besides, since ϵ -dominance is a strategy that can make multiobjective algorithm gain well distributed solutions and has low computational complexity, ϵ -dominance and the incremental tournament local searcher are combined here. The novel memetic multiobjective estimation of distribution algorithm, MMEDA, was proposed accordingly. The algorithm is validated by experiment on twenty-two test problems with and without variable linkages of diverse complexities. Compared with three state-of-the-art multiobjective optimization algorithms, our algorithm achieves comparable results in terms of convergence and diversity metrics.

1. Introduction

Multiobjective optimization usually involves many conflicting, incomparable, and noncommensurable objectives. During the past two decades, multiobjective evolutionary algorithms (MOEAs) have obtained much more interest among optimization community mainly because of the fact that they can be suitably applied to deal simultaneously with a set of possible solutions. A number of evolutionary algorithms have been developed for multiobjective problems, such as strength Pareto evolutionary algorithm (SPEA) [1], Pareto archived evolution strategy (PAES) [2], Pareto envelope-based selection algorithm (PESA) [3, 4], micro-GA [5], nondominated sorting genetic algorithm II (NSGA-II) [6], strength Pareto evolutionary algorithm 2 (SPEA2) [7], multiple objectives with particle swarm optimization (MOPSO) [8], nondominated neighbor immune algorithm (NNIA) [9], and MOEA

with adaptive weight adjustment (MOEA/D-AWA) [10]. Following the recent review of evolutionary multiobjective optimization fields [11, 12], NSGA-II and SPEA2 can be considered as two representatives of state-of-the-art MOEAs in current multiobjective optimization community.

The current MOEAs research mainly focuses on the following several highly related issues, such as fitness assignment, diversity maintenance, external population, combination of MOEA and local search, new dominance scheme, and many-objective problems. However, there are little fresh work done on how to generate new solutions and most current MOEAs directly adopt traditional crossover and mutation operators. Based on the works [13, 14], we can gain multiobjective problems (MOPs) with variable linkages causing trouble to MOEAs with traditional crossover and mutation operators. It seems that it is urgent to design new scheme to generate new solutions.

Estimation of distribution algorithms (EDAs) are a new computing paradigm in evolutionary computation. A posterior probability distribution model based on globally statistical information from the selected solutions is built to generate new solutions for next generation. This new class of algorithms generalizes genetic algorithms by replacing the crossover and mutation operators by learning and sampling the probability distribution of the promising individuals of population per iteration. Working in such a way, the relationships between the variables involved in the problem domain are explicitly and effectively captured and exploited. Several EDAs have been proposed for MOPs. Khan [15] proposed multiobjective BOA (mBOA) and multiobjective hierarchical BOA (mhBOA) by combining the model building and model sampling procedures of BOA [16] and hierarchical BOA (hBOA) [17] with the selection procedure of NSGA-II. They compared the performance of mBOA and mhBOA with that of NSGA-II on a class of bounded difficult additively separable deceptive and hierarchically deceptive functions. Bosman and Thierens [18] combined IDEAs with nondominated tournament selection and clustering, and they used clustering to split the population into subpopulation and separate models were built for each subpopulation. Laumanns and Ocenasek [19] combined mixed BOA with the selection and replacement procedures of SPEA2. The algorithm was tested on some knapsack problems, and it was shown to dominate NSGA-II and SPEA2 in most instances. Zhou et al. [20] present a way, named multiobjective EDA based on decomposition, which utilizes decomposition based techniques and probabilistic model based methods, to tackle the traveling salesman problem. However, these MOEDAs do not involve how the Pareto set distributes in the decision space.

The Pareto set of a continuous MOP is a piecewise continuous $(m - 1)$ -dimensional manifold, where m is the number of the objectives, which have been investigated and applied by several scholars [21, 22]. Zhang et al. [14] exploited the property explicitly and proposed regularity model based multiobjective estimation of distribution algorithm, called RM-MEDA, a regularity model based EDA for continuous MOPs. They have validated that RM-MEDA can effectively deal with MOPs with variable linkages and admitted that the performance of RM-MEDA declines if a MOP has many local Pareto fronts [14, 23].

As Ong et al. [24, 25] suggested, in recent years, the development of hybrid genetic algorithm is one of the most significant trends in the field of metaheuristics. Methods of this kind hybridize recombination operators with local heuristics. Such a hybrid algorithm is also called a memetic algorithm. It is clearly shown that memetic algorithms have higher search ability than traditional EMO algorithms [26]. Ishibuchi and Murata [27] are the first two scholars to propose a multiobjective genetic local search algorithm. Afterward, Xu et al. [28] proposed GLSA, Jaskiewicz [29] proposed MOGLS, and Liu et al. [30] proposed FMOPSO. As forenamed, RM-MEDA extracts globally statistical information to build the probability distribution model and performs weakly in local search. If we hybridize an effective local search operator with RM-MEDA, we may get balance

between exploration and exploitation in the search space. For this end, an incremental tournament local searcher operator is proposed in our study, which biases solutions with high isolation value. Our algorithm is called multiobjective memetic estimation of distribution algorithm.

In order to keep well-spread Pareto-optimal solutions, ϵ -dominance is employed in our algorithm. The ϵ -dominance does not allow two solutions with a difference less than ϵ_i in the i th objective to be nondominated to each other, thereby allowing a good diversity to be maintained in a population. Besides, the method is quite pragmatic because it allows the user to choose a suitable ϵ_i depending on the desired resolution in the i th objective. Deb et al. [31] have validated that the diversity metric of ϵ -MOEA is slightly better than that of NSGA-II based on crowding distance.

Moreover, in [14], Zhang et al. proposed ten multiobjective problems with variable linkages. However, distribution of these ten problems is linear or almost linear in decision space. Besides, RM-MEDA introduces local principal component analysis to build probability distribution model, which is linear or local-linear distribution in decision space. Therefore, most of these problems may be easy for RM-MEDA. For this end, two more difficulty problems are proposed in this paper, which more accord with manifold distribution in the decision space by introducing nonlinear mapping into variables.

In the remainder of the paper, we briefly mention notations and definitions in Section 2. Thereafter, in Section 3, we briefly describe RM-MEDA. Section 4 presents the proposed MMEDA. Section 5 describes our proposed two problems. In Section 6, the experimental study is demonstrated. Finally, we outline the conclusions of this paper.

2. Definitions and Notations

In our paper, we consider the following continuous MOP:

$$\min f(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_m(\mathbf{x}))^T, \quad (1)$$

where $\mathbf{x} \in \Omega \subseteq R^N$, \mathbf{x} is a decision variable vector, and Ω is a continuous search space. $f : \mathbf{x} \rightarrow R^m$ is the map of decision variable space to m real valued objective space. The objectives in a MOP conflict each other, and no single solution can optimize all the objectives at the same time. The Pareto front/Pareto set (PF/PS) is set of all the optimal tradeoff solutions in the objective/decision space.

Definition 1 (Pareto dominance). There is a vector $\mathbf{u} = (u_1, u_2, \dots, u_m)$, which is said to dominate $\mathbf{v} = (v_1, v_2, \dots, v_m)$ (denoted by $\mathbf{u} < \mathbf{v}$), if and only if \mathbf{u} is partially less than \mathbf{v} , which is equal to this expression: $\forall i \in \{1, \dots, m\}, u_i \leq v_i \wedge \exists i \in \{1, \dots, m\} : u_i < v_i$.

Definition 2 (Pareto optimality). A point $\mathbf{x}^* \in \Omega$ is a random optimal point and if for every $\mathbf{x} \in \Omega$ and $\mathbf{I} = (1, 2, \dots, m)$ either $\forall i \in \mathbf{I}, f_i(\mathbf{x}) = f_i^*(\mathbf{x})$ or there is at least one $i \in \mathbf{I}$ such that $f_i(\mathbf{x}) > f_i^*(\mathbf{x})$. In other words, this definition says that \mathbf{x}^* is Pareto optimal if there are no feasible vectors \mathbf{x} which can decrease some criterion without causing a simultaneous increase in at least one other criterion.

Step 1. Initialization: Set $t = 0$, generate initial population $\mathbf{P}(0)$ and evaluate them.
Step 2. Termination: If $t > G_{\max}$ is satisfied, export $\mathbf{P}(t)$ as the output of the algorithm, and stop, else go to step 3.
Step 3. Modeling: Perform local PCA to partition $\mathbf{P}(t)$ into K disjoint clusters S^1, S^2, \dots, S^K . For each cluster S^j , build model (2) by (4) and (5).
Step 4. Sampling: Sample new population $\mathbf{O}(t)$ from model (2) and evaluate $\mathbf{O}(t)$.
Step 5. Non-dominated Sorting and Crowding Distance Assignment: Use the famous fast non-dominated sorting procedure and select solutions at first several low ranks from $\mathbf{P}(t)$ and $\mathbf{O}(t)$. At the critical rank, the crowding distance computation is employed to select individuals with higher values of crowding distance. Then, $\mathbf{P}(t + 1)$ is created.
Step 6. set $t := t + 1$ and go to Step 2.

ALGORITHM 1: The main loop of RM-MEDA.

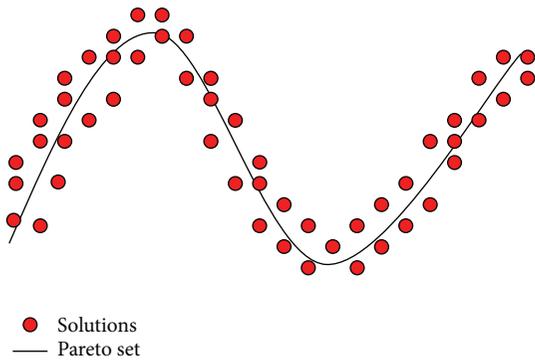


FIGURE 1: Illustration of individual solutions scattered around PS in the decision space.

3. The Framework of RM-MEDA

Under some smoothness assumptions, it could be induced from the Karush-Kuhn-Tucker condition that the PS of (1) defines a piecewise continuous $(m - 1)$ -dimensional manifold, where m is the number of objectives. In [14], they give us the following model to illustrate individual solutions scattered around the PS in the decision space:

$$\xi = \zeta + \nu, \tag{2}$$

where ζ is uniformly distributed over a piecewise continuous $(m - 1)$ -dimensional manifold. ν is an n -dimensional zero-mean noise vector and n is the number of decision variables. Figure 1 illustrates the basic idea of RM-MEDA.

In [14], piecewise $(m - 1)$ -dimensional linear models are used to approximate model ζ in (2). Local principal component analysis is applied to partition population. In RM-MEDA, the number of clusters is an experimental parameter and Zhang sets it to be 5. The solutions of each cluster are used to build a statistical model by principal component analysis and we could get the parameters of each model and noise ν in (2). New trial solutions are sampled from each local model.

As mentioned above, offspring solutions are generated by the statistical model and how to build the model is crucial to this algorithm. In RM-MEDA, they first partition population $P(t)$ into disjoint clusters S^1, S^2, \dots, S^K by local principal component analysis, and more details about partition can be

found in [32]. For each cluster S^j , let $\bar{\mathbf{x}}^j$ be its mean and \mathbf{U}_i^j its i th principal component. Compute following two equations:

$$\begin{aligned} a_i^j &= \min (\mathbf{x} - \bar{\mathbf{x}}^j)^T \mathbf{U}_i^j, \\ b_i^j &= \max (\mathbf{x} - \bar{\mathbf{x}}^j)^T \mathbf{U}_i^j, \quad \mathbf{x} \in S^j \end{aligned} \tag{3}$$

for $i = 1, 2, \dots, m$. Then, set

$$\begin{aligned} \Psi^j &= \left\{ \mathbf{x} \in \frac{R^n}{\mathbf{x}} = \bar{\mathbf{x}}^j + \sum \alpha_i \mathbf{U}_i^j, a_i^j - 0.25 (b_i^j - a_i^j) \right. \\ &\quad \left. \leq \alpha_i \leq b_i^j + 0.25 (b_i^j - a_i^j), i = 1, 2, \dots, m - 1 \right\}. \end{aligned} \tag{4}$$

Let λ_i^j be the i th largest eigenvalue of covariance matrix of points in S^j , and $\nu \sim N(0, \delta_j \mathbf{I})$:

$$\delta_j = \frac{1}{n - m + 1} (\lambda_m^j + \lambda_{m+1}^j + \dots + \lambda_n^j). \tag{5}$$

From (4) and (5), we can get the model of (2) for each cluster, and new offspring solutions are sampled from the model. The flowchart of RM-MEDA is illustrated in Algorithm 1.

RM-MEDA is a novel and efficient multiobjective estimation of distribution algorithms and has been validated by multiobjective problems with variable linkages. However, it may fail in locating the global PF if a MOP has many local PFs [23], and it has not been tested by famous ZDT and DTLZ problems. In the next section, we proposed multiobjective memetic estimation of distribution algorithms, which is a more efficient and effective hybrid multiobjective algorithm.

4. The Proposed Method: MMEDA

4.1. Incremental Tournament Local Searcher (ITLS). It is known that memetic algorithms, combined with EAs and local search heuristics, can be implemented to maintain a balance between exploration and exploitation in the search space. Besides, it is clearly shown by Jaskiewicz [26] that memetic EMO algorithms have higher search ability than traditional EMO algorithms. Several memetic EMO algorithms have been proposed and show high competitive performance. In this paper, an incremental tournament local searcher is proposed and combined with RM-MEDA.

Step 1. Find non-dominated solutions $\mathbf{N}(t)$ from population $\mathbf{P}(t)$, and assign crowding distance to $\mathbf{N}(t)$.
Step 2. Select n_s non-dominated solutions from $\mathbf{N}(t)$ by crowding distance to form tournament pool. If the size of $\mathbf{N}(t)$ is less than n_s , then all of $\mathbf{N}(t)$ are selected.
Step 3. Perform tournament selection with tournament size n_t from the tournament pool by crowding distance and N_s non-dominated solutions are selected, called active subpopulation.
Step 4. Solutions from active subpopulation and tournament pool are selected randomly to implement simulated binary crossover operator and polynomial mutation. All of the new solutions are merged into new population $\mathbf{O}(t)$.

PSEUDOCODE 1: The main pseudocode of ITLS.

Following Liu et al.'s recent reviews [30], the issues considered in the design of the local searcher operator include (1) the selection of appropriate search direction; (2) the selection of appropriate solutions for local optimization; and (3) the allocation of computational budget for local search. With these notions in mind, we devise our local searcher as follows.

An active subpopulation is selected by tournament; that is, N_s nondominated solutions are selected before modeling in RM-MEDA per iteration. At first, a tournament pool is built by nondominated solutions with higher value of crowding distance, and the size of tournament pool is n_s . If the number of nondominated solutions found so far is less than n_s , all of the nondominated solutions are included in the tournament pool. Then we select solutions with higher value of crowding distance by performing tournament selection, and the winner solutions are put in the active subpopulation, whose size is N_s . Since N_s is always larger than the size of tournament pool, n_s , we call it incremental tournament local searcher. Then, traditional genetic operators are employed on the active subpopulation for local search. Figure 2 illustrates the basic idea of ITLS.

By Figure 2, we can obtain that if a nondominated solution locates in more isolated regions, there are more offsprings created near the solutions. The aim is that the larger the crowding-distance value of an individual, the more the times the individual will be reproduced. So there exist more chances to do search in less-crowded regions of the tradeoff fronts. Now, we can answer the former three issues of local searcher. The search direction is optimization direction and is illustrated in Figure 2; the solutions for local optimization are biased towards isolated ones; and the allocation of computational resource for local searcher is determined by N_s , the size of active subpopulation.

In order to present ITLS in detail, the procedure is outlined in Pseudocode 1. Before we describe the ITLS, let us give some more notations. The tournament scale is n_t , and population $P(t)$ is from RM-MEDA before modeling. Nondominated solutions from $P(t)$ are denoted by $N(t)$.

4.2. The Adopted ϵ -Dominance. It is clear from the existing studies that there are two distinct goals in the development of an MOEA: the first one is convergence to the true Pareto-optimal front; then the second one is maintenance of a well-distributed set of nondominated solutions. The proposed ITLS in former subsection corresponds to the first goal. In

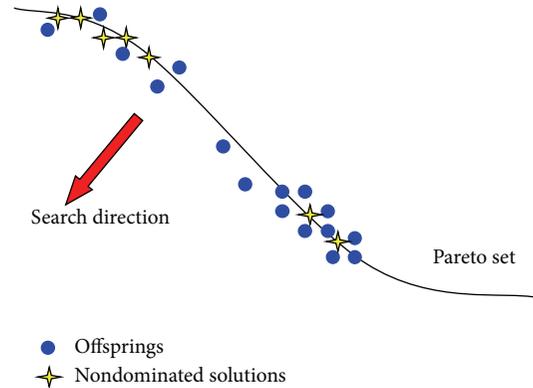


FIGURE 2: Illustration of basic idea of ITLS.

order to maintain well-spread Pareto-optimal solutions, we introduce the ϵ -dominance in our paper.

The ϵ -dominance does not allow two solutions with a difference less than ϵ_i in the i th objective to be nondominated to each other, thereby allowing a good diversity to be maintained in a population. By ϵ -dominance, the search space is divided into a number of grids (or hyperboxes) and the diversity is maintained by ensuring that a grid or hyperbox can be occupied by only one solution. In our MMEDA, there are two coevolving populations: an internal population $P(t)$ and an archive population $N(t)$. The archive population is updated based on the ϵ -dominance concept, whereas a usual domination concept is used to update the internal population. Every solution in the archive is assigned an identification array ($\mathbf{B} = (B_1, B_2, \dots, B_m)^T$, where m is the total number of objectives) as follows:

$$B_j(\mathbf{f}) = \left\lfloor \frac{(f_j - f_j^{\min})}{\epsilon_j} \right\rfloor \quad \text{for } \min f(\mathbf{x}) \text{ in (1),} \quad (6)$$

where f_j^{\min} is the minimum possible value of the j th objective (default value of it is $f_j^{\min} = 0$) and ϵ_j is the allowable tolerance in the j th objective below which two values are meaningless to the user. With the identification arrays calculated for the offspring and each archive member, we can decide how the archive population updates. More details of ϵ -dominance can be found in [31].

Step 1. Initialization: Set $t = 0$, generate initial population $\mathbf{P}(0)$ and evaluate it.

Step 2. Update non-dominated population: Identify non-dominated solutions from population $\mathbf{P}(t)$, and update archive $\mathbf{N}(t)$ by ϵ dominance.

Step 3. Termination: If $t > G_{\max}$ is satisfied, export $\mathbf{N}(t)$ as the output of the algorithm, and Stop, else go to step 3.

Step 3. ITLS: Perform incremental tournament local searcher to generate N_s new solutions $\mathbf{N}_s(t)$.

Step 4. Modeling: Perform the $(m-1)$ -d local PCA to partition $\mathbf{P}(t)$ into K disjoint clusters S^1, S^2, \dots, S^K . For each cluster S^j , build model (2) by (4) and (5).

Step 5. Sampling: Sample new population $\mathbf{O}(t)$ from model (2) and evaluate $\mathbf{O}(t)$.

Step 6. Update current population: select N solutions from $\mathbf{P}(t) \cup \mathbf{O}(t) \cup \mathbf{N}_s(t)$ to create $\mathbf{P}(t+1)$.

Step 7. set $t := t + 1$ and go to Step 2.

ALGORITHM 2: The details of MMEDA.

4.3. Details of MMEDA. As mentioned above, RM-MEDA extracts globally statistical information to build the probability distribution model and then samples it to generate offspring, which emphasizes global statistics ability, yet brings the trouble of weak performance in local search. However, several memetic MOEAs have been proposed and implemented to maintain a balance between exploration and exploitation in search space, which is often crucial to the success of the search and optimization process. For this end, a local search operator, called ITLS, is proposed to combine with RM-MEDA. Besides, in order to maintain well-spread Pareto-optimal solutions, ϵ -dominance is employed in our algorithm, whose name is MMEDA. The proposed algorithm is hybrid with global search and local search, and the success of it is due to the tradeoff between the exploration ability of RM-MEDA and the exploitation ability of ITLS. The details of MMEDA are in Algorithm 2.

5. Two Novel Multiobjective Problems with Variable Linkages

A number of test problems for multiobjective optimization have been designed by some scholars, such as SCH by Schaffer [33], KUR by Kursawe [34], FON by Fonseca and Fleming [35], ZDTs by Zitzler et al. [36], and DTLZs by Deb et al. [37]. These MOPs have been used in a number of significant past studies of EMO to test an optimization methodology. Following Deb's recent review of multiobjective test problems, many of the existing test problems are separable variable-wise or possess linear functions of the variable, which may not provide adequate difficulties to an EMO methodology. Zhang et al. [14] realized the point and proposed ten multiobjective problems with variable linkages (F1~F10 in Table 1).

Analyzing the ten problems proposed by [14], we can get that $\mathbf{x}_i = \mathbf{x}_1$, $i = 2, \dots, 30$, for F1~F4, and $\mathbf{x}_i = \sqrt{\mathbf{x}_1}$, $i = 2, \dots, 30$, for F5~F10, when all the solutions converge to Pareto-optimal fronts. The scheme of introducing variable linkages proposed in the ten problems can be regarded as variable linear or near-linear mapping. Besides, RM-MEDA is based on probability distributions model built by local principal component analysis, which is also a linear or local-linear statistical model. Consequently, RM-MEDA may be efficient to most of these problems. In order to investigate the performance of the optimization methodology, more

problems should be employed and tested. For this end, we introduce twenty-two multiobjective problems with and without variable linkages in our experiment in Table 1.

The variable linkages in our proposed problems are based on the following nonlinear mapping on the variables:

$$\mathbf{x}_1 \longrightarrow \mathbf{x}_1, \quad \mathbf{x}_i \longrightarrow \sin(\pi \mathbf{x}_i) - \mathbf{x}_1, \quad i = 2, \dots, n. \quad (7)$$

In Figure 1, we can obtain that the Pareto set, in the decision space, of a continuous multiobjective optimization problem is a piecewise continuous $(m-1)$ -dimensional manifold, and it seems that the ten problems (F1~F10) may be a simple and special case of the regularity. The problem introduced by nonlinear mapping in (7) seems more accordant with the regularity. Furthermore, there is no common rules of why we choose $\sin()$ function, and other nonlinear function can be used, and more difficulty problems can be proposed by modifying the period of the $\sin()$. The details of two advanced problems are in Table 1 (AF1 and AF2).

6. Experimental Study

6.1. Test Problems. The first five ZDT problems were developed by Zitzler et al. [36] (so called ZDT problems), and the next five DTLZ problems were defined by Deb et al. [37] (so called DTLZ problems). These problems have been cited in a number of significant past studies in EMO community and they can test evolutionary multiobjective optimization algorithms in different aspects. Furthermore, ten problems with variable linkages proposed by Zhang et al. [14] have also been introduced in our paper, which can bring trouble to most of variable-wise EMO methodology, and have been validated by Zhang et al. Lastly, two more difficult problems proposed by ourselves are also presented in Table 1.

It is necessary to note that the performance of an MOEA in tackling multiobjective constrained optimization problems may largely depend on the constraint-handling technique used [38], so we do not mention side-constrained problems in this study.

6.2. Performance Metrics. Zitzler et al. [39] suggested that for a k -objective optimization problem, at least k performances are needed to compare two or more solutions and an infinite number of metrics to compare two or more sets of solutions.

TABLE I: Test instances.

Instance	Variable	Objectives	Number of evaluations
ZDT1	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})} \right]$ $g(\mathbf{x}) = 1 + 9 \left(\sum_{i=2}^n x_i \right) / (n - 1)$	50 000
ZDT2	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - (f_1(\mathbf{x})/g(\mathbf{x}))^2 \right]$ $g(\mathbf{x}) = 1 + 9 \left(\sum_{i=2}^n x_i \right) / (n - 1)$	50 000
ZDT3	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})} - f_1(\mathbf{x})/g(\mathbf{x}) \sin(10\pi x_1) \right]$ $g(\mathbf{x}) = 1 + 9 \left(\sum_{i=2}^n x_i \right) / (n - 1)$	50 000
ZDT4	$x_{1 \in [0,1]}$ $x_{i \in [-5,5]}$ $i = 2, \dots, 10$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})} \right]$ $g(\mathbf{x}) = 1 + 10(n - 1) + \sum_{i=2}^n [x_i^2 - 10 \cos(4\pi x_i)]$	50 000
ZDT6	$[0, 1]^n$ $n = 10$	$f_1(\mathbf{x}) = 1 - \exp(-4x_1) \sin^6(6\pi x_1)$ $f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - (f_1(\mathbf{x})/g(\mathbf{x}))^2 \right]$ $g(\mathbf{x}) = 1 + 9 \left[\left(\sum_{i=2}^n x_i \right) / 9 \right]^{0.25}$	50 000
DTLZ1	$[0, 1]^n$ $n = 7$ $\mathbf{x}_M = [0, 1]^5$	$f_1(\mathbf{x}) = 0.5x_1x_2(1 + g(\mathbf{x}))$ $f_2(\mathbf{x}) = 0.5x_1(1 - x_2)(1 + g(\mathbf{x}))$ $f_3(\mathbf{x}) = 0.5(1 - x_1)(1 + g(\mathbf{x}))$ $g(\mathbf{x}) = 100 \left[\mathbf{x}_M + \sum_{x_i \in \mathbf{x}_M} \left((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5)) \right) \right]$	60 000
DTLZ2	$[0, 1]^n$ $n = 12$	$f_1(\mathbf{x}) = \cos(0.5\pi x_1) \cos(0.5\pi x_2)(1 + g)$ $f_2(\mathbf{x}) = \cos(0.5\pi x_1) \sin(0.5\pi x_2)(1 + g)$ $f_3(\mathbf{x}) = \sin(0.5\pi x_1)(1 + g)$ $g(\mathbf{x}) = \sum_{i=3}^n (x_i - 0.5)^2$	50 000
DTLZ3	$[0, 1]^n$ $n = 12$	$f_1(\mathbf{x}) = \cos(0.5\pi x_1) \cos(0.5\pi x_2)(1 + g)$ $f_2(\mathbf{x}) = \cos(0.5\pi x_1) \sin(0.5\pi x_2)(1 + g)$ $f_3(\mathbf{x}) = \sin(0.5\pi x_1)(1 + g)$ $g(\mathbf{x}) = 100 \left[\mathbf{x}_M + \sum_{i=3}^n \left((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5)) \right) \right]$	50 000
DTLZ4	$[0, 1]^n$ $n = 12$	$f_1(\mathbf{x}) = \cos(0.5\pi x_1^\alpha) \cos(0.5\pi x_2^\alpha)(1 + g)$ $f_2(\mathbf{x}) = \cos(0.5\pi x_1^\alpha) \sin(0.5\pi x_2^\alpha)(1 + g)$ $f_3(\mathbf{x}) = \sin(0.5\pi x_1^\alpha)(1 + g)$ $g(\mathbf{x}) = \sum_{i=3}^n (x_i - 0.5)^2 \quad \alpha = 100$	50 000
DTLZ6	$[0, 1]^n$ $n = 22$ $\mathbf{x}_M = [0, 1]^{20}$	$f_1(\mathbf{x}) = x_1 \quad f_2(\mathbf{x}) = x_2 \quad f_3(\mathbf{x}) = (1 + g(\mathbf{x}_M))h$ $g(\mathbf{x}_M) = 1 + 0.45 \sum x_i \quad x_i \in \mathbf{x}_M$ $h = 3 - f_1 / (1 + g) (1 + \sin(3\pi f_1)) - f_2 / (1 + g) (1 + \sin(3\pi f_2))$	50 000
F1	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = x_1 \quad f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})} \right]$ $g(\mathbf{x}) = 1 + 9 \left(\sum_{i=2}^n (x_i - x_1)^2 \right) / (n - 1)$	15 000
F2	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = x_1 \quad f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - (f_1(\mathbf{x})/g(\mathbf{x}))^2 \right]$ $g(\mathbf{x}) = 1 + 9 \left(\sum_{i=2}^n (x_i - x_1)^2 \right) / (n - 1)$	15 000
F3	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = 1 - \exp(-4x_1) \sin^6(6\pi x_1)$ $f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - (f_1(\mathbf{x})/g(\mathbf{x}))^2 \right]$ $g(\mathbf{x}) = 1 + 9 \left[\left(\sum_{i=2}^n (x_i - x_1)^2 \right) / 9 \right]^{0.25}$	100 000
F4	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = \cos(0.5\pi x_1) \cos(0.5\pi x_2)(1 + g)$ $f_2(\mathbf{x}) = \cos(0.5\pi x_1) \sin(0.5\pi x_2)(1 + g)$ $f_3(\mathbf{x}) = \sin(0.5\pi x_1)(1 + g)$ $g(\mathbf{x}) = \sum_{i=3}^n (x_i - x_1)^2$	35 000
F5	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = x_1 \quad f_2(\mathbf{x}) = g(\mathbf{x}) \left[1 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})} \right]$ $g(\mathbf{x}) = 1 + 9 \left(\sum_{i=2}^n (x_i^2 - x_1^2) \right) / (n - 1)$	15 000

TABLE I: Continued.

Instance	Variable	Objectives	Number of evaluations
F6	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x}) [1 - (f_1(\mathbf{x})/g(\mathbf{x}))^2]$ $g(\mathbf{x}) = 1 + 9 \left(\sum_{i=2}^n (x_i^2 - x_1)^2 \right) / (n - 1)$	15 000
F7	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = 1 - \exp(-4x_1) \sin^6(6\pi x_1)$ $f_2(\mathbf{x}) = g(\mathbf{x}) [1 - (f_1(\mathbf{x})/g(\mathbf{x}))^2]$ $g(\mathbf{x}) = 1 + 9 \left[\left(\sum_{i=2}^n (x_i^2 - x_1)^2 \right) / 9 \right]^{0.25}$	100 000
F8	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = \cos(0.5\pi x_1) \cos(0.5\pi x_2)(1 + g)$ $f_2(\mathbf{x}) = \cos(0.5\pi x_1) \sin(0.5\pi x_2)(1 + g)$ $f_3(\mathbf{x}) = \sin(0.5\pi x_1)(1 + g)$ $g(\mathbf{x}) = \sum_{i=3}^n (x_i^2 - x_1)^2$	35 000
F9	$[0, 1] \times [0, 10]^{n-1}$ $n = 30$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x}) [1 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})}]$ $g(\mathbf{x}) = 0.00025 \sum_{i=2}^n (x_i^2 - x_1)^2 - \prod_{i=2}^n \cos((x_i^2)/\sqrt{i-1}) + 2$	100 000
F10	$[0, 1] \times [0, 10]^{n-1}$ $n = 30$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x}) [1 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})}]$ $g(\mathbf{x}) = 1 + 10(n-1) + \sum_{i=2}^n [(x_i^2 - x_1)^2 - 10 \cos(2\pi(x_i^2 - x_1))]$	100 000
AF1	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = x_1$ $f_2(\mathbf{x}) = g(\mathbf{x}) [1 - \sqrt{f_1(\mathbf{x})/g(\mathbf{x})}]$ $g(\mathbf{x}) = 1 + 9 \left(\sum_{i=2}^n (100 \sin(\pi x_i) - x_1)^2 \right) / (n - 1)$	30 000
AF2	$[0, 1]^n$ $n = 30$	$f_1(\mathbf{x}) = 1 - \exp(-4x_1) \sin^6(6\pi x_1)$ $f_2(\mathbf{x}) = g(\mathbf{x}) [1 - (f_1(\mathbf{x})/g(\mathbf{x}))^2]$ $g(\mathbf{x}) = 1 + 9 \left[\left(\sum_{i=2}^n (6(\sin \pi x_i - x_1))^2 \right) / 9 \right]^{0.25}$	60 000

Coello Coello et al. [8] proposed three issues to allow a quantitative assessment of the performance of a multiobjective optimization algorithm. As we know, it is a common task for any multiobjective optimization algorithm to find solutions as close as possible to the Pareto front and to make them as diverse as possible in the obtained nondominated front. Furthermore, the latter case includes maximizing the spread and the uniformity of solutions found in the final Pareto front. For this end, three metrics are employed in our paper to investigate the performance of the algorithm. As [14], we apply inverted generation distance to the final Pareto-optimal set obtained by an MOEA to evaluate its convergence and spread performance. We also adopt convergence metric proposed by Deb et al. [6] to measure the convergence to the final solutions. Finally, in order to check the uniformity of Pareto-optimal solutions we get in final generation and spacing devised by Schott [40] metric is employed in our paper. The three metrics are summarized as follows.

6.2.1. *Inverted Generation Distance.* Let P^* be a set of uniformly distributed points in the objective space along the PF. Let P be an approximation to the PF; the inverted generational distance from P^* to P is defined as

$$D(P^*, P) = \frac{\sum_{v \in P^*} d(v, P)}{|P^*|}, \quad (8)$$

where $d(v, P)$ is the minimum Euclidean distance between v and the points in P . The inverted generation distance denotes the metric convergence and spread, which represents the distance between the set of the true Pareto-optimal fronts and converged Pareto solutions obtained by EMOAs.

6.2.2. *Convergence Metric.* Metric Y is used to estimate how far the elements in the set of nondominated solutions found so far are from those in the Pareto-optimal set, and it is defined as

$$Y(P, P^*) = \frac{\sum_{v \in P} d(v, P^*)}{|P|}, \quad (9)$$

where $d(v, P^*)$ is the minimum Euclidean distance between v and the points in P^* . Since multiobjective algorithms can be tested on problems having a known set of Pareto-optimal solutions, the calculation of the two metrics is possible.

6.2.3. *Spacing.* Let P be the final approximate Pareto-optimal set obtained by an MOEA. The function S is as follows:

$$S = \sqrt{\frac{1}{h-1} \sum_{i=1}^h (\bar{d} - d_i)^2}, \quad (10)$$

where $d_i = \min_j (|f_1^i(x) - f_1^j(x)| + \dots + |f_m^i(x) - f_m^j(x)|)$, \bar{d} is the mean of all d_i , and h is the number of nondominated solutions found so far. A value of zero for this metric indicates that all members of the Pareto front currently available are equidistantly spaced.

6.3. *The Compared Algorithms.* As Deb et al.'s suggestion in [13], some EMO procedures with variable-wise recombination operators do not perform as well as those with vector-wise operators; besides, PCX-NSGA-II [13] and GDE3 [41] are recommended to handle linkages-based multiobjective optimization problems. Zhang et al. [14] compared RM-MEDA with GDE3, PCX-NSGA-II, and MIDEA [42]. They

concluded that RM-MEDA performed better than GDE3, PCX-NSGA-II, and MIDEA on some test instances with variable linkages, and MIDEA performed slightly badly among the four algorithms. Overall considering former works [13, 14], we choose RM-MEDA, GDE3, and PCX-NSGA-II as the comparisons with MMEDA.

The third evolution version of generalized differential evolution (GDE3) is another updated version of original DE, which modifies earlier GDE version using a growing population and nondominated sorting with pruning of non-dominated solutions to decrease the population size at each generation. The procedure is similar to NSGA-II except that the simulator binary crossover is replaced with a differential evolution operator. The code of GDE3 used in comparison is written in MATLAB by the authors.

Parent-centric recombination (PCX) is a real parameter genetic operator in [43]. Deb et al. [13] introduced the PCX recombination operator in NSGA-II and validated that PCX-based NSGA-II performs better on some problem with variable linkages. The details of PCX-NSGA-II can be found in [13]. The code of PCX-NSGA-II is programmed by ourselves by referring to the code of SBX-NSGA-II and G3PCX from KanGAL (<http://www.iitk.ac.in//kangal/>).

We have discussed so much about RM-MEDA in the former section in our paper. Here, we acknowledge the help of Professor Qingfu Zhang and Dr. Aimin Zhou for sharing the code of RM-MEDA with us and their insightful comments.

6.4. Experimental Setup. In our experiments, the code is programmed in MATLAB. The source code of MMEDA can be obtained from the authors upon request. All the simulations run at a personal computer with P-IV 3.0 G CPU and 2 G RAM. The experimental setting is as follows. Firstly, in GDE3, both CR and F in the differential operator are set to be 1 for F1~F10 and AF1~AF3, which have been investigated by Deb et al. [13] and Zhang et al. [14] on MOPs with variable linkages. Furthermore, by referring to the involved literatures [44, 45], CR and F are set to be 0.2 for ZDTs and DTLZ2, DTLZ4, and DTLZ6 except 0.5 for DTLZ1 and DTLZ3. Secondly, in PCX-NSGA-II, σ in PCX is set to be 0.4 for all the test instances which work well in studies in [13]. Thirdly, in RM-MEDA, all the parameters' setting is the same as the original paper; that is, in local PCA algorithm, K is set to be 5, and extension rate is 0.25. Finally, in MMEDA, the size of tournament pool, n_s , is 30, tournament scale is 2, and the size of active subpopulation, N_s , is 40 for all test instances. Besides, the crossover probability of $p_c = 0.9$ and a mutation probability of $p_m = 1/r$ (where r is the number of decision variables for real-coded GAs) for test instances. We choose $\eta_c = 15$ and $\eta_m = 20$, which are similar to Deb et al.'s setting in [6].

Furthermore, the fourth column in Table 1 is the number of total evaluations for each test instance, respectively, which is followed by some significant past studies in this area [9, 14, 37, 46]. Indexes of the different algorithms are shown in Table 2. The ϵ values for different test instances are described in Table 3, which are similar to ϵ -MOEA [31] and ϵ -ODEMO [47]. Besides, we select 500 for two objective problems and

TABLE 2: Indexes of the different algorithms.

Index	1	2	3	4
Algorithms	MMEDA	RM-MEDA	GDE3	PCX-NSGA-II

1000 for three objective problems with evenly distributed points in Pareto-optimal front, and these points are denoted by P^* .

6.5. Experimental Results of Multiobjective Problems without Variable Linkages. Deb et al. [13] and Zhang et al. [14] have investigated and validated that GDE3, PCX-NSGA-II, and RM-MEDA performed better than other EMOAs on multi-objective problems with variable linkages. However, whether these algorithms could get the same good performance on multiobjective problems without variable linkages is still not investigated so far. For this end, we give the experimental comparison of MMEDA, RM-MEDA, GDE3, and PCX-NSGA-II on famous ZDT and DTLZ problems in this section.

Figure 3 shows the Pareto fronts obtained from our algorithm in a random single run. These problems without variable linkages are rarely tested by estimation of distribution algorithms, and we can see that our method could get fairish results on famous ZDT and DTLZ test instances. Besides, seeing DTLZ2, DTLZ3, and DTLZ4 in Figure 3, we can gain that some extreme points of the Pareto front, as well as points located in segments of the Pareto front that are almost horizontal or vertical, are lost. This is the curse of ϵ -dominance, which has been investigated by Deb et al. [31] and Hernández-Díaz et al. [48].

Next, we investigate statistical results of the four algorithms in our paper in 30 independent runs on each test problem, which are in the form of box plots [49]. In a notched box plot, the notches represent a robust estimate of the uncertainty about the medians for box-to-box comparison. Symbol “+” denotes outlier (Figure 4).

The 30-variable ZDT1 problem has a convex Pareto-optimal front, while ZDT2 has a nonconvex Pareto-optimal front, and ZDT3 provides difficulties by its discontinuities. Many MOEAs have achieved very good results on these problems in both goals of multiobjective optimization (convergence to the true Pareto front and uniform spread of solutions along the front). The results of the problems ZDT1, ZDT2, and ZDT3 (Figures 3 and 4) show that MMEDA achieves good results, which are comparable to the results of RM-MEDA, GDE3, and PCX-NSGA-II; however, MMEDA is not the best in all metrics of the four algorithms. On the first three test problems we cannot see a meaningful difference in performance of the four algorithms. If it is exigent to find which is best, we can see that RM-MEDA is slightly better than the other three algorithms in terms of spacing, while it performs a little poor in convergence metric. Besides, MMEDA and GDE3 are a little better in terms of convergence metric, while they achieve a little worse in spacing metric of these three problems.

ZDT4 is a hard optimization problem with many (21^9) local Pareto fronts that tend to mislead the optimization algorithm. In Figure 5, we can see that RM-MEDA has

TABLE 3: The ϵ values for different test instances.

ZDT1	[0.0075, 0.0075]	DTLZ2	[0.045, 0.045, 0.03]	F4	[0.045, 0.045, 0.03]	F10	[0.0075, 0.0075]
ZDT2	[0.0075, 0.0075]	DTLZ3	[0.045, 0.045, 0.03]	F5	[0.0075, 0.0075]	AF1	[0.0075, 0.0075]
ZDT3	[0.0025, 0.0035]	DTLZ4	[0.045, 0.045, 0.03]	F6	[0.0075, 0.0075]	AF2	[0.0075, 0.0075]
ZDT4	[0.0075, 0.0075]	DTLZ6	[0.035, 0.035, 0.035]	F7	[0.0075, 0.0075]	AF3	[0.0075, 0.0075]
ZDT6	[0.0075, 0.0075]	F1	[0.0075, 0.0075]	F8	[0.045, 0.045, 0.03]		
DTLZ1	[0.02, 0.02, 0.05]	F2	[0.0075, 0.0075]	F9	[0.0075, 0.0075]		

difficulty in locating the global true Pareto-optimal front. Besides, MMEDA, GDE3, and PCX-NSGA-II produce a very similar convergence and diversity measure. Furthermore, it seems that MMEDA demonstrates the best in terms of convergence and diversity by box plots in small scale.

With respect to the 10-variable test problem ZDT6, there are two major difficulties. The first one is thin density of solutions towards the Pareto front and the second one is nonuniform spread of solutions along the front. The Pareto-optimal solutions of DTLZ1 with $(11^5 - 1)$ local Pareto fronts lie on a three-dimensional plane satisfying $f_1 + f_2 + f_3 = 0.5$ (Figure 3). In Figure 5, we can obtain that MMEDA and GDE3 achieve good convergence and diversity measures of ZDT6, followed by PCX-NSGA-II and RM-MEDA, while, for DTLZ1, it seems that MMEDA gives both the best convergence and diversity measure in the four algorithms.

On the whole, we can obtain that MMEDA seems to be the best algorithm on these three problems. However, if we do not employ local searcher, EDAs based on probability global statistical information may perform weakly in approximating the Pareto-optimal fronts of the three problems.

Problems of DTLZ2, DTLZ3, and DTLZ4 are three-objective test instances with spherical Pareto-optimal front (see Figure 3). Note that DTLZ3 has lots of local Pareto fronts, DTLZ4 emphasizes nonuniformity, and DTLZ6 has 2^{19} disconnected Pareto-optimal regions in the search space. Figure 6 shows the performance metrics of the four algorithms on the four problems, respectively. We can find that MMEDA is the best in terms of both convergence and diversity on DTLZ3 and DTLZ4. Furthermore, for DTLZ3, MMEDA performs much better than the other three algorithms even though DTLZ3 has $(3^{10} - 1)$ local Pareto-optimal fronts. Gong et al. [9], Deb et al. [37], and Khare et al. [46] claimed that, for DTLZ3, NSGA-II, SPEA2, and PESA-II could not quite converge on to the true Pareto-optimal fronts in 500 generations (50 000 function evaluations). In our paper, we have found that GDE3, PCX-NSGA-II, and RM-MEDA also did badly in solving DTLZ3.

For convergence metric, it seems that GDE3 achieves the best measure in the four algorithms on DTLZ2 and DTLZ6. Besides, for diversity metric, MMEDA is the best on the four problems, which is the strongpoint of ϵ -dominance. Since the ϵ -dominance does not allow two solutions with a difference of ϵ_i in the i th objective to be mutually nondominated to each other, it will be usually not possible to obtain the extreme corners of the Pareto-optimal front. Although there is such shortcoming of ϵ -dominance, we could still get better spacing

measurement than that of RM-MEDA, GDE3, and PCX-NSGA-II, which are based on crowding distance proposed by Deb et al. [6].

Overall considering the ten famous two and three objective problems without variable linkages, we can conclude that MMEDA produces a good convergence and diversity metrics with the exception of ZDT1, ZDT2, and ZDT3. GDE3 obtains comparative results for most of problems except DTLZ1 and DTLZ3. Besides, PCX-NSGA-II and RM-MEDA are not better than MMEDA and GDE3 on most of the ten problems. However, the mechanism of RM-MEDA is based on global statistical information, and strongpoints of RM-MEDA are discovering linkages of variables and holding the spread of the final solutions. Besides, there are no variable linkages in the former ten problems. Therefore, it is not fair for RM-MEDA. In [14], it has been validated that RM-MEDA performs better than GDE3 and PCX-NSGA-II on some multiobjective problems with variable linkages. In the next subsection, we will give experimental results of our algorithms compared with the other three algorithms on twelve multiobjective problems with variable linkages.

6.6. Experimental Results of Multiobjective Problems with Variable Linkages. In this subsection, experimental results of twelve multiobjective problems with variable linkage are presented. The representation of experimental results is similar to that of Section 6.5; that is, 30 independent runs are performed on each test problem. The statistical results of our selected three metrics are shown by box plots too.

Figure 7 shows the Pareto fronts obtained by our algorithm on F1~F10 and AF1, AF2 in a random single run. We can see that MMEDA obtains good results of F1, F2, F3, F4, F5, F6, F9, AF1, and AF2. However, MMEDA performs a little poor on F7, F8, and F10, especially on F10, of which MMEDA cannot converge to the true Pareto-optimal fronts. In [14], RM-MEDA is also trapped in local Pareto front on F10. Furthermore, for GDE3 and PCX-NSGA-II, they converge to a small region of global Pareto fronts of F10, which is called "genetic drift." Next, we will give experimental comparison with MMEDA, RM-EDA, GDE3, and PCX-NSGA-II by box plots for further research.

Since F1 and F5 are variants of ZDT1 by introducing linear and nonlinear variable linkages into them, respectively, we put the statistical results of them in the same figure (Figure 8) for comparison, which are similar to F2 and F6. Note that the metric inverted generation distance should be first choice for comparison because inverted generation

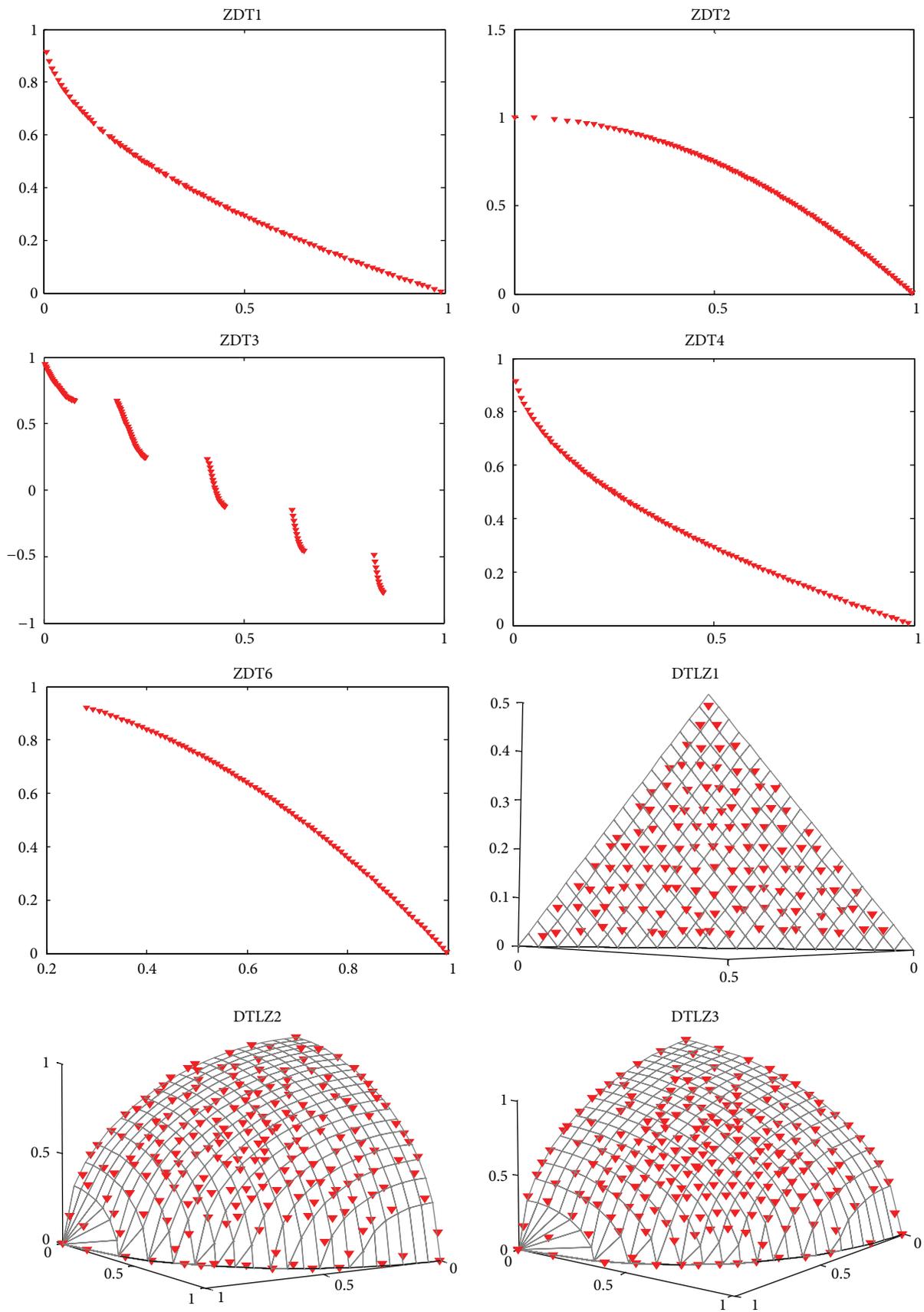


FIGURE 3: Continued.

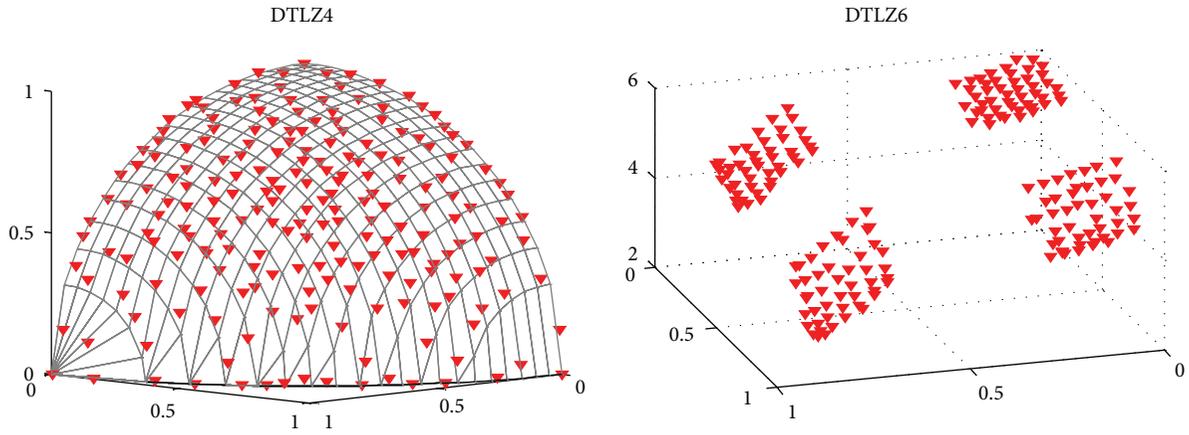


FIGURE 3: Pareto fronts obtained by MMEDA on ZDT and DTLZ problems in our paper, respectively.

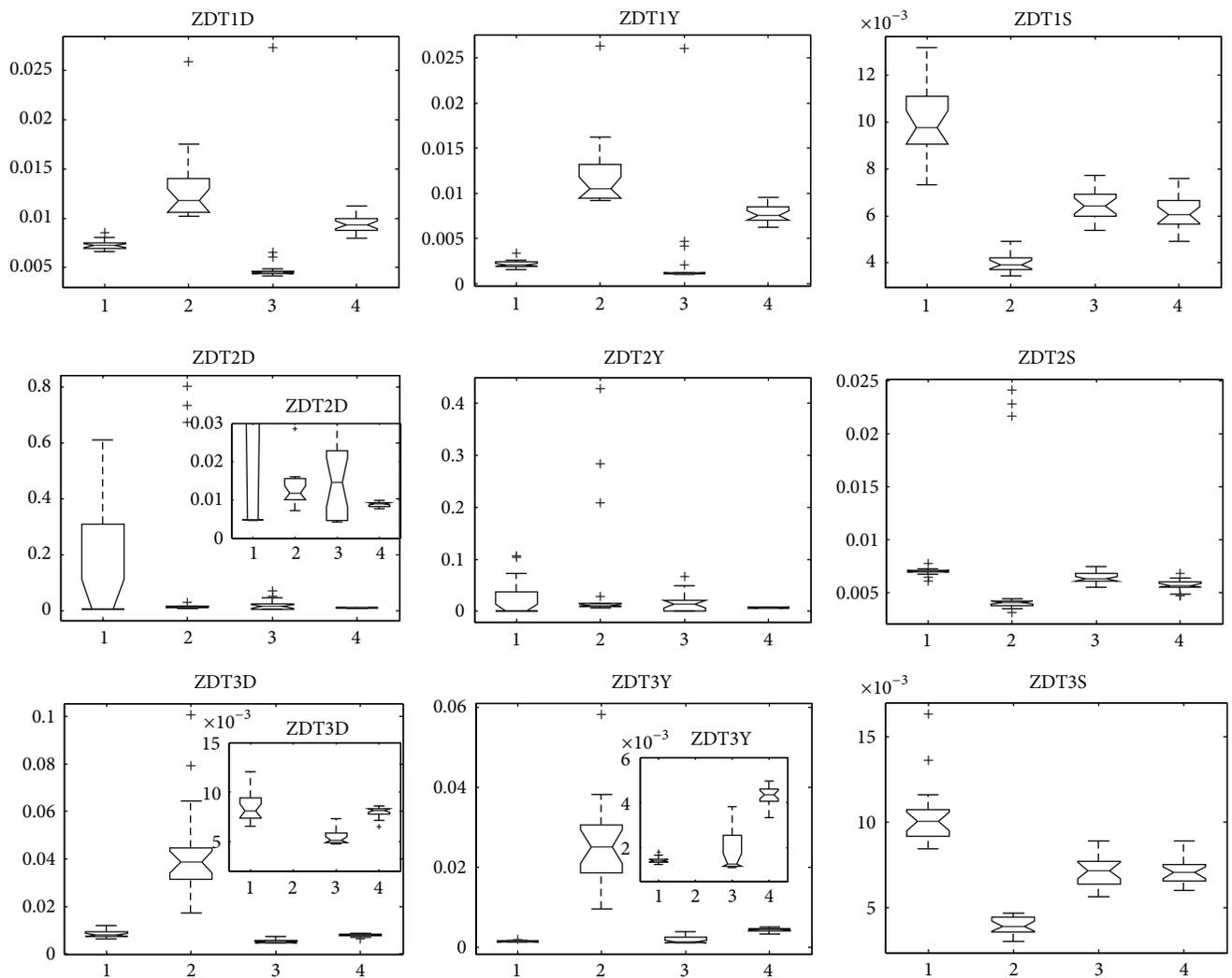


FIGURE 4: Statistical values of inverted generation distance (“D” in ZDTXD for short), convergence (“Y” in ZDTXY for short), and spacing (“S” in ZDTXS for short) for ZDT1, ZDT2, and ZDT3 obtained by MMEDA, RM-MEDA, GDE3, and PCX-NSGA-II, respectively.

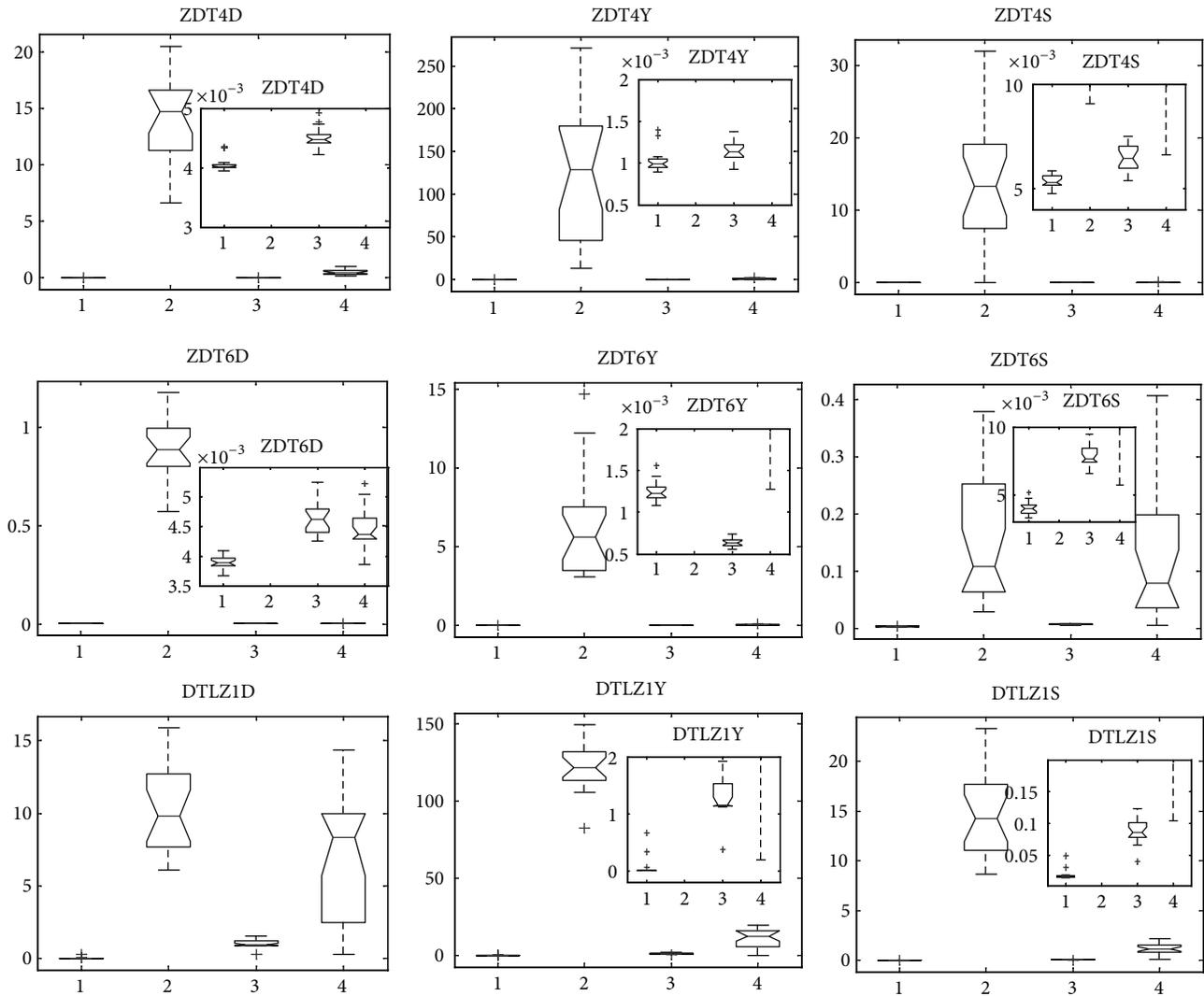


FIGURE 5: Statistical values of inverted generation distance (“D” for short), convergence (“Y” for short), and spacing (“S” for short) for ZDT4, ZDT6, and DTLZ1 obtained by MMEDA, RM-MEDA, GDE3, and PCX-NSGA-II, respectively.

distance can measure both convergence and spread. If we ignore inverted generation distance, we cannot get rational and comprehensive comparison. Taking F6, for example, if we only concentrate on the metric convergence and spacing, it seems that PCX-NSGA-II is the best choice for this problem. Nevertheless, analyzing F6D in Figure 8, the result of F6 is in a small region of whole Pareto- optimal front, which is called “genetic drift” by Goldberg and Segrest [50] and Fonseca and Fleming [51]. Zhang et al. got the same results of PCX-NSGA-II on F6 [14]. For this end, inverted generation distance is foundational one among the three metrics.

In terms of convergence metric of F1 and F2, MMEDA obtains best performance, while it performs a little poor in spacing metric. Furthermore, MMEDA and RM-MEDA perform better than GDE3 and PCX-NSGA-II on the four problems in terms of spread and convergence. Although

PCX-NSGA-II seems better in spacing metric on F5 and F6, it experiences difficulty in spread.

With respect to F3 and F7, since they are variants of ZDT6 by introducing linear and nonlinear variable linkages into them, respectively, there are three major difficulties of them. The first two difficulties are the same as ZDT6 as described above, while the third one is linkage relations between variable mappings. Therefore, F3 and F7 have more difficulty than F1, F2, F5, and F6. Figure 9 shows the performance measures on F3 and F7. We can obtain that RM-MEDA and MMEDA achieve better performance than GDE3 and PCX-NSGA-II in terms of spread, convergence, and uniformity. This might be the reason why GDE3 and PCX-NSGA-II have no efficient mechanism for discovering variable relation, yet RM-MEDA and MMEDA hold the property by building the probability distribution model based on manifold distribution of Pareto

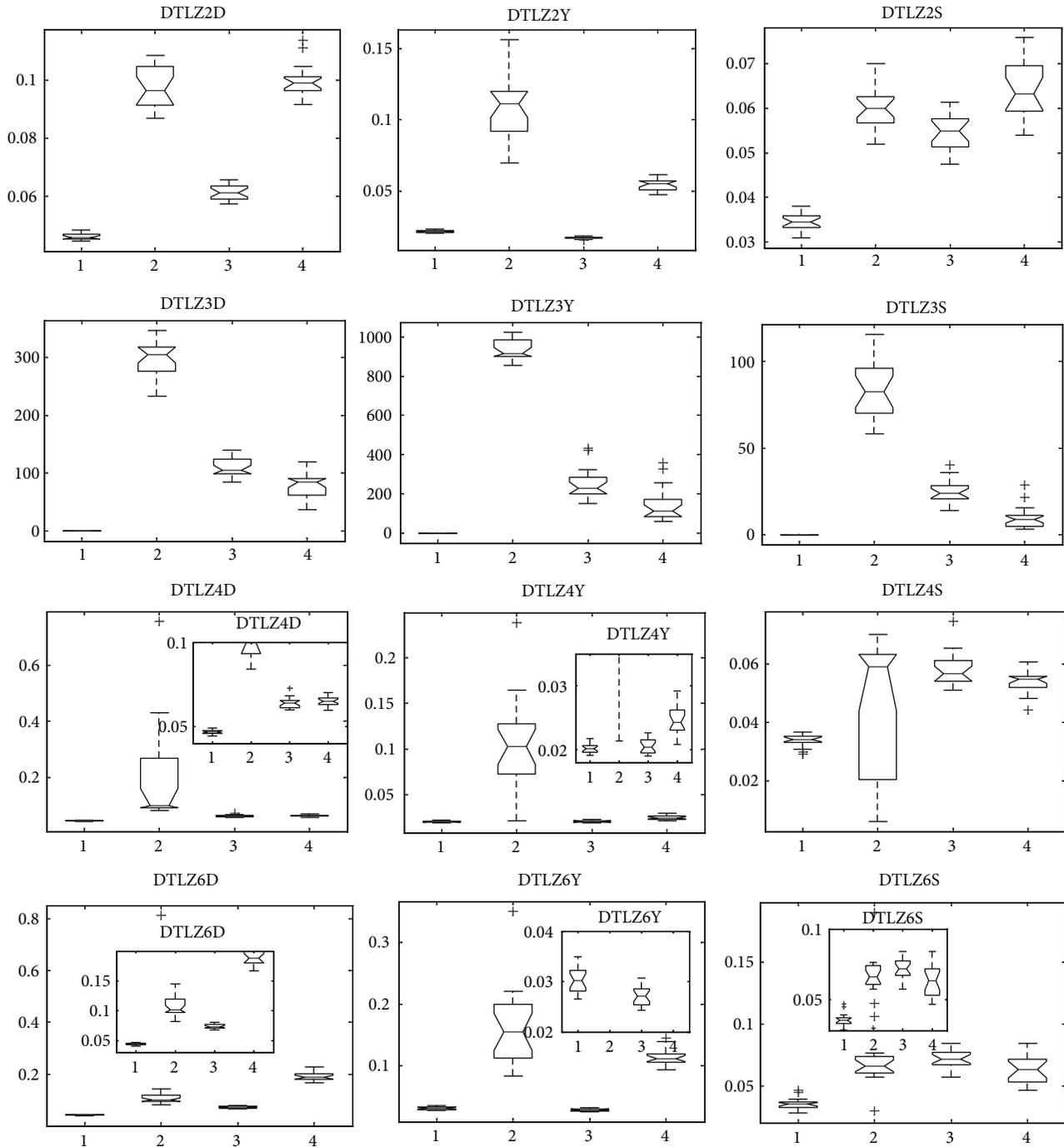


FIGURE 6: Statistical values of inverted generation distance (“D” in DTLZXD for short), convergence (“Y” in DTLZXY for short), and spacing (“S” in DTLZXS for short) for DTLZ2, DTLZ3, DTLZ4, and DTLZ6 obtained by MMEDA, RM-MEDA, GDE3, and PCX-NSGA-II, respectively.

set in decision space, which have been described in detail in Section 3. Finally, it seems that RM-MEDA is slightly better than MMEDA on the two problems.

F4 and F8 are variants of DTLZ2, and Figure 10 is the statistical results of them. We can obtain that MMEDA is slightly better than other three algorithms in our paper. Besides, GDE3 shows comparative results on F4, followed by

RM-MEDA. In [14], Zhang et al. admitted that GDE3 slightly outperforms better than RM-MEDA on some problems and pointed out that the reason might be that RM-MEDA does not directly use the local information of previous solutions in generating new solutions. Here, we give one possible answer, MMEDA, the hybridization of RM-MEDA and an efficient local searcher. For F8, GDE3 and PCX-NSGA-II both

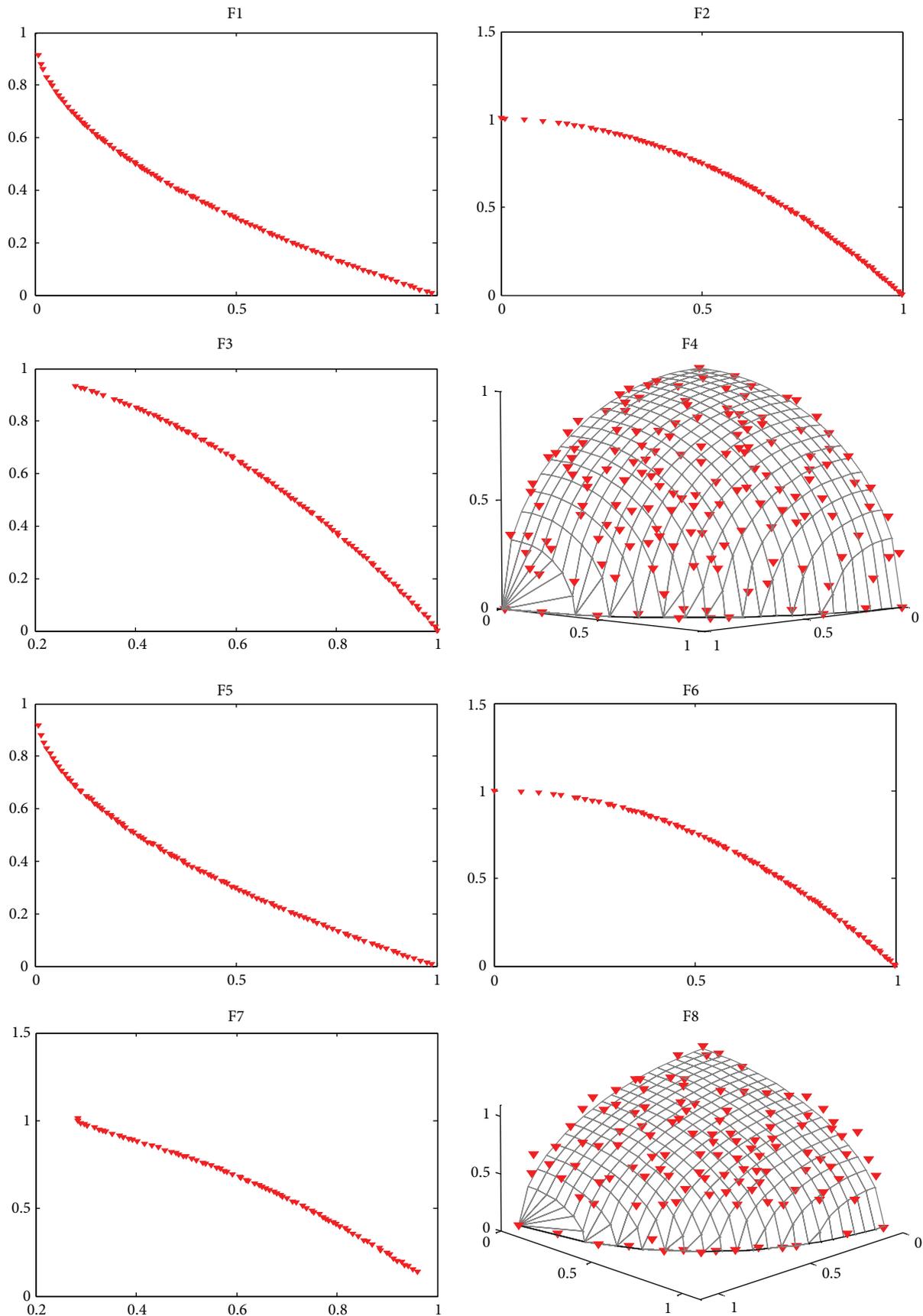


FIGURE 7: Continued.

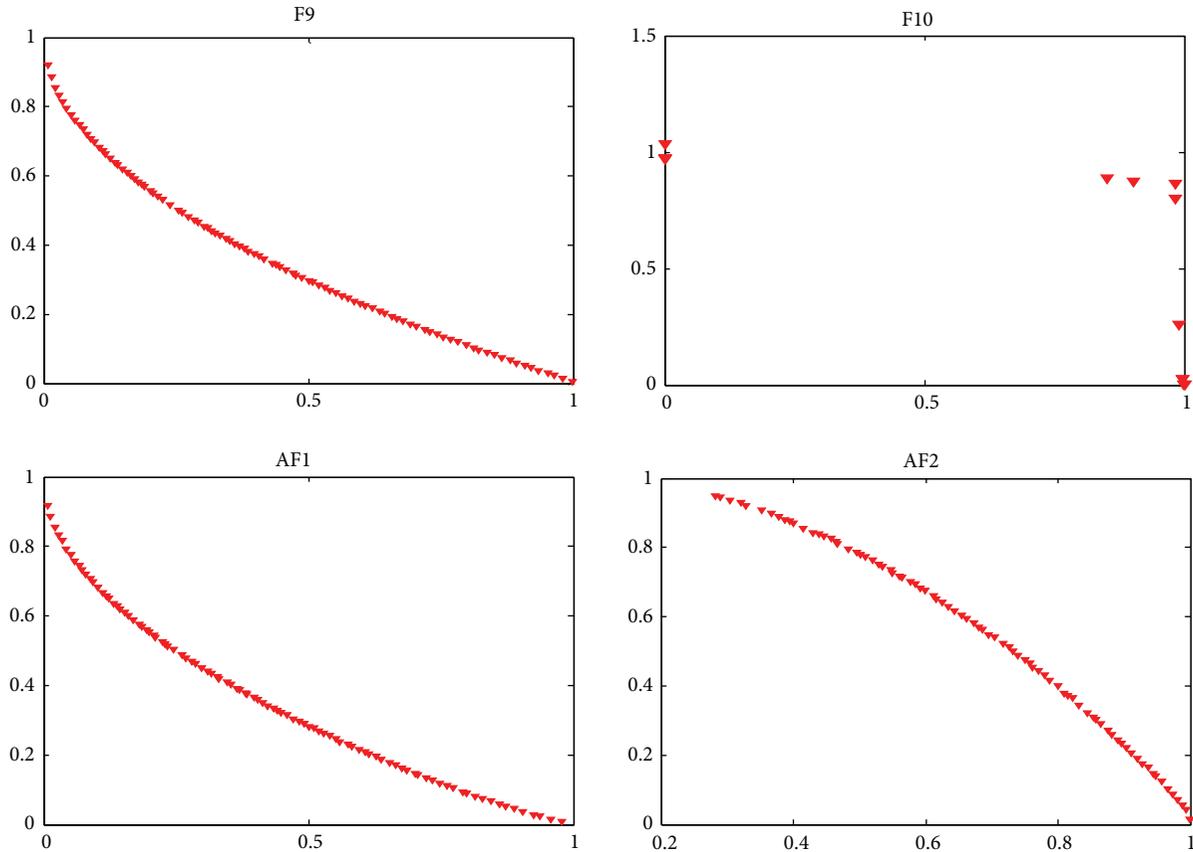


FIGURE 7: Pareto fronts obtained by MMEDA on F1~F10 and AF1, AF2 in our paper, respectively.

converge to a curve and several points, which is “genetic drift” in multiobjective optimization. Figure 11 is the result of a single run of GDE3, and PCX-NSGA-II has similar results with GDE3. In multiobjective optimization, genetic drift means that finite population tends to converge to small regions of efficient set.

F9 is a variant of multimodal with Griewank function, while F10 is a variant of multimodal with Rastrigin function, and all of them are introduced into nonlinear mapping on variables. Figure 12 is the statistical results of them by the four algorithms in our study. We can obtain that MMEDA and RM-MEDA get very similar performance of F9, and they get better measurement than GDE3 and PCX-NSGA-II. However, with respect to F10, it seems that MMEDA and PCX-NSGA-II are the best two algorithms in terms of the three metrics. However, it is not true to them. In Figure 7 (F10), we can see that all the solutions obtained by MMEDA converge to a small region near the global Pareto-optimal front, which is similar to PCX-NSGA-II [14]. They are trapped in “genetic drift.” Besides, the final Pareto fronts obtained by RM-MEDA and GDE3 on F10 are far from the Pareto-optimal front, which can be validated by inverted generation distance (F10D in Figure 12) and convergence metric (F10Y in Figure 12). They are stagnated at local Pareto fronts. In a word, we have to admit that all the four algorithms cannot solve the problem efficiently.

By comparison, we can see that MMEDA is best in convergence and spread on AF1 and AF2, followed by RM-MEDA and GDE3. For PCX-NSGA-II (Figure 13), it cannot converge to the true Pareto-optimal front of the two problems. From these two problems, we can see that MMEDA, which hybrids with ITLS, is a competitive algorithm for some problems.

However, overall considering the experimental results of the twelve problems with variable linkages, RM-MEDA and MMEDA are two more efficient and effective algorithms than traditional EAs for most of the problems. For EDAs, this new class of algorithms generalizes genetic algorithms by replacing the crossover and mutation operators by learning and sampling the probability distribution of the promising individuals of population, and the relationships between the variables involved in the problem domain are explicitly and effectively captured and exploited. Besides, the regularity that the distribution of Pareto set in the decision space is a piecewise continuous $(m - 1)$ -dimensional manifold is hybrid into MMEDA and RM-MEDA. Therefore, we can conclude that MMEDA and RM-MEDA should be better GDE3 and PCX-NSGA-II. Besides, from two problems (F3, F7) we can see that RM-MEDA is better than MMEDA in all metrics. However, in terms of convergence, MMEDA is slightly better than RM-MEDA on nine problems (F1, F2, F4, F5, F6, F8, F10, AF1, and AF2) and the effectiveness of our ITLS seems

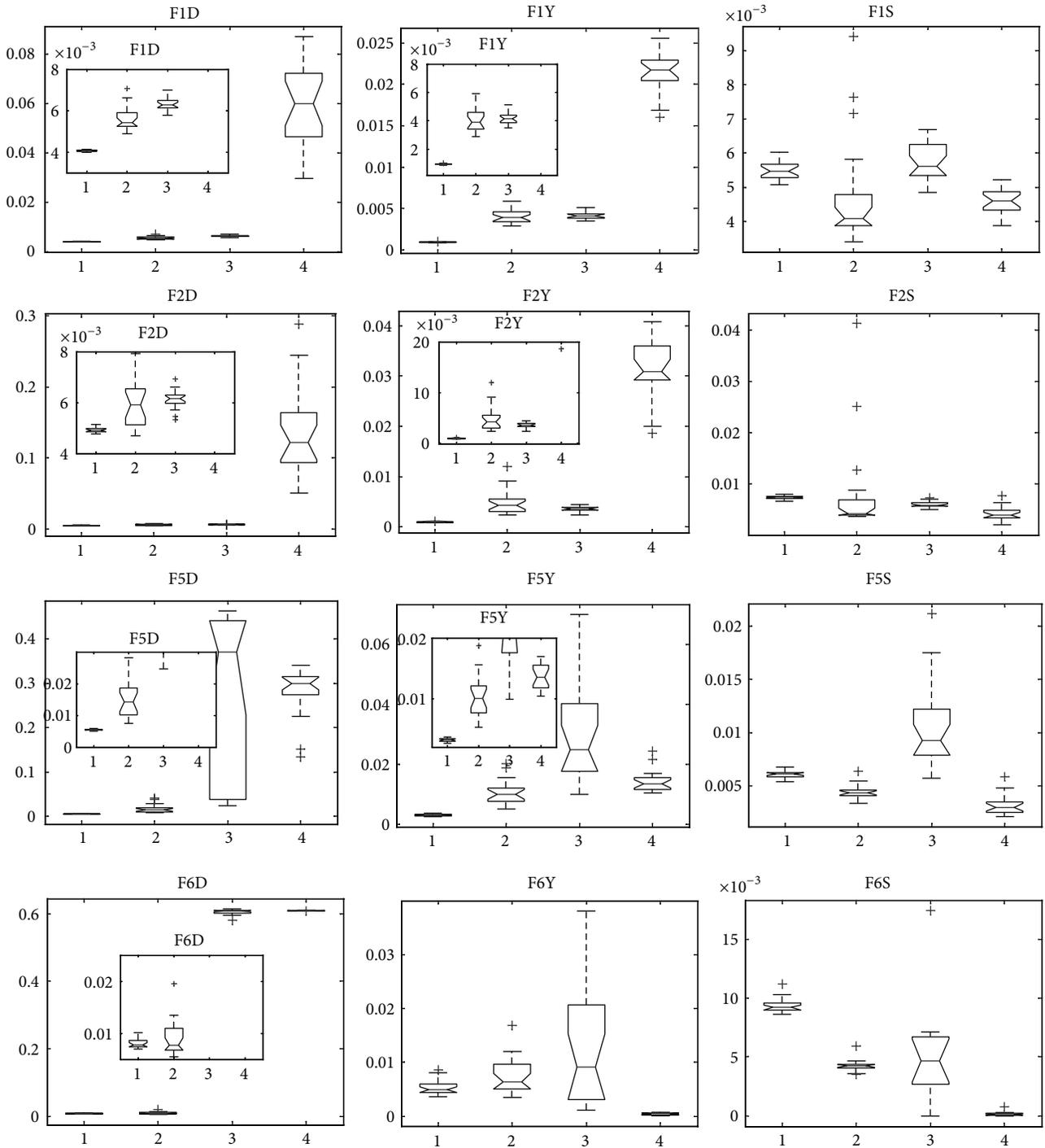


FIGURE 8: Statistical values of inverted generation distance (“D” in FXD for short), convergence (“Y” in FXY for short), and spacing (“S” in FXS for short) for F1, F5, F2, and F6 obtained by MMEDA, RM-MEDA, GDE3, and PCX-NSGA-II, respectively.

to play a very significant role for the above nine problems. But, for seven problems (F1~F3, F5~F7, and F9), RM-MEDA outperforms MMEDA in terms of spacing metric.

As we know, convergence to the true Pareto-optimal front and maintenance of a well-distributed set of nondominated solutions are two challenges for multiobjective optimization community. However, if solutions obtained by an algorithm

cannot converge to the true Pareto-optimal fronts, they are stagnated in local optimal fronts. It is useless to consider uniformity. Besides, if solutions obtained by an EMOA only converge to a small region of the whole Pareto-optimal front, they have trouble in “genetic drift.” We cannot make a comprehensive view of the EMOA if we only take convergence and spacing metrics into account. Since the inverted

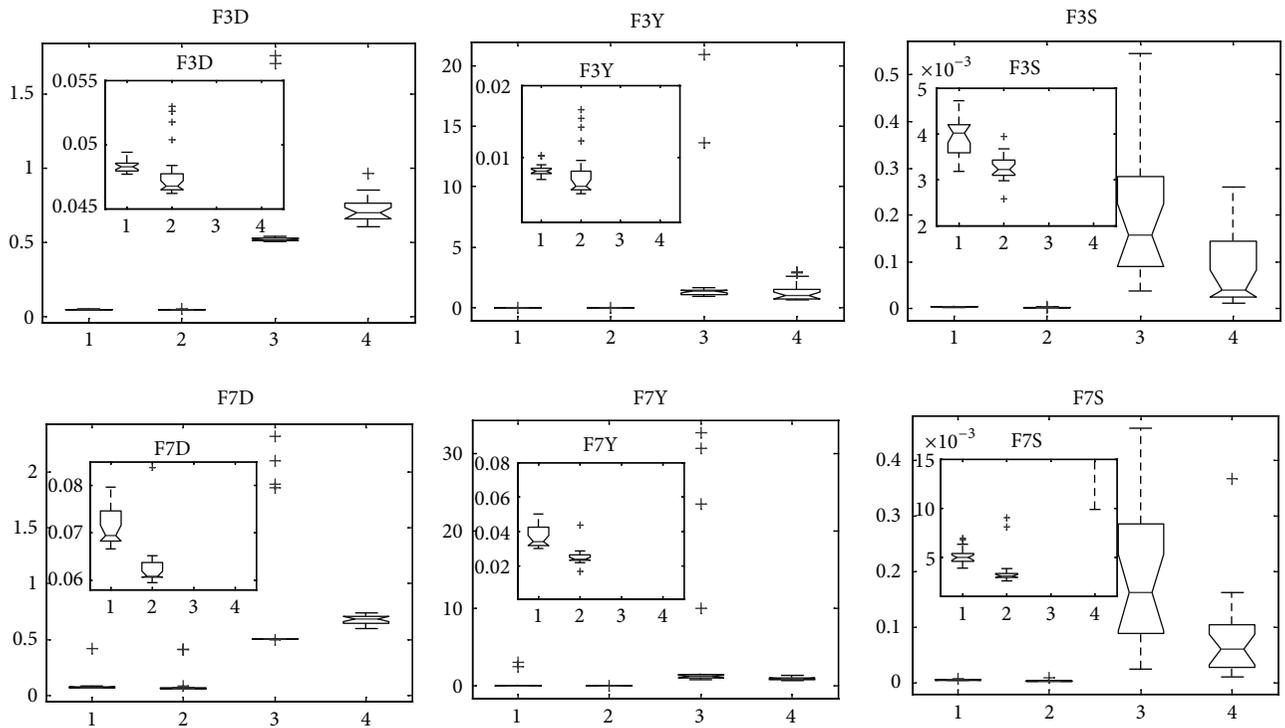


FIGURE 9: Statistical values of inverted generation distance (“D” in FXD for short), convergence (“Y” in FXY for short), and spacing (“S” in FXS for short) for F3 and F7 obtained by MMEDA, RM-MEDA, GDE3, and PCX-NSGA-II, respectively.

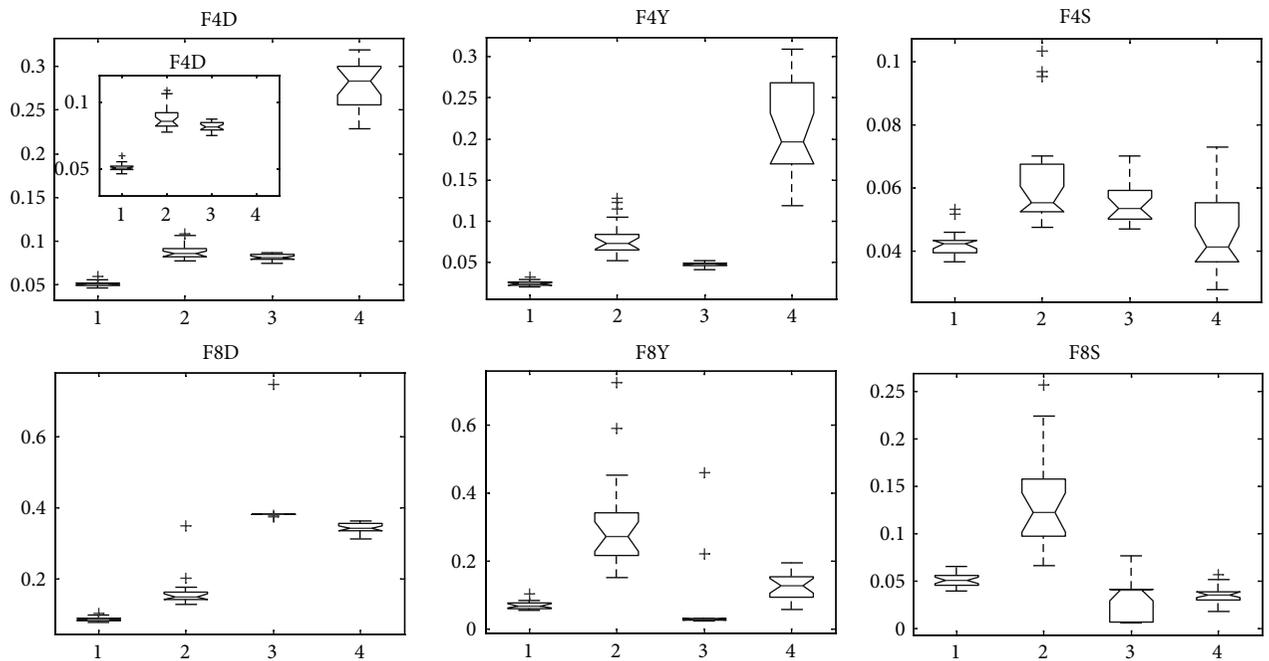


FIGURE 10: Statistical values of inverted generation distance (“D” in FXD for short), convergence (“Y” in FXY for short), and spacing (“S” in FXS for short) for F4 and F8 obtained by MMEDA, RM-MEDA, GDE3, and PCX-NSGA-II, respectively.

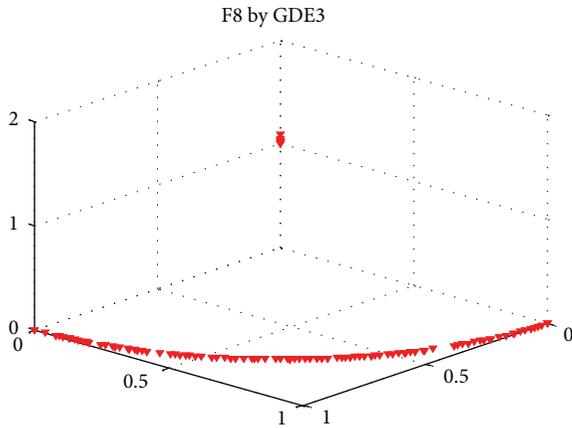


FIGURE 11: Pareto fronts obtained by GDE3 on F8 in a single run.

generation can measure both convergence and spread of solutions obtained by an EMOA, it is used as remedy for the shortcomings of generation distance and spacing metrics. Next, we judge the statistical results by combined metrics.

Finally, with respect to the twenty-two multiobjective problems with and without variable linkages, we can conclude the following.

- (1) In terms of convergence and spread, MMEDA achieves the best measurement among the four algorithms on ZDT4, DTLZ1, DTLZ3, DTLZ4, F1, F2, F4, F5, F8, F9, AF1, and AF2 out of the twenty-two problems while RM-MEDA performs best on F3, F6, and F7. GDE3 does best on ZDT1, ZDT3, ZDT6, DTLZ2, and DTLZ6. PCX-NSGA-II seems to obtain the best result of ZDT2.
- (2) In terms of spread and spacing, MMEDA performs the best performance on ZDT4, ZDT6, DTLZ1, DTLZ2, DTLZ3, DTLZ4, DTLZ6, F4, F8, AF1, and AF2, while RM-MEDA is the best on ZDT1, ZDT2, ZDT3, F1, F3, F5, F6, F7, and F9. GDE3 achieves the best on F2.

Overall considering the three metrics, we can get that MMEDA turns out to be the best compromise among the four MOEAs considered in this study. However, MMEDA is possibly trapped in “genetic drift,” and we can get it from the results of F10 in Figure 3. Therefore, much work is needed to balance the exploration ability of global model and the exploitation ability of local searcher, or more efficient EDAs based on other machine learning techniques should be proposed.

In one word, depending on these empirical results and detailed analysis, we can draw that MMEDA is an efficient algorithm in solving multiobjective optimization problems with and without variable linkages.

7. Concluding Remarks

We have proposed a novel multiobjective algorithm based on the manifold distribution of Pareto set in the decision space,

an efficient local searcher (ITLS), and ϵ -dominance. Besides, two more difficult multiobjective problems by introducing nonlinear mapping into variable linkages are proposed. Following Deb et al. and Zhang et al.’s recent review of current evolutionary multiobjective optimization fields [13, 14] multiobjective problems with variable linkages have more difficulty for most current state-of-the-art MOEAs. Consequently, in order to test our proposed algorithm, MMEDA, twenty-two multiobjective problems with and without variable linkages are employed, and RM-MEDA, GDE3, and PCX-NSGA-II are used in comparison. The results show that our algorithms get competitive results in terms of convergence and diversity metrics.

However, there is much work to do in multiobjective optimization community. An issue that should be addressed in the future research is ϵ -dominance. In our study, it seems that spacing metric of some two-objective problems (such as ZDT1 and ZDT3) is not better than PCX-NSGA-II and GDE3, which are based on crowding distance in SBX-NSGA-II [6], and Deb et al. [31] have validated by experiments that ϵ -MOEAs get better spacing metric than NSGA-II based on crowding distance. There are two major reasons of this. The first is that we introduce a local searcher (ITLS) and the number of iterations will be less than the other three algorithms in our study under the same limit of total evaluations. The second is that some extreme points of the Pareto front, as well as points located in segments of the Pareto front that are almost horizontal or vertical, are lost. This is the curse of ϵ -dominance. Some scholars have proposed Pareto adaptive ϵ -dominance [48], which is an advanced version of ϵ -dominance. But the loss of extreme solutions still exists. So much work is still needed to propose new version of dominance.

Another important topic is “genetic drift,” which has been studied by some researchers, such as Goldberg and Segrest [50], Fonseca and Fleming [51], and Srinivas and Deb [52]. At first, fitness sharing techniques are proposed to prevent genetic drift in multimodal function optimization and multiobjective optimization [52]. Later on, fitness assignment and crowding distance are proposed by Deb et al. [6] and Zitzler et al. [7] for diversity maintenance. Nowadays, some state-of-the-art MOEAs have still trouble in “genetic drift” on some multiobjective problems with variable linkages [13, 14]. So how we can design new efficient and effective scheme to maintain diversity is an important issue of modern EMO community.

How to advance the local search ability of modern MOEAs is another worthwhile work. Gong et al. [9], Deb et al. [37], and Khare et al. [46] validated that, for DTLZ3, NSGA-II, SPEA2, and PESA-II, which are representative of state-of-the-art MOEAs in multiobjective optimization community, could not quite converge on to the true Pareto-optimal fronts in 500 generations (50 000 function evaluations). We have found that GDE3, PCX-NSGA-II, and RM-MEDA also did badly in solving DTLZ3. DTLZ3 may be only one case of problems with many local Pareto fronts, and, possibly, there are many complicated problems like DTLZ3. Therefore, it requires designing more efficient and effective algorithms. Memetic algorithm may be one potential answer to the

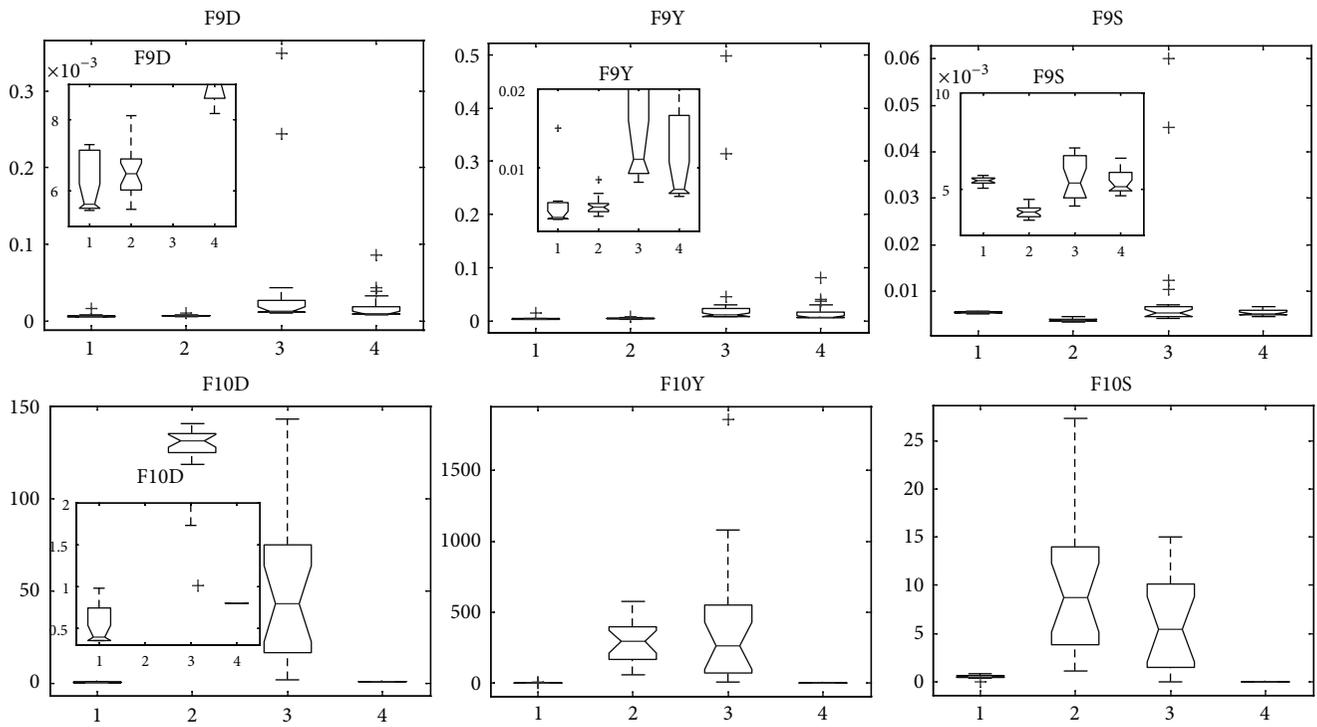


FIGURE 12: Statistical values of inverted generation distance (“D” in FXD for short), convergence (“Y” in FXY for short), and spacing (“S” in FXS for short) for F9 and F10 obtained by MMEDA, RM-MEDA, GDE3, and PCX-NSGA-II, respectively.

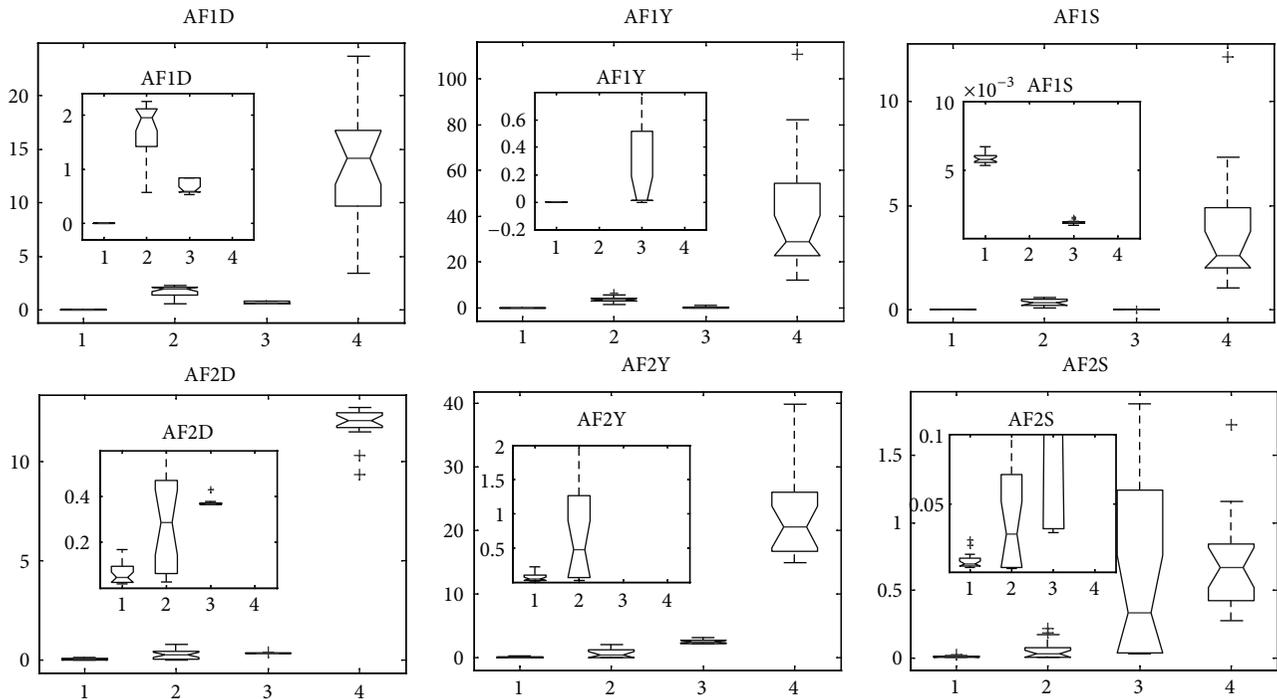


FIGURE 13: Statistical values of inverted generation distance (“D” in AFXD for short), convergence (“Y” in AFXY for short), and spacing (“S” in AFXS for short) for AF1 and AF2 obtained by MMEDA, RM-MEDA, GDE3, and PCX-NSGA-II, respectively.

problem, and several multiobjective memetic algorithms have been proposed by some scholars [27, 29, 53], which have shown comparative performance than traditional EMOAs. A more efficient and effective hybrid algorithm should be addressed in the future research.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grants nos. 61100173, 61100009, 61272283, 61201416, and 61304082) and the Foundation of Xi'an University of Technology (no. 116-211306).

References

- [1] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.
- [2] J. D. Knowles and D. W. Corne, "Approximating the non-dominated front using the pareto archived evolution strategy," *Evolutionary Computation*, vol. 8, no. 2, pp. 149–172, 2000.
- [3] D. W. Corne, J. D. Knowles, and M. J. Oates, "The Pareto-envelope based selection algorithm for multiobjective optimization," in *Parallel Problem Solving from Nature PPSN VI*, vol. 1917 of *Lecture Notes in Computer Science*, pp. 869–878, Springer, New York, NY, USA, 2000.
- [4] D. W. Corne, N. R. Jerram, J. D. Knowles, and M. J. Oates, "PESA-II: region-based selection in evolutionary multiobjective optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '01)*, pp. 283–290, Morgan Kaufmann Publishers, San Francisco, Calif, USA, 2001.
- [5] C. A. Coello Coello and G. T. Pulido, "Multiobjective optimization using a micro-genetic algorithm," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '01)*, pp. 274–282, Morgan Kaufmann, San Francisco, Calif, USA, 2001.
- [6] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.
- [7] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: improving the strength Pareto evolutionary algorithm," in *Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems*, pp. 95–100, Athens, Greece, 2002.
- [8] C. A. Coello Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 256–279, 2004.
- [9] M. Gong, L. Jiao, H. Du, and L. Bo, "Multiobjective immune algorithm with nondominated neighbor-based selection," *Evolutionary Computation*, vol. 16, no. 2, pp. 225–255, 2008.
- [10] Y. T. Qi, X. L. Ma, F. Liu, L. C. Jiao, J. Y. Sun, and J. S. Wu, "MOEA/D with adaptive weight adjustment," *Evolutionary Computation*, vol. 22, no. 2, pp. 231–264, 2014.
- [11] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhangd, "Multiobjective evolutionary algorithms: a survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 32–49, 2011.
- [12] C. A. Coello Coello, "Evolutionary multi-objective optimization: a historical view of the field," *IEEE Computational Intelligence Magazine*, vol. 1, no. 1, pp. 28–36, 2006.
- [13] K. Deb, A. Sinha, and S. Kukkonen, "Multi-objective test problems, linkages, and evolutionary methodologies," in *Proceeding of the 8th Annual Genetic and Evolutionary Computation Conference (GECCO '06)*, pp. 1141–1148, Washington, DC, USA, July 2006.
- [14] Q. F. Zhang, A. M. Zhou, and Y. C. Jin, "RM-MEDA: a regularity model based multiobjective estimation of distribution algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 41–63, 2008.
- [15] N. Khan, *Bayesian optimization algorithms for multiobjective and hierarchically difficult problems [M.S. thesis]*, University of Illinois at Urbana-Champaign, Urbana, Ill, USA, 2003.
- [16] M. Pelikan, D. E. Goldberg, and E. Cant-Paz, "BOA: the Bayesian optimization algorithm," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '99)*, W. Banzhaf, Ed., Illigal Report No. 99003, pp. 525–532, Morgan Kaufmann, San Francisco, Calif, USA, 1999.
- [17] M. Pelikan, *Hierarchical Bayesian Optimization Algorithm: Toward a New Generation of Evolutionary Algorithms*, Springer, New York, NY, USA, 2005.
- [18] P. A. N. Bosman and D. Thierens, "Multi-objective optimization with diversity preserving mixture-based iterated density estimation evolutionary algorithms," *International Journal of Approximate Reasoning*, vol. 31, no. 3, pp. 259–289, 2002.
- [19] M. Laumanns and J. Ocenasek, "Bayesian optimization algorithm for multiobjective optimization," in *Parallel Problem Solving from Nature—PPSN VII*, Lecture Notes in Computer Science, pp. 298–307, Springer, 2009.
- [20] A. Zhou, F. Gao, and G. Zhang, "A decomposition based estimation of distribution algorithm for multiobjective traveling salesman problems," *Computers & Mathematics with Applications*, vol. 66, no. 10, pp. 1857–1868, 2013.
- [21] O. Schütze, S. Mostaghim, M. Dellnitz, and J. Teich, "Covering Pareto sets by multilevel evolutionary subdivision techniques," in *Proceedings of the 2nd International Conference on Evolutionary Multi-Criterion Optimization (EMO '03)*, vol. 2632 of *Lecture Notes in Computer Science*, pp. 118–132, Faro, Portugal, 2003.
- [22] Y. Jin and B. Sendhoff, "Connectedness, regularity and the success of local search in evolutionary multiobjective optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC '03)*, pp. 1910–1917, Canberra, Australia, 2003.
- [23] A. M. Zhou, Q. F. Zhang, Y. C. Jin, B. Sendhoff, and E. Tsang, "Global multiobjective optimization via estimation of distribution algorithm with biased initialization and crossover," in *Proceedings of the 9th Annual Genetic and Evolutionary Computation Conference (GECCO '07)*, pp. 617–623, London, UK, July 2007.
- [24] S. Jiang, Y. S. Ong, J. Zhang, and L. Feng, "Consistencies or contradictions of performance metrics in multiobjective optimization," *IEEE Transactions on Cybernetics*, 2014.
- [25] Y. S. Ong and A. J. Keane, "Meta-lamarckian learning in memetic algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 2, pp. 99–110, 2004.

- [26] A. Jaszkievicz, "On the performance of multiple-objective genetic local search on the 0/1 knapsack problem—a comparative experiment," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 402–412, 2002.
- [27] H. Ishibuchi, T. Yoshida, and T. Murata, "Balance between genetic search and local search in memetic algorithms for multi-objective permutation flowshop scheduling," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 204–223, 2003.
- [28] Y. Xu, R. Qu, and R. Li, "A simulated annealing based genetic local search algorithm for multi-objective multicast routing problems," *Annals of Operations Research*, vol. 206, no. 1, pp. 527–555, 2013.
- [29] A. Jaszkievicz, "Genetic local search for multi-objective combinatorial optimization," *European Journal of Operational Research*, vol. 137, no. 1, pp. 50–71, 2002.
- [30] D. Liu, K. C. Tan, C. K. Goh, and W. K. Ho, "A multiobjective memetic algorithm based on particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 37, no. 1, pp. 42–50, 2007.
- [31] K. Deb, M. Mohan, and S. Mishra, "Evaluating the ϵ -domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions," *Evolutionary Computation*, vol. 13, no. 4, pp. 501–525, 2005.
- [32] N. Kambhatla and T. K. Leen, "Dimension reduction by local principal component analysis," *Neural Computation*, vol. 9, no. 7, pp. 1493–1516, 1997.
- [33] J. D. Schaffer, *Some experiments in machine learning using vector evaluated genetic algorithms [Ph.D. thesis]*, Vanderbilt University, Nashville, Tenn, USA, 1984.
- [34] F. Kursawe, "A variant of evolution strategies for vector optimization," in *Parallel Problem Solving from Nature: Proceedings of the 1st Workshop, PPSN I Dortmund, FRG, October 1–3, 1990*, vol. 496 of *Lecture Notes in Computer Science*, pp. 193–197, Springer, Berlin, Germany, 1991.
- [35] C. M. Fonseca and P. J. Fleming, "An overview of evolutionary algorithms in multiobjective optimization," *Evolutionary Computation Journal*, vol. 3, no. 1, pp. 1–16, 1995.
- [36] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evolutionary Computation Journal*, vol. 8, no. 2, pp. 173–195, 2000.
- [37] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization*, A. Abraham, L. Jain, and R. Goldberg, Eds., pp. 105–145, Springer, London, UK, 2005.
- [38] D. A. Van Veldhuizen, *Multiobjective evolutionary algorithms: classification, analyses, and new innovations [Ph.D. thesis]*, Faculty of the Graduate School of Engineering of the Air Force Institute of Technology, Air University, AFIT/DS/ENG, 1999.
- [39] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.
- [40] J. R. Schott, *Fault tolerant design using single and multicriteria genetic algorithm optimization [M.S. thesis]*, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Mass, USA, 1995.
- [41] S. Kukkonen and J. Lampinen, "GDE3: the third evolution step of generalized differential evolution," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '05)*, pp. 443–450, Edinburgh, UK, September 2005.
- [42] P. A. N. Bosman and D. Thierens, "The naive MIDEA: a baseline multi-objective EA," in *Proceedings of the 3rd International Conference on Evolutionary Multi-Criterion Optimization (EMO '05)*, vol. 3410 of *Lecture Notes in Computer Science*, pp. 428–442, Guanajuato, Mexico, March 2005.
- [43] K. Deb, A. Anand, and D. Joshi, "A computationally efficient evolutionary algorithm for real-parameter optimization," *Evolutionary Computation*, vol. 10, no. 4, pp. 371–395, 2002.
- [44] S. Kukkonen and J. Lampinen, "An empirical study of control parameters for the third version of generalized differential evolution (GDE3)," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '06)*, pp. 2002–2009, Vancouver, Canada, July 2006.
- [45] D. Zaharie, "Critical values for the control parameters of differential evolution algorithms," in *Proceedings of the 8th International Conference on Soft Computing (Mendel '02)*, pp. 62–67, Brno, Czech Republic, June 2002.
- [46] V. Khare, X. Yao, and K. Deb, "Performance scaling of multi-objective evolutionary algorithms," in *Evolutionary Multi-Criterion Optimization: Proceedings of the 2nd International Conference, EMO 2003, Faro, Portugal, April 8–11, 2003*, vol. 2632 of *Lecture Notes in Computer Science*, pp. 376–390, Springer, Berlin, Germany, 2003.
- [47] H. C. Zhi, Y. G. Wen, and Q. H. Yong, "A novel differential evolution algorithm based ϵ -domination and orthogonal design method for multiobjective optimization," in *Evolutionary Multi-Objective Optimization*, vol. 4403 of *Lecture Notes in Computer Science*, pp. 286–301, Springer, New York, NY, USA, 2007.
- [48] A. G. Hernández-Díaz, L. V. Santana-Quintero, C. A. C. Coello, and J. Molina, "Pareto-adaptive ϵ -dominance," *Evolutionary Computation*, vol. 15, no. 4, pp. 493–517, 2007.
- [49] R. McGill, J. W. Tukey, and W. A. Larsen, "Variations of boxplots," *The American Statistician*, vol. 32, pp. 12–16, 1978.
- [50] D. E. Goldberg and P. Segrest, "Finite markov chain analysis of genetic algorithms," in *Proceedings of the 2nd International Conference on Genetic Algorithms on Genetic algorithms and their application*, J. J. Grefenstette, Ed., pp. 1–8, Hillsides, NJ, USA, 1987.
- [51] C. M. Fonseca and P. J. Fleming, "Genetic algorithms for multi-objective optimization: formulation, discussion and generalization," in *Proceedings of 5th International Conference on Genetic Algorithms*, S. Forrest, Ed., pp. 416–423, Morgan Kaufmann, San Mateo, Calif, USA, 1993.
- [52] N. Srinivas and K. Deb, "Multiobjective optimization using non-dominated sorting in genetic algorithms," *Evolutionary Computation Journal*, vol. 2, no. 3, pp. 221–248, 1995.
- [53] L. Ke, Q. Zhang, and R. Battiti, "Hybridization of decomposition and local search for multiobjective optimization," *IEEE Transactions on Cybernetics*, pp. 1–13, 2014.

Research Article

Null Steering of Adaptive Beamforming Using Linear Constraint Minimum Variance Assisted by Particle Swarm Optimization, Dynamic Mutated Artificial Immune System, and Gravitational Search Algorithm

Soodabeh Darzi,¹ Tiong Sieh Kiong,² Mohammad Tariqul Islam,¹ Mahamod Ismail,¹ Salehin Kibria,³ and Balasem Salem²

¹ Department of Electrical, Electronic & Systems Engineering, Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia

² Power Engineering Center & Center of System and Machine Intelligence, College of Engineering, Universiti Tenaga Nasional, 43000 Kajang, Selangor, Malaysia

³ Space Science Center (ANGKASA), Universiti Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia

Correspondence should be addressed to Mohammad Tariqul Islam; titareq@gmail.com

Received 22 April 2014; Revised 2 July 2014; Accepted 2 July 2014; Published 22 July 2014

Academic Editor: Su Fong Chien

Copyright © 2014 Soodabeh Darzi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Linear constraint minimum variance (LCMV) is one of the adaptive beamforming techniques that is commonly applied to cancel interfering signals and steer or produce a strong beam to the desired signal through its computed weight vectors. However, weights computed by LCMV usually are not able to form the radiation beam towards the target user precisely and not good enough to reduce the interference by placing null at the interference sources. It is difficult to improve and optimize the LCMV beamforming technique through conventional empirical approach. To provide a solution to this problem, artificial intelligence (AI) technique is explored in order to enhance the LCMV beamforming ability. In this paper, particle swarm optimization (PSO), dynamic mutated artificial immune system (DM-AIS), and gravitational search algorithm (GSA) are incorporated into the existing LCMV technique in order to improve the weights of LCMV. The simulation result demonstrates that received signal to interference and noise ratio (SINR) of target user can be significantly improved by the integration of PSO, DM-AIS, and GSA in LCMV through the suppression of interference in undesired direction. Furthermore, the proposed GSA can be applied as a more effective technique in LCMV beamforming optimization as compared to the PSO technique. The algorithms were implemented using Matlab program.

1. Introduction

Adaptive beamforming was inaugurated to evolution in aerospace and military applications with technology firmly fixed on phased-array via the electronically steered antennas [1]. Adaptive antennas were then supposed appropriate to solve the cochannel interference and multipath fading problem for mobile wireless communication. Adaptive antenna array was created in the 1950s by Howells identified as the intermediate frequency side lobe canceller (SLC) [2]. Although SLC technique encompasses the ability of automatic interference nulling, it was not fully adaptive due to fixed pattern for main beam and few controlled elements for ancillary array. This technology simplified the improvement of qualified adaptive array in 1965 by Applebaum. This

algorithm was employed to increase the signal-to-noise ratio (SNR) in order to have more efficiency of adaptive antenna. Meanwhile, another adaptive array technique known as linear constraint minimum variance (LCMV) was created by Windrow [3]. This algorithm was developed based on the conventional minimum mean square error (MMSE) for the automatic adjustment of array weights with the privilege of simplicity. Low convergence rate is the main disadvantage of LCMV technique that makes it inappropriate for some applications, while the advantage of it is only that it requires the direction of arrival (DOA) for maximizing the SNR. Formerly, different research works have been presented, which used LCMV for beamforming applications [3–5]. According to the characteristics of LCMV beamforming technique, it has a weak beam pattern and low signal to

interference-noise ratio (SINR) value. Solving this problem through conventional empirical approach is very difficult, time consuming, and sometimes in the applied case unmanageable. Consequently, many metaheuristics and exploratory methods have been settled to get the best results for these types of difficulties. Previous studies show that the employment of metaheuristics algorithm has been growing instead of exhaustive and exact procedures. In this regard, approaches such as genetic algorithms (GA) [6, 7], artificial bee colony (ABC) [8, 9], differential evolution (DE) [10], particle swarm optimization (PSO) [11, 12], ant colony optimization (ACO) [13–15], tabu search (TS) [8, 16, 17], artificial immune system (AIS) [18], and clonal selection (CS) [19, 20] have been used to solve a variety of problems in order to improve various issues in antenna system. According to the above-mentioned study, the main goal of this paper is to optimize the LCMV beamforming weights by PSO and gravitational search algorithm (GSA) technique to improve the performance of system in the expect of SINR.

In this investigation, PSO, DM-AIS, and GSA have been applied in uniform linear antenna arrays with 0.5λ spacing between adjacent elements at a frequency of 2.3 GHz. The rest of this paper is organized as follows. Section 2 introduces system model which contains the basics of adaptive beamforming, LCMV technique, and its algorithm. The artificial intelligence (AI) techniques including PSO, DM-AIS, and GSA are summarized in Section 3. Section 4 shows the application of presented AI in LCMV technique. Simulation results are reported in Section 5, and finally Section 6 concludes this investigation.

2. System Model

In this section, the mathematical formulation of the design model for an adaptive beamforming and LCMV technique will be presented in detail.

2.1. Adaptive Beamforming. The beamforming technique attempts to make beam toward the signal of interest (SOI) and produce null toward the direction of signals not of interest (SNOI). Signal processing technique automatically adjusts incoming SOIs and SNOIs from collected information and weight. The outputs of the individual sensors were linearly combined after being scaled with the corresponding weights. This process optimizes the antenna array to achieve maximum gain in the direction of the desired signal and nulls in the direction of interferers. For a beamformer, the output at any time n , $y(n)$ is given by a linear combination of the data at M antennas, with $x(n)$ being the input vector, $w(n)$ the weight vector, and H the Hermitian transpose. Consider

$$y(n) = w^H(n) x(n). \quad (1)$$

Weight vector $w(n)$ can be defined as follows

$$w(n) = \sum_{N=0}^{M-1} w_n, \quad (2)$$

$$x(n) = \sum_{n=0}^{M-1} X_n.$$

The weight vector at time $n + 1$ for any system that uses the direct gradient vector for upgrading weight vector and avoid the matrix inverse process can be as follows

$$W(n+1) = W(n) + \frac{1}{2}\mu [\nabla J(n)], \quad (3)$$

where μ is the step size parameter, which controls the speed of convergence and lies between 0 and 1. While the smallest quantity of μ assists the cost function superior estimation and sluggish concurrence, the huge quantity of it may result in a quick union. Nevertheless, the constancy over the minimum value could disappear. Consider

$$0 < \mu < \frac{1}{\lambda}. \quad (4)$$

Estimation of gradient vector is written as

$$\nabla J(n) = -2p(n) + 2R(n)W(n),$$

$$R(n) = X(n)X^H(n), \quad (5)$$

$$P(n) = d(n) * X(n),$$

where R is the covariance matrix and p is the cross-correlation vector.

By integrating (5) into (3), the weight vector can be derived as follows

$$W(n+1) = W(n) + \mu [p(n) - R(n)W(n)]$$

$$= W(n) + \mu X(n) [d^*(n) - X(n)W(n)] \quad (6)$$

$$= W(n) + \mu X e^*(n).$$

The following three formulas further define the desired signal as

$$y(n) = w^H(n) x(n),$$

$$e(n) = d(n) \cdot y(n) W(n+1) \quad (7)$$

$$= W(n) + \mu X(n) e^*(n).$$

2.2. Conventional LCMV Beamforming. Numerous algorithms were introduced for the design of an adaptive beamformer [21]. One of the popular approaches for adaptive beamforming was generated by Windrow [3]. His technique leads to an adaptive beamformer with the LCMV. The form of radiation beam in LCMV depends on the knowledge of the received desired signal. Moreover, LCMV technique can estimate the source of interference and control radiation lobe in order to accumulate high power to the desired direction. Although this system does not have to know the radiated power of the desired signal or the direction of interference and white noise, it is capable of suppressing the disturbances as much as possible. The total signal is given by

$$x_{\text{Total}} = x_s(t) + x_i(t) + x_n(t), \quad (8)$$

where x_s is the desired signal, x_i is the interference signal, x_n is the noise signal added from Gaussian noise, and $x_i + x_n$ is the undesired signal.

As mentioned in Section 2.1, the array output can be written as follow

$$y = w^H x. \quad (9)$$

The output power is given by

$$p = \{E|y|^2\} = E \{w^H x x^H w\} = w^H E \{x x^H\} w = w^H R, \quad (10)$$

where the R covariance matrix should be $(M, 1)$ for the received signal x .

The LCMV algorithm depends on the steering vector, based on the incident angle that is obtained from the incident wave on each element. The optimum weights are selected to minimize the array output power while maintaining unity gain in the look direction. The LCMV adaptive algorithm is shown in formula (11) when a constant beam pattern is required. Consider

$$\min \{w^H R_{xx} w\} \text{ conditional on } w^H a(\theta_0) = g, \quad (11)$$

where g is the complex scalar.

Steering vector $a(\theta)$ is given by [22]

$$a(\theta) = \begin{bmatrix} 1 \\ \exp \left\{ j \frac{2\pi}{\lambda} (\sin \theta_i) d \right\} \\ \exp \left\{ j \frac{2\pi}{\lambda} (\sin \theta_i) (m-1) d \right\} \end{bmatrix}, \quad (12)$$

where θ_i is the desired angle, d is the space between elements of antenna, and m is number of elements. All assumptions in this work will be shown later in Sections 4 and 5.

The optimization weight vectors can be calculated by the following formula [22, 23]:

$$W_{\text{LCMV}} = g * \frac{R_{xx}^{-1} a(\theta_0)}{a^H(\theta_0) R_{xx}^{-1} a(\theta_0)}. \quad (13)$$

That means M number of weights as below will be obtained by using M number of adaptive antenna elements. Consider

$$w_{\text{MVDR}} = \begin{bmatrix} w_1 \\ w_2 \\ \dots \\ w_M \end{bmatrix}. \quad (14)$$

The total noise, containing uncorrelated noise and interferences, is decreased by the minimization procedure. Notably, continually sustaining the output signal, the minimization of the total interference and noise is the same as maximizing the output SINR. The number of interference sources should be equal to or less than $M-2$ since an array with M elements has only $M-1$ degrees of freedom and has been employed by the constraint in the look direction for cancelling interferences in order to maximize the SINR. The LCMV algorithm is unsuitable when the users spread in all directions. This is called a multipath environment due to capability of increasing

sensitivity in only one direction [24]. The multipath interference occurs where numerous scatterers neighbor in the same base station and users are in populated urban regions. Consequently, LCMV beamforming technique may have an inappropriately low null level, in the situation of undesired interference signals which may have significant effect on performance of the system.

3. Methodology

In this section, the PSO, DM-AIS, and GSA are summarized as a basis to describe the proposed model which is used to solve adaptive beamforming problems in LCMV technique.

3.1. Particle Swarm Optimization (PSO). Particle swarm optimization (PSO) is a heuristic robust stochastic optimization technique that was introduced by Kennedy and Eberhart [25]. This method is inspired from the simulation of the social behaviors of animals (flock of birds). PSO updates the population of particles by adding an operator based on the fitness information achieved from the environment. Therefore, the individuals of the population would be estimated to move to the improved solution. The position and velocity of every population member are updated via the following formula [26]:

$$v_i^d(t+1) = w v_i^d(t) + c_1 R_1 (pbest_i^d(t) - p_i^d(t)) + c_2 R_2 (gbest(t)^d - p_i^d(t)), \quad (15)$$

$$p_i^d(t+1) = p_i^d(t) + v_i^d(t+1). \quad (16)$$

In the mentioned formula, velocity is v_i^d with dimension d and particle i , p_i^d shows the position of particle i , t is the iteration number, and d represents the dimension of search space. R_1 and R_2 are uniform random variables as the random cognitive coefficient and random social coefficient, respectively, that are applied to make a randomized characteristic in the search region. w is the inertia weight to balance the local and global search abilities of the particles. c_1 and c_2 are cognitive and social coefficient, $pbest$ is the individual best solution for i th particle, and $gbest$ represents the global best solution for the population. The positions are updated according to formula (16) and their fitnesses are evaluated. This process is repeated until a termination criterion is met. The work in [27] has suggested six different steps for solving the problem by using the PSO as below.

Step 1. Generate randomly the particles and their velocity in the dimensional search space.

Step 2. Evaluate the fitness function for each one.

Step 3. Update $pbest$ for each position; if the $pbest$ value is lower than current fitness value of the particle, replace it by current value.

Step 4. Update g_{best} ; if the g_{best} value is lower than current fitness value of the particle, replace it by current value.

Step 5. Update the velocity and position of each particle using (15) and (16), respectively.

Step 6. Repeat Steps 2–5, until the maximum number of iteration is met. The explanations on how PSO is used to optimize the weights of LCMV technique is given in Section 4.

3.2. Dynamic Mutated Artificial Immune System (DM-AIS).

One of the metaheuristic algorithms is artificial immune systems (AIS) [28, 29] with significant popularity in the field of optimization. AIS is mimicking the behavior of human immune system towards foreign elements in a host body. The human immune system makes active antibodies when antigens attack the body. In addition, the human immune system produces great amount of antibodies that fix powerfully to a specific antigen through its cloning process. The mutation rate of cloned antibodies is inversely proportionate to the affinity of antigens. Thus, the lowest affinity antibodies will result in the highest mutation rates. The general steps of AIS are described as below.

- (i) Initialize of random solutions (antibodies).
- (ii) Calculate fitness for each solution.
- (iii) Store the best fitness value and its associated solution.
- (iv) Select a subset of antibodies for cloning.
- (v) Apply mutation for each cloned antibody.
- (vi) Calculate fitness for each cloned antibody after mutation.
- (vii) Compare new fitness with best fitness value.
- (viii) If fitness of cloned antibody better than previous fitness, replace cloned antibody as antibody.
- (ix) Repeat until termination criteria are met.

DM-AIS [18] is a type of AIS algorithm with a new dynamic mutation function. The new population of antibodies is derived from the fitness value based on dynamic mutation rate. The new dynamic mutation rate is then able to improve the convergence rate of the antibody solution.

3.3. Gravitational Search Algorithm (GSA). One of the latest search algorithms for heuristic population is gravitational search algorithm (GSA). GSA is employed as an artificial world of masses following the gravitation of Newtonian laws [30]. All the GSA search agents (individuals) can be considered as objects and their masses are the factors of evaluate them.

Gravity force is the cause of the movement of all objects globally toward the objects with heavier masses. It means that optimum solutions of the problems are represented by the heavy masses. In this method, the new position and velocity of agent i will be upgraded based on the formula below:

$$\begin{aligned} v_i^d(t+1) &= \text{rand}_i \times v_i^d(t) + a_i^d(t), \\ p_i^d(t+1) &= p_i^d(t) + v_i^d(t+1). \end{aligned} \quad (17)$$

In the mentioned formula, velocity of i th agent is v_i^d in dimension d , p_i^d is position of i th agent at iteration number t . rand_i represents a random variable in order to provide a randomized characteristic for the search space as well as enhancement of the finding the global optimal chance, a_i^d is the acceleration of agent i in dimension d , and could be obtained as below:

$$a_i^d(t) = \sum_{j \in k_{best}, j \neq i} \text{rand}_i G(t) \frac{M_j(t)}{R_{i,j}(t) + \varepsilon} (p_j^d(t) - p_i^d(t)), \quad (18)$$

where a_i^d is the acceleration of i th agent in dimension d and rand_j is random value; according to formula (19), the gravitational constant at time t is $G(t)$; M_j is mass of j th agent represented in formula (20); ε is a minor element to prevent division by zero and the $R_{i,j}(t)$ is the Euclidean distance that is represented as $R_{i,j}(t) = \|p_i(t), p_j(t)\|_2$. It is valued to reference that we employ R as a replacement of R^2 in formula (18) due to the tests offered in [30] which illustrate that R offers better output in comparison with R^2 . k_{best} is a control function that is able to advance the performance of GSA. This function controls the exploitation and exploration with initialization to k_0 at the starting and reducing with each iteration [30]. k_0 is adjusted to N (overall amount of agents) and is reduced to 1 linearly. In formula (18), the gravitational constant $G(t)$ is a reducing function of time when it is set to G_0 at the starting and will be decreased exponentially as shown in formula below:

$$G(t) = G_0 \times \exp\left(-\beta \times \frac{t}{t_{\max}}\right). \quad (19)$$

In formula (19), β is a gradient constant value, t is the current repetition, and t_{\max} is the maximum repetition number. Furthermore, the mass of agents in formula (18) is examined with employing the formula (20) as a below:

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^N m_j(t)}, \quad (20)$$

in which

$$m_i(t) = \frac{\text{fit}_i(t) - \text{worst}(t)}{\text{best}(t) - \text{worst}(t)}. \quad (21)$$

In formula (21), $\text{fit}_i(t)$ is the fitness value of the agent i at time t . $\text{worst}(t)$ and $\text{best}(t)$ are the worst and best fitness of all agents, respectively.

4. Proposed LCMV Beamforming Assisted by Gravitational Search Algorithm, Dynamic Mutated Artificial Immune System, and Particle Swarm Optimization

The PSO, DM-AIS, and GSA were utilized to enhance the SINR value of the LCMV beamforming technique in this paper. The smart antenna will try to optimize via PSO, DM-AIS, and GSA iteration process to make deep null at the interference sources in order to get the maximum SINR.

TABLE 1: Used parameters of PSO and GSA briefly in this study.

Parameter	Description	PSO	GSA
N	Population size	10	10
d	Dimension of the search space	4,8	4,8
t_{\max}	Maximum iteration	500	500
W	Weight	Start weight = 0.9 End weight = 0.4	N.A
c_1	Cognitive coefficient	2	N.A
R_1	Random cognitive coefficient	Rand(0, 1)	N.A
c_2	Social coefficient	2	N.A
R_2	Random social coefficient	Rand(0, 1)	N.A
G_0	Initial value of gravitational constant	N.A	100
β	Gradient constant	N.A	20
ε	Zero offset constant	N.A	$2.22e - 16$

In these algorithms, the w (weight vector) will be used as the system population. These algorithms will initiate by generating the N particles, which is indicated by W_N weight vectors. Moreover, the number of produced weight vectors is dependent on the population size P_{size} . For the first generation, the first set of weight vectors W_1 is obtained from the computation of the conventional LCMV weight vector. The weight vectors in every particle contain an M number of weight vectors, depending on the antenna elements used or the number of sensors. The weight vectors in the population of any iteration can be illustrated in matrix format as a below:

$$W_{NM} = \begin{bmatrix} W_{\text{lcmv}1} & W_{\text{lcmv}2} & \cdots & W_{\text{lcmv}M} \\ W_{11} & W_{12} & \cdots & W_{1M} \\ W_{21} & W_{22} & \cdots & W_{2M} \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ W_{n1} & W_{n2} & \cdots & W_{nM} \end{bmatrix}, \quad (22)$$

where W_{NM} are the weight vectors of total population N with M sensors in each antenna and W_{lcmv} are the weight vectors from LCMV beamformer.

PSO, DM-AIS, and GSA: $p_i^d = W_n^M$, where $d = M$, $i = n$, and i th Particle at d th dimension can be presented as n th weight at dimension M .

Each set of particles W has amplitude and phase ($A\angle\theta$) to steer the radiation beam toward its target user and place the deep null toward the interference sources to achieve the optimum SINR. The best weight vector is determined according to the fitness value obtained from fitness function as shown below. Consider

$$\text{Fitness.Function (FF)} = \frac{P_{\text{User}}}{\sum_{n=1}^N P_{\text{Inter},n} + \text{Noise}}, \quad (23)$$

where P_{User} is the power of target user, P_{Inter} is the power of interference, and n is the number of interference sources.

Table 1 shows the used parameters of PSO and GSA in this study. The parameters of GSA are chosen according to

the guidelines and recommendations presented in [30]. These configurations of GSA have also been utilized extensively after the development of GSA [31–34]. The linearly decreasing inertia coefficient of PSO was chosen to allow initial exploration while not degrading the convergence rate. The cognitive and social coefficients were chosen based on the recommendations in the literature [35, 36]. The maximum iterations for DM-AIS are 500 for fair comparison with GSA and PSO. DM-AIS parameters implemented in this study were the same as in [18].

5. Experimental Results and Discussion

The effectiveness of the proposed PSO, DM-AIS, and GSA techniques in LCMV in comparison with the conventional LCMV beamforming is investigated by considering four cases of different interferences and elements in this section. All cases have one user at 0° , while the number of interferences and elements changes in each case. The first case is interference signal at 20° by using 4-element patch antenna; the second case is two interference signals at 20° and 60° by using 4-element patch antenna. Third and fourth cases include same interference at cases 1 and 2 but using 8-element patch antenna. Optimization result of PSO and GSA are obtained from 10 cycle's simulation and the best results are plotted.

5.1. Case 1: One User, One Interference, Four Elements. One interference source at 20° and user at 0° by using 4-element patch antenna has been assumed in the first case study. The simulation result is shown in Figure 1.

Table 2 compares the power values of the above-mentioned four methods. It is obvious that the GSA method performs better than PSO and DM-AIS with the same number of iterations. It means GSA is superior as compared to PSO and DM-AIS because it creates deeper null in the interference direction while increasing power at desired direction. This result also demonstrates that the improvements of SINR are 52.01%, 53.63%, and 54.14%, respectively, by PSO, DM-AIS, and GSA as compared to conventional LCMV method.

TABLE 2: Comparison of weight vectors, power values, and SINR calculation for conventional LCMV, PSO-LCMV, DM-AIS-LCMV, and GSA-LCMV for user at 0° with interference at 20° by using 4 elements.

Algorithm	Weights	Power at 0° (W)	Power at 20° (W)	SINR (dB)
LCMV	$-0.0034 - 0.9316i$	4.35	4.07×10^{-5}	50.27
	$-2.1723 - 1.8276i$			
	$-2.1723 + 1.827i$			
	$-0.0034 + 0.9316i$			
PSO	$-1.0592 - 0.3784i$	4.38	1×10^{-15}	76.42
	$-1.1478 - 0.3784i$			
	$-1.8144 + 1.27657i$			
	$0.0176 + 1.2784i$			
DM-AIS	$1.2962 + 0.9929i$	5.31	6.29×10^{-10}	77.23
	$0.3293 + 0.9559i$			
	$1.2026 + 0.3361i$			
	$1.9141 + 0.1246i$			
GSA	$0.6579 + 0.8473i$	5.65	5.03×10^{-10}	77.50
	$1.3924 + 1.1853i$			
	$0.6916 + 0.5625i$			
	$2.5054 - 0.4960i$			

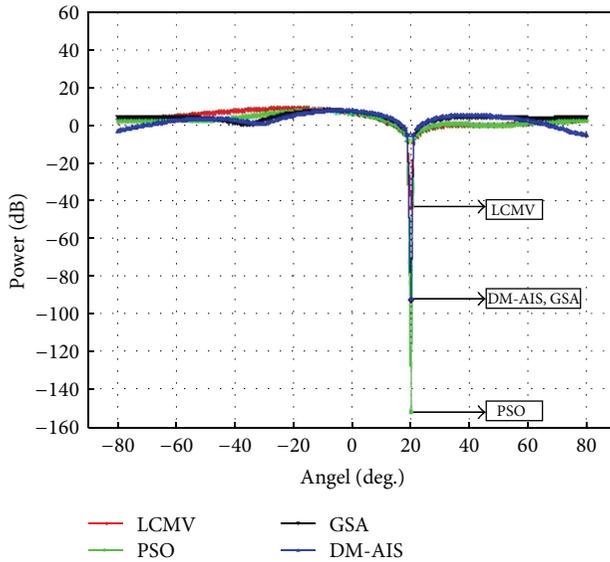


FIGURE 1: Comparison of performance of power response LCMV, PSO-LCMV, DM-AIS-LCMV, and GSA-LCMV for user at 0° with interference at 20° by using 4 elements.

5.2. Case 2: One User, Two Interferences, Four Elements. Two interference sources at 20° , 60° by using 4-element patch antenna and user at 0° has been assumed in the second case study. The simulation outcome is displayed in Figure 2.

Table 3 compares the power values and SINR of the methods mentioned previously. The GSA method achieves better performance with the same iterations in Case 2 also. This shows that the proposed methods can achieve better nulls than conventional LCMV for two interference cases. In addition, this table presents the ratio of SINR 4.40 dB through conventional LCMV while the PSO, DM-AIS, and GSA improve these values 1176.81%, 1556.59%, and 1732.95%, respectively.

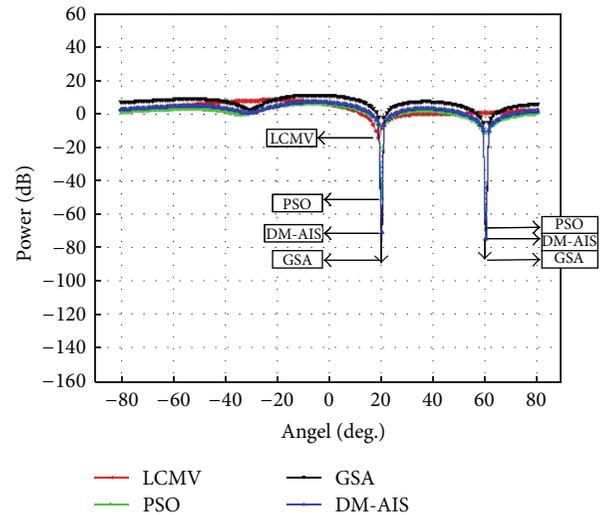


FIGURE 2: Comparison of performance of power response LCMV, PSO-LCMV, DM-AIS-LCMV, and GSA-LCMV for user at 0° with interference at 20° and 60° by using 4 elements.

5.3. Case 3: One User, One Interference, Eight Elements. One interference at 20° by using 8-element patch antenna and user at 0° has been assumed in the third case study.

The results of Case 3 are consistent with the findings of previous two cases. GSA-LCMV outperforms conventional LCMV, DM-AIS-LCMV, and PSO-LCMV significantly for eight-element array also, as illustrated in Figure 3. Thus, the superiority of GSA over other techniques is independent of number of array elements. Table 4 demonstrates the ratio of SINR 38.93 dB through conventional LCMV, while the PSO, DM-AIS, and GSA improve these values 49.73%, 72.61%, and 105.26%, respectively.

5.4. Case 4: One User, Two Interferences, and Eight Elements. Two interference sources at 20° , 60° by using 8-element patch

TABLE 3: Comparison of weight vectors, power values, and SINR calculation for conventional LCMV, PSO-LCMV, DM-AIS-LCMV, and GSA-LCMV for user at 0° with interference at 20° and 60° by using 4 elements.

Algorithm	Weights	Power at 0° (W)	Power at 20° (W)	Power at 60° (W)	SINR (dB)
LCMV	$0.0067 - 0.7860i$ $-1.7890 - 1.6179i$ $-1.7888 + 1.6184i$ $0.0069 + 0.7860i$	3.56	1.5×10^{-1}	1.14×10^0	4.40
PSO	$0.2620 + 1.0535i$ $0.1924 + 0.6244i$ $0.5209 + 0.8704i$ $1.3205 + 0.3789i$	3.72	8.69×10^{-6}	1.54×10^{-7}	56.18
DM-AIS	$0.7471 + 1.2004i$ $0.4117 + 0.5250i$ $0.7398 + 0.8941i$ $1.7452 + 0.1232i$	4.56	9.44×10^{-8}	3.96×10^{-8}	72.89
GSA	$-4.1832 - 1.6647i$ $-2.5456 + 0.9557i$ $-1.8412 + 0.4129i$ $-3.3839 + 2.0752i$	12.08	1.78×10^{-9}	2.18×10^{-9}	80.65

TABLE 4: Comparison of weight vectors, power values, and SINR calculation for conventional LCMV, PSO-LCMV, DM-AIS-LCMV, and GSA-LCMV for user at 0° with interference at 20° by using 8 elements.

Algorithm	Weights	Power at 0° (W)	Power at 20° (W)	SINR (dB)
LCMV	$0.0317 + 0.1445i$ $0.1194 + 0.0909i$ $-0.2264 + 0.1311i$ $0.0990 - 0.2212i$ $0.1017 + 0.2525i$ $-0.1896 - 0.12725i$ $0.1060 - 0.1115i$ $0.0227 - 0.1350i$	6.90×10^{-2}	8.71×10^{-6}	38.93
PSO	$0.0317 + 0.1445i$ $0.1194 + 0.0909i$ $-0.2263 + 0.1311i$ $0.0990 - 0.2212i$ $0.1017 + 0.2525i$ $-0.1897 - 0.1271i$ $0.1061 - 0.1115i$ $0.0227 - 0.1349i$	6.92×10^{-2}	2.23×10^{-9}	58.30
DM-AIS	$-0.0864 + 0.7410i$ $0.9922 + 0.1975i$ $1.6119 + 0.4745i$ $0.1607 + 0.5461i$ $1.3573 + 0.2932i$ $-0.1193 + 0.6657i$ $0.9244 + 0.7284i$ $0.0049 + 0.5399i$	6.40×10^0	1.11×10^{-6}	67.20
GSA	$-0.6737 + 0.8571i$ $0.6194 - 0.0643i$ $1.3861 + 0.6505i$ $0.8473 + 0.8198i$ $1.1431 + 0.9980i$ $1.5379 + 1.0542i$ $1.8363 + 0.9692i$ $1.3944 + 0.42148i$	9.90×10^0	6.30×10^{-10}	79.92

TABLE 5: Comparison of weight vectors, power values, and SINR calculation for conventional LCMV, PSO-LCMV, DM-AIS-LCMV, and GSA-LCMV for user at 0° with interference at 20° , 60° by using 8 elements.

Algorithm	Weights	Power at 0° (W)	Power at 20° (W)	Power at 60° (W)	SINR (dB)
LCMV	$-0.6890 + 0.4136i$ $0.2415 + 1.1050i$ $-0.6307 - 0.3236i$ $-0.2050 + 0.4525i$ $-0.1819 - 0.4779i$ $-0.5573 + 0.2462i$ $0.2360 - 1.2187i$ $-0.7214 - 0.4358i$	2.51	2.98×10^{-5}	7.66×10^{-5}	43.73
PSO	$-0.6004 + 0.5241i$ $0.2293 + 1.1021i$ $-0.5799 - 0.3262i$ $-0.5170 + 0.9217i$ $-0.1831 - 0.3784i$ $-1.0433 + 0.2464i$ $0.0982 - 0.3784i$ $-0.6887 - 0.3784i$	3.54	1×10^{-15}	1×10^{-15}	75.49
DM-AIS	$-0.4890 + 0.5137i$ $-0.2718 + 0.6251i$ $0.0371 + 0.4669i$ $0.2083 + 0.6415i$ $-0.6586 + 0.7074i$ $-0.1554 + 0.7897i$ $0.3603 + 0.5786i$ $0.3198 + 0.5346i$	4.90	1.70×10^{-9}	1.79×10^{-9}	76.75
GSA	$-0.3417 + 0.1177i$ $0.1586 + 1.0271i$ $1.1506 + 0.3389i$ $1.8499 + 0.6928i$ $1.5246 + 1.0174i$ $0.7917 + 0.3530i$ $1.5273 + 0.7368i$ $1.4564 + 0.8934i$	9.62	2.64×10^{-9}	6.36×10^{-10}	79.69

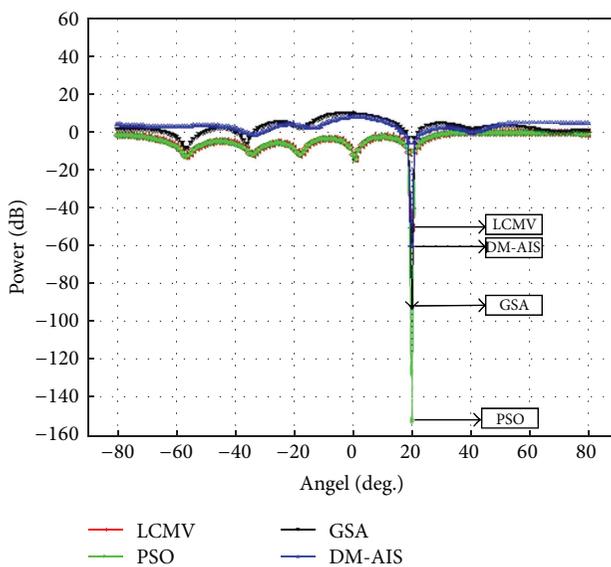


FIGURE 3: Comparison of performance of power response LCMV, PSO-LCMV, DM-AIS-LCMV, and GSA-LCMV for user at 0° with interference at 20° by using 8 elements.

antenna and user at 0° have been assumed in the fourth case study.

Case 4 confirms the findings of Case 3 in terms of superiority of GSA, as shown in Figure 4. Table 5 shows that GSA-LCMV achieves highest power in desired direction and lower power at nulls than conventional LCMV. PSO is known to have premature convergence issues. Thus, GSA is able to consistently outperform PSO in array optimization problems. Besides, this result also illustrates that the development of SINR is 72.62%, 75.50%, and 82.22%, respectively, by PSO, DM-AIS, and GSA as compared to conventional LCMV technique. The DM-AIS was developed to improve the performance of AIS for beamforming applications [18]. Based on the results presented in this section, it converges to significantly better solutions than PSO consistently. However, GSA still outperforms DM-AIS with superior SINR in all scenarios.

6. Conclusion

This paper applied PSO, DM-AIS, and GSA in linear antenna arrays with different number of elements to control the nulling of interference and the directionality towards the

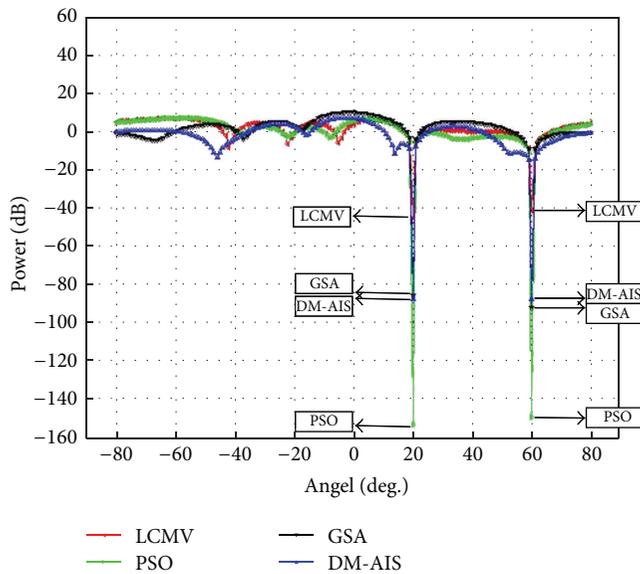


FIGURE 4: Comparison of performance of power response LCMV, PSO-LCMV, DM-AIS-LCMV, and GSA-LCMV for user at 0° with interference at 20° , 60° by using 8 elements.

desired signal. The results of the LCMV assisted by proposed approaches were compared with conventional LCMV, and the effectiveness of the proposed approaches in minimizing the power of interference and increasing SINR was observed. The result of LCMV beamforming assisted by GSA algorithm is better than PSO and DM-AIS algorithm and also its conventional LCMV beamforming algorithm. This new proposed GSA-LCMV can be useful for smart antenna for the radiation pattern synthesis because of its good accuracy and not requiring complicated mathematical functions.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

- [1] M. Barrett and R. Arnott, "Adaptive antennas for mobile communications," *IEEE Electronics and Communication Engineering Journal*, vol. 6, no. 4, pp. 203–214, 1994.
- [2] J. Litva and T. K.-Y. Lo, *Digital Beamforming in Wireless Communications*, Artech House Mobile Communications, 1996.
- [3] M. Souden, J. Benesty, and S. Affes, "A study of the LCMV and MVDR noise reduction filters," *IEEE Transactions on Signal Processing*, vol. 58, no. 9, pp. 4925–4935, 2010.
- [4] K. M. Buckley and L. J. Griffiths, "Spatial filtering with broadband minimum variance beamformers," *IEEE Antennas and Propagation Society International Symposium*, vol. 34, no. 5, pp. 1322–1323, 1986.
- [5] D. Z. Filho, Ch. C. Cavalcante, and J. M. T. Romano, "Adaptive LCMV beamforming avoiding DOA estimation for packet-like wireless systems," in *Proceedings of the International Telecommunications Symposium*, 2002.
- [6] T. S. Kiong, M. Ismail, and A. Hassan, "WCDMA forward link capacity improvement by using adaptive antenna with genetic algorithm assisted MDPC beamforming technique," *Journal of Applied Sciences*, vol. 6, no. 8, pp. 1766–1773, 2006.
- [7] Z. Xu, H. Li, Q.-Z. Liu, and J.-Y. Li, "Pattern synthesis of conformal antenna array by the hybrid genetic algorithm," *Progress in Electromagnetics Research*, vol. 79, pp. 75–90, 2008.
- [8] O. Kaid Omar, F. Debbat, and A. Boudghene Stambouli, "Null steering beamforming using hybrid algorithm based on honey bees mating optimization and tabu search in adaptive antenna array," *Progress in Electromagnetics Research C*, vol. 32, pp. 65–80, 2012.
- [9] K. Guney and M. Onay, "Amplitude-only pattern nulling of linear antenna arrays with the use of bees algorithm," *Progress in Electromagnetics Research*, vol. 70, pp. 21–36, 2007.
- [10] S. W. Yang, Y. B. Gan, and A. Y. Qing, "Antenna-array pattern nulling using a differential evolution algorithm," *International Journal of RF and Microwave Computer-Aided Engineering*, vol. 14, no. 1, pp. 57–63, 2004.
- [11] Z. D. Zaharis and T. V. Yioultis, "A novel adaptive beamforming technique applied on linear antenna arrays using adaptive mutated Boolean PSO," *Progress in Electromagnetics Research*, vol. 117, pp. 165–179, 2011.
- [12] M. M. Khodier and C. G. Christodoulou, "Linear array geometry synthesis with minimum sidelobe level and null control using particle swarm optimization," *IEEE Transactions on Antennas and Propagation*, vol. 53, no. 8, pp. 2674–2679, 2005.
- [13] N. Karaboga, K. Güneş, and A. Akdagli, "Null steering of linear antenna arrays with use of modified touring ant colony optimization algorithm," *International Journal of RF and Microwave Computer-Aided Engineering*, vol. 12, no. 4, pp. 375–383, 2002.
- [14] D. Karaboga, K. Guney, and A. Akdagli, "Antenna array pattern nulling by controlling both amplitude and phase using modified touring ant colony optimization algorithm," *International Journal of Electronics*, vol. 91, no. 4, pp. 241–251, 2004.
- [15] A. A. Akdagli, K. Guney, and D. Karaboga, "Touring ant colony optimization algorithm for shaped-beam pattern synthesis of linear antenna," *Electromagnetics Research*, vol. 26, no. 8, pp. 615–628, 2006.
- [16] K. Guney and A. Akdagli, "Null steering of linear antenna arrays using modified tabu search algorithm," *Progress in Electromagnetics Research*, vol. 33, pp. 167–182, 2001.
- [17] A. Akdagli and K. Guney, "Shaped-beam pattern synthesis of equally and unequally spaced linear antenna arrays using a modified tabu search algorithm," *Microwave and Optical Technology Letters*, vol. 36, no. 1, pp. 16–20, 2003.
- [18] S. K. Tiong, B. Salem, S. P. Koh, K. P. Sankar, and S. Darzi, "Minimum variance distortionless response (MVDR) beamformer with enhanced nulling level control via dynamic mutated artificial immune system," *The Scientific World Journal*, vol. 2014, Article ID 164053, 9 pages, 2014.
- [19] B. Babayigit, A. Akdagli, and K. Guney, "A clonal selection algorithm for null synthesizing of linear antenna arrays by amplitude control," *Journal of Electromagnetic Waves and Applications*, vol. 20, no. 8, pp. 1007–1020, 2006.
- [20] K. Guney, A. Akdagli, and B. Babayigit, "Shaped-beam pattern synthesis of linear antenna arrays with the use of a clonal selection algorithm," *Neural Network World*, vol. 16, no. 6, pp. 489–501, 2006.
- [21] B. Salem, S. K. Tiong, and S. P. Koh, "Beamforming algorithms technique by using MVDR and LCMV," *World Applied Programming*, vol. 2, no. 5, pp. 315–324, 2012.

- [22] R. Compton, *Adaptive Antennas Concept and Performance*, Prentice Hall, 2011.
- [23] G. Renzhou, "Suppressing radio frequency interferences with adaptive beamformer based on weight iterative algorithm," in *Proceedings of the IET Conference on Wireless, Mobile and Sensor Networks (CCWMSN '07)*, pp. 648–651, 2009.
- [24] A. Ali, R. L. Ali, and A. Rehman, "An improved gain vector to enhance convergence characteristics of recursive least squares algorithm," *International Journal of Hybrid Information Technology*, vol. 4, pp. 99–107, 2011.
- [25] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.
- [26] H. Wang, H. Sun, C. Li, S. Rahnamayan, and J. Pan, "Diversity enhanced particle swarm optimization with neighborhood search," *Information Sciences*, vol. 223, pp. 119–135, 2013.
- [27] G. He and N. Huang, "A modified particle swarm optimization algorithm with applications," *Applied Mathematics and Computation*, vol. 219, no. 3, pp. 1053–1060, 2012.
- [28] C. A. Coello Coell and N. Cruz Cortes, "An approach to solve multi objective optimization problem based on artificial immune system," in *Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS '02)*, pp. 212–221, University of Kent at Canterbury, Canterbury, UK, 2002.
- [29] C. A. C. Coello and N. C. Cortés, "Solving multiobjective optimization problems using an artificial immune system," *Genetic Programming and Evolvable Machines*, vol. 6, no. 2, pp. 163–190, 2005.
- [30] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.
- [31] G. Binjie and P. Feng, "Modified gravitational search algorithm with particle memory ability and its application," *International Journal of Innovative Computing*, vol. 9, no. 11, pp. 4531–4544, 2013.
- [32] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi, "BGSA: binary gravitational search algorithm," *Natural Computing*, vol. 9, no. 3, pp. 727–745, 2010.
- [33] M. Khajehzadeh, M. R. Taha, and M. Eslami, "Efficient gravitational search algorithm for optimum design of retaining walls," *Structural Engineering and Mechanics*, vol. 45, no. 1, pp. 111–127, 2013.
- [34] M. Khajehzadeh, M. R. Taha, A. El-Shafie, and M. Eslami, "A modified gravitational search algorithm for slope stability analysis," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 8, pp. 1589–1597, 2012.
- [35] A. Banks, J. Vincent, and C. Anyakoha, "A review of particle swarm optimization—II. Hybridisation, combinatorial, multi-criteria and constrained optimization, and indicative applications," *Natural Computing*, vol. 7, no. 1, pp. 109–124, 2008.
- [36] P. Sun, H. Sun, W. Feng, Q. Zhao, and H. Zhao, "A study of acceleration coefficients in particle swarm optimization algorithm based on CPSO," in *Proceedings of the 2nd International Conference on Information Engineering and Computer Science (ICIECS '10)*, pp. 1–4, 2010.

Research Article

A Cuckoo Search Algorithm for Multimodal Optimization

Erik Cuevas and Adolfo Reyna-Orta

Departamento de Electronica, Universidad de Guadalajara, CUCEI, Avenida Revolución 1500, 44430 Guadalajara, JAL, Mexico

Correspondence should be addressed to Erik Cuevas; erik.cuevas@cucei.udg.mx

Received 23 April 2014; Accepted 5 May 2014; Published 22 July 2014

Academic Editor: Xin-She Yang

Copyright © 2014 E. Cuevas and A. Reyna-Orta. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Interest in multimodal optimization is expanding rapidly, since many practical engineering problems demand the localization of multiple optima within a search space. On the other hand, the cuckoo search (CS) algorithm is a simple and effective global optimization algorithm which can not be directly applied to solve multimodal optimization problems. This paper proposes a new multimodal optimization algorithm called the multimodal cuckoo search (MCS). Under MCS, the original CS is enhanced with multimodal capacities by means of (1) the incorporation of a memory mechanism to efficiently register potential local optima according to their fitness value and the distance to other potential solutions, (2) the modification of the original CS individual selection strategy to accelerate the detection process of new local minima, and (3) the inclusion of a depuration procedure to cyclically eliminate duplicated memory elements. The performance of the proposed approach is compared to several state-of-the-art multimodal optimization algorithms considering a benchmark suite of fourteen multimodal problems. Experimental results indicate that the proposed strategy is capable of providing better and even a more consistent performance over existing well-known multimodal algorithms for the majority of test problems yet avoiding any serious computational deterioration.

1. Introduction

Optimization is a field with applications in many areas of science, engineering, economics, and others, where mathematical modelling is used [1]. In general, the goal is to find an acceptable solution of an objective function defined over a given search space. Optimization algorithms are usually broadly divided into deterministic and stochastic ones [2]. Since deterministic methods only provide a theoretical guarantee of locating a local minimum for the objective function, they often face great difficulties in solving optimization problems [3]. On the other hand, stochastic methods are usually faster in locating a global optimum [4]. Moreover, they adapt easily to black-box formulations and extremely ill-behaved functions, whereas deterministic methods usually rest on at least some theoretical assumptions about the problem formulation and its analytical properties (such as Lipschitz continuity) [5].

Evolutionary algorithms (EA), which are considered to be members of the stochastic group, have been developed by a combination of rules and randomness that mimics several natural phenomena. Such phenomena include evolutionary processes such as the evolutionary algorithm (EA) proposed

by Fogel et al. [6], de Jong [7], and Koza [8]; the genetic algorithm (GA) proposed by Holland [9] and Goldberg [10]; the artificial immune system proposed by de Castro and von Zuben [11]; and the differential evolution algorithm (DE) proposed by Storn and Price [12]. Some other methods which are based on physical processes include simulated annealing proposed by Kirkpatrick et al. [13], the electromagnetism-like algorithm proposed by Birbil and Fang [14], and the gravitational search algorithm proposed by Rashedi et al. [15]. Also, there are other methods based on the animal-behavior phenomena such as the particle swarm optimization (PSO) algorithm proposed by Kennedy and Eberhart [16] and the ant colony optimization (ACO) algorithm proposed by Dorigo et al. [17].

Most of research work on EA aims for locating the global optimum [18]. Despite its best performance, a global optimum may be integrated by parameter values that are considered impractical or prohibitively expensive, limiting their adoption into a real-world application. Therefore, from a practical point of view, it is desirable to have access to not only the global optimum but also as many local optima as possible (ideally all of them). Under such circumstances, a local optimum with acceptable performance quality and

modest cost may be preferred over a costly global solution with marginally better performance [19]. The process of finding the global optimum and multiple local optima is known as multimodal optimization.

EA perform well for locating a single optimum but fail to provide multiple solutions [18]. Several methods have been introduced into the EA scheme to achieve multimodal optimization, such as fitness sharing [20–22], deterministic crowding [23], probabilistic crowding [22, 24], clustering based niching [25], clearing procedure [26], species conserving genetic algorithm [27], and elitist-population strategies [28]. However, most of these methods have difficulties that need to be overcome before they can be employed successfully to multimodal applications. Some identified problems include difficulties in tuning some niching parameters, difficulties in maintaining discovered solutions in a run, extra computational overheads, and poor scalability when dimensionality is high. An additional problem represents the fact that such methods are devised for extending the search capacities of popular EA such as GA and PSO, which fail in finding a balance between exploration and exploitation, mainly for multimodal functions [29]. Furthermore, they do not explore the whole region effectively and often suffer premature convergence or loss of diversity.

As alternative approaches, other researchers have employed artificial immune systems (AIS) to solve multimodal optimization problems. Some examples are the clonal selection algorithm [30] and the artificial immune network (AiNet) [31, 32]. Both approaches use operators and structures which attempt to find multiple solutions by mimicking the natural immune system's behavior.

Every EA needs to address the issue of exploration and exploitation of the search space [33]. Exploration is the process of visiting entirely new points of a search space, whilst exploitation is the process of refining those points within the neighborhood of previously visited locations in order to improve their solution quality. Pure exploration degrades the precision of the evolutionary process but increases its capacity to find new potential solutions. On the other hand, pure exploitation allows refining existent solutions but adversely driving the process to local optimal solutions.

Multimodal optimization requires a sufficient amount of exploration of the population agents in hyperspace so that all the local and global attractors can be successfully and quickly detected [34, 35]. However, an efficient multimodal optimization algorithm should exhibit not only good exploration tendency but also good exploitative power, especially during the last stages of the search, because it must ensure accurately a distributed convergence to different optima in the landscape. Therefore, the ability of an EA to find multiple solutions depends on its capacity to reach a good balance between the exploitation of found-so-far elements and the exploration of the search space [36]. So far, the exploration-exploitation dilemma has been an unsolved issue within the framework of EA.

Recently, a novel nature-inspired algorithm, called the cuckoo search (CS) algorithm [37], has been proposed for solving complex optimization problems. The CS algorithm is based on the obligate brood-parasitic strategy of some cuckoo

species. One of the most powerful features of CS is the use of Lévy flights to generate new candidate solutions. Under this approach, candidate solutions are modified by employing many small changes and occasionally large jumps. As a result, CS can substantially improve the relationship between exploration and exploitation, still enhancing its search capabilities [38]. Recent studies show that CS is potentially far more efficient than PSO and GA [39]. Such characteristics have motivated the use of CS to solve different sorts of engineering problems such as mesh generation [40], embedded systems [41], steel frame design [42], scheduling problems [43], thermodynamics [44], and distribution networks [45].

This paper presents a new multimodal optimization algorithm called the multimodal cuckoo search (MCS). The method combines the CS algorithm with a new memory mechanism which allows an efficient registering of potential local optima according to their fitness value and the distance to other potential solutions. The original CS selection strategy is mainly conducted by an elitist decision where the best individuals prevail. In order to accelerate the detection process of potential local minima in our method, the selection strategy is modified to be influenced by individuals that are contained in the memory mechanism. During each generation, eggs (individuals) that exhibit different positions are included in the memory. Since such individuals could represent to the same local optimum, a depuration procedure is also incorporated to cyclically eliminate duplicated memory elements. The performance of the proposed approach is compared to several state-of-the-art multimodal optimization algorithms considering a benchmark suite of 14 multimodal problems. Experimental results indicate that the proposed strategy is capable of providing better and even more consistent performance over the existing well-known multimodal algorithms for the majority of test problems avoiding any serious computational deterioration.

The paper is organized as follows. Section 2 explains the cuckoo search (CS) algorithm, while Section 3 presents the proposed MCS approach. Section 4 exhibits the experimental set and its performance results. Finally, Section 5 establishes some concluding remarks.

2. Cuckoo Search (CS) Method

CS is one of the latest nature-inspired algorithms, developed by Yang and Deb [37]. CS is based on the brood parasitism of some cuckoo species. In addition, this algorithm is enhanced by the so-called Lévy flights [46], rather than by simple isotropic random walks. Recent studies show that CS is potentially far more efficient than PSO and GA [39].

Cuckoo birds lay their eggs in the nests of other host birds (usually other species) with amazing abilities such as selecting nests containing recently laid eggs and removing existing eggs to increase the hatching probability of their own eggs. Some of the host birds are able to combat this parasitic behavior of cuckoos and throw out the discovered alien eggs or build a new nest in a distinct location. This cuckoo breeding analogy is used to develop the CS algorithm. Natural systems are complex, and therefore they cannot be modeled exactly by a computer algorithm in its basic form. Simplification of

natural systems is necessary for successful implementation in computer algorithms. Yang and Deb [39] simplified the cuckoo reproduction process into three idealized rules.

- (1) An egg represents a solution and is stored in a nest. An artificial cuckoo can lay only one egg at a time.
- (2) The cuckoo bird searches for the most suitable nest to lay the eggs in (solution) to maximize its eggs' survival rate. An elitist selection strategy is applied, so that only high-quality eggs (best solutions near the optimal value) which are more similar to the host bird's eggs have the opportunity to develop (next generation) and become mature cuckoos.
- (3) The number of host nests (population) is fixed. The host bird can discover the alien egg (worse solutions away from the optimal value) with a probability of $p_a \in [0, 1]$, and these eggs are thrown away or the nest is abandoned and a completely new nest is built in a new location. Otherwise, the egg matures and lives to the next generation. New eggs (solutions) laid by a cuckoo choose the nest by Lévy flights around the current best solutions.

From the implementation point of view, in the CS operation, a population, $\mathbf{E}^k (\{\mathbf{e}_1^k, \mathbf{e}_2^k, \dots, \mathbf{e}_N^k\})$, of N eggs (individuals) is evolved from the initial point ($k = 0$) to a total gen number iterations ($k = 2 \cdot \text{gen}$). Each egg, $\mathbf{e}_i^k (i \in [1, \dots, N])$, represents an n -dimensional vector, $\{e_{i,1}^k, e_{i,2}^k, \dots, e_{i,n}^k\}$, where each dimension corresponds to a decision variable of the optimization problem to be solved. The quality of each egg, \mathbf{e}_i^k (candidate solution), is evaluated by using an objective function, $f(\mathbf{e}_i^k)$, whose final result represents the fitness value of \mathbf{e}_i^k . Three different operators define the evolution process of CS: (A) Lévy flight, (B) replacement of some nests by constructing new solutions, and (C) elitist selection strategy.

2.1. Lévy Flight (A). One of the most powerful features of cuckoo search is the use of Lévy flights to generate new candidate solutions (eggs). Under this approach, a new candidate solution, $\mathbf{e}_i^{k+1} (i \in [1, \dots, N])$, is produced by perturbing the current \mathbf{e}_i^k with a change of position \mathbf{c}_i . In order to obtain \mathbf{c}_i , a random step, \mathbf{s}_i , is generated by a symmetric Lévy distribution. For producing \mathbf{s}_i , Mantegna's algorithm [47] is employed as follows:

$$\mathbf{s}_i = \frac{\mathbf{u}}{|\mathbf{v}|^{1/\beta}}, \tag{1}$$

where $\mathbf{u} (\{u_1, \dots, u_n\})$ and $\mathbf{v} (\{v_1, \dots, v_n\})$ are n -dimensional vectors and $\beta = 3/2$. Each element of \mathbf{u} and \mathbf{v} is calculated by considering the following normal distributions:

$$u \sim N(0, \sigma_u^2), \quad v \sim N(0, \sigma_v^2), \tag{2}$$

$$\sigma_u = \left(\frac{\Gamma(1 + \beta) \cdot \sin(\pi \cdot \beta/2)}{\Gamma((1 + \beta)/2) \cdot \beta \cdot 2^{(\beta-1)/2}} \right)^{1/\beta}, \quad \sigma_v = 1,$$

where $\Gamma(\cdot)$ represents the gamma distribution. Once \mathbf{s}_i has been calculated, the required change of position \mathbf{c}_i is computed as follows:

$$\mathbf{c}_i = 0.01 \cdot \mathbf{s}_i \oplus (\mathbf{e}_i^k - \mathbf{e}^{\text{best}}), \tag{3}$$

where the product \oplus denotes entrywise multiplications whereas \mathbf{e}^{best} is the best solution (egg) seen so far in terms of its fitness value. Finally, the new candidate solution, \mathbf{e}_i^{k+1} , is calculated by using

$$\mathbf{e}_i^{k+1} = \mathbf{e}_i^k + \mathbf{c}_i. \tag{4}$$

2.2. Replacement of Some Nests by Constructing New Solutions (B). Under this operation, a set of individuals (eggs) are probabilistically selected and replaced with a new value. Each individual, $\mathbf{e}_i^k (i \in [1, \dots, N])$, can be selected with a probability of $p_a \in [0, 1]$. In order to implement this operation, a uniform random number, r_1 , is generated within the range $[0, 1]$. If r_1 is less than p_a , the individual \mathbf{e}_i^k is selected and modified according to (5). Otherwise, \mathbf{e}_i^k remains without change. This operation can be resumed by the following model:

$$\mathbf{e}_i^{k+1} = \begin{cases} \mathbf{e}_i^k + \text{rand} \cdot (\mathbf{e}_{d_1}^k - \mathbf{e}_{d_2}^k), & \text{with probability } p_a, \\ \mathbf{e}_i^k, & \text{with probability } (1 - p_a), \end{cases} \tag{5}$$

where rand is a random number normally distributed, whereas d_1 and d_2 are random integers from 1 to N .

2.3. Elitist Selection Strategy (C). After producing \mathbf{e}_i^{k+1} either by operator A or by operator B, it must be compared with its past value \mathbf{e}_i^k . If the fitness value of \mathbf{e}_i^{k+1} is better than \mathbf{e}_i^k , then \mathbf{e}_i^{k+1} is accepted as the final solution. Otherwise, \mathbf{e}_i^k is retained. This procedure can be resumed by the following statement:

$$\mathbf{e}_i^{k+1} = \begin{cases} \mathbf{e}_i^{k+1}, & \text{if } f(\mathbf{e}_i^{k+1}) < f(\mathbf{e}_i^k), \\ \mathbf{e}_i^k, & \text{otherwise.} \end{cases} \tag{6}$$

This elitist selection strategy denotes that only high-quality eggs (best solutions near the optimal value) which are more similar to the host bird's eggs have the opportunity to develop (next generation) and become mature cuckoos.

2.4. Complete CS Algorithm. CS is a relatively simple algorithm with only three adjustable parameters: p_a , the population size, N , and the number of generations gen . According to Yang and Deb [39], the convergence rate of the algorithm is not strongly affected by the value of p_a and it is suggested to use $p_a = 0.25$. The operation of CS is divided in two parts: initialization and the evolution process. In the initialization ($k = 0$), the first population, $\mathbf{E}^0 (\{\mathbf{e}_1^0, \mathbf{e}_2^0, \dots, \mathbf{e}_N^0\})$, is produced. The values, $\{e_{i,1}^0, e_{i,2}^0, \dots, e_{i,n}^0\}$, of each individual, \mathbf{e}_i^k , are randomly and uniformly distributed between the prespecified

(1) Input: p_a, N and gen	
(2) Initialize $\mathbf{E}^0 (k = 0)$	
(3) until ($k = 2 \cdot \text{gen}$)	
(5) $\mathbf{E}^{k+1} \leftarrow \text{OperatorA}(\mathbf{E}^k)$	Section 2.1
(6) $\mathbf{E}^{k+1} \leftarrow \text{OperatorC}(\mathbf{E}^k, \mathbf{E}^{k+1})$	Section 2.3
(7) $\mathbf{E}^{k+2} \leftarrow \text{OperatorB}(\mathbf{E}^{k+1})$	Section 2.2
(8) $\mathbf{E}^{k+1} \leftarrow \text{OperatorC}(\mathbf{E}^{k+1}, \mathbf{E}^{k+2})$	Section 2.3
(9) end until	

ALGORITHM 1: Cuckoo search (CS) algorithm.

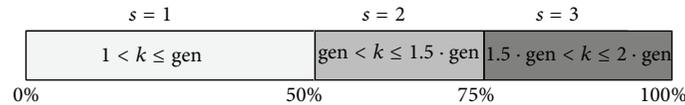


FIGURE 1: Division of the evolution process according to MCS.

lower initial parameter bound, b_j^{low} , and the upper initial parameter bound, b_j^{high} . One has

$$e_{i,j}^0 = b_j^{\text{low}} + \text{rand} \cdot (b_j^{\text{high}} - b_j^{\text{low}}); \quad (7)$$

$$i = 1, 2, \dots, N; \quad j = 1, 2, \dots, n.$$

In the evolution process, operators A (Lévy flight), B (replacement of some nests by constructing new solutions), and C (elitist selection strategy) are iteratively applied until the number of iterations $k = 2 \cdot \text{gen}$ has been reached. The complete CS procedure is illustrated in Algorithm 1.

From Algorithm 1, it is important to remark that the elitist selection strategy (C) is used two times, just after operator A or operator B is executed.

3. The Multimodal Cuckoo Search (MCS)

In CS, individuals emulate eggs which interact in a biological system by using evolutionary operations based on the breeding behavior of some cuckoo species. One of the most powerful features of CS is the use of Lévy flights to generate new candidate solutions. Under this approach, candidate solutions are modified by employing many small changes and occasionally large jumps. As a result, CS can substantially improve the relationship between exploration and exploitation, still enhancing its search capabilities. Despite such characteristics, the CS method still fails to provide multiple solutions in a single execution. In the proposed MCS approach, the original CS is adapted to include multimodal capacities. In particular, this adaptation contemplates (1) the incorporation of a memory mechanism to efficiently register potential local optima according to their fitness value and the distance to other potential solutions, (2) the modification of the original CS individual selection strategy to accelerate the detection process of new local minima, and (3) the inclusion of a depuration procedure to cyclically eliminate duplicated memory elements.

In order to implement these modifications, the proposed MCS divides the evolution process in three asymmetric states. The first state ($s = 1$) includes 0 to 50% of the evolution process. The second state ($s = 2$) involves 50 to 75%. Finally, the third state ($s = 3$) lasts from 75 to 100%. The idea of this division is that the algorithm can react in a different manner depending on the current state. Therefore, in the beginning of the evolutionary process, exploration can be privileged, while, at the end of the optimization process, exploitation can be favored. Figure 1 illustrates the division of the evolution process according to MCS.

The next sections examine the operators suggested by MCS as adaptations of CS to provide multimodal capacities. These operators are (D) the memory mechanism, (E) new selection strategy, and (F) depuration procedure.

3.1. Memory Mechanism (D). In the MCS evolution process, a population, $\mathbf{E}^k (\{\mathbf{e}_1^k, \mathbf{e}_2^k, \dots, \mathbf{e}_N^k\})$, of N eggs (individuals) is evolved from the initial point ($k = 0$) to a total gen number iterations ($k = 2 \cdot \text{gen}$). Each egg, \mathbf{e}_i^k ($i \in [1, \dots, N]$), represents an n -dimensional vector, $\{e_{i,1}^k, e_{i,2}^k, \dots, e_{i,n}^k\}$, where each dimension corresponds to a decision variable of the optimization problem to be solved. The quality of each egg, \mathbf{e}_i^k (candidate solution), is evaluated by using an objective function, $f(\mathbf{e}_i^k)$, whose final result represents the fitness value of \mathbf{e}_i^k . During the evolution process, MCS maintains also the best, $\mathbf{e}^{\text{best},k}$, and the worst, $\mathbf{e}^{\text{worst},k}$, eggs seen so far, such that

$$\mathbf{e}^{\text{best},k} = \arg \min_{i \in \{1, 2, \dots, N\}, a \in \{1, 2, \dots, k\}} (f(\mathbf{e}_i^a)), \quad (8)$$

$$\mathbf{e}^{\text{worst},k} = \arg \min_{i \in \{1, 2, \dots, N\}, a \in \{1, 2, \dots, k\}} (f(\mathbf{e}_i^a)).$$

Global and local optima possess two important characteristics: (1) they have a significant good fitness value and (2)

they represent the best fitness value inside a determined neighborhood. Therefore, the memory mechanism allows efficiently registering potential global and local optima during the evolution process, involving a memory array, \mathbf{M} , and a storage procedure. \mathbf{M} stores the potential global and local optima, $\{\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_T\}$, during the evolution process, with T being the number of elements so far that are contained in the memory \mathbf{M} . On the other hand, the storage procedure indicates the rules that the eggs, $\{\mathbf{e}_1^k, \mathbf{e}_2^k, \dots, \mathbf{e}_N^k\}$, must fulfill in order to be captured as memory elements. The memory mechanism operates in two phases: initialization and capture.

3.1.1. Initialization Phase. This phase is applied only once within the optimization process. Such an operation is achieved in the null iteration ($k = 0$) of the evolution process. In the initialization phase, the best egg, \mathbf{e}_B , of \mathbf{E}^0 , in terms of its fitness value, is stored in the memory \mathbf{M} ($\mathbf{m}_1 = \mathbf{e}_B$), where $\mathbf{e}_B = \arg \min_{i \in \{1, 2, \dots, N\}} (f(\mathbf{e}_i^0))$, for a minimization problem.

3.1.2. Capture Phase. This phase is applied from the first ($k = 1$) iteration to the last iteration ($k = 2, 3, \dots, 2 \cdot \text{gen}$), at the end of each operator (A and B). At this stage, eggs, $\{\mathbf{e}_1^k, \mathbf{e}_2^k, \dots, \mathbf{e}_N^k\}$, corresponding to potential global and local optima are efficiently registered as memory elements, $\{\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_T\}$, according to their fitness value and the distance to other potential solutions. In the operation, each egg, \mathbf{e}_i^k , of \mathbf{E}^k is tested in order to evaluate if it must be captured as a memory element. The test considers two rules:

- (1) significant fitness value rule and
- (2) nonsignificant fitness value rule.

Significant Fitness Value Rule. Under this rule, the solution quality of \mathbf{e}_i^k is evaluated according to the worst element, $\mathbf{m}^{\text{worst}}$, that is contained in the memory \mathbf{M} , where $\mathbf{m}^{\text{worst}} = \arg \max_{i \in \{1, 2, \dots, T\}} (f(\mathbf{m}_i))$, in case of a minimization problem. If the fitness value of \mathbf{e}_i^k is better than $\mathbf{m}^{\text{worst}}$ ($f(\mathbf{e}_i^k) < f(\mathbf{m}^{\text{worst}})$), \mathbf{e}_i^k is considered potential global and local optima. The next step is to decide whether \mathbf{e}_i^k represents a new optimum or it is very similar to an existent memory element, $\{\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_T\}$ (if it is already contained in the memory \mathbf{M}). Such a decision is specified by an acceptance probability function, $\text{Pr}(\delta_{i,u}, s)$, that depends, on one side, on the distances $\delta_{i,u}$ from \mathbf{e}_i^k to the nearest memory element \mathbf{m}_u and, on the other side, on the current state s of the evolution process (1, 2, and 3). Under $\text{Pr}(\delta_{i,u}, s)$, the probability that \mathbf{e}_i^k would be part of \mathbf{M} increases as the distance $\delta_{i,u}$ enlarges. Similarly, the probability that \mathbf{e}_i^k would be similar to an existent memory element $\{\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_T\}$ increases as $\delta_{i,u}$ decreases. On the other hand, the indicator s that relates a numeric value with the state of the evolution process is gradually modified during the algorithm to reduce the likelihood of accepting inferior solutions. The idea is that in the beginning of the evolutionary process (exploration), large distance differences can be considered, while only small distance differences are tolerated at the end of the optimization process.

In order to implement this procedure, the normalized distance $\delta_{i,q}$ ($q \in [1, \dots, T]$) is calculated from \mathbf{e}_i^k to all the elements of the memory \mathbf{M} $\{\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_T\}$. $\delta_{i,q}$ is computed as follows:

$$\delta_{i,q} = \sqrt{\left(\frac{e_{i,1}^k - m_{q,1}}{b_1^{\text{high}} - b_1^{\text{low}}}\right)^2 + \left(\frac{e_{i,2}^k - m_{q,2}}{b_2^{\text{high}} - b_2^{\text{low}}}\right)^2 + \dots + \left(\frac{e_{i,n}^k - m_{q,n}}{b_n^{\text{high}} - b_n^{\text{low}}}\right)^2} \tag{9}$$

where $\{m_{q,1}, m_{q,2}, \dots, m_{q,n}\}$ represent the n components of the memory element \mathbf{m}_q , whereas b_j^{high} and b_j^{low} indicate the low j parameter bound and the upper j parameter bound ($j \in [1, \dots, n]$), respectively. One important property of the normalized distance $\delta_{i,q}$ is that its values fall into the interval $[0, 1]$.

By using the normalized distance $\delta_{i,q}$ the nearest memory element \mathbf{m}_u to \mathbf{e}_i^k is defined, with $\mathbf{m}_u = \arg \min_{j \in \{1, 2, \dots, T\}} (\delta_{i,j})$. Then, the acceptance probability function $\text{Pr}(\delta_{i,u}, s)$ is calculated by using the following expression:

$$\text{Pr}(\delta_{i,u}, s) = (\delta_{i,u})^s \tag{10}$$

In order to decide whether \mathbf{e}_i^k represents a new optimum or it is very similar to an existent memory element, a uniform random number r_1 is generated within the range $[0, 1]$. If r_1 is less than $\text{Pr}(\delta_{i,u}, s)$, the egg \mathbf{e}_i^k is included in the memory \mathbf{M} as a new optimum. Otherwise, it is considered that \mathbf{e}_i^k is similar

to \mathbf{m}_u . Under such circumstances, the memory \mathbf{M} is updated by the competition between \mathbf{e}_i^k and \mathbf{m}_u , according to their corresponding fitness values. Therefore, \mathbf{e}_i^k would replace \mathbf{m}_u in case $f(\mathbf{e}_i^k)$ is better than $f(\mathbf{m}_u)$. On the other hand, if $f(\mathbf{m}_u)$ is better than $f(\mathbf{e}_i^k)$, \mathbf{m}_u remains with no change. The complete procedure of the significant fitness value rule can be resumed by the following statement:

$$\mathbf{M} = \begin{cases} \mathbf{m}_{T+1} = \mathbf{e}_i^k, \\ \text{with probability } \text{Pr}(\delta_{i,u}, s), \\ \mathbf{m}_u = \mathbf{e}_i^k \text{ if } f(\mathbf{e}_i^k) < f(\mathbf{m}_u), \\ \text{with probability } 1 - \text{Pr}(\delta_{i,u}, s). \end{cases} \tag{11}$$

In order to demonstrate the significant fitness value rule process, Figure 2 illustrates a simple minimization problem that involves a two-dimensional function, $f(\mathbf{x})$ ($\mathbf{x} = \{x_1, x_2\}$). As an example, it assumed a population, \mathbf{E}^k , of two different particles ($\mathbf{e}_1^k, \mathbf{e}_2^k$), a memory with two memory

(1)	Input: $\mathbf{e}_i^k, \mathbf{e}^{\text{best},k}, \mathbf{e}^{\text{worst},k}$	
(2)	Calculate $p(\mathbf{e}_i^k, \mathbf{e}^{\text{best},k}, \mathbf{e}^{\text{worst},k}) = 1 - (f(\mathbf{e}_i^k) - f(\mathbf{e}^{\text{best},k})) / (f(\mathbf{e}^{\text{worst},k}) - f(\mathbf{e}^{\text{best},k}))$	
(3)	Calculate $P(p) = \begin{cases} p & 0.5 \leq p \leq 1 \\ 0 & 0 \leq p < 0.5 \end{cases}$	
(5)	if (rand(0, 1) $\leq P$) then	
(6)	\mathbf{e}_i^k is considered a local optimum	With probability P
(7)	else	
(8)	\mathbf{e}_i^k is ignored	With probability $1 - P$
(9)	end if	

ALGORITHM 2: Nonsignificant fitness value rule procedure.

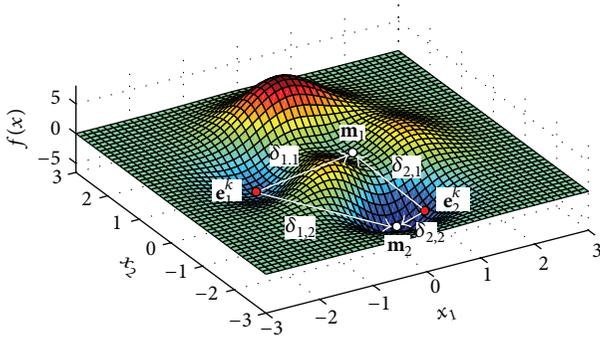
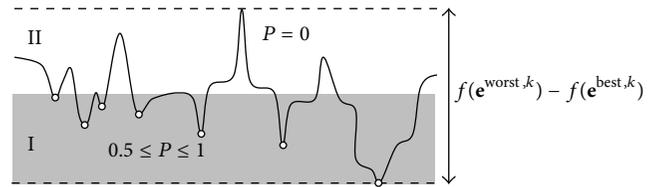


FIGURE 2: Graphical illustration of the significant fitness value rule process.

elements $(\mathbf{m}_1, \mathbf{m}_2)$, and the execution of the first state ($s = 1$). According to Figure 2, both particles \mathbf{e}_1^k and \mathbf{e}_2^k maintain a better fitness value than \mathbf{m}_1 , which possesses the worst fitness value of the memory elements. Under such conditions, the significant fitness value rule must be applied to both particles. In case of \mathbf{e}_1^k , the first step is to calculate the correspondent distances $\delta_{1,1}$ and $\delta_{1,2}$. \mathbf{m}_1 represents the nearest memory element to \mathbf{e}_1^k . Then, the acceptance probability function $\text{Pr}(\delta_{1,1}, 1)$ is calculated by using (10). Since the value of $\text{Pr}(\delta_{1,1}, 1)$ is high, there exists a great probability that \mathbf{e}_1^k becomes the next memory element ($\mathbf{m}_3 = \mathbf{e}_1^k$). On the other hand, for \mathbf{e}_2^k , \mathbf{m}_2 represents the nearest memory element. As $\text{Pr}(\delta_{2,2}, 1)$ is very low, there exists a great probability that \mathbf{e}_2^k competes with \mathbf{m}_2 for a place within \mathbf{M} . In such a case, \mathbf{m}_2 remains with no change considering that $f(\mathbf{m}_2) < f(\mathbf{e}_2^k)$.

Nonsignificant Fitness Value Rule. Different to the significant fitness value rule, the nonsignificant fitness value rule allows capturing local optima with low fitness values. It operates if the fitness value of \mathbf{e}_i^k is worse than $\mathbf{m}^{\text{worst}}$ ($f(\mathbf{e}_i^k) \geq f(\mathbf{m}^{\text{worst}})$). Under such conditions, it is necessary, as a first step, to test which particles could represent local optima and which must be ignored as a consequence of their very low fitness value. Then, if the particle represents a possible local optimum, its inclusion inside the memory \mathbf{M} is explored.

The decision on whether \mathbf{e}_i^k represents a new local optimum or not is specified by a probability function, P ,

FIGURE 3: Effect of the probability function P in a simple example.

which is based on the relationship between $f(\mathbf{e}_i^k)$ and the so far valid fitness value interval ($f(\mathbf{e}^{\text{worst},k}) - f(\mathbf{e}^{\text{best},k})$). Therefore, the probability function P is defined as follows:

$$p(\mathbf{e}_i^k, \mathbf{e}^{\text{best},k}, \mathbf{e}^{\text{worst},k}) = 1 - \frac{f(\mathbf{e}_i^k) - f(\mathbf{e}^{\text{best},k})}{f(\mathbf{e}^{\text{worst},k}) - f(\mathbf{e}^{\text{best},k})}, \quad (12)$$

$$P(p) = \begin{cases} p, & 0.5 \leq p \leq 1, \\ 0, & 0 \leq p < 0.5, \end{cases}$$

where $\mathbf{e}^{\text{best},k}$ and $\mathbf{e}^{\text{worst},k}$ represent the best and worst eggs seen so far, respectively. In order to decide whether \mathbf{p}_i^k represents a new local optimum or it must be ignored, a uniform random number, r_2 , is generated within the range $[0, 1]$. If r_2 is less than P , the egg \mathbf{e}_i^k is considered to be a new local optimum. Otherwise, it must be ignored. Under P , the so far valid fitness value interval ($f(\mathbf{e}^{\text{worst},k}) - f(\mathbf{e}^{\text{best},k})$) is divided into two sections: I and II (see Figure 3). Considering this division, the function P assigns a valid probability (greater than zero) only to those eggs that fall into the zone of the best individuals (part I) in terms of their fitness value. Such a probability value increases as the fitness value improves. The complete procedure can be reviewed in Algorithm 2.

If the particle represents a possible local optimum, its inclusion inside the memory \mathbf{M} is explored. In order to consider if \mathbf{e}_i^k could represent a new memory element, another procedure that is similar to the significant fitness value rule process is applied. Therefore, the normalized distance $\delta_{i,q}$ ($q \in [1, \dots, T]$) is calculated from \mathbf{p}_i^k to all the elements of the memory $\mathbf{M} \{\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_T\}$, according to (9). Afterwards, the nearest distance $\delta_{i,u}$ to \mathbf{e}_i^k is determined.

Then, by using $\Pr(\delta_{i,u}, s)$ (10), the following rule can be thus applied:

$$\mathbf{M} = \begin{cases} \mathbf{m}_{T+1} = \mathbf{e}_i^k, & \text{with probability } \Pr(\delta_{i,u}, s), \\ \text{no change,} & \text{with probability } 1 - \Pr(\delta_{i,u}, s). \end{cases} \quad (13)$$

Under this rule, a uniform random number, r_3 , is generated within the range $[0, 1]$. If r_3 is less than $\Pr(\delta_{i,u}, s)$, the egg \mathbf{e}_i^k is included in the memory \mathbf{M} as a new optimum. Otherwise, the memory does not change.

3.2. New Selection Strategy (E). The original CS selection strategy is mainly conducted by an elitist decision where the best individuals in the current population prevail. Such an operation, defined in this paper as operator C (Section 2.3), is executed two times, just after operators A and B in the original CS method. This effect allows incorporating interesting convergence properties to CS when the objective considers only one optimum. However, in case of multiple-optimum detection, such a strategy is not appropriate. Therefore, in order to accelerate the detection process of potential local minima in our method, the selection strategy is modified to be influenced by the individuals contained in the memory \mathbf{M} .

Under the new selection procedure (operator E), the final population \mathbf{E}^{k+1} is built by considering the first N element from the memory \mathbf{M} instead of using the best individuals between the currents \mathbf{E}^{k+1} and \mathbf{E}^k . In case of the number of elements in \mathbf{M} is less than N , the rest of the individuals are completed by considering the best elements from the current \mathbf{E}^{k+1} .

3.3. Depuration Procedure (F). During the evolution process, the memory \mathbf{M} stores several individuals (eggs). Since such individuals could represent the same local optimum, a depuration procedure is incorporated at the end of each state s (1, 2, and 3) to eliminate similar memory elements. The inclusion of this procedure allows (a) reducing the computational overhead during each state and (b) improving the search strategy by considering only significant memory elements.

Memory elements tend to concentrate on optimal points (good fitness values), whereas element concentrations are enclosed by areas holding bad fitness values. The main idea in the depuration procedure is to find the distances among concentrations. Such distances, considered as depuration ratios, are later employed to delete all elements inside them, except for the best element in terms of their fitness values.

The method used by the depuration procedure in order to determine the distance between two concentrations is based on the element comparison between the concentration corresponding to the best element and the concentration of the nearest optimum in the memory. In the process, the best element \mathbf{m}^{best} in the memory is compared to a memory element, \mathbf{m}_b , which belongs to one of both concentrations (where $\mathbf{m}^{\text{best}} = \arg \min_{i \in \{1, 2, \dots, T\}} (f(\mathbf{m}_i))$). If the fitness value of the medium point, $f((\mathbf{m}^{\text{best}} + \mathbf{m}_b)/2)$, between both is not worse than both, $(f(\mathbf{m}^{\text{best}}), f(\mathbf{m}_b))$, the element \mathbf{m}_b is part of

the same concentration of \mathbf{m}^{best} . However, if $f((\mathbf{m}^{\text{best}} + \mathbf{m}_b)/2)$ is worse than both, the element \mathbf{m}_b is considered as part of the nearest concentration. Therefore, if \mathbf{m}_b and \mathbf{m}^{best} belong to different concentrations, the Euclidian distance between \mathbf{m}_b and \mathbf{m}^{best} can be considered as a depuration ratio. In order to avoid the unintentional deletion of elements in the nearest concentration, the depuration ratio D_R is lightly shortened. Thus, the depuration ratio r is defined as follows:

$$D_R = 0.85 \cdot \|\mathbf{m}^{\text{best}} - \mathbf{m}_b\|. \quad (14)$$

The proposed depuration procedure only considers the depuration ratio r between the concentration of the best element and the nearest concentration. In order to determine all ratios, preprocessing and postprocessing methods must be incorporated and iteratively executed.

The preprocessing method must (1) obtain the best element \mathbf{m}^{best} from the memory in terms of its fitness value, (2) calculate the Euclidian distances from the best element to the rest of the elements in the memory, and (3) sort the distances according to their magnitude. This set of tasks allows identification of both concentrations: the one belonging to the best element and that belonging to the nearest optimum, so they must be executed before the depuration ratio D_R calculation. Such concentrations are represented by the elements with the shortest distances to \mathbf{m}^{best} . Once D_R has been calculated, it is necessary to remove all the elements belonging to the concentration of the best element. This task is executed as a postprocessing method in order to configure the memory for the next step. Therefore, the complete depuration procedure can be represented as an iterative process that at each step determines the distance of the concentration of the best element with regard to the concentration of the nearest optimum.

A special case can be considered when only one concentration is contained within the memory. This case can happen because the optimization problem has only one optimum or because all the other concentrations have been already detected. Under such circumstances, the condition where $f((\mathbf{m}^{\text{best}} + \mathbf{m}_b)/2)$ is worse than $f(\mathbf{m}^{\text{best}})$ and $f(\mathbf{m}_b)$ would be never fulfilled.

In order to find the distances among concentrations, the depuration procedure is conducted in Procedure 1.

At the end of the above procedure, the vector \mathbf{Y} will contain the depurated memory which would be used in the next state or as a final result of the multimodal problem.

In order to illustrate the depuration procedure, Figure 4 shows a simple minimization problem that involves two different optimal points (concentrations). As an example, it assumed a memory, \mathbf{M} , with six memory elements whose positions are shown in Figure 4(a). According to the depuration procedure, the first step is (1) to build the vector \mathbf{Z} and (2) to calculate the corresponding distance $\Delta_{1,j}^a$ among the elements. Following such operation, the vector \mathbf{Z} and the set of distances are configured as $\mathbf{Z} = \{\mathbf{m}_5, \mathbf{m}_1, \mathbf{m}_3, \mathbf{m}_4, \mathbf{m}_6, \mathbf{m}_2\}$ and $\{\Delta_{1,2}^1, \Delta_{1,3}^2, \Delta_{1,5}^3, \Delta_{1,4}^4, \Delta_{1,6}^5\}$, respectively. Figure 4(b) shows the configuration of \mathbf{X} where, for sake of easiness, only the two distances $\Delta_{1,2}^1$ and $\Delta_{1,5}^3$ have been represented. Then,

- (1) Define two new temporal vectors \mathbf{Z} and \mathbf{Y} . The vector \mathbf{Z} will hold the results of the iterative operations whereas \mathbf{Y} will contain the final memory configuration. The vector \mathbf{Z} is initialized with the elements of \mathbf{M} that have been sorted according to their fitness values, so that the first element represents the best one. On other hand, \mathbf{Y} is initialized empty.
- (2) Store the best element \mathbf{z}_1 of the current \mathbf{Z} in \mathbf{Y} .
- (3) Calculate the Euclidian distances $\Delta_{1,j}$ between \mathbf{z}_1 and the rest of elements from \mathbf{Z} ($j \in \{2, \dots, |\mathbf{Z}|\}$), where $|\mathbf{Z}|$ represents the number of elements in \mathbf{Z} .
- (4) Sort the distances $\Delta_{1,j}$ according to their magnitude. Therefore, a new index a is incorporated to each distance $\Delta_{1,j}^a$, where a indicate the place of $\Delta_{1,j}$ after the sorting operation. ($a = 1$ represents the shortest distance).
- (5) Calculate the deputation ratio D_R :


```

for  $q = 1$  to  $|\mathbf{Z}| - 1$ 
  Obtain the element  $\mathbf{z}_j$  corresponding to the distance  $\Delta_{1,j}^a$ 
  Compute  $f((\mathbf{z}_1 + \mathbf{z}_j)/2)$ 
  if  $(f((\mathbf{z}_1 + \mathbf{z}_j)/2) > f(\mathbf{z}_1) \text{ and } f((\mathbf{z}_1 + \mathbf{z}_j)/2) > f(\mathbf{z}_j))$ 
     $D_R = 0.85 \cdot \|\mathbf{x}_1 - \mathbf{x}_j\|$ 
    break
  end if
  if  $q = |\mathbf{Z}| - 1$ 
    There is only one concentration
  end if
end for

```
- (6) Remove all elements inside D_R from \mathbf{Z} .
- (7) Sort the elements of \mathbf{Z} according to their fitness values.
- (8) Stop, if there are more concentrations, otherwise return to Step 2.

PROCEDURE 1

- | | | |
|------|--|-------------|
| (1) | Input: p_a, N and gen | |
| (2) | Initialize $\mathbf{E}^0 (k = 0)$ | |
| (3) | until $(k = 2 \cdot \text{gen})$ | |
| (5) | $\mathbf{E}^{k+1} \leftarrow \text{OperatorA}(\mathbf{E}^k)$ | Section 2.1 |
| (6) | $\mathbf{M} \leftarrow \text{OperatorD}(\mathbf{E}^{k+1})$ | Section 3.1 |
| (7) | $\mathbf{E}^{k+1} \leftarrow \text{OperatorE}(\mathbf{M}, \mathbf{E}^{k+1})$ | Section 3.2 |
| (8) | $\mathbf{E}^{k+2} \leftarrow \text{OperatorB}(\mathbf{E}^{k+1})$ | Section 2.2 |
| (9) | $\mathbf{M} \leftarrow \text{OperatorD}(\mathbf{E}^{k+2})$ | Section 3.1 |
| (10) | $\mathbf{E}^{k+2} \leftarrow \text{OperatorE}(\mathbf{M}, \mathbf{E}^{k+2})$ | Section 3.2 |
| (11) | if $(s \text{ has changed})$ | |
| (12) | $\mathbf{M} \leftarrow \text{OperatorF}(\mathbf{M})$ | Section 3.3 |
| (13) | end if | |
| (14) | end until | |

ALGORITHM 3: Multimodal cuckoo search (MCS) algorithm.

the deputation ratio R is calculated. This process is an iterative computation that begins with the shortest distance $\Delta_{1,2}^1$. The distance $\Delta_{1,2}^1$ (see Figure 4(c)), corresponding to \mathbf{z}_1 and \mathbf{z}_2 , produces the evaluation of their medium point u ($(\mathbf{z}_1 + \mathbf{z}_2)/2$). Since $f(u)$ is worse than $f(\mathbf{z}_1)$ but not worse than $f(\mathbf{z}_2)$, the element \mathbf{z}_2 is considered to be part of the same concentration as \mathbf{z}_1 . The same conclusion is obtained for $\Delta_{1,3}^2$ in case of \mathbf{z}_3 , after considering the point v . For $\Delta_{1,5}^3$, the point w is produced. Since $f(w)$ is worse than $f(\mathbf{z}_1)$ and $f(\mathbf{z}_5)$, the element \mathbf{z}_5 is considered to be part of the concentration corresponding to the next optimum. The iterative process ends here, after assuming that the same result is produced with $\Delta_{1,4}^4$ and $\Delta_{1,6}^5$, for \mathbf{z}_4 and \mathbf{z}_6 , respectively. Therefore, the

deputation ratio D_R is calculated as 85% of the distances $\Delta_{1,5}^3$. Once the elements inside of D_R have been removed from \mathbf{Z} , the same process is applied to the new \mathbf{Z} . As a result, the final configuration of the memory is shown in Figure 4(d).

3.4. Complete MCS Algorithm. Once the new operators (D) memory mechanism, (E) new selection strategy, and (F) deputation procedure have been defined, the proposed MCS algorithm can be summarized by Algorithm 3. The new algorithm combines operators defined in the original CS with the new ones. Despite these new operators, the MCS maintains the same three adjustable parameters (p_a , N , and gen) compared to the original CS method.

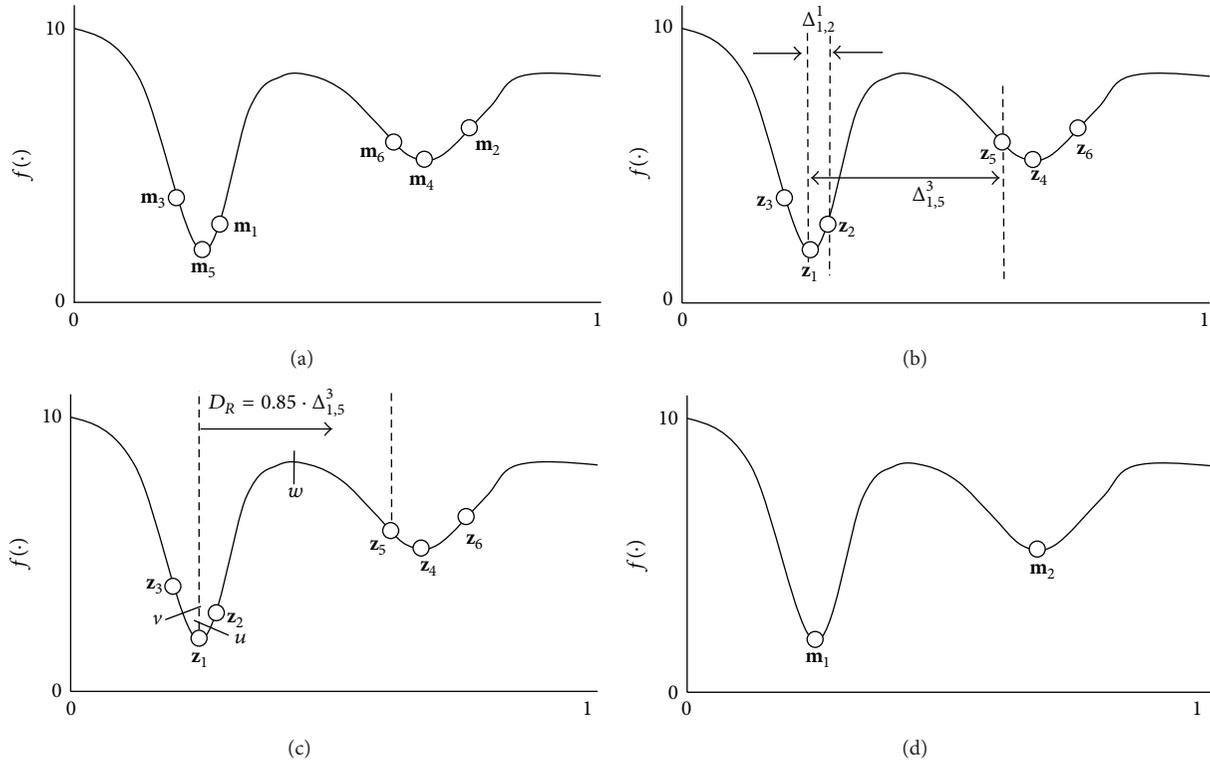


FIGURE 4: Depuration procedure. (a) Initial memory configuration, (b) vector Z and distances $\Delta_{1,j}^a$, (c) the determination of the depuration ratio R , and (d) the final memory configuration.

4. Experimental Results

This section presents the performance of the proposed algorithm beginning from Section 4.1 that describes the experimental methodology. For the sake of clarity, results are divided into two sections, Section 4.2 and Section 4.3, which report the comparison between the MCS experimental results and the outcomes produced by other multimodal metaheuristic algorithms.

4.1. Experimental Methodology. This section examines the performance of the proposed MCS by using a test suite of fourteen benchmark functions with different complexities. Table 3 in the appendix presents the benchmark functions used in our experimental study. In the table, **NO** indicates the number of optimal points in the function and **S** indicates the search space (subset of R^2). The experimental suite contains some representative, complicated, and multimodal functions with several local optima. Such functions are considered complex entities to be optimized, as they are particularly challenging to the applicability and efficiency of multimodal metaheuristic algorithms. A detailed description of each function is given in the appendix.

In the study, five performance indexes are compared: the effective peak number (EPN), the maximum peak ratio (MPR), the peak accuracy (PA), the distance accuracy (DA), and the number of function evaluations (NFE). The first four indexes assess the accuracy of the solution, whereas the last measures the computational cost.

The effective peak number (EPN) expresses the amount of detected peaks. An optimum \mathbf{o}_j is considered as detected if the distance between the identified solution \mathbf{z}_j and the optimum \mathbf{o}_j is less than 0.01 ($\|\mathbf{o}_j - \mathbf{z}_j\| < 0.01$). The maximum peak ratio (MPR) is used to evaluate the quality and the number of identified optima. It is defined as follows:

$$MPR = \frac{\sum_{i=1}^t f(\mathbf{z}_i)}{\sum_{j=1}^q f(\mathbf{o}_j)}, \tag{15}$$

where t represents the number of identified solutions (identified optima) for the algorithm under testing and q represents the number of true optima contained in the function. The peak accuracy (PA) specifies the total error produced between the identified solutions and the true optima. Therefore, PA is calculated as follows:

$$PA = \sum_{j=1}^q |f(\mathbf{o}_j) - f(\mathbf{z}_j)|. \tag{16}$$

Peak accuracy (PA) may lead to erroneous results, mainly if the peaks are close to each other or hold an identical height. Under such circumstances, the distance accuracy (DA) is used to avoid such error. DA is computed as PA, but fitness values are replaced by the Euclidian distance. DA is thus defined by the following model:

$$DA = \sum_{j=1}^q \|\mathbf{o}_j - \mathbf{z}_j\|. \tag{17}$$

The number of function evaluations (NFE) indicates the total number of function computations that have been calculated by the algorithm under testing, through the overall optimization process.

The experiments compare the performance of MCS against the crowding differential evolution (CDE) [22], the fitness sharing differential evolution (SDE) [21, 22], the clearing procedure (CP) [26], the elitist-population strategy (AEGA) [28], the clonal selection algorithm (CSA) [30], and the artificial immune network (AiNet) [31].

Since the approach solves real-valued multimodal functions and a fair comparison must be assured, we have used for the GA approaches a consistent real coding variable representation and uniform operators for crossover and mutation. The crossover probability of $P_c = 0.8$ and the mutation probability of $P_m = 0.1$ have been used. We have employed the standard tournament selection operator with tournament size = 2 for implementing the sequential fitness sharing, the clearing procedure, and the elitist-population strategy (AEGA). On the other hand, the parameter values for the AiNet algorithm have been defined as suggested in [31], with the mutation strength of $\beta = 100$, the suppression threshold of $\sigma_{s(\text{AiNet})} = 0.2$, and the update rate of $d = 40\%$. Algorithms based on DE use a scaling factor of $F = 0.5$ and a crossover probability of $P_c = 0.9$. The crowding DE employs a crowding factor of $CF = 50$ and the sharing DE considers $\alpha = 1.0$ with a share radius of $\sigma_{\text{share}} = 0.1$.

In case of the MCS algorithm, the parameters are set to $p_a = 0.25$, the population size is $N = 50$, and the number of generations is $\text{gen} = 500$. Once they have been all experimentally determined, they are kept for all the test functions through all experiments.

To avoid relating the optimization results to the choice of a particular initial population and to conduct fair comparisons, we perform each test 50 times, starting from various randomly selected points in the search domain as it is commonly done in the literature.

All algorithms have been tested in MatLAB® over the same Dell Optiplex GX520 computer with a Pentium-4 2.66G-HZ processor, running Windows XP operating system over 1 Gb of memory. The sections below present experimental results for multimodal optimization problems which have been divided into two groups. The first one considers functions f_1-f_7 , while the second gathers functions f_8-f_{14} .

4.2. Comparing MCS Performance for Functions f_1-f_7 . This section presents a performance comparison for different algorithms solving the multimodal problems f_1-f_7 that are shown in Table 3. The aim is to determine whether MCS is more efficient and effective than other existing algorithms for finding all multiple optima of f_1-f_7 . All the algorithms employ a population size of 50 individuals using 500 successive generations.

Table 1 provides a summarized performance comparison among several algorithms in terms of the effective peak number (EPN), the maximum peak ratio (MPR), the peak accuracy (PA), the distance accuracy (DA), and the number

of function evaluations (NFE). The results are averaged by considering 50 different executions.

Considering the EPN index, in all functions f_1-f_7 , MCS always finds better or equally optimal solutions. Analyzing results of function f_1 , the CDE, AEGA, and the MCS algorithms reach all optima. In case of function f_2 , only CSA and AiNet have not been able to detect all the optima values each time. Considering function f_3 , only MCS can detect all optima at each run. In case of function f_4 , most of the algorithms detect only half of the total optima but MCS can recognize most of them. Analyzing results of function f_5 , CDE, CP, CSA, and AiNet present a similar performance whereas SDE, AEGA, and MCS obtain the best EPN values. In case of f_6 , almost all algorithms present a similar performance; however, only the CDE, CP, and MCS algorithms have been able to detect all optima. Considering function f_7 , the MCS algorithm is able to detect most of the optima whereas the rest of the methods reach different performance levels.

By analyzing the MPR index in Table 1, MCS has reached the best performance for all the multimodal problems. On the other hand, the rest of the algorithms present different accuracy levels, with CDE and SDE being the most consistent.

Considering the PA index, MCS presents the best performance. Since PA evaluates the accumulative differences of fitness values, it could drastically change when one or several peaks are not detected (function f_3) or when the function under testing presents peaks with high values (function f_4). For the case of the DA index in Table 1, it is evident that the MCS algorithm presents the best performance providing the shortest distances among the detected optima.

Analyzing the NFE measure in Table 1, it is clear that CSA and AiNet need fewer function evaluations than other algorithms considering the same termination criterion. This fact is explained by considering that both algorithms do not implement any additional process in order to detect multiple optima. On the other hand, the MCS method maintains a slightly higher number of function evaluations than CSA and AiNet due to the inclusion of the depuration procedure. The rest of the algorithms present a considerable higher NFE value.

It can be easily deduced from such results that the MCS algorithm is able to produce better search locations (i.e., a better compromise between exploration and exploitation) in a more efficient and effective way than other multimodal search strategies by using an acceptable number of function evaluations.

4.3. Comparing MCS Performance for Functions f_8-f_{14} . This section presents a performance comparison for different algorithms solving the multimodal problems f_8-f_{14} that are shown in Table 3. The aim is to determine whether MCS is more efficient and effective than its competitors for finding multiple optima in f_8-f_{14} . All the algorithms employ a population size of 50 individuals using 500 successive generations. Table 2 provides a summarized performance comparison among several algorithms in terms of the effective peak number (EPN), the maximum peak ratio (MPR), the peak accuracy (PA), the distance accuracy (DA), and the number

TABLE 1: Performance comparison among multimodal optimization algorithms for the test functions f_1-f_7 . For all the parameters, numbers in parentheses are the standard deviations.

Function	Algorithm	EPN	MPR	PA	DA	NFE
f_1	CDE	3 (0)	0.9996 (0.0004)	0.0995 (0.1343)	0.0305 (0.0169)	27432 (1432)
	SDE	2.96 (0.18)	0.9863 (0.0523)	1.3053 (0.8843)	0.1343 (0.0483)	31435 (2342)
	CP	2.93 (0.25)	0.9725 (0.0894)	1.3776 (1.0120)	0.1432 (0.0445)	34267 (4345)
	AEGA	3 (0)	0.9932 (0.0054)	0.0991 (0.2133)	0.1031 (0.0065)	30323 (2316)
	CSA	2.91 (0.20)	0.9127 (0.0587)	1.4211 (1.0624)	0.2188 (0.0072)	25050 (0)
	AiNet	2.94 (0.20)	0.9002 (0.0901)	1.3839 (1.0214)	0.1760 (0.0067)	25050 (0)
	MCS	3 (0)	1 (0)	0.0005 (0.0001)	0.0007 (0.0002)	25433 (54)
f_2	CDE	12 (0)	1 (0)	0.0015 (0.0010)	0.2993 (0.0804)	26321 (1934)
	SDE	12 (0)	1 (0)	0.0018 (0.0006)	0.3883 (0.0657)	32563 (1453)
	CP	12 (0)	1 (0)	0.0009 (0.0003)	0.2694 (0.0506)	30324 (3521)
	AEGA	12 (0)	0.9987 (0.0011)	0.0988 (0.0097)	0.3225 (0.0058)	29954 (1987)
	CSA	11.92 (0.41)	0.9011 (0.0091)	0.1055 (0.0121)	0.4257 (0.0096)	25050 (0)
	AiNet	11.96 (0.30)	0.9256 (0.0074)	0.0996 (0.0105)	0.3239 (0.0081)	25050 (0)
	MCS	12 (0)	1 (0)	0.0001 (0.0001)	0.0073 (0.0002)	25188 (42)
f_3	CDE	23.03 (1.77)	0.8780 (0.0956)	180.47 (265.54)	9.3611 (6.4667)	28654 (2050)
	SDE	20.06 (2.59)	0.6980 (0.1552)	155.52 (184.59)	14.892 (7.5935)	31432 (1017)
	CP	21.03 (1.90)	0.7586 (0.1125)	192.32 (146.21)	11.195 (3.1490)	32843 (2070)
	AEGA	20.45 (1.21)	0.7128 (0.1493)	134.47 (157.37)	16.176 (8.0751)	30965 (2154)
	CSA	18.02 (2.41)	0.5875 (0.1641)	185.64 (104.24)	21.057 (10.105)	25050 (0)
	AiNet	19.24 (2.01)	0.6123 (0.1247)	179.35 (164.37)	18.180 (9.1112)	25050 (0)
	MCS	24.66 (1.01)	0.9634 (0.0397)	2.9408 (4.3888)	15.521 (8.0834)	25211 (37)
f_4	CDE	3.46 (1.00)	0.4929 (0.1419)	395.46 (305.01)	210.940 (72.99)	29473 (3021)
	SDE	3.73 (0.86)	0.5301 (0.1268)	544.48 (124.11)	206.65 (160.84)	33421 (1342)
	CP	3.26 (0.63)	0.4622 (0.0869)	192.32 (146.21)	199.41 (68.434)	29342 (1543)
	AEGA	3.51 (0.52)	0.5031 (0.0754)	188.23 (101.54)	187.21 (33.211)	32756 (1759)
	CSA	3.12 (0.11)	0.4187 (0.0464)	257.54 (157.18)	278.14 (47.120)	25050 (0)
	AiNet	3.20 (0.47)	0.5164 (0.0357)	197.24 (86.21)	178.23 (29.191)	25050 (0)
	MCS	6.26 (0.82)	0.8919 (0.1214)	41.864 (16.63)	39.938 (12.962)	25361 (81)
f_5	CDE	22.96 (2.25)	0.4953 (0.0496)	0.2348 (0.0269)	17.83 (7.1214)	28543 (1345)
	SDE	31.40 (2.35)	0.6775 (0.0503)	0.7005 (0.0849)	3.9430 (0.9270)	30543 (1576)
	CP	21.33 (2.00)	0.4599 (0.0436)	1.3189 (0.5179)	10.766 (1.9245)	28743 (2001)
	AEGA	30.11 (2.01)	0.6557 (0.0127)	0.8674 (0.0296)	2.870 (1.6781)	29765 (1911)
	CSA	24.79 (3.14)	0.5107 (0.0308)	0.2121 (0.0187)	8.7451 (3.470)	25050 (0)
	AiNet	26.57 (2.35)	0.5005 (0.0471)	0.2087 (0.0324)	6.472 (2.4187)	25050 (0)
	MCS	33.03 (2.07)	0.8535 (0.0251)	0.1617 (0.0283)	4.6012 (1.4206)	25159 (49)
f_6	CDE	6 (0)	0.9786 (0.0157)	0.1280 (0.0942)	0.1231 (0.0182)	30234 (2410)
	SDE	5.86 (0.43)	0.9185 (0.0685)	0.3842 (0.1049)	0.1701 (0.0222)	31453 (1154)
	CP	6 (0)	0.9423 (0.0123)	0.3460 (0.0741)	0.1633 (0.0149)	30231 (832)
	AEGA	5.11 (0.64)	0.8945 (0.0387)	0.4004 (0.0879)	0.1224 (0.0101)	31932 (943)
	CSA	4.97 (0.24)	0.8174 (0.0631)	0.4797 (0.0257)	0.1295 (0.0054)	25050 (0)
	AiNet	5.23 (1)	0.9012 (0.0197)	0.3974 (0.0702)	0.1197 (0.0054)	25050 (0)
	MCS	6 (0)	0.9993 (0.0002)	0.0037 (0.0014)	0.0006 (0.0002)	25463 (37)
f_7	CDE	30.36 (2.77)	0.6200 (0.0566)	2.2053 (1.8321)	330.51 (47.531)	33423 (1021)
	SDE	35.06 (5.15)	0.7162 (0.1051)	1.9537 (0.9290)	243.39 (140.04)	32832 (995)
	CP	35.06 (3.98)	0.7164 (0.0812)	2.4810 (1.4355)	250.11 (78.194)	31923 (834)
	AEGA	32.51 (2.59)	0.7004 (0.0692)	2.0751 (0.9561)	278.78 (46.225)	33821 (1032)
	CSA	31.78 (1.14)	0.6764 (0.4100)	1.9408 (0.9471)	347.21 (38.147)	25050 (0)
	AiNet	34.42 (1.80)	0.7237 (0.0257)	1.8632 (0.0754)	261.27 (61.217)	25050 (0)
	MCS	38.86 (1.54)	0.8014 (0.0313)	0.2290 (0.0166)	49.53 (7.1533)	25643 (97)

TABLE 2: Performance comparison among multimodal optimization algorithms for the test functions f_8 - f_{14} . For all the parameters, numbers in parentheses are the standard deviations.

Function	Algorithm	EPN	MPR	PA	DA	NFE
f_8	CDE	24.16 (2.77)	0.9682 (0.0318)	2.4873 (2.4891)	0.8291 (0.8296)	28453 (2345)
	SDE	18.56 (2.51)	0.4655 (0.0636)	30.21 (43.132)	2.1162 (0.6697)	31328 (945)
	CP	8.80 (1.95)	0.2222 (0.0509)	60.52 (56.056)	6.9411 (0.9500)	30743 (1032)
	AEGA	15.67 (2.21)	0.3934 (0.0534)	40.56 (10.111)	3.2132 (0.2313)	32045 (684)
	CSA	14.54 (3.12)	0.3323 (0.0431)	48.34 (8.343)	3.8232 (0.4521)	25050 (0)
	AiNet	16.78 (2.63)	0.4264 (0.0321)	37.32 (10.432)	2.9832 (0.5493)	25050 (0)
	MCS	24.73 (0.49)	0.9898 (0.0170)	0.900 (1.4771)	0.2584 (0.1275)	25582 (74)
f_9	CDE	2.1 (0.20)	0.7833 (0.0211)	23.235 (7.348)	2.9354 (0.3521)	30074 (1621)
	SDE	2.3 (0.31)	0.8245 (0.0145)	20.745 (8.012)	2.6731 (0.8621)	31497 (245)
	CP	2.4 (0.25)	0.8753 (0.0301)	18.563 (5.865)	2.3031 (0.7732)	29746 (1114)
	AEGA	2.1 (0.10)	0.7879 (0.0174)	22.349 (6.231)	3.0021 (0.6431)	30986 (1027)
	CSA	2 (0)	0.7098 (0.0025)	32.859 (8.659)	3.1432 (0.5431)	25050 (0)
	AiNet	2 (0)	0.7165 (0.0076)	31.655 (6.087)	3.2265 (0.3467)	25050 (0)
	MCS	4.74 (0.25)	0.9154 (0.0163)	2.3515 (2.511)	0.0109 (0.0428)	26043 (112)
f_{10}	CDE	4.12 (0.78)	0.7285 (0.0342)	3.546 (1.034)	3.0132 (0.5321)	29597 (1034)
	SDE	4.64 (0.54)	0.7893 (0.0532)	3.054 (1.127)	2.864 (0.3271)	32743 (964)
	CP	4 (0)	0.7092 (0.0298)	3.881 (1.154)	3.3412 (0.4829)	28463 (1142)
	AEGA	3.43 (0.33)	0.6734 (0.0745)	4.212 (1.312)	3.9121 (0.8732)	29172 (1044)
	CSA	3.76 (0.51)	0.6975 (0.0828)	4.002 (1.197)	3.5821 (0.7498)	25050 (0)
	AiNet	4 (0)	0.7085 (0.0385)	3.797 (1.002)	3.3002 (0.6496)	25050 (0)
	MCS	6.82 (0.75)	0.9274 (0.0137)	0.423 (0.064)	0.6842 (0.0598)	25873 (88)
f_{11}	CDE	10.36 (1.60)	0.8572 (0.1344)	1.859 (0.952)	0.5237 (0.0321)	34156 (2321)
	SDE	10.36 (2.04)	0.8573 (0.1702)	1.268 (0.581)	0.6927 (0.0921)	32132 (975)
	CP	9.16 (1.76)	0.7577 (0.1462)	2.536 (0.890)	0.6550 (0.0440)	30863 (1002)
	AEGA	8.34 (1.32)	0.6954 (0.1021)	4.432 (1.232)	0.7021 (0.0231)	31534 (852)
	CSA	8 (0)	0.6532 (0.1378)	4.892 (1.003)	0.7832 (0.0432)	25050 (0)
	AiNet	8 (0)	0.6438 (0.2172)	4.921 (1.102)	0.7753 (0.0326)	25050 (0)
	MCS	12 (0)	0.9998 (0.0003)	0.011 (0.008)	0.0060 (0.0012)	25789 (121)
f_{12}	CDE	6.21 (1.54)	0.6986 (0.1893)	4.029 (1.401)	5.1514 (1.0351)	31456 (975)
	SDE	5.34 (2.03)	0.5812 (0.1992)	5.075 (1.071)	6.0117 (1.1517)	32481 (1002)
	CP	6.04 (0.61)	0.6312 (0.1771)	4.657 (1.321)	5.3177 (1.7517)	33123 (563)
	AEGA	4 (0)	0.4112 (0.0343)	6.341 (1.034)	7.8751 (1.652)	32634 (843)
	CSA	4 (0)	0.3998 (0.0212)	6.151 (1.121)	7.7976 (1.0043)	25050 (0)
	AiNet	4 (0)	0.4034 (0.0973)	6.003 (1.735)	7.6613 (1.1219)	25050 (0)
	MCS	9.65 (1.45)	0.9411 (0.0087)	0.015 (0.009)	0.1043 (0.0864)	25832 (65)
f_{13}	CDE	13 (0)	1 (0)	0.010 (0.003)	0.031 (0.0098)	31572 (962)
	SDE	13 (0)	1 (0)	0.008 (0.004)	0.021 (0.0065)	33435 (1201)
	CP	13 (0)	1 (0)	0.015 (0.002)	0.037 (0.0065)	31834 (799)
	AEGA	10.66 (1.21)	0.8323 (0.0343)	0.088 (0.033)	0.096 (0.0098)	32845 (1182)
	CSA	8.94 (2.34)	0.7998 (0.0564)	0.110 (0.088)	0.113 (0.0104)	25050 (0)
	AiNet	10.32 (1.52)	0.8297 (0.0206)	0.098 (0.075)	0.087 (0.0086)	25050 (0)
	MCS	13 (0)	0.9997 (0.0134)	0.011 (0.007)	0.023 (0.0016)	25740 (101)
f_{14}	CDE	3.04 (1.34)	0.6675 (0.0754)	0.809 (0.101)	176.54 (21.23)	32273 (1004)
	SDE	3.55 (0.56)	0.7017 (0.0487)	0.675 (0.079)	115.43 (34.21)	30372 (965)
	CP	2.87 (1.23)	0.6123 (0.0861)	1.081 (0.201)	202.65 (42.81)	31534 (1298)
	AEGA	3 (0)	0.6686 (0.0542)	0.894 (0.076)	150.32 (57.31)	29985 (1745)
	CSA	3 (0)	0.6691 (0.0231)	0.897 (0.045)	161.57 (27.92)	25050 (0)
	AiNet	3.50 (0.25)	0.7001 (0.0765)	0.668 (0.097)	121.43 (43.12)	25050 (0)
	MCS	7.13 (0.81)	0.9859 (0.0094)	0.023 (0.010)	17.62 (4.13)	25786 (92)

TABLE 3: Test functions used in the experimental study.

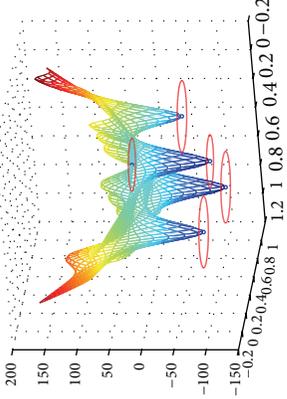
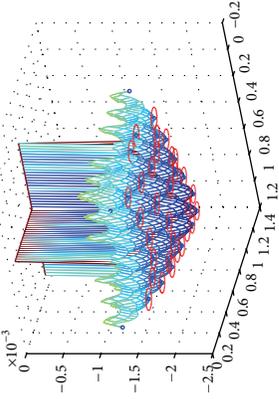
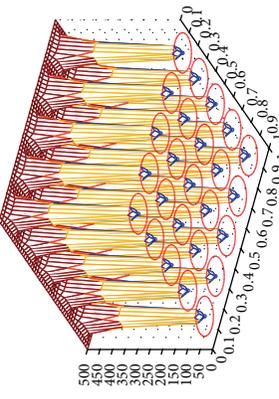
$f(x)$ ($x = \{x_1, x_2\}$)	S	NO	Graph
<p>Bird</p> $f_1(\mathbf{x}) = \sin(x_1) \cdot e^{(1-\cos(x_2))^2} + \cos(x_2) \cdot e^{(1-\sin(x_1))^2} + (x_1 - x_2)^2$	$[-2\pi, 2\pi]$	3	
<p>Cross in tray</p> $f_2(\mathbf{x}) = -0.0001 \cdot \left(\sin(x_1) \cdot \sin(x_2) \cdot e^{\left 100 - \sqrt{ x_1^2 - x_2^2 } \right } + 1 \right)^{0.1}$	$[-10, 10]$	12	
<p>DeJongs5</p> $f_3(\mathbf{x}) = \left\{ 0.002 + \sum_{i=-2}^2 \sum_{j=-2}^2 \left[5(i+1) + j + 3 + (x_1 - 16j)^6 + (x_2 - 16i)^6 \right]^{-1} \right\}$	$[-40, 40]$	25	

TABLE 3: Continued.

$f(x)$ ($x = \{x_1, x_2\}$)	S	NO	Graph
<p>Eggholder</p> $f_4(\mathbf{x}) = -(x_2 + 47) \sin \left(\sqrt{\left x_2 + \frac{x_1}{2} + 47 \right } \right) - x_1 \sin \left(\sqrt{\left x_1 + \frac{x_2}{2} + 47 \right } \right)$	[-512, 512]	7	
<p>Vincent</p> $f_5(\mathbf{x}) = - \sum_{i=1}^n \sin(10 \cdot \log(x_i))$	[0.25, 10]	36	
<p>Roots</p> $f_6(\mathbf{x}) = -(1 + (x_1 + x_2)^6 - 1)^{-1}$	[-2, 2]	6	

TABLE 3: Continued.

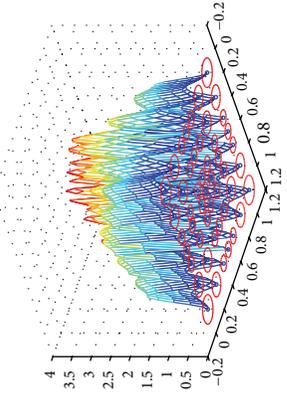
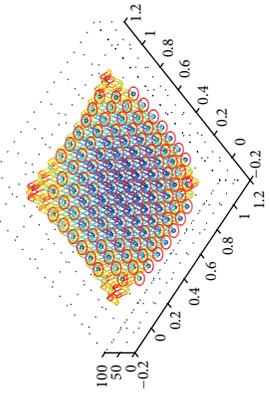
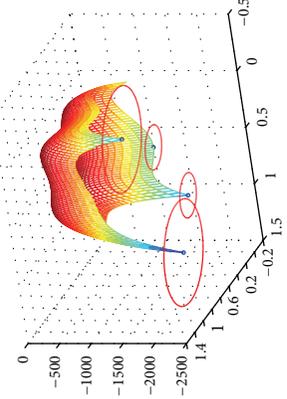
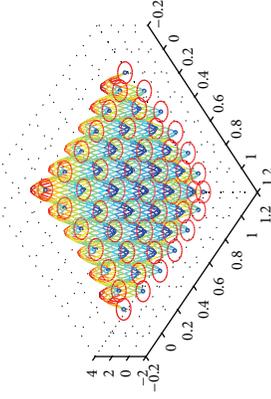
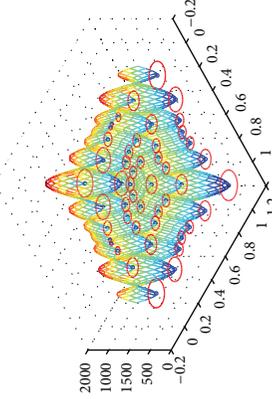
$f(\mathbf{x})$ ($x = \{x_1, x_2\}$)	S	NO	Graph
<p>Hilly</p> $f_7(\mathbf{x}) = 10 \left[e^{-(x_1 /50)} \left(1 - \cos \left(\frac{6}{100^{3/4}} \pi x_1 ^{3/4} \right) \right) + e^{-(x_2 /250)} \left(1 - \cos \left(\frac{6}{100^{3/4}} \pi x_2 ^{3/4} \right) \right) + 2 \left(e^{-((b-x_1)^2 + (b-x_2)^2)/50} \right) \right]$ <p>with $b = ((5/6) \cdot 100^{3/4})^{4/3}$</p>	[-100, 100]	48	
<p>Rastrigin</p> $f_8(\mathbf{x}) = \sum_{i=1}^n x_i^2 - 10 \cos(2\pi x_i)$	[-5.12, 5.12]	25	
<p>Himmelmblau</p> $f_9(\mathbf{x}) = -(x_1^2 + x_2 - 11)^2 - (x_1 + x_2^2 - 7)^2$	[-6, 6]	5	

TABLE 3: Continued.

$f(\mathbf{x})$ ($x = \{x_1, x_2\}$)	S	NO	Graph
<p>Foxholes</p> $f_{10}(\mathbf{x}) = - \sum_{i=1}^{30} \left(\sum_{j=1}^n [(x_j - a_{ij})^2 + c_j] \right)^{-1}$	[0, 10]	8	
<p>Guichi f4</p> $f_{11}(\mathbf{x}) = - (x_1 \sin(4\pi x_1) - x_2 \sin(4\pi x_2 + \pi))$	[-2, 2]	12	
<p>Holder Table</p> $f_{12}(\mathbf{x}) = - \sin(x_1) \cos(x_2) e^{ 1 - \sqrt{ x_1^2 + x_2^2} /\pi} $	[-10, 10]	12	

TABLE 3: Continued.

$f(x)$ ($x = \{x_1, x_2\}$)	S	NO	Graph
<p>Rastrigin_49m</p> $f_{13}(x) = \sum_{i=1}^n x_i^2 - 18 \cos(2\pi x_i)$	[-1, 1]	13	
<p>Schwefel</p> $f_{14}(x) = 418.9829 \cdot n + \sum_{i=1}^n -x_i \sin(\sqrt{ x_i })$	[-500, 500]	8	

of function evaluations (NFE). The results are averaged by considering 50 different executions.

The goal of multimodal optimizers is to find as many as possible global optima and good local optima. The main objective in these experiments is to determine whether MCS is able to find not only optima with prominent fitness value, but also optima with low fitness values. Table 2 provides a summary of the performance comparison among the different algorithms.

Considering the EPN measure, it is observed that MCS finds more optimal solutions for the multimodal problems f_8-f_{14} than the other methods. Analyzing function f_8 , only MCS can detect all optima whereas CP, AEGA, CSA, and AiNet exhibit the worst EPN performance.

Functions f_9-f_{12} represent a set of special cases which contain a few prominent optima (with good fitness value). However, such functions present also several optima with bad fitness values. In these functions, MCS is able to detect the highest number of optimum points. On the contrary, the rest of algorithms can find only prominent optima.

For function f_{13} , four algorithms (CDE, SDE, CP, and MCS) can recognize all optima for each execution. In case of function f_{14} , numerous optima are featured with different fitness values. However, MCS still can detect most of the optima.

In terms of number of the maximum peak ratios (MPR), MCS has obtained the best score for all the multimodal problems. On the other hand, the rest of the algorithms present different accuracy levels.

A close inspection of Table 2 also reveals that the proposed MCS approach is able to achieve the smallest PA and DA values in comparison to all other methods.

Similar conclusions to those in Section 4.2 can be established regarding the number of function evaluations (NFE). All results demonstrate that MCS achieves the overall best balance in comparison to other algorithms, in terms of both the detection accuracy and the number of function evaluations.

5. Conclusions

The cuckoo search (CS) algorithm has been recently presented as a new heuristic algorithm with good results in real-valued optimization problems. In CS, individuals emulate eggs (contained in nests) which interact in a biological system by using evolutionary operations based on the breeding behavior of some cuckoo species. One of the most powerful features of CS is the use of Lévy flights to generate new candidate solutions. Under this approach, candidate solutions are modified by employing many small changes and occasionally large jumps. As a result, CS can substantially improve the relationship between exploration and exploitation, still enhancing its search capabilities. Despite such characteristics, the CS method still fails to provide multiple solutions in a single execution. In order to overcome such inconvenience, this paper proposes a new multimodal optimization algorithm called the multimodal cuckoo search (MCS). Under MCS, the original CS is enhanced with multimodal capacities by means of (1) incorporation of a memory mechanism to efficiently

register potential local optima according to their fitness value and the distance to other potential solutions, (2) modification of the original CS individual selection strategy to accelerate the detection process of new local minima, and (3) inclusion of a depuration procedure to cyclically eliminate duplicated memory elements.

MCS has been experimentally evaluated over a test suite of the fourteen benchmark multimodal functions. The performance of MCS has been compared to some other existing algorithms including the crowding differential evolution (CDE) [22], the fitness sharing differential evolution (SDE) [21, 22], the clearing procedure (CP) [26], the elitist-population strategy (AEGA) [28], the clonal selection algorithm (CSA) [30], and the artificial immune network (AiNet) [31]. All experiments have demonstrated that MCS generally outperforms all other multimodal metaheuristic algorithms in terms of both the detection accuracy and the number of function evaluations. The remarkable performance of MCS is explained by two different features: (i) operators (such as Lévy flight) allow a better exploration of the search space, increasing the capacity to find multiple optima, and (ii) the diversity of solutions contained in the memory M in the context of multimodal optimization is maintained and further improved through an efficient mechanism.

Appendix

For list of benchmark functions, see Table 3.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The proposed algorithm is part of the optimization system used by a biped robot supported under the Grant CONACYT CB 181053.

References

- [1] P. M. Pardalos, H. E. Romeijn, and H. Tuy, "Recent developments and trends in global optimization," *Journal of Computational and Applied Mathematics*, vol. 124, no. 1-2, pp. 209-228, 2000.
- [2] C. A. Floudas, I. G. Akrotirianakis, S. Caratzoulas, C. A. Meyer, and J. Kallrath, "Global optimization in the 21st century: advances and challenges," *Computers & Chemical Engineering*, vol. 29, no. 6, pp. 1185-1202, 2005.
- [3] Y. Ji, K.-C. Zhang, and S.-J. Qu, "A deterministic global optimization algorithm," *Applied Mathematics and Computation*, vol. 185, no. 1, pp. 382-387, 2007.
- [4] A. Georgieva and I. Jordanov, "Global optimization based on novel heuristics, low-discrepancy sequences and genetic algorithms," *European Journal of Operational Research*, vol. 196, no. 2, pp. 413-422, 2009.

- [5] D. Lera and Y. D. Sergeyev, "Lipschitz and Hölder global optimization using space-filling curves," *Applied Numerical Mathematics*, vol. 60, no. 1-2, pp. 115–129, 2010.
- [6] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence through Simulated Evolution*, John Wiley & Sons, Chichester, UK, 1966.
- [7] K. de Jong, *Analysis of the behavior of a class of genetic adaptive systems [Ph.D. thesis]*, University of Michigan, Ann Arbor, Mich, USA, 1975.
- [8] J. R. Koza, "Genetic programming: a paradigm for genetically breeding populations of computer programs to solve problems," Tech. Rep. STAN-CS-90-1314, Stanford University, Stanford, Calif, USA, 1990.
- [9] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.
- [10] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Boston, Mass, USA, 1989.
- [11] L. N. de Castro and F. J. von Zuben, "Artificial immune systems—part I: basic theory and applications," Tech. Rep. TR-DCA 01/99, 1999.
- [12] R. Storn and K. Price, "Differential evolution—a simple and efficient adaptive scheme for global optimisation over continuous spaces," Tech. Rep. TR-95-012, ICSI, Berkeley, Calif, USA, 1995.
- [13] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
- [14] S. I. Birbil and S.-C. Fang, "An electromagnetism-like mechanism for global optimization," *Journal of Global Optimization*, vol. 25, no. 3, pp. 263–282, 2003.
- [15] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "Filter modeling using gravitational search algorithm," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 1, pp. 117–122, 2011.
- [16] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, December 1995.
- [17] M. Dorigo, V. Maniezzo, and A. Colnari, "Positive feedback as a search strategy," Tech. Rep. 91-016, Politecnico di Milano, Milano, Italy, 1991.
- [18] S. Dasa, S. Maity, B.-Y. Qu, and P. N. Suganthan, "Real-parameter evolutionary multimodal optimization—a survey of the state-of-the-art," *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 71–78, 2011.
- [19] K.-C. Wong, C.-H. Wu, R. K. P. Mok, C. Peng, and Z. Zhang, "Evolutionary multimodal optimization using the principle of locality," *Information Sciences*, vol. 194, pp. 138–170, 2012.
- [20] D. Beasley, D. R. Bull, and R. R. Martin, "A sequential niche technique for multimodal function optimization," *Evolutionary Computation*, vol. 1, no. 2, pp. 101–125, 1993.
- [21] B. L. Miller and M. J. Shaw, "Genetic algorithms with dynamic niche sharing for multimodal function optimization," in *Proceedings of the 3rd IEEE International Conference on Evolutionary Computation*, pp. 786–791, May 1996.
- [22] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," in *Proceedings of the Congress on Evolutionary Computation (CEC '04)*, pp. 1382–1389, June 2004.
- [23] S. W. Mahfoud, *Niching methods for genetic algorithms [Ph.D. thesis]*, Illinois Genetic Algorithm Laboratory, University of Illinois, Urbana, Ill, USA, 1995.
- [24] O. J. Mengshoel and D. E. Goldberg, "Probability crowding: deterministic crowding with probabilistic replacement," in *Proceedings of the International Genetic and Evolutionary Computation Conference*, W. Banzhaf, Ed., pp. 409–416, Orlando, Fla, USA, 1999.
- [25] X. Yin and N. Germary, "A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization," in *Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms*, pp. 450–457, 1993.
- [26] A. Petrowski, "A clearing procedure as a niching method for genetic algorithms," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '96)*, pp. 798–803, Nagoya, Japan, May 1996.
- [27] J.-P. Li, M. E. Balazs, G. T. Parks, and P. J. Clarkson, "A species conserving genetic algorithm for multimodal function optimization," *Evolutionary Computation*, vol. 10, no. 3, pp. 207–234, 2002.
- [28] Y. Liang and K.-S. Leung, "Genetic Algorithm with adaptive elitist-population strategies for multimodal function optimization," *Applied Soft Computing Journal*, vol. 11, no. 2, pp. 2017–2034, 2011.
- [29] G. Chen, C. P. Low, and Z. Yang, "Preserving and exploiting genetic diversity in evolutionary programming algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 661–673, 2009.
- [30] L. N. de Castro and F. J. von Zuben, "Learning and optimization using the clonal selection principle," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 3, pp. 239–251, 2002.
- [31] L. N. Castro and J. Timmis, "An artificial immune network for multimodal function optimization," in *Proceedings of the Congress on Evolutionary Computation*, pp. 699–704, Honolulu, Hawaii, USA, 2002.
- [32] Q. Xu, L. Wang, and J. Si, "Predication based immune network for multimodal function optimization," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 4, pp. 495–504, 2010.
- [33] K. C. Tan, S. C. Chiam, A. A. Mamun, and C. K. Goh, "Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization," *European Journal of Operational Research*, vol. 197, no. 2, pp. 701–713, 2009.
- [34] S. Roy, S. M. Islam, S. Das, S. Ghosh, and A. V. Vasilakos, "A simulated weed colony system with subregional differential evolution for multimodal optimization," *Engineering Optimization*, vol. 45, no. 4, pp. 459–481, 2013.
- [35] F. Yahyaie and S. Filizadeh, "A surrogate-model based multimodal optimization algorithm," *Engineering Optimization*, vol. 43, no. 7, pp. 779–799, 2011.
- [36] S. Yazdani, H. Nezamabadi-pour, and S. Kamyab, "A gravitational search algorithm for multimodal optimization," *Swarm and Evolutionary Computation*, vol. 14, pp. 1–14, 2014.
- [37] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature & Biologically Inspired Computing (NABIC '09)*, pp. 210–214, Coimbatore, india, December 2009.
- [38] S. Walton, O. Hassan, K. Morgan, and M. R. Brown, "A review of the development and applications of the Cuckoo search algorithm," in *Swarm Intelligence and Bio-Inspired Computation Theory and Applications*, X.-S. Yang, Z. Cui, R. Xiao, A. H. Gandomi, and M. Karamanoglu, Eds., pp. 257–271, Elsevier, San Diego, Calif, USA, 2013.

- [39] X.-S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343, 2010.
- [40] S. Walton, O. Hassan, K. Morgan, and M. R. Brown, "Modified cuckoo search: a new gradient free optimisation algorithm," *Chaos, Solitons and Fractals*, vol. 44, no. 9, pp. 710–718, 2011.
- [41] A. Kumar and S. Chakarverty, "Design optimization for reliable embedded system using Cuckoo search," in *Proceedings of the 3rd International Conference on Electronics Computer Technology (ICECT '11)*, pp. 264–268, April 2011.
- [42] A. Kaveh and T. Bakhshpoori, "Optimum design of steel frames using Cuckoo search algorithm with Lévy flights," *The Structural Design of Tall and Special Buildings*, vol. 22, no. 13, pp. 1023–1036, 2013.
- [43] L. H. Tein and R. Ramli, "Recent advancements of nurse scheduling models and a potential path," in *Proceedings of 6th IMT-GT Conference on Mathematics, Statistics and Its Applications (ICMSA '10)*, pp. 395–409, 2010.
- [44] V. Bhargava, S. E. K. Fateen, and A. Bonilla-Petriciolet, "Cuckoo search: a new nature-inspired optimization method for phase equilibrium calculations," *Fluid Phase Equilibria*, vol. 337, pp. 191–200, 2013.
- [45] Z. Moravej and A. Akhlaghi, "A novel approach based on cuckoo search for DG allocation in distribution network," *International Journal of Electrical Power and Energy Systems*, vol. 44, no. 1, pp. 672–679, 2013.
- [46] I. Pavlyukevich, "Lévy flights, non-local search and simulated annealing," *Journal of Computational Physics*, vol. 226, no. 2, pp. 1830–1844, 2007.
- [47] R. N. Mantegna, "Fast, accurate algorithm for numerical simulation of Lévy stable stochastic processes," *Physical Review E*, vol. 49, no. 4, pp. 4677–4683, 1994.

Research Article

MAC Protocol for Ad Hoc Networks Using a Genetic Algorithm

Omar Elizarraras, Marco Panduro, Aldo L. Méndez, and Alberto Reyna

Universidad Autónoma de Tamaulipas, UAMRR, Carr. Reynosa-San Fernando S/N, Colonia Arcoiris, 88779 Reynosa, TAMPS, Mexico

Correspondence should be addressed to Aldo L. Méndez; almendez@uat.edu.mx

Received 23 April 2014; Accepted 4 July 2014; Published 21 July 2014

Academic Editor: T. O. Ting

Copyright © 2014 Omar Elizarraras et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The problem of obtaining the transmission rate in an ad hoc network consists in adjusting the power of each node to ensure the signal to interference ratio (SIR) and the energy required to transmit from one node to another is obtained at the same time. Therefore, an optimal transmission rate for each node in a medium access control (MAC) protocol based on CSMA-CDMA (carrier sense multiple access-code division multiple access) for ad hoc networks can be obtained using evolutionary optimization. This work proposes a genetic algorithm for the transmission rate election considering a perfect power control, and our proposition achieves improvement of 10% compared with the scheme that handles the handshaking phase to adjust the transmission rate. Furthermore, this paper proposes a genetic algorithm that solves the problem of power combining, interference, data rate, and energy ensuring the signal to interference ratio in an ad hoc network. The result of the proposed genetic algorithm has a better performance (15%) compared to the CSMA-CDMA protocol without optimizing. Therefore, we show by simulation the effectiveness of the proposed protocol in terms of the throughput.

1. Introduction

In a wireless ad hoc network, the nodes can communicate with each other without the support of infrastructure. Since the wireless channel is shared by all the nodes in the network, a medium access control (MAC) plays an important role in coordinating access among the nodes so that information gets through from one node to another [1]. Usually each node is able to communicate with each other's node when all nodes are spread around a geographic range. However, nodes could spread over larger geographic range than the communication signal can reach. In this case, the nodes could have communication over multiple hops. However, there is only one medium that is shared by all the nodes that are in the same radio communication range and the radio frequency bandwidth is limited. Furthermore, packet collisions are unavoidable due to the fact that traffic arrivals are random and there is nonzero propagation time between transmitters and receivers. Therefore, MAC schemes are used to coordinate the access to the channel in the network [2].

The tendency of the MAC protocols for wireless ad hoc networks is using adaptive systems to adjust the transmission parameters (multirate) and the objective is to maximize

the throughput in the use of the channel. Rate adaptation is indispensable to optimally exploit the scarce wireless resources under instable channel conditions. Rate adaptation consists of assessing the wireless channel conditions and selecting the most appropriate data rate. Moreover, in the MAC protocols, the low throughput in the region of low traffic is because there is no more information for sending (this is not due to errors of the multiuser interference). So the system performance is limited by the access technique used in wireless ad hoc networks [3]. To address this problem, we use a rate adaptation to optimize throughput requirements. Then, the problem can be formulated as an optimization problem; that is, minimize the resources consumption considering the power, interference, data rate, and energy ensuring the signal to interference ratio in an ad hoc network. The multirate in ad hoc networks has been addressed in [4–9]; these protocols were proposed to maximize the throughput by adapting the rate based on the channel, but they do not address the energy issue.

The MAC protocol plays a critical role in a wireless ad hoc network considering bandwidth efficiency, resolving collisions, resources allocation, power transmission, interference, energy, data rate, and distance. In most standardized wireless

ad hoc networks, such as in the widely deployed IEEE 802.11 networks, only one node is allowed to transmit in a given time slot, while in CSMA-CDMA (carrier sense multiple access-code division multiple access) protocol more than one node can transmit; that is, more than one simultaneous transmission can be achieved. Hence, CSMA-CDMA MAC protocol is employed for multihop wireless ad hoc networks [10–13].

Furthermore, power control is applied in a wireless ad hoc network to control transmission range (distance between the source and destination); on the other hand, it is useful to keep low interference levels to the neighboring nodes. Because CDMA systems are interference limited, power control also serves as a tool for interference management in CDMA systems to guarantee success of multiple concurrent transmissions [1]. Another parameter closely related with power transmission is the energy consumption. In critical environments such as military or rescue operations, where ad hoc networks will be typically used, conserving of battery power will be vital in order to make the network operational for long durations. Recharging or replacing batteries will not often be possible. The majority of work in the literature focuses on the protocol design and performance evaluation in terms of traditional metrics such as throughput, delay, power, energy, and distance. In this case, the literature analyzes these terms by separate. Some protocols related to these routing metrics have been proposed in [14–18].

In order to improve the performance (throughput) of an ad hoc network, we use a MAC protocol based on CDMA with an efficient evolutionary algorithm for transmission rate election. In this evolutionary optimization algorithm, we control the transmission rate and handle the spreading factor or processing gain (P_G) for a MAC scheme based on CSMA-CDMA ad hoc networks and we take a perfect control power into account.

Modern evolutionary optimization techniques have aroused great interest among the scientific and technical community in a wide variety of fields for the last years because of their ability to solve problems with a nonlinear and nonconvex dependence of design parameters. Several optimization techniques have emerged in the past two decades that mimic biological evolution or the way biological entities communicate in nature. Some of these algorithms have been used successfully in many electromagnetism and network problems with many constraints and nonlinear processes. The most representative algorithm includes genetic algorithms (GA) [19], among others.

Therefore, in this work, we present a genetic algorithm that solves the problem of power combining, interference, data rate, and energy ensuring the signal to interference ratio in an ad hoc network and we use Dijkstra's algorithm to make the search and discovery of the route.

The remainder of this paper is organized as follows. In Section 2, we introduce the model for the MAC protocol based on CSMA-CDMA considering a perfect power control. The operation of proposed protocol is presented in Section 3. Section 4 presents the genetic algorithm for the transmission rate election. Section 5 presents the simulations performed

and the results obtained. Finally, we present the conclusions of our work in Section 6.

2. MAC Protocol Considering a Perfect Power Control

In a system based on CSMA-CDMA, the CSMA protocol is used to access the channel and reserve it for a communication between two nodes. Instead, the function of CDMA is to assign a code to each node so they can transmit simultaneously.

The CSMA protocol operation is as follows: when a node wants to transmit its data, it monitors the channel state. If the channel is idle for a time period that is equal to a distributed interframes space, the node transmits the request-to-send (RTS) packet after a random period called backoff. If the channel is not idle, the node defers its transmission. If the destination node detects an RTS packet, it responds with a clear-to-send (CTS) packet. The transmission will only start after the CTS packet is correctly received. Moreover, the packets RTS and CTS carry the information of the duration of the packet to be transmitted. This information can be read by each node, which is then able to update their network allocation vector (NAV) containing the information of the period of time in which the channel will remain busy.

In the CDMA protocol, the simultaneous transmission of packet by different nodes is allowed. Therefore, it is necessary for a code assignment mechanism to assign distinct codes to different nodes. Several code assignment mechanisms have been previously proposed [20, 21] and we have used the receiver-transmitter-based mechanism.

Considering the previously mentioned, in this paper, the MAC protocol for ad hoc networks is based on CSMA-CDMA with different transmission rates. Furthermore, we consider a perfect control power and an error-free channel. The operation of the protocol MAC is as follows.

Assuming that node A has generated a packet and it needs to transmit to node B, then

- (1) node A contends for the allocation of a code. This code is selected randomly from a code table;
- (2) node A sends a RTS message informing the neighboring nodes of its intention to transmit, code selected, transmission duration, and destination node (node B). Node A changes its state to receive the response message (CTS) from node B. If node A does not receive the CTS message, it assumes that there was a collision and applies the backoff mechanism. This step (Step 2) will run until it has received the reply from node B or until the lifetime of the packet has expired;
- (3) node B receives the RTS message; it sends the reply message (CTS) and informs the neighboring nodes that will be occupied (attending node A);
- (4) node A determines the number of active neighboring nodes. With this number, it can obtain an optimal transmission rate according to the genetic algorithm. Then, it sends data packet to node B;

- (5) once B receives the packets, it sends an ACK (acknowledgement) message to inform node A that the packets were successfully received and the code is released;
- (6) if A does not receive the ACK message, it returns to Step 2 until the transmission turns successful or until the packet expires.

Furthermore, we need to propose an algorithm to obtain the transmission rate for each node. Therefore, if a node needs to transmit a packet, it calculates the number of neighboring nodes trying to transmit in a time slot. With this number, it is possible to obtain the best transmission rate combination for each node [3]. This combination of transmission rates should maximize the throughput and it is based on the nodes that transmit at different rates as well as the bit error probability (P_b) or bit error rate (BER) as it is often called. The use of the Gaussian approximation to determine P_b is based on the argument that the decision statistic and the multiple access interference may be modeled as a Gaussian random variable. Therefore, considering that the system is limited by interference and noise is neglected, the bit error probability used for a fixed rate of transmission is given by [22]

$$P_b(n) = Q\left(\sqrt{3\frac{P_G}{n-1}}\right), \quad (1)$$

where n is the number of simultaneous nodes, P_G is the processing gain and it is defined as the ratio of the transmission bandwidth to the information bandwidth, and Q is the error function given by

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^{\infty} e^{-u^2/2} du, \quad (2)$$

where (2) is in terms of u , and this indicates the variable of the function to be integrated and defined in [23].

The correctly detecting probability of a packet containing L bits is

$$P_d(n) = [1 - P_b(n)]^L. \quad (3)$$

In the CSMA-CDMA protocol, when the channel has a low traffic, a high processing gain is not necessary in the system because there is a low multiuser interference. Then, the processing gain (protection against multiuser interference) depends on the traffic conditions in the channel. Therefore, the system performance can be improved by using an adaptive transmission rate according to load conditions in the channel. Therefore, through genetic algorithms, it is possible to obtain the rate at which each node must transmit in an ad hoc network compared to [3] where it used exhaustive search.

Up to now, we have analyzed the CSMA-CDMA protocol with perfect power control to obtain the transmission rate for each node. In next section, we present the problem of power combining, interference, data rate, and energy ensuring the signal to interference ratio in a multihop ad hoc wireless network for the MAC protocol based on CSMA-CDMA as medium access technique with the ability to route packets.

3. Proposed Protocol

3.1. Operation. Assuming that node A has generated a packet and it needs to transmit to node B, which is out of node A transmission range, then

- (1) node A contends for the allocation of a code. This code is selected randomly from a code table. If a primary collision is present, then the nodes use a backoff mechanism to retransmit [24];
- (2) node A performs the route discovery based on the minimum distance, the same way as the link state routing protocol (Dijkstra). This means that the node source finds the route to its destination hop by hop through the minimum path until it reaches its destination node;
- (3) once the route is obtained, the algorithm optimizes the power, interference, transmission rate, and energy of the nodes that forward packets to destination node B;
- (4) node A sends the RTS message to the next node of the route. This process is repeated consecutively until it reaches node B. Node A changes its state to receive the CTS message;
- (5) node B receives the RTS message and it responds with a CTS message;
- (6) node A receives the CTS message and starts sending data packets. If node A receives no response from node B, it assumes that there was collision and retransmits the RTS message;
- (7) once the RTS-CTS dialog is completed, B starts to receive data packets and responds with an ACK message to inform node A that the packets have been received successfully;
- (8) if A receives no acknowledgement (ACK) from B, it waits for a random time to retransmit and repeat the process from Step 4;
- (9) once B receives the packets, it sends an ACK message to inform node A that the packets were successfully received and if there are no more packets to send, the code is released.

3.2. System Model. It is considered a wireless ad hoc network where there are M nodes participating in routing and N active nodes in the system. It should be mentioned that M nodes that participate in routing are determined in the phase of path discovery, and each node knows the neighbor nodes. This path discovery is assumed to be executed at the beginning of the simulation. We considered a link state routing algorithm, and all nodes know the detailed information of the network, topology, nodes neighboring, distance between them, and so forth. Moreover, the gain on the communication link between node i and node j is given by g_{ij} . All the g_{ij} are positive. The transmission powers of node i and node j are denoted by p_i and p_j , respectively. The current signal to interference ratio (SIR) [25] between node i

and node j , γ_{ij} , is the ratio between the power received from transmitter i at receiver j and the received interference caused by neighboring nodes plus noise power at receiver j , and it is determined by

$$\gamma_{ij} = \frac{W}{R_{ij}} \frac{g_{ij} P_{ij}}{\sum_{x=1, x \neq j, x \neq i}^N g_{jx} P_{jx} + \sigma_j}, \quad i = 1, 2, 3, \dots, M, \quad (4)$$

where R_{ij} is the data rate used to transmit from node i to node j , W is the total spread spectrum bandwidth occupied by CDMA, σ_j is the noise power at receiver j , and p_{ij} and p_{jx} are, respectively, the received powers of the transmissions between nodes i and j and nodes j and x .

On the other hand, if noise power at receiver j is neglected because its value is very small, then we have

$$\gamma_{ij} = \frac{W}{R_{ij}} \frac{g_{ij} P_{ij}}{\sum_{x=1, x \neq j, x \neq i}^N g_{jx} P_{jx}}, \quad i = 1, 2, 3, \dots, M, \quad (5)$$

and the channel gain is determined by

$$g_{ij} = d_{ij}^{-\alpha}, \quad (6)$$

where d_{ij} is the distance between nodes i and j and α is the attenuation coefficient.

We assume that each node should achieve the target SIR, γ_{ij}^t , from node i along the route to node j as follows:

$$\gamma_{ij} \geq \gamma_{ij}^t, \quad i = 1, 2, \dots, M, \quad i \neq j. \quad (7)$$

A power vector \mathbf{P} and a transmission rate vector \mathbf{R} are used such that they satisfy the criterion expressed in (7), where the power vector must comply as follows:

$$p_{\min} \leq p_i \leq p_{\max}. \quad (8)$$

The \mathbf{R} vector must satisfy the next criterion as follows:

$$R_{\min} \leq R_{ij} \leq R_{\max}. \quad (9)$$

Considering that each node can choose its rate transmission and power within K possibilities, one has

$$\begin{aligned} p_i &\in \{p_i^1, p_i^2, \dots, p_i^K\}, \\ R_i &\in \{r_i^1, r_i^2, \dots, r_i^K\}. \end{aligned} \quad (10)$$

4. Genetic Algorithm

The main purpose of this study is to solve the problem of power combining, interference, data rate, and energy ensuring the signal to interference ratio in an ad hoc network. For this purpose, we propose to use a population-based stochastic procedure denominated genetic algorithm (GA) [19]. We chose this algorithm for its easiness of implementation. The procedure for the used GA technique (Figure 1) is described as follows.

The function Generate Initial Population randomly and uniformly generates a set of individuals.

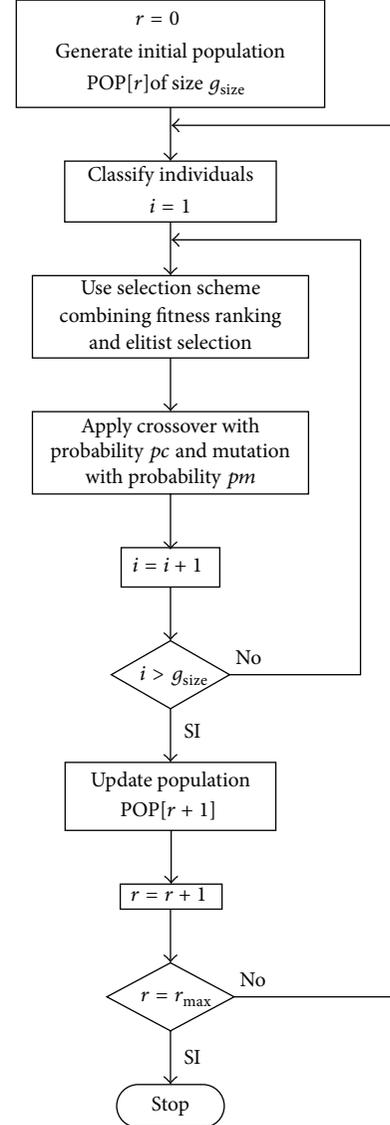


FIGURE 1: Flow chart for the evolutionary optimization algorithm.

The main idea in Classify Individuals is to rank the individuals according to their fitness value.

A selection scheme combining fitness ranking and elitist selection [19] is implemented instead of a common weighted roulette wheel selection.

The function Update Population assigns ranks to individuals in the population generated by the union of parents and children. This is in order to hold the best individuals in each generation. Golberg [19] explains the procedures involved in each step of this algorithm in detail. The individual representations as well as the crossover and mutation operators are explained in the following subsections.

4.1. Individual Representation and Decoding

4.1.1. Perfect Power Control. For a number of desired transmission rate combinations, t , with a population P of size n ,

each individual (combination of transmission rates) is expressed as

$$P = \begin{bmatrix} C_1 \\ C_2 \\ \vdots \\ C_n \end{bmatrix}, \quad (11)$$

where $C_1 \cdots C_n$ represents each individual as potential solution. Moreover, C represents the combinations of transmission rates as individuals. Consider

$$C = [k_{1R} \cdots k_{tR}], \quad (12)$$

where k nodes transmit at t times the basic rate (R).

On the other hand, the bandwidth, W , is given as

$$W = P_G \cdot R, \quad (13)$$

where P_G is processing gain, R is the transmission rate, and the bandwidth (W) must be constant.

So the basic transmission rate is expressed in terms of processing gain as follows:

$$R = \frac{W}{P_G}. \quad (14)$$

Therefore, the individuals in (11) are encoded in binary form so that the individual of population (12) is transformed into a single binary string.

4.1.2. Power Combining. In this case, we consider the transmission rates and powers as individuals and potential solutions to the previously presented problem. Furthermore, they must comply with target SIR (7).

The vector \mathbf{R} and vector \mathbf{P} are transformed into binary numbers or strings; that is, they are encoded in a binary form. The chromosomes A_i and B_i are used to represent the vectors \mathbf{R} and \mathbf{P} , respectively, where i represents an individual. Moreover, $A_i = a_1, a_2, \dots, a_l$, where a_1, a_2, \dots, a_l is the binary representation of length l for the rate transmission R_i , and $B_i = b_1, b_2, \dots, b_l$, where b_1, b_2, \dots, b_l is the binary representation for power p_i [25].

The individuals for power population and rate transmission population are given by

$$A = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_T \end{bmatrix}, \quad (15)$$

$$B = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_T \end{bmatrix},$$

where T is the population size.

4.2. Genetic Operators. The used genetic operators are standard; the well-known two-point crossover [19] along with a single mutation where a locus is randomly selected and the allele is replaced by a random number is uniformly distributed in the feasible region.

4.3. Objective Function

4.3.1. Perfect Power Control. The objective function (of) for maximizing the performance of the transmission rates combination is given by

$$\text{of} = \max S. \quad (16)$$

Evaluating the performance S (throughput) for a transmission rates combination t , considering a perfect power control, is determined as the average of the correctly detecting probability of each node that transmits with different rates [22] as follows:

$$S_t = \sum_t^{t_{\max}} n_{tR} P_{d,tR} L, \quad (17)$$

where S_t is the performance for the given combination of transmission rates, n_{tR} is number of nodes transmitting t times the transmission basic rate R , $P_{d,tR}$ is the correctly detecting probability of a packet transmitted at t times the transmission basic rate R , L is packet length, and t_{\max} is the maximum value that can increase the basic rate.

Therefore, the procedure to find the optimal transmission rate is as follows.

- (1) Generate a population of random combinations of transmission rates.
- (2) Calculate the fitness (S_t) of the population generated.
- (3) Apply genetic operators to the population by keeping the fittest individuals (best solutions).
- (4) Iterate Steps 2 and 3 the number of generations desired.

The result of this algorithm for each number of nodes is stored in a table. Therefore, depending on the number of the nodes that are transmitting in an instant of time, the nodes automatically will select the appropriate transmission rate.

4.3.2. Power Combining. The objective function (of) used to minimize the resources consumption in each hop is as follows:

$$\text{of} = \min (\text{abs}(\gamma_{ij} - \gamma_{ij}^t) + E_{ij}). \quad (18)$$

To evaluate each individual (power and rate) and the energy consumed through a path, we apply (19) (fitness). This equation represents an evaluation function for each possible solution obtained from the transmission rate and power calculated by each node. It is important to mention that, through this formulation, the genetic algorithm will tend to choose those solutions that require lower power, lower

energy, and higher transmission rate, depending on the quality of the link (SIR). Consider

$$\text{Fit}_i = \text{abs}(\gamma_{ij} - \gamma_{ij}^t) + E_{ij}, \quad (19)$$

where E_{ij} is the energy needed to transmit from node i to node j . To calculate the energy, we consider that the power transmission between nodes i and j is p_{ij} , R_{ij} is the transmission rate used to transmit from node i to node j , $P_d(\gamma_{ij})$ is the probability of correctly detecting a packet, with γ_{ij} being equal to the SIR of link (i, j) , and L denotes the packet length. Moreover, the number of transmissions necessary to receive a packet correctly is a random variable, D . If all transmissions are statistically independent, D is a geometric random variable. So the expected value of D is $E[D] = 1/P_d(\gamma_{ij})$. The total transmission time required for correct reception is the random variable DL/R_{ij} . With the transmitted power p_{ij} , the energy expended is the random variable $p_{ij}DL/R_{ij}$ with expected value $E[D]Lp_{ij}/R_{ij} = Lp_{ij}/[R_{ij}P_d(\gamma_{ij})]$. Therefore, energy E_{ij} is obtained by [26]

$$E_{ij} = \frac{LP_{ij}}{R_{ij}P_d(\gamma_{ij})}. \quad (20)$$

Energy (20) is in function of the transmission rate, and $P_d(\gamma_{ij})$ is given by

$$P_c(\gamma_{ij}) = (1 - 2\text{BER}_{ij})^L, \quad (21)$$

where BER_{ij} is the bit error rate for the link (i, j) . The BER is expressed as follows:

$$\text{BER}_{ij} = 0.5e^{-\gamma_{ij}/2}. \quad (22)$$

Therefore, the main steps of the procedure of the proposed optimization algorithm are as follows.

- (1) Generate populations of transmission rate (**R**) and power (**P**).
- (2) Calculate the fitness of the population.
- (3) Manipulate the individuals by genetic operators keeping the elite individuals.
- (4) Repeat Steps 2 and 3 until the desired number of generations has been reached.

5. Simulations and Results

The first simulation is considering a perfect power control and the process of simulation was achieved by a program in MATLAB, on a personal computer, which considers a single hop ad hoc wireless network. We consider 160 nodes that are placed randomly in a square area of side length 150 meters. In this simulation, the nodes generate packets using a Pareto distribution. Additionally, we consider that the nodes are fixed. The preview concepts are applied to the simulation. Furthermore, a chip rate of 4.096 Mcps, four transmission rates (16, 32, 64, and 128 kbps), four processing gains (256,

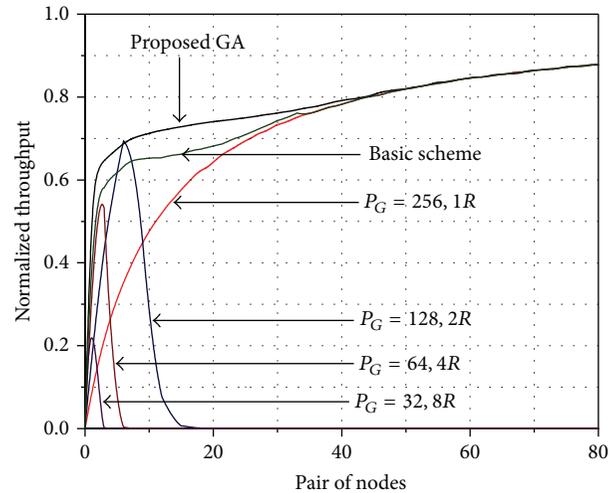


FIGURE 2: Throughput of the CSMA-CDMA protocol considering perfect power control.

128, 64, and 32), and the basic transmission rate of 16 kbps are employed.

For the genetic algorithm implemented, a population size of 50 individuals is used with 50 generations. Furthermore, crossover and mutation rates of 0.9 and 0.05 were used, respectively. These parameters have been selected following the recommendations of De Jong [27] and Grefenstette [28]. Furthermore, elitism is implemented in the genetic algorithm (GA) to ensure that the fitness of the population will not diminish from one generation to another. Elitism guarantees that the best individuals from the current generation are going to survive to the next generation. Therefore, with elitism is united the population of parents and children and half of them selected. The result of the simulation is shown in Figure 2.

Figure 2 shows the throughput behavior with proposed genetic algorithm (GA) for transmission rate election, and we observe a better performance with respect to fixed rates and the throughput is normalized with respect to maximum capacity of the system. Furthermore, Figure 2 illustrates that our proposed GA achieves improvement of 10% as compared with the basic scheme [13]. This is because the transmission rate with the basic scheme is obtained in the handshaking phase, where the node attempts to transmit with the maximum value of rate and if there is a fail in the handshaking phase, then the node decreases its transmission rate. When the node has success in the handshaking phase, it increases its transmission rate. Therefore, the basic scheme is a trial-and-error process. Unlike the basic scheme, in the proposed GA, the optimum transmission rate of each node is obtained and ensures that it will not affect the multiuser interference. Furthermore, during low traffic channel, nodes increase their transmission rates. As traffic increases, processing gain also increases, obtaining as a result a dynamic CDMA system bandwidth control.

The following simulation considers the combination of power, interference, data rate, and energy. The maximum

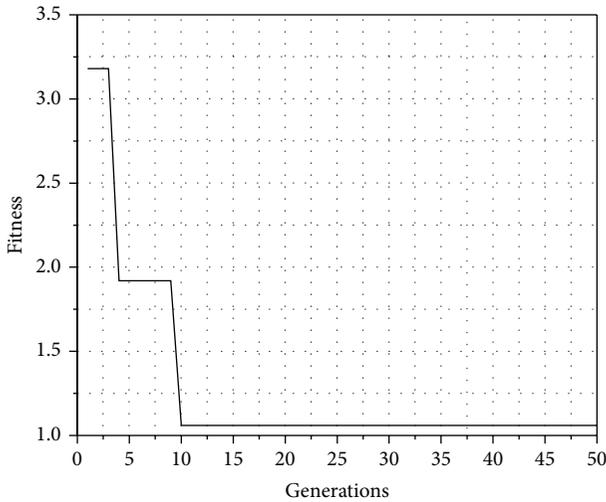


FIGURE 3: Fitness optimization.

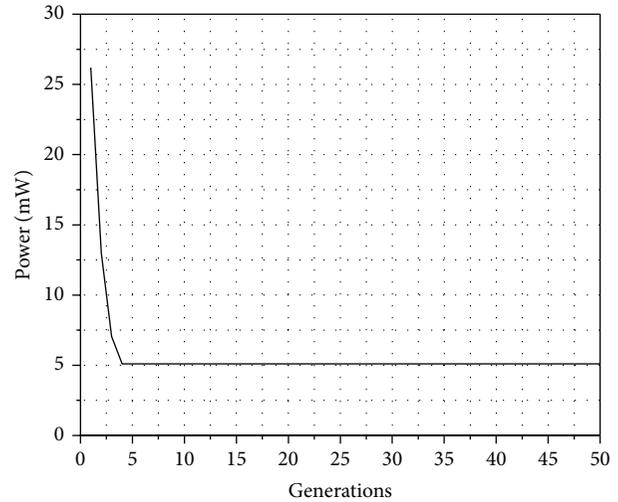


FIGURE 4: Behavior of the average power.

power is 100 mW and the minimum power is 5 mW. The thermal noise is neglected. The target SIR (signal to interference ratio) was set to 3.918 dB. The attenuation coefficient α was set to 2. The channel gain only is in function of the distance. Therefore, fading is not considered. The transmission rates are 1200, 1800, 2400, 4800, 7200, 9600, and 14400 bps. The network was considered as a multihop network for this simulation, in which the routing technique used for routing data packets is based on minimum path. In this work, Dijkstra's algorithm for routing is implemented.

The genetic algorithm assigns powers and transmission rates considering the standard IS-95. The population size is of 50 individuals; crossover probability and mutation are of 0.95 and 0.05, respectively [27, 28]. The termination criterion is 50 generations. The individual length for the power was set to 7 bits and in the case of transmission rate is 3 bits.

An obtained result of the genetic algorithm is shown in Figure 3. This result confirms the effectiveness of the proposed algorithm.

Figure 3 shows that when fitness decreases SIR approaches to the desired value; therefore, the node only uses the resource needed to reach the destination node without interfering with their neighbors. This is shown in Figure 4.

Figure 4 illustrates the behavior of power with respect to generation number using genetic algorithms. It is observed how the power is minimized and thus the node does not cause interference to neighboring nodes or expend more energy than necessary. Furthermore, another consideration of the genetic algorithm is to increase the transmission rate, and this behavior can be seen in Figure 5. The behavior shown in Figure 5 is the result of the genetic algorithm through 50 generations. In this graphic, we can see that the data rate is increased, while the power is decreased (Figure 4), which means that the SIR is fitted to a target with optimum parameters to achieve the best performance and throughput by the MAC protocol.

Furthermore, Figure 6 shows the throughput of proposed protocol where codes are used to grant the channel access

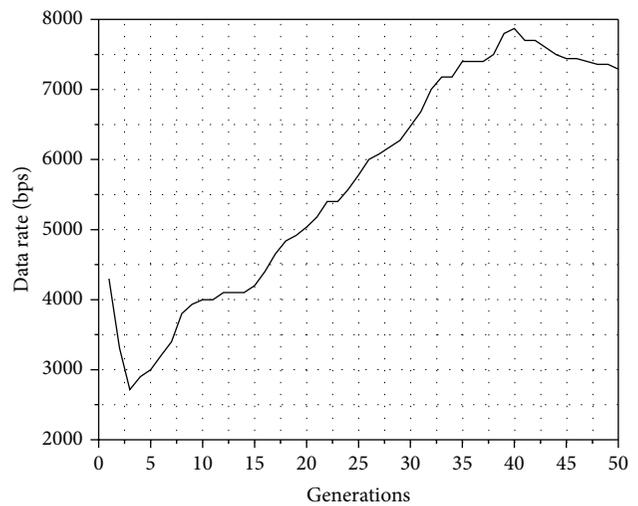


FIGURE 5: Behavior of transmission rate.

and genetic algorithms are applied to guarantee the signal to interference ratio (SIR) for a successful transmission. Figure 6 shows the throughput behavior with proposed genetic algorithm (GA) for transmission multirate. The throughput is normalized with respect to maximum capacity of the system. Furthermore, during low traffic channel, nodes increase their transmission rates. As traffic increases, processing gain also increases, obtaining as a result a dynamic CDMA system bandwidth control. The proposed algorithm is compared with the CSMA-CDMA protocol without optimization. In CSMA-CDMA without optimization, as in [29], the nodes perform the RTS/CTS process and they adjust the power to reach their neighboring nodes but the SIR and the minimum energy are not guaranteed. Therefore, the CSMA-CDMA without optimization has a lower performance compared to our proposed GA (15%).

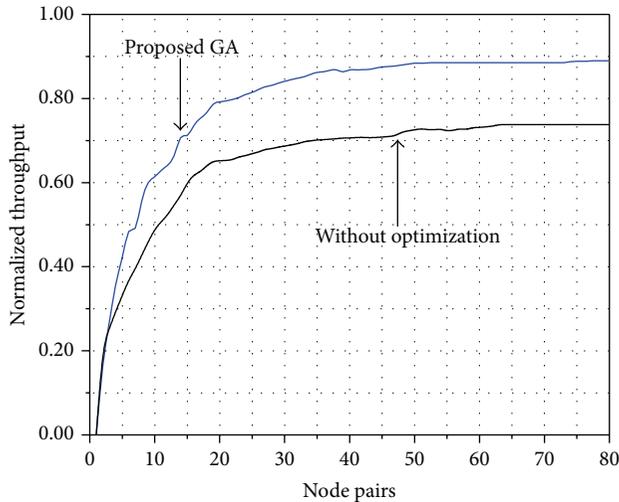


FIGURE 6: Throughput of CDMA ad hoc network using genetic algorithm (GA).

6. Conclusions

This paper has presented a MAC protocol based on CSMA-CDMA considering a perfect power control. A genetic algorithm is used to obtain the optimal transmission rate for each node in an ad hoc network. In this proposed protocol, we control the transmission rate according to the offered traffic. The simulation results demonstrate that proposed MAC protocol performs better than traditional CSMA-CDMA with fixed rates and the proposed protocol outperforms conventional protocols for single hop. Moreover, a shortest path routing protocol is applied in this paper when the nodes communicate with each other using multihop links. The method of genetic algorithms is used to optimize the network resource. On the other hand, the paper presents an efficient MAC protocol based on CSMA-CDMA to achieve high performance.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

- [1] S. Kumar, V. S. Raghavan, and J. Deng, "Medium access control protocols for ad hoc wireless networks: a survey," *Ad Hoc Networks*, vol. 4, no. 3, pp. 326–358, 2006.
- [2] M. Kaynia, N. Jindal, and G. E. Øien, "Improving the performance of wireless ad hoc networks through MAC layer design," *IEEE Transactions on Wireless Communications*, vol. 10, no. 1, pp. 240–252, 2011.
- [3] A. Méndez, M. Panduro, and R. Aquino, "Evaluación de Rendimiento del Protocolo MAC basado en CDMA Para Redes 802.11," *Interciencia*, vol. 34, no. 2, pp. 46–51, 2009.
- [4] F. Mzee Awuor, K. Djouani, and G. Noel, "Joint power and rate adaptation in Ad-Hoc networks based on coupled interference," *Procedia Computer Science*, vol. 5, pp. 272–279, 2011.
- [5] S. Elnoubi, N. Sadek, and O. Hussein, "Basic access adaptation for Ad-Hoc networks," in *Proceedings of the Military Communications Conference (MILCOM '08)*, pp. 1–7, San Diego, Calif, USA, 2008.
- [6] J. He, W. Guan, L. Bai, and K. Chen, "Theoretic analysis of IEEE 802.11 rate adaptation algorithm sampleRate," *IEEE Communications Letters*, vol. 15, no. 5, pp. 524–526, 2011.
- [7] P. Li, Q. Shen, Y. Fang, and H. Zhang, "Power controlled network protocols for Multi-Rate ad hoc networks," *IEEE Transactions on Wireless Communications*, vol. 8, no. 4, pp. 2142–2149, 2009.
- [8] Z. Ji, Y. Yang, J. Zhou, M. Takai, and R. Bagrodia, "Exploiting medium access diversity in rate adaptive wireless LANs," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking (MobiCom '04)*, pp. 345–359, ACM, Philadelphia, Pa, USA, September 2004.
- [9] J. Wang, H. Zhai, Y. Fang, and M. C. Yuang, "Opportunistic media access control and rate adaptation for wireless ad hoc networks," in *Proceedings of the IEEE International Conference on Communications*, pp. 154–158, Paris, France, June 2004.
- [10] D. N. Djonin, A. K. Karmokar, D. V. Djonin, and V. K. Bhargava, "Adaptive rate transmission in ad hoc wireless networks," *IEICE Transactions on Communications*, vol. 87, no. 5, pp. 1385–1392, 2004.
- [11] A. Muqattash, M. Krunz, and W. E. Ryan, "Solving the near-far problem in CDMA-based ad hoc networks," *Ad Hoc Networks*, vol. 1, no. 4, pp. 435–453, 2003.
- [12] T. Su, W. Jeng, and W. Hsieh, "Enhancing the performance of ad hoc wireless networks using code division," *Journal of Internet Technology*, vol. 7, no. 3, pp. 285–292, 2006.
- [13] R. Fantacci, A. Fern, and D. Tarchi, "Medium access control protocol for CDMA ad hoc networks," *Electronics Letters*, vol. 40, no. 18, pp. 1131–1133, 2004.
- [14] L. Qian, D. R. Vaman, and N. Song, "QoS-aware maximally disjoint routing in power-controlled multihop CDMA wireless ad hoc networks," *Eurasip Journal on Wireless Communications and Networking*, vol. 2007, Article ID 53717, 13 pages, 2007.
- [15] N. Nie and C. Comaniciu, "Energy efficient AODV routing in CDMA ad hoc networks using beamforming," *EURASIP Journal on Wireless Communications and Networking*, vol. 2006, Article ID 76709, 2006.
- [16] X. Han, X. Zhang, and J. Lv, "Considering lifetime optimal rate allocation and power control in wireless ad hoc networks," in *Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM '08)*, Dalian, China, October 2008.
- [17] S. Kandukuri and S. Boyd, "Simultaneous rate and power control in multirate multimedia CDMA systems," in *Proceedings of the IEEE 6th International Symposium on Spread Spectrum Techniques and Applications (ISSSTA '00)*, pp. 570–574, Parsippany, NJ, USA, September 2000.
- [18] M. Chiang, C. W. Tan, D. P. Palomar, D. O'Neill, and D. Julian, "Power control by geometric programming," *IEEE Transactions on Wireless Communications*, vol. 6, no. 7, pp. 2640–2650, 2007.
- [19] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.
- [20] L. Hu, "Distributed code assignments for CDMA packet radio networks," *IEEE Transactions on Networking*, vol. 1, no. 6, pp. 668–677, 1993.
- [21] J. J. Garcia-Luna-Aceves and J. Raju, "Distributed assignment of codes for multihop packet-radio networks," in *Proceedings of the MILCOM*, vol. 1, pp. 450–454, Monterrey, Calif, USA, 1997.

- [22] O. Salient and R. Agusti, "Adaptive S-ALOHA CDMA as an alternative way of integrating services in mobile environments," *IEEE Transactions on Vehicular Technology*, vol. 49, no. 3, pp. 936–947, 2000.
- [23] T. S. Rappaport, *Wireless Communications: Principles and Practice*, Prentice Hall, Upper Saddle River, NJ, USA, 2002.
- [24] Z. Jiang and M. Zhou, "Spread spectrum MAC protocol with dynamic rate and collision avoidance for mobile Ad Hoc network," *IEEE Transactions on Vehicular Technology*, vol. 56, no. 5, pp. 3149–3158, 2007.
- [25] Y. D. Lee and K. H. Kim, "Multi-rate power control of CDMA cellular system using genetic algorithm," in *ICMIT: Information Systems and Signal Processing*, vol. 6041 of *Proceedings of SPIE*, September 2005.
- [26] D. J. Goodman and N. B. Mandayam, "Power control for wireless data," *IEEE Personal Communications*, vol. 7, no. 2, pp. 48–54, 2000.
- [27] A. K. De Jong, *An analysis of the behavior of a class of genetic adaptive systems [Ph.D. thesis]*, University of Michigan, Ann Arbor, Mich, USA, 1975.
- [28] J. J. Grefenstette, "Optimization of control parameters for genetic algorithms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 16, no. 1, pp. 122–128, 1986.
- [29] T. Al-Meshhadany and W. Ajib, "New CDMA-based MAC protocol for Ad Hoc networks," in *Proceedings of the IEEE 66th Vehicular Technology Conference (VTC '07-Fall)*, pp. 91–95, Baltimore, Md, USA, October 2007.

Research Article

An Ant Colony Optimization Based Feature Selection for Web Page Classification

Esra Saraç and Selma Ayşe Özel

Department of Computer Engineering, Çukurova University, Balcali, Sarıçam, 01330 Adana, Turkey

Correspondence should be addressed to Selma Ayşe Özel; saozel@cu.edu.tr

Received 25 April 2014; Revised 20 June 2014; Accepted 22 June 2014; Published 17 July 2014

Academic Editor: T. O. Ting

Copyright © 2014 E. Saraç and S. A. Özel. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The increased popularity of the web has caused the inclusion of huge amount of information to the web, and as a result of this explosive information growth, automated web page classification systems are needed to improve search engines' performance. Web pages have a large number of features such as HTML/XML tags, URLs, hyperlinks, and text contents that should be considered during an automated classification process. The aim of this study is to reduce the number of features to be used to improve runtime and accuracy of the classification of web pages. In this study, we used an ant colony optimization (ACO) algorithm to select the best features, and then we applied the well-known C4.5, naive Bayes, and k nearest neighbor classifiers to assign class labels to web pages. We used the WebKB and Conference datasets in our experiments, and we showed that using the ACO for feature selection improves both accuracy and runtime performance of classification. We also showed that the proposed ACO based algorithm can select better features with respect to the well-known information gain and chi square feature selection methods.

1. Introduction

The aim of text classification is to categorize documents into a certain number of predefined classes by using document features. Text classification plays a crucial role in many retrieval and management tasks such as information retrieval, information extraction, document filtering, and building hierarchical directories [1]. When text classification focuses on web pages it is named web classification or web page classification. However, web pages are different from text, and they contain a lot of additional information, such as URLs, links, HTML tags, which are not supported by text documents. Because of this property of web pages, web classification is different from traditional text classification [1].

On the web, classification is used for topic-specific web link selection, analysis of the topical structure of the web, development of web directories, and focused crawling [1]. Previously, people manually constructed some web directories such as Yahoo! [2] and the Open Directory Project [3] and manually assigned class labels to web documents. However,

manual classification is time consuming and needs a lot of human effort, which makes it unscalable with respect to the high growing speed of the web. Therefore, there has been great need for automated web page classification systems [4].

A major problem of the web page classification is the high dimensionality of the feature space. We need to select "good" subsets of features from the original feature space to reduce the dimensionality and to improve the efficiency and run time performance of the classification process [5]. Several approaches such as document frequency, information gain, mutual information, chi square analysis, and term strength have been applied to select proper features for text categorization. According to Yang and Pedersen [6] chi square analysis and information gain are more effective for optimizing classification results, and document frequency is less effective but it is scalable and affordable. In Aghdam et al. [7] nature inspired search and optimization algorithms have been applied for feature selection of text classification problem. According to [7], ant colony optimization and genetic algorithms can choose better features than the information gain and chi square analysis, and performance of ant colony optimization

is better than the genetic algorithm. For this reason, in this study we applied an ant colony optimization, which was originally developed to solve optimization problems, to select the best features from the web pages for accurate and time efficient classification.

Ant colony optimization (ACO) was inspired from the behavior of real ant colonies, and it is used to solve discrete optimization problems. The first ACO system was introduced by Marco Dorigo in his Ph.D. thesis [8] and was called the ant system (AS). The AS is the result of a research on computational intelligence approaches to combinatorial optimization [8]. The AS was initially applied to the travelling salesman problem [9] and then to other hard problems. The original AS is motivated by the natural phenomenon that ants deposit pheromone on the ground in order to mark some favorable path that should be followed by other members of the colony. The aim of the colony is to find the shortest path between a food source and the nest. The behavior of an ant colony is a good example of a self-organized system such that it is based on positive feedback (i.e., the deposit of pheromone) and negative feedback (i.e., the evaporation of pheromone). Theoretically, if the quantity of pheromone remains the same over time on all edges, no route is chosen. However, because of feedback, a slight variation on an edge allows the edge to be chosen. The algorithm moves from an unstable state in which no edge is stronger than another, to a stable state where the route is composed of the strongest edges. Ant colony optimization has a wide application domain; for example, Liu et al. [10] have used ACO for continuous domains.

In this study, a new ACO based feature selection algorithm, which selects best features from web pages, has been proposed. The contributions of this study are summarized as follows.

- (i) In the earlier studies, only the bag of terms approach is used for feature extraction and ACO is applied to select features among the small number of terms. However, in our study each term in the URLs and in the HTML tags is taken as a different feature. Therefore, a feature is represented as <URL><term>, or <tag><term> pair which we call “tagged terms” representation that yields thousands of features to be extracted from web pages. According to our research in the literature, there exists no ACO based study which works on such large scale feature space.
- (ii) In earlier studies, each ant selects features one by one; however, in our study each ant selects a set of features at a time since our feature space is too high, and selecting features one by one increases running time of ACO sharply.
- (iii) In earlier studies, only the effect of using features from bag of terms approach has been studied. In this study, the effect of using features only from URLs, from <title> tags, and tagged terms, as well as bag of terms approach, is investigated. We also study the effects of HTML tags on classification performance.

This paper is organized as follows. In Section 2, we give related work on ACO based feature selection and web page

classification. Section 3 describes our ACO-based feature selection algorithm. Section 4 includes the datasets used, the experiments performed, their results, and discussions. Section 5 concludes the study and gives some future work.

2. Related Work

In this section, we give a brief review of web page classification, feature selection, and ant colony optimization for feature selection.

2.1. Web Page Classification. Classification is the process of assigning predefined class labels to some unseen or test data. For this purpose, a set of labeled data is used to train a classifier which is then used for labeling unseen data. This classification process is also defined as a supervised learning [11]. The process is not different in web page classification such that there is one or more predefined class labels and a classification model assigns class labels to web pages which are in fact hypertext and have many features such as textual tokens, markup tags, URLs, and host names in URLs that are meaningful for classifiers. As web pages have additional properties, their classification has several differences from traditional text classification [1].

Web page classification has some subfields like subject classification and functional classification [1]. In subject classification, classifier is concerned with the content of a web page and tries to determine the “subject” of the web page. For example, categories of online newspapers like finance, sport, and technology are instances of subject classification. Functional classification on the other hand deals with the function or type of the web page. For example, determining whether a web page is a “personal homepage” or a “course page” is an instance of a functional classification. Subject and functional classification are the most popular classification types [1].

Classification can be divided into binary classification and multiclass classification according to the number of classes [1]. In binary classification, there is only one class label. Classifier looks for an instance and assigns it to the specific class or not. Instances of the specific class are called relevant, and the others are named nonrelevant. If there is more than one class, this type of classification is called multiclass classification [1]. The classifier also assigns an instance to one of the multiple classes. In our study, we focus on functional classification of web pages, and we make binary classification since it is the basis of the focused crawlers [12, 13] or topical crawlers [14] of the search engines. The techniques developed in this study can also be used for subject classification and/or multiclass classification of web pages.

2.2. Feature Selection for Web Page/Text Classification. Feature selection is the one of the most important steps in classification systems. Web pages are generally in HTML format. This means that web pages are semistructured data, as they contain HTML tags and hyperlinks in addition to pure text. Because of this property of web pages, feature selection in web page classification is different than traditional text

classification. Feature selection is generally used to reduce dimension of data with tens or hundreds of thousands of features which would be impossible to process further. A major problem of web page classification is the high dimensionality of the feature space. The best feature subset contains the least number of features that most contribute to classification accuracy and efficiency.

To improve the performance of web page classification, several approaches that are imported from feature selection for text classification have been applied. Information gain [11], mutual information [15], document frequency [6], and term strength [16] are the most popular traditional feature selection techniques. Information gain (IG) measures the amount of information in bits about the class prediction, if the only information available is the presence of a feature and the corresponding class distribution. Concretely, it measures the expected reduction in entropy [11].

Mutual information (MI) was first introduced by Shannon [15] in the context of digital communications between discrete random variables and was generalized to continuous random variables. Mutual information is considered as an acceptable measure of relevance between two random variables [17]. Mutual information is a probabilistic method which measures how much information the presence/absence of a term contributes to making the correct classification decision on a class [18].

Document frequency (DF) is the number of documents in which a term occurs in a dataset. It is the simplest criterion for term selection and easily scales to a large dataset with linear computational complexity. It is a simple but effective feature selection method for text categorization [6].

Term strength (TS) has been proposed and evaluated by Wilbur and Sirotkin [16] for vocabulary reduction in text retrieval. Term strength is also used in text categorization [19, 20], such that it predicts term importance based on how commonly a term is likely to appear in "closely-related" documents. TS uses training set of documents to derive document pairs whose measured similarity according to the cosine value of the two document vectors is above a threshold. Then, "term strength" is computed based on the predicted conditional probability that a term occurs in the second half of a pair of related documents given that it occurs in the first half. The above methods namely the IG, the DF, the MI, and the TS have been compared by Yang and Pedersen [6] by using the kNN classifier on the Reuters [21] corpus. According to [6], IG is the most effective method with 98% feature reduction; DF is the simplest method with the lowest cost in computation and it can be credibly used instead of IG if computation of this measure is too expensive.

In addition to traditional feature selection methods, swarm intelligence techniques are also popular to be used for feature selection. In this study, we review the application of ACO for feature selection in general classification problems and web/text classification domains.

2.3. Ant Colony Optimization for General Classification Problems. Jensen and Shen [22] have proposed a hybrid approach

which is a combination of fuzzy rough set and ACO to select features from the Water Treatment Plant database. Fuzzy rough set dependency measure is used for features which are more informative in the currently given selected subset. The number of features is reduced from 38 to 10 with fuzzy rough sets, and then it is further reduced from 10 to 9.5 with ACO. Dependency measure is used as the stopping criteria, and C4.5 is employed for classification. Instead of accuracy and *F*-measure values only the training and testing errors are presented. The error rates in training with no feature reduction, fuzzy rough set reduction, and ant fuzzy rough set reduction are 1.5%, 10.8%, and 6.5%, respectively. They have observed 19.1%, 25.2%, and 22.1% testing errors with no feature reduction, fuzzy rough set reduction, and ant fuzzy rough set reduction, respectively.

Chen et al. [23] have proposed a rough set approach for feature selection based on ACO. Mutual information is used as heuristic information. Feature selection is started with a feature core rather than a random feature which causes complete graph to become in a smaller form. Feature reduction is finished after the core has been found. They have studied the UCI dataset with C4.5 classifier and achieved 98.2% average accuracy for classification.

Huang [24] has used classification accuracy and feature weights of the constructed SVM classifier to design the pheromone update in ACO based feature selection. In this study the UCI and simulated datasets are used and 94.65% average accuracy value is obtained.

Sivagaminathan and Ramakrishnan [25] have proposed a hybrid approach which is a combination of the neural networks and ACO for feature selection. Neural networks are used for error prediction and classification. In the experiments 3, 5, 8, 10, and 12 ants are used for the medical diagnosis dataset which contains 21 to 34 features, and 77.50% average accuracy on breast cancer (Wisconsin prognostic) and 98.22% average accuracy on thyroid disease datasets are achieved.

Vieira et al. [26] have proposed two separate ant colonies combined with fuzzy models for feature selection; one colony is used for minimizing number of features, and the other one is employed to minimize classification error. In this study, the first colony determines the number of features and the second one selects the features. A fuzzy model is used as a classifier and 96.4% average accuracy on breast cancer (Wisconsin prognostic) dataset is observed.

Nemati and Basiri [27] have proposed an ACO based method for a speaker verification system. Gaussian mixture model universal background model (GMM-UBM) is used as a classifier over the TIMIT corpora and equal error rate (EER) of the classification is taken as evaluation criteria. 4.56%, 2.634%, and 3.679% EER values are observed with GMM-UBM, ACO, and GA methods, respectively.

Kabir et al. [28] have combined ACO with neural networks where neural networks are used as classifier. The proposed study contains both wrapper and filter methods such that a probabilistic formula for random selection and determination of subset size is used. Eight well-known cancer datasets are used in the experiments and 98.91% average accuracy is achieved.

Akarsu and Karahoca [29] have used ACO for clustering and feature selection. Ant colony clustering technique is used to segment breast cancer dataset. To remove irrelevant or redundant features from the dataset, sequential backward search technique is applied. Feature selection and clustering algorithms are incorporated as a wrapper. The results showed that the accuracy of the FS-ACO clustering approach is better than the filter approaches. They have compared their study with the clustering algorithm in Weka with respect to the sum of squared error value. According to the experimental evaluation, the sum of squared error is 732 for Weka and 758 for the proposed method.

Al-Ani [30] has proposed an ACO based subset search procedure for speech classification problem. Local importance of a given feature is measured by using the mutual information evaluation function, and only the best k subsets are used to update the pheromone. First iteration starts with m features. The second and following steps start with $m - p$ features that are selected randomly from the previous k -best subsets. p is a numeric value which changes from 1 to $m - 1$. They have studied TIMIT corpora with ANN classifier. The average classification accuracy of ACO, GA, and SFS over all the cases is 84.22%, 83.49%, and 83.19%, respectively.

Wang et al. [31] have developed an ACO based feature selection, which employs SVM classifier, to find the best feature subset for the UCI dataset. The experiments are performed with 5, 10, 15, 20, and 25 ants, and the best feature subset is found with 15 ants. In the experimental evaluation 94.83% average accuracy for Wine dataset and 79.57% average accuracy for Image Segment dataset are observed.

Jain and Singh [32] have modified probability function of ANT with exponential function for feature selection. Two modified ant algorithms are applied to a number of problems and the results are compared with those obtained by applying the original ACO and genetic algorithm, and the same results but with better execution time are observed.

Abd-alsabour and Randall [33] have proposed a wrapper based system for feature selection. In the experiments, UCI dataset and SVM classifier are used. In the proposed feature selection algorithm, the number of selected features is not fixed, so the length of each ant's feature subset may differ. By using the features selected by ants, 1 and 0.8584 classification accuracy values are observed for Wine and Vehicle datasets, respectively.

Rasmy et al. [34] have proposed a hybrid approach for feature selection. They have used ACO for selecting features and ANTMIner for classification. In their ACO, ants start choosing a node (i.e., feature) randomly, after that classifier performance and length of selected feature vectors are adopted as heuristic information for ACO. In this study, UCI dataset having 150 to 560 features is used and the number of features is reduced to 9 and 70 for Diabetes and Analcadata classes, respectively. After feature selection, 94.4% and 85% average classification accuracy is achieved for Sonar and Diabetes datasets, respectively.

2.4. Ant Colony Optimization for Text and Web Classification. AntMiner [35] is the first study that uses the ACO in

the web page classification domain. Holden and Freitas [36] have been inspired by AntMiner [35] and used the ant colony paradigm to find a set of rules that classify the web pages into several categories. They have no prior assumptions about which words in the web pages to be classified can be used as potential discriminators. To reduce data rarity, they use stemming which is a technique in which different grammatical forms of a root word are considered as equivalent such that *help*, *helping*, and *helped* are taken as *help*. Holden and Freitas [36] have also gathered sets of words if they are closely related in the WordNet electronic thesaurus. They have compared their AntMiner with the rule inference algorithms C4.5 and CN2. They have found that AntMiner is comparable in accuracy and forms simpler rules with respect to C4.5 and CN2. The best result of AntMiner is 81.0% classification accuracy and this result is obtained when WordNet generalization is used with Title features.

Aghdam et al. [7] have proposed an ACO based feature selection algorithm for text classification. The features selected by an ant are evaluated according to the classifier performance and the feature subset length. In the experiments, it is assumed that classifier performance is more important than subset length, so that they assign 80% and 20% weights to classifier performance and the subset length, respectively. To measure classification performance, a simple k nearest neighbor classifier is used in the experiments. The performance of the proposed algorithm is compared with the performance of a genetic algorithm, information gain, and chi square analysis on the task of feature selection in Reuters-21578 dataset [21]. Their experimental evaluation showed the superiority of the ACO based feature selection over genetic algorithms, information gain, and chi square analysis methods. They studied only bag of terms feature extraction method and observed 89.08% microaverage F -measure value for Reuters-21578 dataset.

Jensen and Shen [37] have proposed an ACO enhanced fuzzy rough feature selection for web page classification. Terms extracted from web pages are weighted according to the $tf * idf$ weighting scheme. In the proposed ACO based feature selection algorithm, each subset selected by each ant is evaluated by a fuzzy-rough measure of the selected subset. The pheromone values are updated according to this measure and the length of the selected subset. After selecting the best feature set, the web pages are then classified. The experiments are performed on a small dataset which contains 280 web pages collected from Arts & Humanities, Entertainment, Computers & Internet, Health, Business & Economy categories of Yahoo directory and it is observed that ACO based feature selection performs the highest degree of reduction in the feature space with minimal loss of information.

Janaki Meena et al. [38] have used ACO for feature selection and naïve Bayes for classification over the 20 Newsgroup dataset. The ratio between observed frequency and expected frequency of the term is applied as a heuristic measure to the features extracted according to the bag of terms method. Map reduce is used for parallelization. Experiments are performed with 500 ants and 150 iterations.

For talk.politics.mideast dataset, recall and precision values are 0.94 and 0.68, respectively.

Mangai et al. [39] have studied the WebKB dataset. Features are selected with Ward's minimum variance measure, information gain, and $tf * idf$ methods. Ward's minimum variance measure is first used to identify clusters of redundant features in a web page. In each cluster, the best representative features are retained and the others are eliminated. Removing such redundant features helps to minimize the resource utilization during classification. After clustering process, features are selected from these clusters, then kNN, SVM, naïve Bayes, and C4.5 classifiers are used with 10-fold cross validation for classification. Course web pages are used as positive instances and student web pages are used as negative instances. The proposed method of feature selection is compared with other common feature selection methods. Experiments showed that the proposed method performs better than most of the other feature selection methods in terms of reducing the number of features and the classifier training time. 95.00% and 95.65% accuracy values are achieved with kNN and SVM classifiers, respectively.

Our proposed system is different from the above studies in the following respects.

- (i) We use ACO for selecting a predefined number of features from the large feature space extracted from the web pages instead of selecting features one by one. Our feature selection method chooses features as feature groups. Since using a single feature to determine the class of a web page is not enough, also, including features one by one to the selected feature list of each ant increases run time of ACO because all the computations for pheromone update must be repeated for each selection process. However, when we choose a set of features, we perform the necessary computations just once for each selected set.
- (ii) We adopt ACO pheromone update formula for web page classification such that our ants are not blind as in the original ACO's ants; we feed them with $tf * idf$ value of each term. So, they have an idea about terms before selecting features.
- (iii) We have investigated effects of using features from URLs, <title> tags, tagged terms, and bag of terms on the classification performance.
- (iv) We have used larger datasets with larger feature spaces.
- (v) We have also investigated which tags are more important for web page classification.

3. Ant Colony Optimization for Feature Selection

This section includes our ACO based feature selection and classification system. The main structure of the proposed system is shown in Figure 1. Our system consists of feature extraction, ACO based feature selection, and classification components. The training dataset is prepared according to binary class classification problem. From the training dataset,

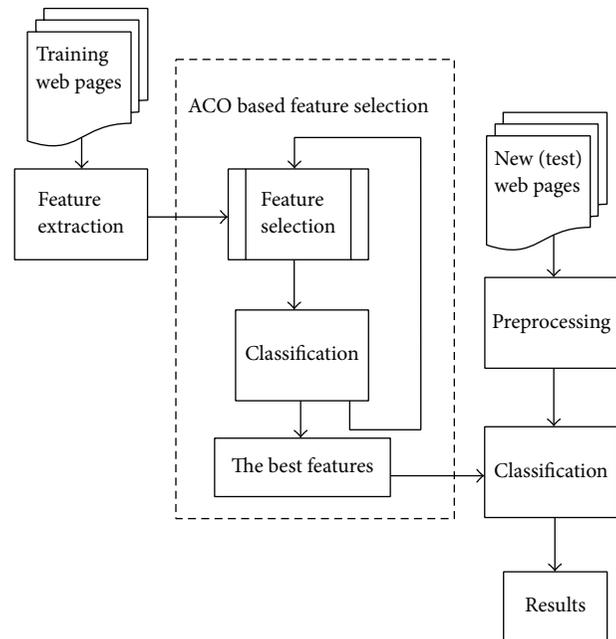


FIGURE 1: Architecture of the proposed system.

features are extracted, after that the best subset of features is selected by our ACO algorithm and then, by using the selected best features, new web pages are classified with Weka [40] data mining software implemented in Java. The components of our proposed system are explained in detail in the following subsections.

3.1. Feature Extraction. In the feature extraction phase all terms from the <title>, <h1>, <h2>, <h3>, <a>, , <i>, , , <p>, and tags which denote title, header at level 1, header at level 2, header at level 3, anchor, bold, italic, emphasize, strong, paragraph, and list item, and additionally URL addresses of web pages are used. According to the experimental results of the earlier studies [41–43], these tags have useful information and should be used during feature extraction. To extract features, all the terms from each of the above mentioned tags and URL addresses of the relevant pages in the training set are taken. After term extraction, stopwords are removed and the remaining terms are stemmed by using the Porter's stemmer [44]. The stemmed terms and their corresponding tags form our whole feature set. The details of the feature extraction step are presented in Section 4.3.

3.2. Feature Selection. After the feature extraction step, the (sub)optimum subset of features is selected with our ACO based feature selection algorithm. The flowchart of our proposed ACO algorithm is presented in Figure 2.

In our proposed method, each feature represents a node, and all nodes are independent of each other. Nodes (i.e. features) are selected according to their selection probability

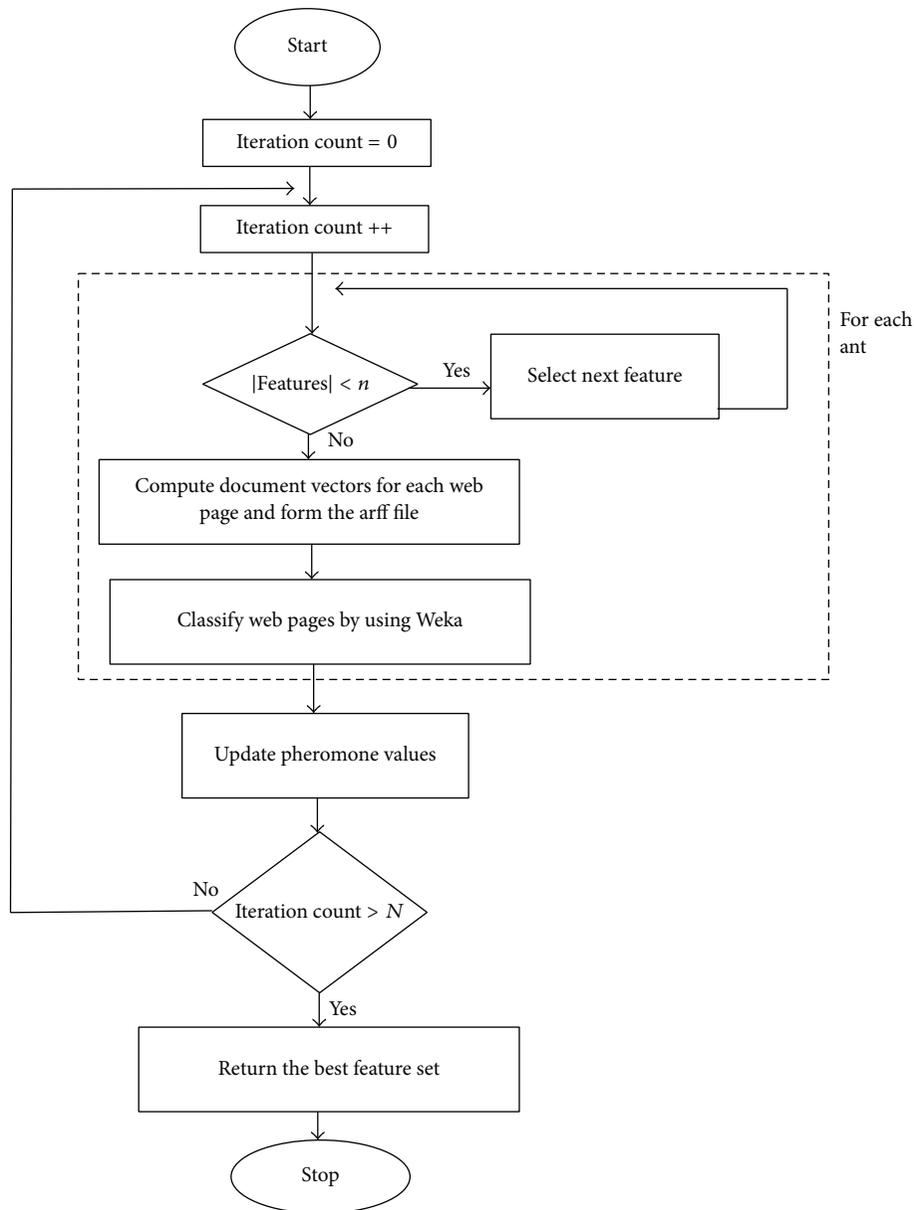


FIGURE 2: Flow chart of the proposed ACO algorithm.

$P_k(i)$ which is given in (1). Initially, all nodes have the same selection probability:

$$P_k(i) = \frac{[\tau(i)]^\alpha [\eta(i)]^\beta}{\sum_{l \in N_i^k} [\tau(l)]^\alpha [\eta(l)]^\beta}, \quad (1)$$

where $\eta(i)$ is equal to the document frequency of feature i which is the number of documents in the training set that contains feature i and represents heuristic information available to the ants. N_i^k is the “feasible” neighborhood of ant k , that is, all features as yet unvisited by ant k . $\tau(i)$ is the pheromone trail value of feature i . Parameters α and β determine the relative influence of heuristic and

pheromone information respectively. The pheromone values, and parameters α and β are initialized according to [45] which have shown that 1 is the best value for α and β , and 10 is suitable for initial pheromone trail value. After all the ants have built a complete tour, the pheromone trail is updated according to the *global update rule* which is given in (2) as

$$\tau(i) = \rho \tau(i) + \sum_{k=1}^n \Delta \tau_k(i), \quad (2)$$

where ρ denotes pheromone evaporation parameter which decays the pheromone trail, and n is the number of ants. According to [45], ρ value is selected as 0.2. The specific

amount of pheromone, $\Delta\tau_k(i)$, that each ant k deposits on the trail is given by (3) as

$$\Delta\tau_k(i) = \begin{cases} 2B_kL_k & \text{if node } i \text{ is used by elitist ant } k \\ B_kL_k & \text{if node } i \text{ is used by any ant } k \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

In (3), L_k is the F -measure value of ant k 's feature subset, and B_k is the unit pheromone value. This means that the higher the F -measure of the ant's selected subset, the more the pheromone deposited on the features used in the subset, and these features are more likely to be selected in the next iteration.

The proposed ACO based feature selection algorithm works as follows. Initially all features have the same selection probability $P_k(i)$ value. According to $P_k(i)$ value of each feature i , an ant k chooses n features. A roulette wheel selection algorithm [46] is used to select each of the n features.

When an ant k chooses n features, the web pages in the training dataset are classified with respect to the selected n features by using the C4.5 classifier (i.e., the J48 classifier) of Weka data mining tool. To classify the web pages in the training dataset, $tf * idf$ values where tf is the term frequency and idf is the inverse document frequency of the selected features for each web page are taken as the feature values.

The J48 classifier is an open source Java implementation of the C4.5 algorithm in the Weka data mining tool. C4.5 is a well-known decision tree algorithm, and it is an extension of Quinlan's earlier ID3 algorithm [47]. It builds decision trees from a set of training data using an extension of information gain known as gain ratio. After making classification by using the selected features, the classification performance is measured with respect to F -measure [48] value which is given in (4) as

$$F\text{-measure} = \frac{2 * \text{recall} * \text{precision}}{\text{recall} + \text{precision}}. \quad (4)$$

F -measure is a combination of precision and recall such that *recall* is the proportion of web pages which are classified as class C_i , among all web pages which truly have class C_i , and *precision* is the proportion of the web pages which truly have class C_i among all those which are classified as class C_i . In earlier studies, researchers measured performance with respect to F -measure value. To comply with the standards on this issue, F -measure value is chosen as the performance metric in this study. The above feature selection and F -measure value computations are repeated for all ants. After these computations, an ant is chosen as an elitist ant which has the highest F -measure value. Then, the pheromone values are updated based on (2) and (3). This process is repeated a predetermined number of times (i.e., N). Finally, the feature subset having the best F -measure value is chosen as the best feature set which can then be used for classifying new (unseen) web pages.

TABLE 1: Train/test distribution of WebKB dataset for binary class classification.

Class	Train	Test
	relevant/nonrelevant	relevant/nonrelevant
Course	846/2822	86/942
Project	840/2822	26/942
Student	1485/2822	43/942
Faculty	1084/2822	42/942

4. Experimental Evaluation and Results

This section includes the datasets, namely, the WebKB and the Conference that were used in this study, the experiments performed, and their results.

4.1. WebKB Dataset. The WebKB dataset [49] is a set of web pages collected by the World Wide Knowledge Base (Web>KB) project of the CMU [50] text learning group and has been downloaded from The 4 Universities Dataset Homepage [51]. These pages are collected from computer science departments of various universities in 1997 and manually classified into seven different classes, namely, student, faculty, staff, department, course, project, and others. For each class, the collection contains web pages from four universities which are Cornell, Texas, Washington, Wisconsin universities, and other miscellaneous pages collected from other universities.

The 8,282 web pages are manually classified into the seven categories such that the student category has 1641 pages, faculty has 1124, staff has 137, department has 182, course has 930, project has 504, and other contains 3764 pages. The class *other* is a collection of pages that are not deemed as the "main page" and are not representing an instance of the previous six classes. The WebKB dataset includes 867 web pages from Cornell University, 827 pages from Texas University, 1205 pages from Washington University, 1263 pages from Wisconsin University, and finally 4120 miscellaneous pages from other universities.

From the WebKB dataset *Project*, *Faculty*, *Student*, and *Course* classes are used in this study. As *Staff* and *Department* classes have small number of positive examples, they are not considered. Training and test datasets are constructed as described in the WebKB project website [51]. For each class, training set includes relevant pages which belong to randomly chosen three universities and others class of the dataset. The fourth university's pages are used in the test phase. Approximately 75% of the irrelevant pages from others class are included in the training set and the remaining 25% of them are included in the test set. The number of web pages in the train and test part of the WebKB dataset, which is used in this study, is given in Table 1. For example, the Course class includes 846 relevant and 2822 irrelevant pages for the training phase and 86 relevant and 942 irrelevant pages for the test phase.

4.2. Conference Dataset. The Conference dataset consists of the computer science related conference homepages that were obtained from the DBLP website [52]. The conference web

TABLE 2: Train/test distribution of the conference dataset.

	Train		Test	
	relevant	nonrelevant	relevant	nonrelevant
Conference	618	1159	206	386

pages are labeled as positive documents in the dataset. To complete the dataset, the short names of the conferences were queried using the Google search engine [53] manually, and the irrelevant pages in the result set were taken as negative documents. The dataset consists of 824 relevant and 1545 irrelevant pages which are approximately 2 times of the relevant pages.

In the Conference dataset, approximately 75% of both relevant and irrelevant pages are taken as the training set, and the remaining 25% of the relevant and irrelevant pages are included into the test set. The number of web pages in the train and test part of the Conference dataset is given in Table 2.

4.3. Feature Extraction. In the feature extraction phase all <title>, <h1>, <h2>, <h3>, <a>, , <i>, , , , and <p> tags, text content, and URL addresses of web pages are used. All the terms from each of the above mentioned tags and URL addresses of the relevant web pages in the training set are taken. After term extraction, stopword removal and Porter's stemming algorithm [44] are applied. Each stemmed term and its corresponding tag or URL pair forms a feature. For example, a term "program" in a <title> tag, in a tag, or in a URL is taken as a different feature and this feature extraction method is called "tagged terms" method. Terms from similar HTML tags, for example, , , , and <i>, are grouped together to reduce the feature space.

In this study, features are selected from four different feature sets for each class. In the first set, features are extracted only from the URL addresses of web pages. Secondly, only <title> tags are used for feature extraction. In the third feature extraction method, all terms that appear in the web pages regardless of their HTML tag are used as features. In other words, a term which appears in the document regardless of its position is taken as a feature. This feature extraction approach is called "bag-of-terms" method. Finally, all terms that appear in each of the above listed HTML tags are used as features. In other words, a term which appears in different HTML tags is taken as a different feature (i.e., tagged terms).

The number of features varies according to the dataset and the feature extraction method used. Numbers of features for each class of all datasets with respect to the feature extraction method used are shown in Table 3. As an example, 33519 features are extracted when tagged terms method are used for the Course class. When only the <title> tag is considered, the number of features extracted reduces to 305 for this class.

4.4. Experimental Setup. In this study, Perl script language was used for the feature extraction phase, and our ACO based feature selection algorithm was implemented in Java programming language under Eclipse environment. The

proposed method was tested under Microsoft Windows 7 operating system. The hardware used in the experiments has 16 GB of RAM and Intel Xenon E5-2643 3.30 GHz processor. Our feature selection method is tested on the Conference and the WebKB datasets. The proposed ACO based feature selection method is run for 250 iterations, since after 250 iterations we observed that there is no improvement on the classification performance. We have determined the number of ants as 30 experimentally, since we observed that 30 ants give satisfactory results for our study.

In our ACO based feature selection algorithm, each ant chooses a predefined number (i.e., n) of features. However, in classical ACO based systems, each ant chooses features one by one. The reasons for choosing n features are that (i) our classification problem has thousands of features and inclusion or removal of one single feature does not make a considerable effect on the classification performance; (ii) choosing features one by one among the thousands of features increases time complexity of the ACO. In the experiments the number n is taken as 10, 100, and 500.

In our ACO based feature selection system, we need a classifier to evaluate the fitness of each set of features selected by each ant. For this purpose, we employed C4.5 classifier since in our previous study [54] we compared classification performance of naive Bayes, kNN (i.e., IBk) and C4.5 (i.e., J48) algorithms of Weka data mining tool, and we observed that C4.5 is the best performer for our datasets.

After determining the best set of features by using our ACO based feature selection algorithm, we use the selected features to classify the web pages in the test datasets. We repeat this process (i.e., ACO based feature selection and then classification of test datasets) 5 times, and the best, worst, and average F -measure values of these 5 runs are presented in the following sections.

4.5. Experiment 1: Selecting Features Only from URLs of Web Pages. Performance of the proposed method for selecting features from only URL addresses of web pages is considered in this experiment. For this purpose, features are extracted only from the URL addresses of web pages in the training datasets. For all classes, n features are selected with our ACO based feature selection algorithm, and then test (unseen) web pages are classified with respect to these selected n features. To classify test web pages, J48 classifier is used. This process is repeated 5 times and the best, worst, and average classification performance of these 5 runs in F -measure values for the selected n features for all datasets is given in Table 4. As Course dataset has less than 500 features extracted from URLs (see Table 3), we only select 100 and 10 features from URLs for Course class.

In Table 4, the best F -measure values are written in bold face for all classes. According to the experimental results presented in Table 4, we observed that the WebKB dataset includes meaningful URL addresses so that, for all n values, web pages from the Course, Project, Student, and Faculty classes are classified at 100% accuracy by using the ACO selected features from the URLs of the web pages. However, this observation is not true for the Conference dataset, as

TABLE 3: Number of features for all classes with respect to feature extraction methods.

Class	Feature extraction method			
	Tagged terms	Bag of terms	<title> tag	URL
Course	33519	16344	305	479
Project	30856	15307	596	686
Student	49452	22245	1987	1557
Faculty	47376	24641	1502	1208
Conference	34952	18572	890	1115

TABLE 4: The best, worst, and average F -measure values of classification using ACO selected n features from URLs for all classes.

No. of features selected (n)	F -measure	Dataset				
		Course	Student	Project	Faculty	Conference
10	Best	1	1	1	1	0.782
	Avg.	1	0.911	0.993	1	0.664
	Worst	1	0.822	0.986	1	0.545
100	Best	1	1	1	1	0.883
	Avg.	1	1	1	1	0.745
	Worst	1	1	1	1	0.606
500	Best	—	1	1	1	0.842
	Avg.	—	1	1	1	0.794
	Worst	—	1	1	1	0.745

TABLE 5: The best, worst, and average F -measure values of classification using ACO selected n features from <title> tags for all classes.

No. of features selected (n)	F -measure	Dataset				
		Course	Student	Project	Faculty	Conference
10	Best	0.983	0.92	0.891	0.947	0.736
	Avg.	0.966	0.903	0.883	0.901	0.712
	Worst	0.948	0.885	0.875	0.854	0.687
100	Best	0.983	0.92	0.883	0.939	0.739
	Avg.	0.976	0.916	0.879	0.930	0.713
	Worst	0.980	0.911	0.869	0.921	0.687
500	Best	—	0.92	0.883	0.94	0.741
	Avg.	—	0.915	0.879	0.933	0.678
	Worst	—	0.91	0.869	0.922	0.710

the best F -measure value obtained is 0.883. So, we can say that the Conference dataset has less meaningful URL addresses with respect to the WebKB dataset. There is no considerable difference between varying numbers of features for the WebKB dataset. For the Conference dataset, on the other hand, reducing the number of features also reduces the average F -measure values of classification of the test web pages.

4.6. Experiment 2: Selecting Features Only from <title> Tags.

Performance of the proposed method using only <title> tags of Web pages is considered in this experiment. For this purpose, features are extracted only from the <title> tags of web pages in the training datasets. Similar to the first experiment, each ant selects n features from the whole feature set that are obtained from <title> tags, and our ACO algorithm returns the best n features. After that web pages in

the test dataset are classified with respect to these selected n features by using J48 classifier. This process is repeated 5 times and the best, worst, and average classification performance for these 5 runs in F -measure value for the selected n features for all datasets is given in Table 5. As Course dataset has less than 500 features extracted from <title> tags (see Table 3), we only select 100 and 10 features from <title> tags for Course class.

In Table 5, the best F -measure values are written in bold face for all classes. According to the results presented in Table 5, the F -measure values of the classification of the test web pages by using the ACO selected n features are also high for the WebKB dataset, which implies that the WebKB dataset includes meaningful title declarations. However, the title declarations of the Conference dataset are not meaningful as the WebKB dataset. In this experiment, the reduced number of features affects average F -measure values negatively for the Course, Student, and Faculty classes; however, for the

TABLE 6: The best, worst, and average F -measure values of classification using ACO selected n features from bag of terms method for all classes.

No. of features selected (n)	F -measure	Dataset				
		Course	Student	Project	Faculty	Conference
10	Best	0.926	0.896	0.95	0.862	0.933
	Avg.	0.880	0.780	0.944	0.836	0.926
	Worst	0.834	0.665	0.947	0.811	0.92
100	Best	0.895	0.840	0.934	0.928	0.948
	Avg.	0.873	0.821	0.903	0.918	0.940
	Worst	0.851	0.802	0.872	0.908	0.932
500	Best	0.960	0.973	0.981	0.921	0.952
	Avg.	0.955	0.969	0.964	0.918	0.949
	Worst	0.950	0.966	0.947	0.915	0.946

Project and Conference classes the average F -measure values are improved when the number of features is reduced.

4.7. Experiment 3: Selecting Features from Bag of Terms Method. In this experiment, features are selected among terms which are extracted by using bag of terms method. In this method, only the terms regardless of their position or tag are taken as features. Our ACO based feature selection algorithm runs 5 times for each dataset, and we obtain 5 sets of best features. Then by using the selected set of features, we classify the test web pages with J48 classifier. The best, worst, and average classification performance in F -measure value of these 5 runs for classifying the test web pages for all datasets is given in Table 6.

In Table 6, the best F -measure values are written in bold face for all classes. When we compare Tables 4, 5, and 6, we observed that text contents in web pages are more meaningful for classification than URL addresses and titles for the Conference dataset. However, in the WebKB dataset, URL addresses have better features for classification than page contents and titles of web pages. As in the previous experiments, F -measure values change with the number of features and the average classification performance decreases when the number of features is decreased.

4.8. Experiment 4: Selecting Features from Tagged Terms Method. In this experiment, features are selected among terms which are extracted by using tagged terms method such that each term in each HTML tag is taken as a different feature. As URLs have very discriminating terms for the WebKB dataset, terms from URLs are not taken in this experiment. Our ACO based feature selection algorithm is run 5 times and we obtain 5 sets of best features for each case. Then by using the selected set of features, we classify the test web pages by using the J48 classifier. The best, worst, and average classification performance in F -measure value of these 5 runs for ACO selected n features is presented in Table 7.

In Table 7, the best F -measure values are written in bold face. According to Table 7, as n decreases classification performance increases for the Course, Student, Project, and Faculty classes. However, for the Conference dataset, average

classification performance is the best for the medium and high n values (i.e., 100 and 500).

When the results of these four experiments are compared, we observed that the WebKB dataset (Course, Project, Student, and Faculty classes) is classified with 100% accuracy (i.e., F measure values are 1.0), and the classification performance of the Conference dataset is also satisfactory (i.e., F -measure values are up to 0.952). For the WebKB dataset, when the number of selected features is small (i.e., $n = 10$), the best feature extraction methods are URL only and tagged terms method. For the Conference dataset, on the other hand, the best feature extraction method is bag of terms and then tagged terms method. This result has occurred because of the fact that the URLs of the WebKB dataset have class specific terms; for the Conference dataset on the other hand, URLs do not have class specific terms. Since tagged terms feature extraction method is successful in general cases, we use this feature extraction method in the rest of the experiments.

4.9. Experiment 5: Using Other Classifiers for the Test Phase. C4.5 decision tree classifier, which is named J48 in Weka, is employed in our ACO based feature selection system to compute the fitness of the selected features and therefore to update pheromone values and selection probabilities of features. In the test phase of the previous experiments, we also applied J48 classifier to classify the test (unseen) web pages. In this experiment, our aim is to show whether our ACO based selected features give good classification performance with other well-known classifiers, namely, the kNN and the naïve Bayes. For this purpose, we employ kNN, which is named IBk in Weka, and naïve Bayes classifiers in the test phase. In the ACO-based feature selection phase, we employed J48 classifier. It is also possible to employ kNN or naïve Bayes classifiers in the ACO-based feature selection algorithm; however, according to our previous study [54], J48 has better classification performance with respect to kNN and naïve Bayes for our datasets and that is why we prefer J48 in the feature selection process. Using kNN or naïve Bayes in the ACO based feature selection may be considered for the future work. The best, worst, and average classification performance in F -measure values of the 5 runs is presented in Tables 8 and 9.

TABLE 7: The best, worst, and average F -measure values of classification using ACO selected n features from tagged terms method for all classes.

No. of features selected (n)	F -measure	Dataset				
		Course	Student	Project	Faculty	Conference
10	Best	0.963	0.94	0.954	0.965	0.921
	Avg.	0.957	0.938	0.95	0.952	0.904
	Worst	0.954	0.937	0.941	0.946	0.877
100	Best	0.902	0.82	0.923	0.928	0.94
	Avg.	0.881	0.808	0.887	0.914	0.932
	Worst	0.865	0.79	0.845	0.869	0.911
500	Best	0.9	0.35	0.91	0.356	0.936
	Avg.	0.87	0.345	0.896	0.355	0.932
	Worst	0.885	0.345	0.884	0.354	0.929

TABLE 8: The best, worst, and average F -measure values of classification using ACO selected n features from tagged terms for all classes when IBk classifier is used in the test phase.

No. of features selected (n)	F -measure	Dataset				
		Course	Student	Project	Faculty	Conference
10	Best	0.948	0.929	0.937	0.948	0.906
	Avg.	0.943	0.914	0.92	0.936	0.888
	Worst	0.929	0.909	0.896	0.915	0.842
100	Best	0.95	0.887	0.918	0.965	0.866
	Avg.	0.94	0.872	0.906	0.951	0.861
	Worst	0.932	0.858	0.896	0.941	0.855
500	Best	0.951	0.902	0.918	0.94	0.844
	Avg.	0.944	0.899	0.911	0.938	0.84
	Worst	0.937	0.898	0.908	0.936	0.837

According to Table 8, as the number of features selected decreases, average classification performance increases for the Project, Student, and Conference datasets when IBk classifier is used in the test phase. For the Faculty dataset, the best classification performance is obtained when 100 features are selected.

For the naïve Bayes classifier, as the number of selected features decreases, the classification performance increases as shown in Table 9. According to Tables 7, 8, and 9 the features selected by our ACO based feature selection algorithm which uses J48 classifier give satisfactory classification performance when these features are used to classify new web pages with IBk and naïve Bayes classifiers. However, when the results presented in Tables 7, 8, and 9 are compared, we can say that using J48 in the test phase yields more satisfactory classification, since we choose features in ACO according to the J48 classifier.

4.10. Experiment 6: Distribution of Tags in the Selected Feature Subsets. In this section, we have investigated the distribution of tags in the ACO selected subset of features from tagged terms method for each class. Figures 3, 4, 5, 6, and 7 show tag distributions for the ACO selected features for these five classes. Since URLs are very dominant features for the WebKB dataset as it can be seen from Table 4, we did not include features from URLs in this experiment.

According to Figures 3, 4, 5, 6, and 7, when the number of selected features is small (i.e., 10 to 100) the most discriminative features are obtained from h1, title, and anchor tags. As the number of selected features increases (i.e., 500), features extracted from anchor tag and body text become more dominating. According to our experimental results we observed that using small number of features (10 to 100) is enough to make a good classification for most of our datasets, so instead of using all the features extracted from the web pages, it is enough to use features extracted from URL, h1, title, and anchor tags.

4.11. Experiment 7: Effect of ACO Based Feature Selection on Classification Performance in Terms of Accuracy and Running Time. In this section, effect of the proposed ACO based feature selection algorithm on the classification performance has been investigated. For this purpose, F -measure values and running time of the classification of the test web pages using the C4.5 classifier with and without the ACO based feature selection are compared. The results of this experiment are presented in Table 10 and Figure 8. In Table 10, F -measure values of classification of test web pages with ACO based feature selection and without making any feature selection (i.e., by using all features) are compared for features extracted from tagged terms, URLs, and <title> tags.

TABLE 9: The best, worst, and average F -measure values of classification using ACO selected n features from tagged terms for all classes when naïve Bayes classifier is used in the test phase.

No. of features selected (n)	F -measure	Dataset				
		Course	Student	Project	Faculty	Conference
10	Best	0.933	0.864	0.917	0.936	0.897
	Avg.	0.856	0.863	0.88	0.926	0.855
	Worst	0.618	0.859	0.809	0.909	0.838
100	Best	0.868	0.38	0.912	0.875	0.797
	Avg.	0.858	0.354	0.901	0.762	0.695
	Worst	0.829	0.324	0.889	0.602	0.645
500	Best	0.543	0.54	0.502	0.527	0.826
	Avg.	0.51	0.539	0.484	0.51	0.821
	Worst	0.478	0.53	0.472	0.501	0.816

TABLE 10: F -measure values of the C4.5 classifier with and without making any feature selection.

Dataset	Feature extraction methods					
	Tagged terms		Title		URL	
	ACO selected features	Without feature selection	ACO selected features	Without feature selection	ACO selected features	Without feature selection
Course	0.963	0.909	0.983	0.898	1	1
Faculty	0.965	0.911	0.947	0.920	1	1
Project	0.954	0.355	0.891	0.985	1	1
Student	0.940	0.337	0.920	0.679	1	1
Conference	0.940	0.930	0.741	0.372	0.883	0.972

When we compared the results presented in Table 10, we can say that classification performance in terms of F -measure increases when we used features selected by our ACO based feature selection method from tagged terms. For the features extracted from title tags, classification performance increases with our ACO based feature selection method for the Course, Student, Faculty, and Conference datasets, but F -measure value decreases for the Project dataset. For the features extracted from URLs both using all features and making ACO based feature selection give satisfactory classification performance.

In Figure 8, time required to classify test web pages for all classes when C4.5 classifier is used are displayed. As it can be easily seen from the figure, making feature selection reduces the time required to classify new (unseen) web pages sharply without making reduction in classification accuracy (Table 10).

4.12. Experiment 8: Comparison of the Proposed Method with the Well-Known Feature Selection Methods. In this experiment, we compared our ACO based feature selection method with two well-known feature selection methods that are information gain and chi square analysis. To accomplish this, we select 10, 100, and 500 features extracted from tagged terms method with information gain (IG) and chi square (Chi) feature selection methods. After that, we classify test datasets by using the selected features. The classification performance in F -measure values of our ACO based method,

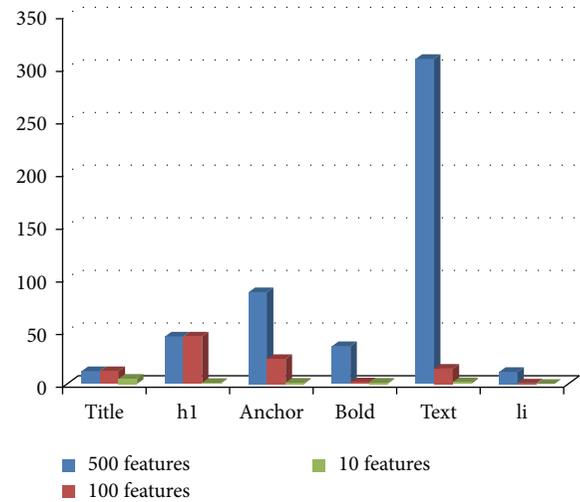


FIGURE 3: Distribution of the ACO selected tags for course class.

chi Square, and information gain methods is presented in Table 11.

The best F -measure values are written in bold face for all cases in Table 11. According to the results given in Table 11, our ACO based feature selection algorithm can select better features with respect to chi square and information gain especially when the number of selected feature is small (i.e., less than 500). As shown in Table 11, when the number of selected features increases, the classification performance of

TABLE 11: Comparison of the proposed method with well-known feature selection methods.

Datasets	No. of selected features								
	10			100			500		
	IG	Chi	ACO	IG	Chi	ACO	IG	Chi	ACO
Faculty	0.89	0.93	0.96	0.90	0.91	0.92	0.90	0.90	0.35
Course	0.91	0.88	0.96	0.36	0.36	0.90	0.90	0.91	0.90
Student	0.80	0.79	0.94	0.42	0.42	0.82	0.34	0.34	0.35
Project	0.83	0.90	0.95	0.39	0.39	0.92	0.35	0.35	0.91
Conference	0.93	0.57	0.92	0.94	0.94	0.94	0.93	0.93	0.93

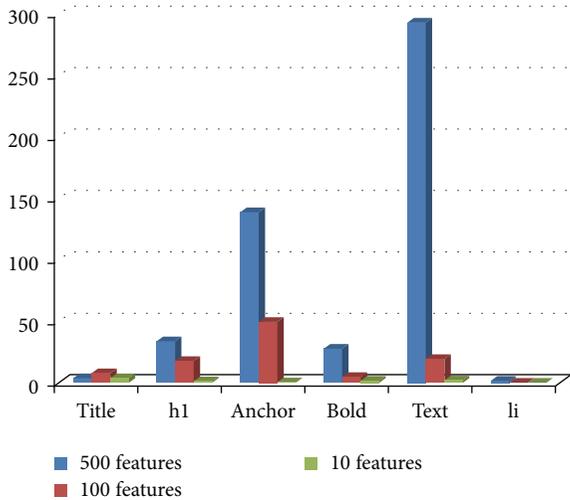


FIGURE 4: Distribution of the ACO selected tags for project class.

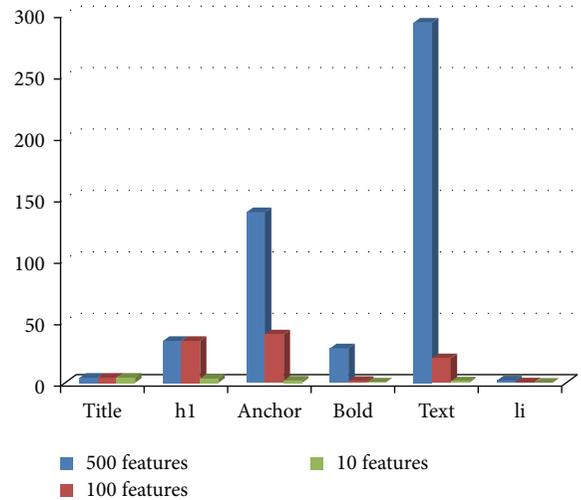


FIGURE 6: Distribution of the ACO selected tags for student class.

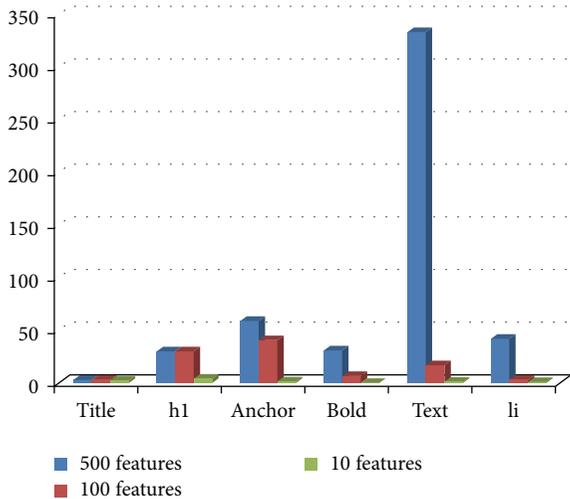


FIGURE 5: Distribution of the ACO selected tags for faculty class.

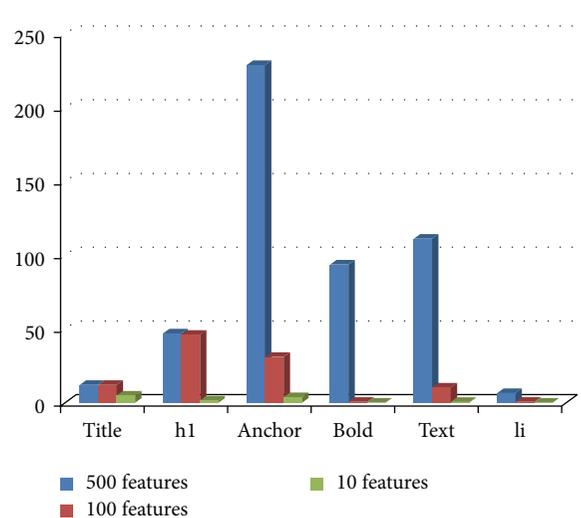


FIGURE 7: Distribution of the ACO selected tags for conference class.

test data decreases, so it is better to use small number of features in terms of classification accuracy and running time (i.e., Figure 8).

4.13. Comparison of the Proposed Method with Earlier Studies. Several ACO based feature selection algorithms have been

proposed for the UCI dataset. The authors of [22–26, 28, 29, 31, 33, 34] have reported that their proposed ACO based algorithms increase the performance of classification. According to our experimental results, we also observed that ACO based feature selection algorithm improves the

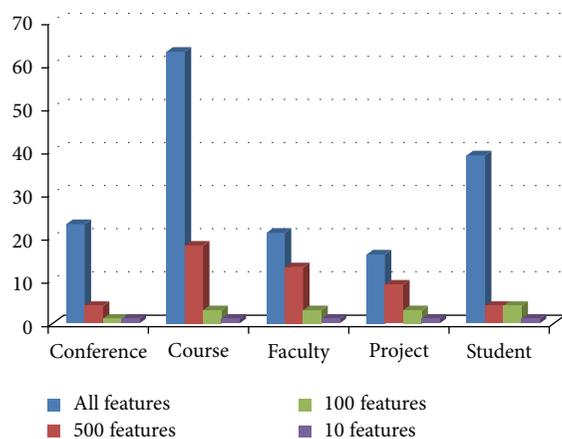


FIGURE 8: Comparison of time required to classify test pages.

classification performance in terms of classification accuracy and time for the WebKB and Conference datasets in general.

In [55], sequential n-grams are used to extract features from the URLs of web pages. The selected features are then classified with maximum entropy and support vector machine separately. The average F -measure value is 0.525 for the multiclass classification of the WebKB dataset. Our average F -measure value is 1.0 for WebKB dataset when features are selected from URLs and binary classification is made. Based on these results, we can say that our ACO based feature selection with binary class classification has better classification performance for the WebKB dataset.

Özel [56] has used tagged terms as features with a GA based classifier. URL addresses are not used in feature extraction step. Average F -measure values are 0.9 and 0.7 for the Course and the Student classes of the WebKB dataset. In our proposed method, on the other hand, average F -measure value is increased up to 1.0 for the Course and Student classes. This comparison shows that features from URLs affect the classification performance positively. In addition to this, using ACO based feature selection and then applying the C4.5 classifier perform better than a GA based classifier.

Jiang [57] has proposed a text classification algorithm that combines a k-means clustering scheme with a variation of expectation maximization (EM), and the proposed method can learn from a very small number of labeled samples and a large quantity of unlabeled data. Experimental results show that the average F -measure value is 0.7 for WebKB dataset in multiclass classification [57]. This result shows that our ACO-based algorithm with binary classification performs better since it yields higher F -measure value.

Joachims [58] has used transductive support vector machines on WebKB dataset with binary class classification and for feature extraction. Bag-of-terms method is used. According to the experimental results of this study, average F -measure values are reported as 0.938, 0.537, 0.184, and 0.838 for the Course, Faculty, Project, and Student classes, respectively. Also, these results show that our proposed algorithm has better performance with respect to the SVM algorithm.

Mangai et al. [39] have used Ward's minimum variance measure and information gain for feature selection from the WebKB dataset. 95% accuracy is achieved for Course dataset with a kNN classifier. However, when we use our ACO-based feature selection algorithm and then apply a kNN classifier to test web pages from the Course dataset, we also obtained 0.951 F -measure value. When we apply C4.5 classifier after our ACO-based feature selection algorithm, the F -measure value increases up to 1.

5. Conclusion

In this study we have developed an ACO-based feature selection system for the web page classification problem. We have used four kinds of feature extraction methods that are, namely, URL only, <title> tag only, bag of terms, and tagged terms. After the feature extraction step, in our ACO based feature selection algorithm, for each ant we select a predefined number of features, and then we evaluate the performance of the selection made by each ant by using the C4.5 classifier. According to the performance of the selected features, we update pheromone values and selection probabilities of the features. This process is repeated until the best features are selected. After selecting the best features, classification of the new (unseen) web pages is made by using the selected feature set. Experimental evaluation shows that using tagged terms as feature extraction method gives good classification performance on the average cases with respect to using bag-of-terms, URL alone, or <title> tag alone methods. Our ACO based feature selection system is able to select better features with respect to well-known information gain and chi square selection methods. The proposed system is effective in reducing the number of features so that it is suitable for classification of high dimensional data. By reducing the feature space, our system also reduces the time required to classify new web pages sharply without loss of accuracy in classification. As future work, performance of our ACO-based feature selection system may be evaluated for the multiclass case.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work was supported by Çukurova University Academic Research Project Unit under Grant no MMF2013D10 and by The Scientific and Technological Research Council of Turkey (TÜBİTAK) scholarship 2211-C.

References

- [1] X. Qi and B. D. Davison, "Web page classification: features and algorithms," *ACM Computing Surveys*, vol. 41, no. 2, article 12, 2009.
- [2] Yahoo!, <https://maktoob.yahoo.com/?p=us>.

- [3] Open direct Project, <http://www.dmoz.org/>.
- [4] C. Chen, H. Lee, and C. Tan, "An intelligent web-page classifier with fair feature-subset selection," *Engineering Applications of Artificial Intelligence*, vol. 19, no. 8, pp. 967–978, 2006.
- [5] W. Shang, H. Huang, H. Zhu, Y. Lin, Y. Qu, and Z. Wang, "A novel feature selection algorithm for text categorization," *Expert Systems with Applications*, vol. 33, no. 1, pp. 1–5, 2007.
- [6] Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Proceedings of the 14th International Conference on Machine Learning (ICML '97)*, pp. 412–420, Nashville, Tenn, USA, July 1997.
- [7] M. H. Aghdam, N. Ghasem-Aghaee, and M. E. Basiri, "Text feature selection using ant colony optimization," *Expert Systems with Applications*, vol. 36, no. 3, pp. 6843–6853, 2009.
- [8] M. Dorigo, *Optimization, learning and natural algorithms [Ph. D. thesis]*, Politecnico di Milano, Milan, Italy, 1992.
- [9] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 26, no. 1, pp. 29–41, 1996.
- [10] L. Liu, Y. Dai, and J. Gao, "Ant colony optimization algorithm for continuous domains based on position distribution model of ant colony foraging," *The Scientific World Journal*, vol. 2014, Article ID 428539, 9 pages, 2014.
- [11] T. M. Mitchell, *Machine Learning*, McGraw-Hill, New York, NY, USA, 1st edition, 1997.
- [12] S. Chakrabarti, M. van den Berg, and B. Dom, "Focused crawling: a new approach to topic-specific Web resource discovery," *Computer Networks*, vol. 31, no. 11–16, pp. 1623–1640, 1999.
- [13] S. A. Özel and E. Saraç, "Focused crawler for finding professional events based on user interests," in *Proceedings of the 23rd International Symposium on Computer and Information Sciences (ISCIS '08)*, pp. 441–444, Istanbul, Turkey, October 2008.
- [14] F. Menczer and R. K. Belew, "Adaptive information agents in distributed textual environments," in *Proceedings of the 2nd International Conference on Autonomous Agents (AGENTS '98)*, pp. 157–164, Minneapolis, Minn, USA, May 1998.
- [15] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, pp. 379–423, 1948.
- [16] W. J. Wilbur and K. Sirotkin, "The automatic identification of stop words," *Journal of Information Science*, vol. 18, no. 1, pp. 45–55, 1992.
- [17] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley-Interscience Press, New York, NY, USA, 1991.
- [18] S. Guiasu, *Information Theory with Applications*, McGraw-Hill Press, New York, NY, USA, 1st edition, 1977.
- [19] Y. Yang, "Noise reduction in a statistical approach to text categorization," in *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 256–263, July 1995.
- [20] Y. Yang and W. J. Wilbur, "Using corpus statistics to remove redundant words in text categorization," *Journal of the American Society for Information Science*, vol. 47, no. 5, pp. 357–369, 1996.
- [21] Reuters Dataset, <http://kdd.ics.uci.edu/databases/reuters21578/reuters21578.html>.
- [22] R. Jensen and Q. Shen, "Fuzzy-rough data reduction with ant colony optimization," *Fuzzy Sets and Systems*, vol. 149, no. 1, pp. 5–20, 2005.
- [23] Y. Chen, D. Miao, and R. Wang, "A rough set approach to feature selection based on ant colony optimization," *Pattern Recognition Letters*, vol. 31, no. 3, pp. 226–233, 2010.
- [24] C. L. Huang, "ACO-based hybrid classification system with feature subset selection and model parameters optimization," *Neurocomputing*, vol. 73, no. 1–3, pp. 438–448, 2009.
- [25] R. K. Sivagaminathan and S. Ramakrishnan, "A hybrid approach for feature subset selection using neural networks and ant colony optimization," *Expert Systems with Applications*, vol. 33, no. 1, pp. 49–60, 2007.
- [26] S. M. Vieira, J. M. C. Sousa, and T. A. Runkler, "Two cooperative ant colonies for feature selection using fuzzy models," *Expert Systems with Applications*, vol. 37, no. 4, pp. 2714–2723, 2010.
- [27] S. Nemati and M. E. Basiri, "Text-independent speaker verification using ant colony optimization-based selected features," *Expert Systems with Applications*, vol. 38, no. 1, pp. 620–630, 2011.
- [28] M. M. Kabir, M. Shahjahan, and K. Murase, "A new hybrid ant colony optimization algorithm for feature selection," *Expert Systems with Applications*, vol. 39, no. 3, pp. 3747–3763, 2012.
- [29] E. Akarsu and A. Karahoca, "Simultaneous feature selection and ant colony clustering," *Procedia Computer Science*, vol. 3, pp. 1432–1438, 2011.
- [30] A. Al-Ani, "Ant colony optimization for feature subset selection," *World Academy of Science, Engineering and Technology*, vol. 1, no. 4, 2007.
- [31] W. Wang, Y. Jiang, and S. W. Chen, "Feature subset selection based on ant colony optimization and support vector machine," in *Proceedings of the 7th WSEAS International Conference of Signal Processing, Computational Geometry & Artificial Vision*, p. 182, Athens, Greece, August 2007.
- [32] N. Jain and J. P. Singh, "Modification of ant algorithm for feature selection," in *Proceedings of the International Conference on Control Automation, Communication and Energy Conservation (INCACEC '09)*, June 2009.
- [33] N. Abd-alsabour and M. Randall, "Feature selection for classification using an ant colony system," in *Proceedings of the 6th IEEE International Conference on e-Science Workshops (e-ScienceW '10)*, pp. 86–91, Brisbane, Australia, December 2010.
- [34] M. H. Rasmy, M. El-Beltagy, M. Saleh, and B. Mostafa, "A hybridized approach for feature selection using ant colony optimization and ant-miner for classification," in *Proceeding of the 8th International Conference on Informatics and Systems (INFOS '12)*, pp. BIO-211–BIO-219, Cairo, Egypt, May 2012.
- [35] R. S. Parpinelli, H. S. Lopes, and A. Freitas, "An ant colony algorithm for classification rule discovery," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 321–332, 2002.
- [36] N. Holden and A. A. Freitas, "Web page classification with an ant colony algorithm," in *Parallel Problem Solving from Nature—PPSN VIII*, vol. 3242 of *Lecture Notes in Computer Science*, pp. 1092–1102, Springer, Berlin, Germany, 2004.
- [37] R. Jensen and Q. Shen, "Web page classification with aco-enhanced fuzzy-rough feature selection," vol. 4259 of *Lecture Notes in Artificial Intelligence*, pp. 147–156, 2006.
- [38] M. Janaki Meena, K. R. Chandran, A. Karthik, and A. Vijay Samuel, "An enhanced ACO algorithm to select features for text categorization and its parallelization," *Expert Systems with Applications*, vol. 39, no. 5, pp. 5861–5871, 2012.
- [39] J. A. Mangai, V. S. Kumar, and S. A. Balamurugan, "A novel feature selection framework for automatic web page classification," *International Journal of Automation and Computing*, vol. 9, no. 4, pp. 442–448, 2012.
- [40] WEKA, <http://www.cs.waikato.ac.nz/~ml/weka>.

- [41] S. Kim and B. Zhang, "Genetic mining of HTML structures for effective web-document retrieval," *Applied Intelligence*, vol. 18, no. 3, pp. 243–256, 2003.
- [42] A. Ribeiro, V. Fresno, M. C. Garcia-Alegre, and D. Guinea, "Web page classification: a soft computing approach," *Lecture Notes in Artificial Intelligence*, vol. 2663, pp. 103–112, 2003.
- [43] A. Trotman, "Choosing document structure weights," *Information Processing & Management*, vol. 41, no. 2, pp. 243–264, 2005.
- [44] M. F. Porter, "An algorithm for suffix stripping," *Program*, vol. 14, no. 3, pp. 130–137, 1980.
- [45] M. Dorigo and T. Stützle, *Ant Colony Optimization*, The MIT Press, 2004.
- [46] T. Bäck, *Evolutionary Algorithms in Theory and Practice*, Oxford University Press, New York, NY, USA, 1996.
- [47] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, Calif, USA, 1st edition, 1993.
- [48] C. J. van Rijsbergen, *Information Retrieval*, Butterworth-Heinemann, London, UK, 2nd edition, 1979.
- [49] M. Craven, D. DiPasquo, D. Freitag et al., "Learning to extract symbolic knowledge from the World Wide Web," in *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI '98)*, pp. 509–516, AAAI Press, July 1998.
- [50] CMU, <http://www.cs.cmu.edu/>.
- [51] WebKB, <http://www.cs.cmu.edu/~webkb/>.
- [52] DBLP, <http://www.informatik.uni-trier.de/~ley/db/>.
- [53] Google, <http://www.google.com>.
- [54] E. Saraç and S. A. Özel, "URL tabanlı web sayfası sınıflandırma," in *Akıllı Sistemlerde Yenilikler ve Uygulamaları Sempozyumu (ASYU '10)*, pp. 13–18, 2010.
- [55] M.-Y. Kan and H. O. N. Thi, "Fast webpage classification using URL features," in *Proceedings of the 14th ACM International Conference on Information and Knowledge Management (CIKM '05)*, pp. 325–326, New York, NY, USA, November 2005.
- [56] S. A. Özel, "A Web page classification system based on a genetic algorithm using tagged-terms as features," *Expert Systems with Applications*, vol. 38, no. 4, pp. 3407–3415, 2011.
- [57] E. P. Jiang, "Learning to integrate unlabeled data in text classification," in *Proceedings of the 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT '10)*, pp. 82–86, Chengdu, China, July 2010.
- [58] T. Joachims, "Transductive inference for text classification using support vector machines," in *Proceedings of the 16th International Conference on Machine Learning*, pp. 200–209, 1999.

Research Article

Reinforcement Learning for Routing in Cognitive Radio Ad Hoc Networks

Hasan A. A. Al-Rawi,¹ Kok-Lim Alvin Yau,¹ Hafizal Mohamad,²
Nordin Ramli,² and Wahidah Hashim²

¹ Department of Computer Science and Networked Systems, Sunway University, No. 5 Jalan Universiti, Bandar Sunway, 46150 Petaling Jaya, Selangor, Malaysia

² Wireless Network and Protocol Research Lab, MIMOS Berhad, Technology Park Malaysia, 57000 Kuala Lumpur, Malaysia

Correspondence should be addressed to Kok-Lim Alvin Yau; koklimy@sunway.edu.my

Received 13 April 2014; Accepted 31 May 2014; Published 16 July 2014

Academic Editor: Su Fong Chien

Copyright © 2014 Hasan A. A. Al-Rawi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Cognitive radio (CR) enables unlicensed users (or secondary users, SUs) to sense for and exploit underutilized licensed spectrum owned by the licensed users (or primary users, PUs). Reinforcement learning (RL) is an artificial intelligence approach that enables a node to observe, learn, and make appropriate decisions on action selection in order to maximize network performance. Routing enables a source node to search for a least-cost route to its destination node. While there have been increasing efforts to enhance the traditional RL approach for routing in wireless networks, this research area remains largely unexplored in the domain of routing in CR networks. This paper applies RL in routing and investigates the effects of various features of RL (i.e., reward function, exploitation, and exploration, as well as learning rate) through simulation. New approaches and recommendations are proposed to enhance the features in order to improve the network performance brought about by RL to routing. Simulation results show that the RL parameters of the reward function, exploitation, and exploration, as well as learning rate, must be well regulated, and the new approaches proposed in this paper improves SUs' network performance without significantly jeopardizing PUs' network performance, specifically SUs' interference to PUs.

1. Introduction

Cognitive radio (CR) has been proposed to enable unlicensed users (or secondary users, SUs) to exploit the underutilized licensed channels (or white spaces) owned by the licensed users (or primary users, PUs). Most traditional routing schemes adopt a rule-based approach [1] in which each node keeps and follows a set of predefined rules in its action selection for different network conditions; and this may not suit CR due to its intrinsic characteristic of the *dynamicity* and *unpredictability* of the network conditions (i.e., PUs' activities and channel quality) which require context awareness and intelligence [2–4]. Context awareness enables a SU node to observe the operating environment; while intelligence enables the SU node to learn and make action selection that maximizes network performance as time goes by. These

capabilities are essential as the rule-based approach may not be feasible to define actions for all possible sets of network conditions in CR networks (CRNs).

Reinforcement learning (RL) [5], which is an artificial intelligence approach, has been applied to achieve context awareness and intelligence in CRNs [2]. This article presents a simulation study on the application of RL to routing in CRNs. Firstly, the traditional RL approach is applied in a routing scheme, which we call Cognitive Radio Q-routing (CRQ-routing). Next, a RL feature, namely reward function, is investigated. An enhanced RL-based routing scheme called weighted cognitive radio Q-routing (WCRQ-routing) is proposed. Subsequently, other two RL features, namely, exploitation and exploration, as well as learning rate, are investigated. The network performance of RL-based routing schemes can be enhanced by regulating the RL

features; hence, new enhancements are proposed for the reward function, exploitation, and exploration, as well as learning rate.

CRQ-routing is a spectrum-aware scheme that finds least-cost routes in CRNs taking into account the dynamicity and unpredictability of the channel availability and channel quality. Simulation results show that CRQ-routing and its enhancements minimize SUs' interference to PUs, SUs' end-to-end delay, and SUs' packet loss rate, as well as maximizing SUs' throughput.

Our contributions are as follows.

- (i) Section 3 presents CRQ-routing which applies the traditional RL approach.
- (ii) Section 4 investigates different reward representations for network performance enhancement. In addition to CRQ-routing, this section investigates a variant of the reward function which we conveniently call WCRQ-routing. WCRQ-routing applies a weight factor ω to adjust the tradeoff between PUs' and SUs' network performance. Performance enhancement achieved by CRQ-routing and WCRQ-routing for different PU utilization levels (PULs) and packet error rate (PER) are compared with the traditional shortest path (SP) routing scheme and the optimal primary user-aware shortest path (PASP) routing scheme.
- (iii) Section 5 investigates the effects of exploitation and exploration on network performance. A simple and pragmatic exploration approach called dynamic softmax (DS) is proposed to dynamically regulate the frequency of exploration according to the dynamicity of the operating environment. Performance enhancement achieved by DS is compared with two traditional exploration approaches, namely, ϵ -greedy and softmax.
- (iv) Section 6 investigates the effects of learning rate on network performance. A simple and pragmatic learning rate adjustment approach called the counterapproach (CA), which is based on the traditional win-or-learn-fast policy hill climbing (or win-lose) [6], is proposed to dynamically regulate the learning rate according to the dynamicity of the operating environment. Performance enhancement achieved by CA is compared with the traditional approach, namely, win-lose.

Simulation experiment, results, and discussions are presented in each of the sections. Finally, Section 7 concludes this paper.

2. Related Work

While most researches focus on the enhancement of either PUs' or SUs' network performance [7–10], this paper focuses on both PUs' and SUs'. CRQ-routing minimizes SUs' interference to PUs without causing significant detrimental effects on SUs' network-wide performance.

In [7–10], either one or both of the following requirements are applicable. Firstly, information on PUs' and SUs' physical locations is essential. The associated challenges are additional energy consumption, increased hardware cost, and the availability of the physical location information in indoor scenarios [2]. Secondly, network-wide information such as link cost is essential; however, it is difficult to obtain up-to-date information for the entire network in the presence of dynamicity and unpredictability of the channel availability and channel quality in CRNs. For instance, a reactive routing scheme requires a SU destination node to confirm a route prior to data transmission; however, due to the dynamicity and unpredictability of the channel availability and channel quality, a new route may have expired before routing information reaches the SU destination node. CRQ-routing does not require geographical and network-wide information, and it adopts a per-hop routing approach (rather than an end-to-end routing approach) that enables each SU intermediate node to make routing decision for a single hop to its next-hop node based on local information.

The application of RL to routing schemes in CRNs has been limited, such as [11], although it has been shown to improve routing performance in various traditional wireless networks [12, 13]. In [11], the SUs' network performance is shown to be enhanced, while in CRQ-routing, both PUs' and SUs' network performances are enhanced. Using RL, CRQ-routing integrates route discovery mechanism with channel selection. CRQ-routing is a multipath routing scheme that enables a SU node to maintain multiple routes, and this can be well incorporated into RL through its feature called exploration. Generally speaking, the existence of multiple routes helps to enhance network reliability and to achieve load balancing among various routes. This is because a SU can automatically switch its route to another one during route recovery in the event of route failure.

3. CRQ-Routing: Application of the Traditional Reinforcement Learning Approach to Routing

This section presents CRQ-routing that takes account of the PUs' and SUs' network performance by minimizing SUs' interference to PUs along a route without significantly jeopardizing SUs' network-wide performance. It applies a traditional RL approach called Q-learning [14, 15], which is a popular RL approach. CRQ-routing enables a SU to observe its local operating environment regularly and subsequently to learn an action selection policy through exploring various routes, and finally to choose routes with enhanced network performance (i.e., lower SUs' interference to PUs, lower SUs' end-to-end delay, lower SUs' packet loss rate, and higher SUs' throughput). Generally speaking, RL enables a SU node to

- (a) estimate the dynamic link cost. This allows a SU to learn about and adapt to the local network conditions (i.e., PUs' activities and channel quality) which are dynamic and unpredictable in nature,

TABLE 1: CRQ-routing model embedded at SU node i .

State	$s_t^i \in S = \{1, 2, \dots, N-1\}$, each state s_t^i representing a SU destination node n . N represents the number of SUs in the entire network.
Action	$a_t^i \in A^i = \{1, 2, \dots, J\}$, each action a_t^i representing the selection of a SU next-hop node j along with its operating channel. J represents the number of SU i 's neighboring SU nodes.
Cost	$r_t^i(a_t^i)$ represents the link-layer delay incurred to successfully deliver a packet from SU node i to SU neighbor node $a_t^i = j$, including retransmission delays as a result of PU-SU packet collision and packet loss.

- (b) search for the best-possible route using information observed from the local operating environment and information received from neighboring nodes,
- (c) incorporate a wide range of factors that can affect the routing performance into consideration, including both PUs' activities and channel quality.

By choosing links with lower link-layer delay, which is the time duration required to deliver a SU's packet to a next-hop node successfully, SUs' interference to PUs can be reduced and SUs' end-to-end network performance can be enhanced. Note that the link-layer delay includes the time duration incurred by retransmission as a result of PU-SU packet collisions. The RL model for CRQ-routing is shown in Table 1, and it is embedded in a SU node i . Using CRQ-routing, each SU chooses a next-hop node and channel pair as part of a route. There are three key representations for the RL model as follows.

- (i) State $s_t^i \in S = \{1, 2, \dots, N-1\}$ represents a SU destination node n , where N represents the number of SUs in the entire network.
- (ii) Action $a_t^i \in A^i = \{1, 2, \dots, J\}$ represents the selection of a next-hop SU neighbor node j along with its operating channel, where J represents the number of SU i 's neighboring SU nodes.
- (iii) Cost $r_t^i(a_t^i)$, which indicates the consequence upon taking action a_t^i , represents the link-layer delay of a SU communication node pair, namely nodes i and j . The link-layer delay includes retransmission delays caused by packet loss and PU-SU packet collision. Hence, the end-to-end delay (or the accumulated link-layer delay) reflects the accumulated SUs' interference to PUs; and by achieving lower delay, the interference level can be reduced.

Each SU node i keeps track of Q -value $Q_t^i(s_t^i, a_t^i)$, which relates the three representations, in its Q -table (i.e., routing table). For each state-action pair, the Q -value represents the accumulated link-layer delay of a route leading to SU destination node s_t^i by choosing a SU next-hop node $a_t^i = j$. At time t , SU i selects a SU next-hop node $a_t^i = j$ as part of a route to reach destination node s_t^i ; and upon a successful transmission at time $t+1$, it receives an end-to-end delay estimate for the route, namely $Q_t^j(s_t^j, k)$, from node j and estimates the link-layer delay $r_{t+1}^i(j)$. Note that the link-layer delay is dynamic and unpredictable due to the nature of the PUs' activities in which there are different levels of PUL. In

general, the link-layer delay increases with PUL in a channel. The Q -value $Q_t^i(s_t^i, a_t^i = j)$ is updated as follows:

$$Q_{t+1}^i(s_t^i, j) \leftarrow (1 - \alpha) Q_t^i(s_t^i, j) + \alpha \left(r_{t+1}^i(j) + \min_{k \in A^j} Q_t^j(s_t^j, k) \right), \quad (1)$$

where $0 \leq \alpha \leq 1$ is the learning rate and node $k \in A^j$ is an upstream node of SU node j . A SU node i adopts a policy $\pi_{t+1}^i(s_t^i)$ that chooses a SU next-hop node with the minimum cost as follows:

$$\pi_{t+1}^i(s_t^i) = \underset{a \in A^i}{\operatorname{argmin}} \left(Q_t^i(s_t^i, a) \right). \quad (2)$$

In the next section, we present a variant of the reward representation in RL to further enhance PUs' and SUs' network performance.

4. WCRQ-Routing: A Variant of the Reward Representation

WCRQ-routing incorporates a weight factor into the reward representation of CRQ-routing to adjust the tradeoff between PUs' and SUs' network performance. WCRQ-routing provides further enhancement on SUs' network performance without jeopardizing PUs' network performance. The main difference between WCRQ-routing and CRQ-routing is the cost (or negative reward) representation as follows.

- (i) CRQ-routing enables a SU to learn about the accumulated cost in terms of link-layer delay along a route.
- (ii) WCRQ-routing enables a SU to learn about the accumulated cost in terms of the number of SUs' packet retransmissions and the packet queue length of SUs along a route. Hence, WCRQ-routing enables a SU to take account of packet retransmission which further improves the PUs' network performance, and network performance which further improves the SUs' network performance. WCRQ-routing also incorporates a weight factor ω which adjusts the tradeoff between PUs' and SUs' network performance.

The RL model for WCRQ-routing is shown in Table 2, and it is embedded in a SU node i . The state and action representations are similar for CRQ-routing (see Table 1) and WCRQ-routing, and so only the reward representation is shown. The reward representation for the RL model is as follows.

TABLE 2: Reward representation for WCRQ-routing model embedded at SU node i .

Cost	$r_t^i(a_t^i) = \omega r_t^{i,j} + (1 - \omega)q_t^j$ where $r_t^{i,j}$ represents the number of retransmissions for a packet sent from SU node i to SU neighbor node j at time t , while q_t^j represents the number of packets in the queue of SU neighbor node j . Weight factor $\omega = [0, 1]$ is used to adjust the tradeoff between PUs' and SUs' network performance.
------	---

- (i) Cost $r_t^i(a_t^i) = \omega r_t^{i,j} + (1 - \omega)q_t^j$ has two components:
- (1) $r_t^{i,j}$ represents the number of retransmissions for a packet sent from SU node i to SU neighbor node j at time t as a result of PU-SU packet collisions and SU packet loss and
 - (2) q_t^j represents the number of packets in the queue of SU neighbor node j . Both $r_t^{i,j}$ and q_t^j values in reward $r_t^i(a_t^i)$ are normalized to $[0, 1]$. The weight factor $\omega = [0, 1]$ adjusts the tradeoff between PUs' and SUs' network performance.

Similar to CRQ-routing, each SU node i keeps track of Q -values $Q_t^i(s_t^i, a_t^i)$ in its Q -table (i.e., routing table). For each state-action pair, the Q -value represents a weighted cost that takes account of the number of packet retransmissions and packet queue length of SUs along a route leading to SU destination node s_t^i by choosing a SU next-hop node a_t^i . At time t , SU i selects a SU next-hop node $a_t^i = j$ as part of a route to reach destination node s_t^i and upon a successful transmission, it receives an estimate of the weighted cost for the route, namely $Q_t^j(s_t^j, k)$, from node j and estimates reward $r_{t+1}^i(j)$, which takes into account the number of retransmissions for a packet and the number of packets in the queue of SU neighbor node j at time $t + 1$. Note that both number of retransmissions for a packet and number of packets in the queue of a SU's neighbor node are dynamic and unpredictable due to the nature of the PUs' activities in which there are different levels of PUL. In general, $r_{t+1}^i(j)$ increases with PUL in a channel. The Q -value $Q_t^i(s_t^i, a_t^i)$ is updated using (1) and a SU next-hop node with the minimum cost is chosen using (2).

The rest of this section is organized as follows. Section 4.1 presents simulation setup and parameters. Section 4.2 presents a comparison of network performance achieved by CRQ-routing, WCRQ-routing, and two baseline routing schemes, namely, shortest path (SP) and PU aware shortest path (PASP) routing schemes. Section 4.3 presents the effects of the weight factor ω in reward representation on network performance.

4.1. Simulation Setup and Parameters. Figure 1 shows the system model which is a multihop CRN with N SUs and K PUs [16]. Each PU transmits in one of the K different channels. In Figure 1, we consider a SU source node A and a SU destination node G .

We compare the network performance achieved by CRQ-routing and WCRQ-routing with nonlearning approaches, specifically, the traditional SP routing and the optimal PASP routing approaches. SP routing selects a route that has the minimum number of hops and this has been shown to improve the end-to-end network performance in traditional networks. PASP routing selects a route that has the minimum

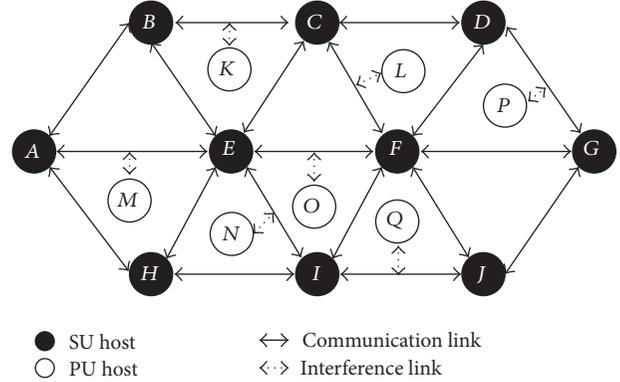


FIGURE 1: Network scenario.

accumulated amount of PUs' activities. This means that the route encounters the least number of PUs, and so the PUL along the route may be the lowest. However, PASP routing may not be feasible in practice as it is a centralized approach that requires network-wide information of PUL for each link and channel. Nevertheless, PASP routing is an optimal approach that minimizes SUs' interference to PUs and so it serves as a good comparison in our simulation study.

There are four network performance metrics as follows.

- (i) *PU-SU collision probability* indicates the level of SUs' interference to PUs. Reducing this metric improves the PUs' network performance. This metric is a ratio of the number of PU-SU collisions to the number of SUs' packet transmissions.
- (ii) *SU end-to-end delay* includes the transmission, processing, backoff, and queuing delays along a route.
- (iii) *SU packet loss rate* is a ratio of the number of packet loss to the total number of packets sent.
- (iv) *SU throughput* is the number of arriving packets per second (pps) at the SU destination node.

The simulation compares the aforementioned performance metrics with respect to different levels of PULs for CRQ-routing, WCRQ-routing, SP routing, and PASP routing. The dynamicity of the PUs' activities (or PUL) is represented by PU arrival rate μ_{PUL} , and each scenario has a certain level of unpredictability of the PUs' activities (or the standard deviation of PU arrival rate) which is represented by σ_{PUL} .

Table 3 shows a summary of the simulation parameters and values. Generally speaking, each simulation is run for 100 seconds and repeated 50 times with different random seeds. The simulation reporting interval is 1 second which indicates that, at each second of the simulation running time, a mean value of the simulation result is calculated. The default

TABLE 3: Simulation parameters and values.

Category	Parameter	Value
SU	SU's transmission delay, d_{SU}^{tr}	1.0 ms
	Processing delay, d_{SU}^{pr}	1.0 ms
	Mean arrival rate, λ_{SU}	0.6
	Learning rate, α	0.5
	WCRQ-routing weight factor, ω	0.5
PU	PU's transmission delay, d_{PU}^{tr}	1.2 ms
	Mean arrival rate (or PUL), μ_{PUL}	[0.0, 1.0]
	Standard deviation, σ_{PUL}	{0.0, 0.4, 0.8}
Channel	Mean PER, μ_{PER}	0.05
	Standard deviation of PER, σ_{PER}	0.025

number of SUs is $N = 10$, and PUs is $K = 19$. Each PU does not change its channel and operates in distinctive channels. Each PU activity in channel $k \in K$ is modeled as a Poisson process with mean $\lambda_{PUL}^k(\mu_{PUL}, \sigma_{PUL})$, which is assigned using Box-Muller transform [17] according to an expected mean $\mu_{PUL} = [0, 1]$ and standard deviation $\sigma_{PUL} = [0, 1]$. The standard deviation of PUL is $\sigma_{PUL} \in \{0.0, 0.4, 0.8\}$, which indicates low, medium, and high levels of unpredictability of the PUs' activities, and these values are chosen due to their significant effects to the results. Since the focus is on the comparison of network performance in regard to PUL, we assume the channels have low level of noise with $\mu_{PER} = 0.05$ and $\sigma_{PER} = 0.025$. Packets are generated at the SU source node using Poisson process with a mean arrival rate of $\lambda_{SU} = 0.6$. Since the PUs have higher priority than SUs, their transmissions take longer; specifically, the SUs' transmission delay is $d_{SU}^{tr} = 1.0$ ms and the PUs' transmission delay is $d_{PU}^{tr} = 1.2$ ms. In order to model a simple queuing delay d_{SU}^{qu} , we assume a finite packet queue size of 1000 packets in each SU node with a constant processing delay $d_{SU}^{pr} = 1.0$ ms. The Q-values are initialized to 0 in order to encourage exploration at the start of the simulation. The SU learning rate is $\alpha = 0.5$ (see (1)). For WCRQ-routing, the weight factor is set to $\omega = 0.5$ so that there is a balanced tradeoff between PUs' and SUs' network performance. The effects of ω on the network performance are presented in Section 4.3.

4.2. Comparison of CRQ-Routing, WCRQ-Routing, SP, and PASP Routing Schemes. We present simulation results for the four performance metrics in this section.

4.2.1. SUs' Interference to PUs. When the standard deviation of PUL is low $\sigma_{PUL} = 0$, most next-hop node (or link) and channel pairs have the same PU mean arrival rate μ_{PUL} , so all routing schemes achieve similar probability of PU-SU packet collisions across a CRN (see Figure 2(a)). When the unpredictability level of PUL is $\sigma_{PUL} = 0.4$, the link and channel pairs have greater difference in the levels of PU mean arrival rate μ_{PUL} , WCRQ-routing, and CRQ-routing choose routes that minimize SUs' interference to PUs. Both WCRQ-routing and CRQ-routing reduce collisions with PUs for up to 19% compared to SP routing, whereas the empirical PASP

routing scheme, which provides the best results, reduces collisions with PUs for up to 30% compared to SP routing (see Figure 2(b)). When $\sigma_{PUL} = 0.8$, the link and channel pairs have the greatest difference in the levels of PU mean arrival rate μ_{PUL} . Similar trends to the network scenario of $\sigma_{PUL} = 0.4$ are observed although collisions with PUs have generally increased due to the increased unpredictability of PUs' activities (see Figure 2(c)).

Generally speaking, both WCRQ-routing and CRQ-routing achieves almost similar performance in terms of SUs' interference to PUs; and the SUs' interference to PUs increases with the PU mean arrival rate μ_{PUL} and the standard deviation of PUL σ_{PUL} .

4.2.2. SU End-to-End Delay. When the standard deviation of PUL is low $\sigma_{PUL} = 0$, most link and channel pairs have the same PU mean arrival rate μ_{PUL} . The SU end-to-end delay increases with PU mean arrival rate μ_{PUL} (see Figure 3(a)). When the PU mean arrival rate μ_{PUL} is low, the SU end-to-end delay consists of mainly transmission and queuing delays, and when μ_{PUL} becomes higher (i.e., $\mu_{PUL} > 0.3$), the retransmission and backoff delays caused by PU-SU packet collisions and overflow of SUs' packet queues increase contributing to higher SU end-to-end delay. WCRQ-routing reduces SU end-to-end delay for up to 72% compared to the other routing schemes. This is because WCRQ-routing takes into account the queue length of SUs along the route, so it chooses routes with lower network congestion contributing to lower SU end-to-end delay.

When the unpredictability level of PUL increases from $\sigma_{PUL} = 0.4$ to $\sigma_{PUL} = 0.8$, the link and channel pairs have greater difference in the levels of PU mean arrival rate μ_{PUL} . WCRQ-routing chooses routes that minimize the number of packet retransmissions caused by PU-SU packet collisions and overflow of SUs' packet queues contributing to lower SU end-to-end delay (see Figures 3(b) and 3(c)). With greater unpredictability level of PUL σ_{PUL} , it is easier for WCRQ-routing to find a route with better performance; hence, the lower SU end-to-end delay when $\sigma_{PUL} = 0.8$ in Figure 3(c) compared to $\sigma_{PUL} = 0.4$ in Figure 3(b), and WCRQ-routing achieves lower SU end-to-end delay for up to 89% when $\sigma_{PUL} = 0.8$ compared to other routing schemes. Additionally,

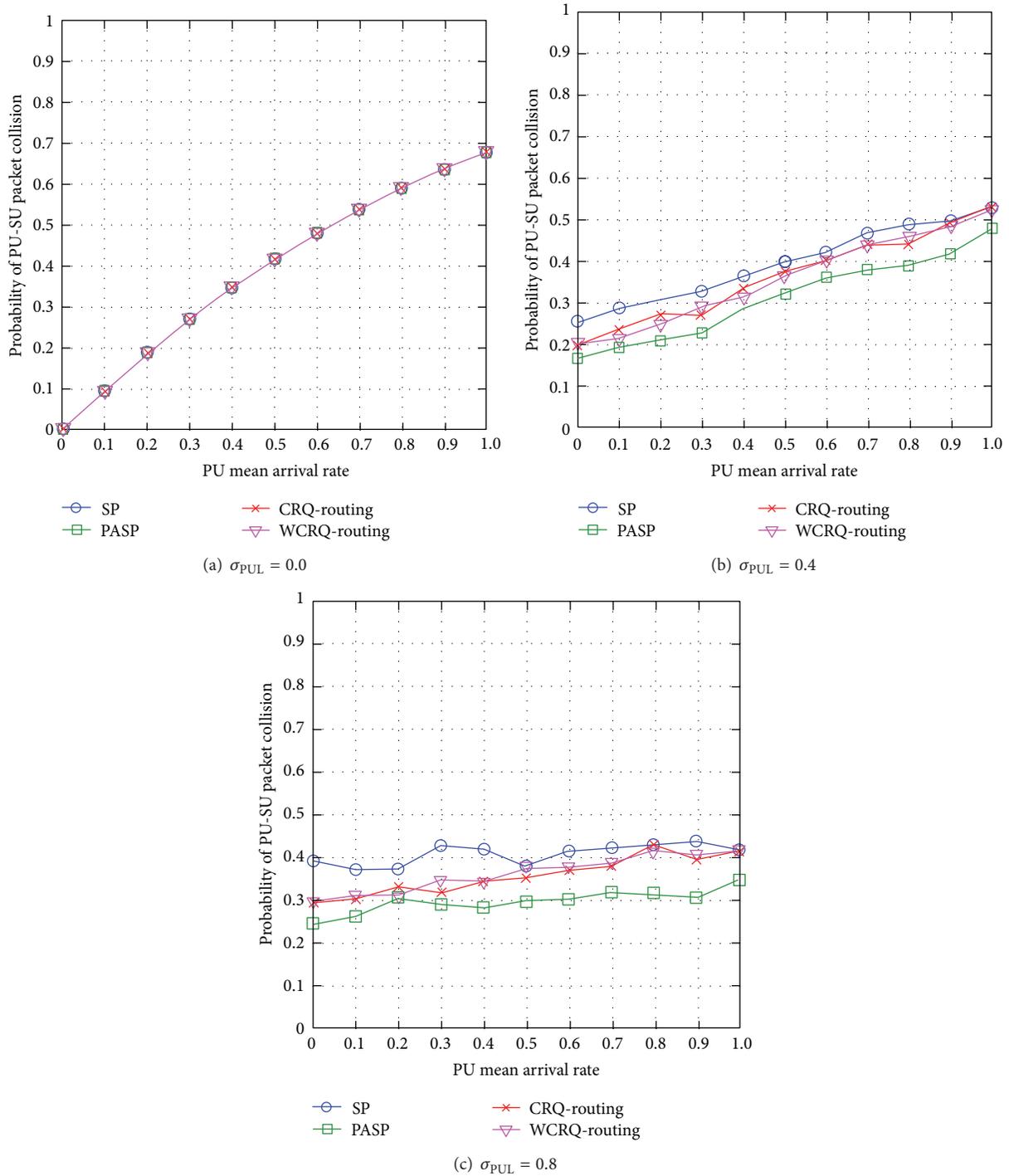


FIGURE 2: SUs' interference to PUs for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

there are two main observations. Firstly, the fluctuations of SU end-to-end delay are observed because the routes of SP routing and PASP routing are static as they are unaware of the unpredictability of PUL, while the routes of CRQ-routing and WCRQ-routing are dynamic in nature. Secondly, PASP routing and CRQ-routing deteriorate to the network performance of SP routing with increasing PU mean arrival

rate μ_{PUL} because both schemes take account of PU mean arrival rate μ_{PUL} only in routing decision, and so when μ_{PUL} becomes higher, they may choose longer routes resulting in higher SU end-to-end delay. In contrast, WCRQ-routing takes account of network congestion as well.

Generally speaking, WCRQ-routing reduces SU end-to-end delay, and the metric increases with the PU mean arrival

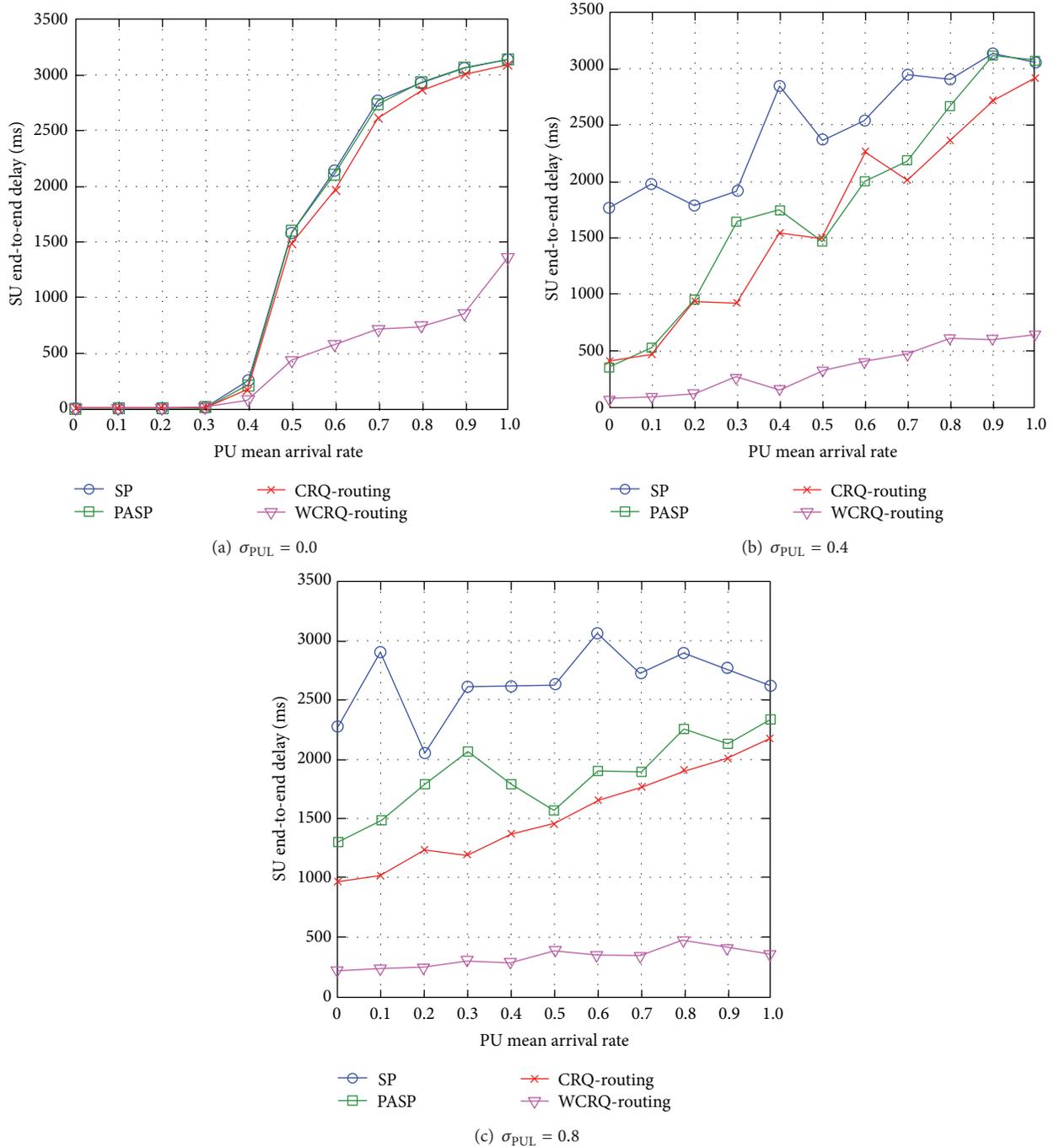


FIGURE 3: SU end-to-end delay for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

rate μ_{PUL} and reduces with the standard deviation of PUL σ_{PUL} .

4.2.3. *SU Packet Loss*. When the standard deviation of PUL is low $\sigma_{PUL} = 0$, WCRQ-routing achieves load-balancing among the available routes as it takes into account the queue length of SUs along a route, so it reduces network congestion, and hence there is lower SU packet loss of up to 16% compared to the other routing schemes (see Figure 4(a)).

When the unpredictability level of PUL is $\sigma_{PUL} = 0.4$ and $\sigma_{PUL} = 0.8$, WCRQ-routing achieves almost similar network performance to PASP routing with lower packet loss for up to 56% compared to SP routing and 25% compared to CRQ-routing when $\sigma_{PUL} = 0.8$ (see Figures 4(b) and 4(c)). This is because WCRQ-routing chooses routes with lower network congestion in order to minimize packet retransmission caused by PU-SU packet collisions and overflow of SU's packet queues leading to lower number of packet

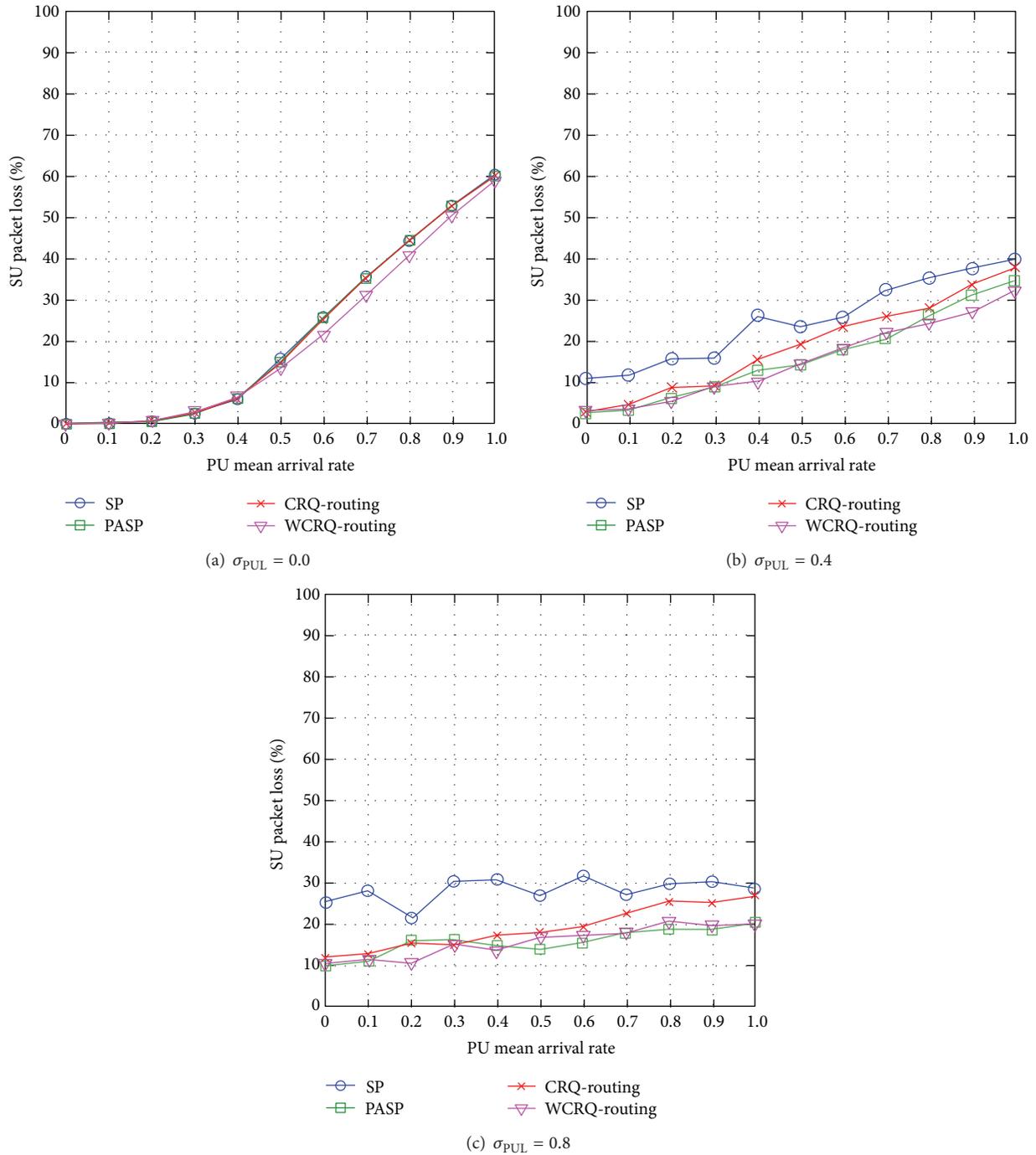


FIGURE 4: SU packet loss for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

retransmissions. Both network scenarios of $\sigma_{PUL} = 0.4$ and $\sigma_{PUL} = 0.8$ share almost similar trends although the SU packet loss has generally decreased in the case of $\sigma_{PUL} = 0.8$.

Generally speaking, WCRQ-routing reduces SU packet loss, and the metric increases with the PU mean arrival rate μ_{PUL} and reduces with the standard deviation of PUL σ_{PUL} .

4.2.4. SU Throughput. When the standard deviation of PUL is low $\sigma_{PUL} = 0$, WCRQ-routing achieves load-balancing

among available routes as it takes into account the queue length of SUs along a route, so it reduces network congestion, and hence there is higher SU throughput of up to 11% compared to the other routing schemes (see Figure 5(a)).

When the unpredictability level of PUL is $\sigma_{PUL} = 0.4$ and $\sigma_{PUL} = 0.8$, WCRQ-routing achieves higher SU throughput of up to 27% compared to SP routing, up to 12% compared to CRQ-routing, and up to 9% compared to PASP routing when $\sigma_{PUL} = 0.8$ (see Figures 5(b) and 5(c)). This is because

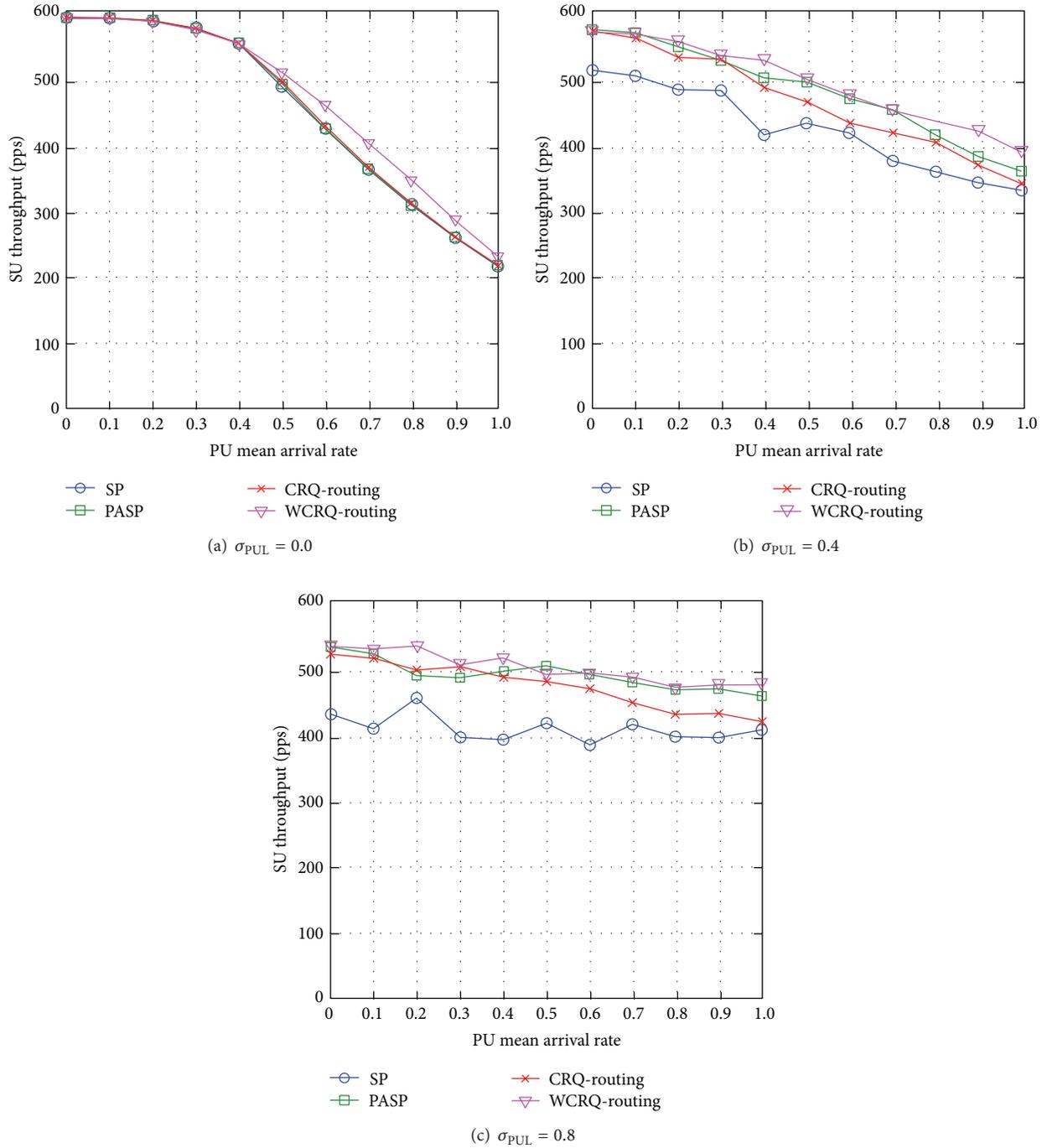


FIGURE 5: SU throughput for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

WCRQ-routing chooses routes with lower network congestion in order to minimize packet retransmission caused by PU-SU packet collisions and overflow of SU's packet queues leading to higher SU throughput.

Generally speaking, WCRQ-routing increases SU throughput, and the metric reduces with the PU mean arrival rate μ_{PUL} and reduces with the standard deviation of PUL σ_{PUL} .

4.2.5. Section Summary. We summarize simulation outcomes for the comparison of CRQ-routing, WCRQ-routing, SP, and PASP routing schemes as follows.

- (i) Network performance degrades as the dynamicity of PUs' activities μ_{PUL} increases.
- (ii) WCRQ-routing and CRQ-routing minimize SUs' interference to PUs with respect to PUL in the

TABLE 4: Simulation parameters and values for investigating the effects of weight factor in reward representation.

Category	Parameter	Value
SU	Mean arrival rate, λ_{SU}	0.8
PU	Mean arrival rate (PUL), μ_{PUL}	{0.2, 0.4}
	Standard deviation, σ_{PUL}	0.5
Channel	Mean PER, μ_{PER}	{0.2, 0.4}
	Standard deviation of PER, σ_{PER}	0.2

presence of dynamicity and unpredictability of the channel availability.

- (iii) WCRQ-routing enhances network performance of SUs compared to other routing schemes including lower SU end-to-end delay, lower SU packet loss, and higher SU throughput. Using a weight factor in WCRQ-routing, the cost represents two factors including the number of packet retransmissions and the packet queue length of SUs.

4.3. Effects of Weight Factor in Reward Representation. This section investigates the effects of the weight factor ω of WCRQ-routing on network performance. WCRQ-routing applies a weight factor ω to adjust the tradeoff between PUs' and SUs' network performance. Based on Table 2, with a higher value of ω , there is greater consideration on $r_t^{i,j}$ which represents the number of retransmissions for a packet sent from SU node i to SU neighbor node j at time t and indicates the probability of PU-SU packet collision, and lesser consideration on SU neighbor node j 's queue length q_t^j which indicates SU network congestion. This means a higher value of weight factor ω improves the PUs' network performance, while a lower value of ω improves the SUs' network performance.

While Table 3 presents the default simulation parameters and values, Table 4 presents the specific simulation parameters and values for this investigation. Generally speaking, we simplify the simulation values in order to focus on the effects of weight factor ω . This explains why we consider two different PUL μ_{PUL} and PER μ_{PER} values only and a fixed value of the standard deviation of PUL σ_{PUL} . We increase the network congestion level with increased SU mean arrival rate to $\lambda_{\text{SU}} = 0.8$.

In this section, we assume that when PU-SU packet collision occurs, a SU packet is transmitted successfully while the PU's packet is lost; therefore, the PUs' activities can be easily affected and it is prone to SUs' interference. We present the simulation results for the four performance metrics in the rest of this section.

4.3.1. SUs' Interference to PUs. Figure 6 shows that the SUs' interference to PUs decreases with the weight factor ω of WCRQ-routing. Hence, a higher value of weight factor ω improves the PUs' network performance, while a lower value of ω improves SUs' network performance. In addition, with respect to PUL and PER, Figure 6 shows that PUL has greater effects on PUs' network performance compared to

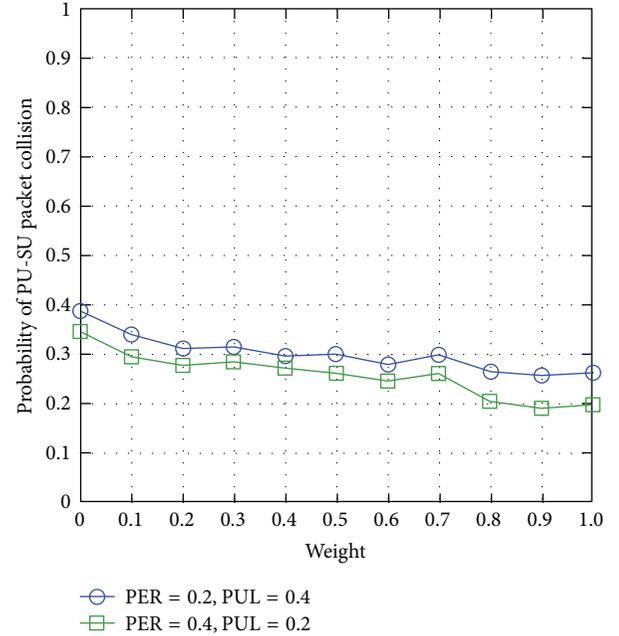


FIGURE 6: SUs' interference to PUs for varying weight factor ω .

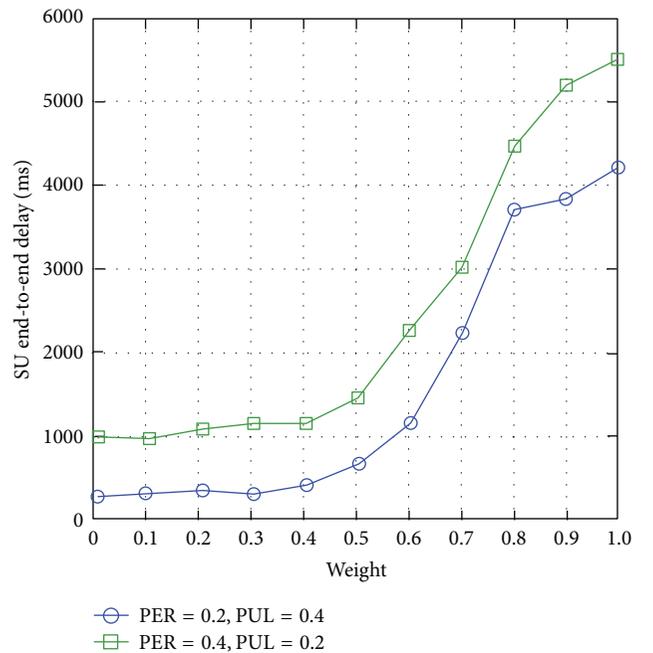


FIGURE 7: SU end-to-end delay for varying weight factor ω .

PER because PUL has a direct effect on the PU-SU packet collisions; hence, higher PULs indicate higher probability of PU-SU collisions.

4.3.2. SU End-to-End Delay. Figure 7 shows that the SU end-to-end delay increases with the weight factor ω of WCRQ-routing. Hence, a higher value of weight factor ω improves the PUs' network performance; while a lower value of ω improves

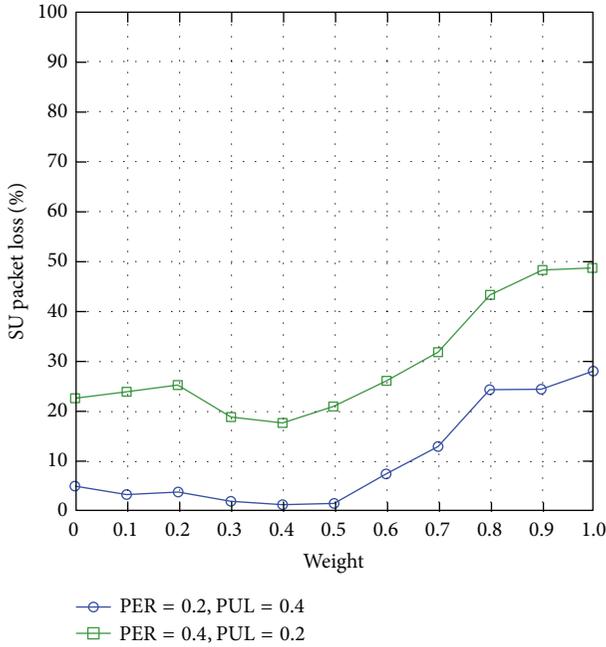


FIGURE 8: SU packet loss for varying weight factor ω .

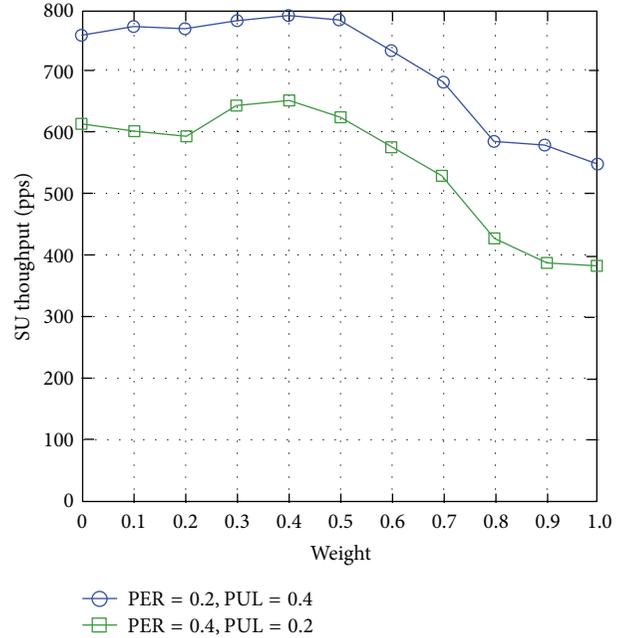


FIGURE 9: SU throughput for varying weight factor ω .

SUs' network performance. In addition, with respect to PUL and PER, Figure 7 shows that PER has greater effects on SUs' network performance compared to PUL because PER has a direct effect on the packet length and a SU's packet can be transmitted successfully during a PU-SU packet collision. With a higher value of PER (or a noisier channel), SU network congestion increases due to increasing number of SU packet retransmissions leading to higher SU end-to-end delay.

4.3.3. SU Packet Loss. Figure 8 shows that the SU packet loss increases with the weight factor ω of WCRQ-routing. The explanations leading to this circumstance are similar to those found in the investigation of SU end-to-end delay (see Section 4.3.2). Interestingly, Figure 8 shows that when weight factor is $\omega \cong 0.5$, WCRQ-routing achieves the lowest packet loss. At $\omega = 0.5$, a SU node i gives equal consideration to both link quality $r_t^{i,j}$, which aims to avoid routes with higher PUs' activities in order to reduce SUs' backoff delays, and SU j 's queue length q_t^j , both of which reduce the overflow of SUs' queues and SUs' packet loss.

4.3.4. SU Throughput. Figure 9 shows that the SU throughput reduces with the weight factor ω of WCRQ-routing. The explanations leading to this circumstance are similar to those found in the investigation of SU end-to-end delay (see Section 4.3.2); and when weight factor is $\omega \cong 0.5$, WCRQ-routing achieves the highest throughput.

4.3.5. Section Summary. We summarize simulation outcomes for the effects of weight factor in reward representation as follows.

- (i) A higher value of weight factor ω improves the PUs' network performance; while a lower value of ω improves the SUs' network performance.
- (ii) A balanced weight factor $\omega \cong 0.5$ achieves the best-possible SUs' network performance, particularly lower SU packet loss and higher SU throughput.

5. Enhancement of Exploration Mechanism

Traditionally, during route selection, there are two types of actions, namely, exploitation and exploration. Exploitation selects the best-known route $a_t^i = \operatorname{argmin}_{a \in A} Q_t^i(s_t^i, a)$, which has the lowest cost, in order to improve network performance. Exploration selects a random route $a_t^i \in A$ in order to improve knowledge, specifically, the estimation of Q -values for various routes. Two traditional exploration schemes are ϵ -greedy and softmax [5], and these schemes have been applied to regulate the exploration probability. A well-balanced tradeoff between exploitation and exploration helps to maximize network performance as time goes by. There have been some limited efforts to investigate this tradeoff in wireless networks; and this investigation applies to routing in CRNs. We present an overview of exploration probability and the traditional (called ϵ -greedy and softmax) and our proposed (called dynamic softmax) exploration approaches to dynamically regulate the exploration probability, as well as simulation results for the four performance metrics, in the rest of this section.

5.1. An Overview of Exploitation and Exploration. While route exploitation may seem to improve network performance as it forwards SUs' packets using the best-known route, it may cause network congestion and subsequently degrade SUs' network performance. On the other hand, route

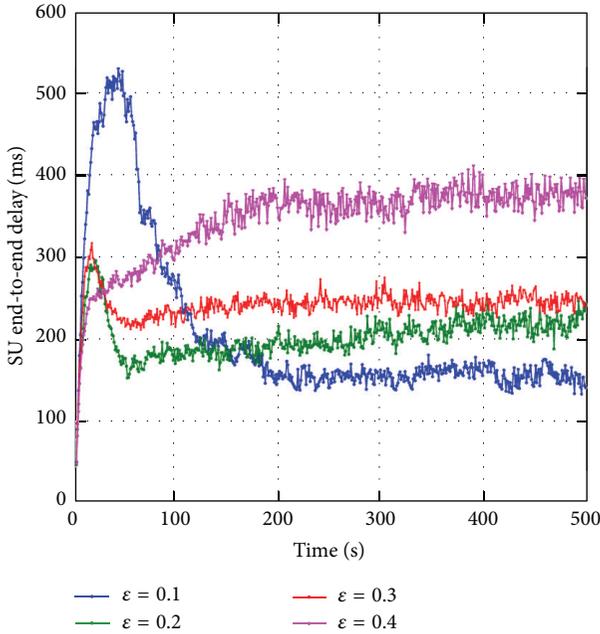


FIGURE 10: SU end-to-end delay for varying exploration probability ε .

exploration reduces traffic load on the best-known route, and so it may increase the convergence rate to an optimal route in a dynamic and unpredictable operating environment. However, if the exploration probability is high, forwarding the SUs' packets along nonoptimal routes may degrade SUs' network performance causing higher end-to-end delay and packet loss, as well as lower throughput.

We present simulation results of a preliminary investigation to show the effects of exploration probability ε on the convergence rate of SUs' network performance, particularly SU end-to-end delay. Note that more details on simulation setup and parameters are presented in later section. Using ε -greedy, a SU explores with a small probability ε and exploits with probability $1 - \varepsilon$. Convergence rate is the time duration for a SU node to find an optimal or near-optimal route, which provides a stable and enhanced SUs' network performance. Figure 10 shows that, as the exploration probability increases, the convergence rate increases (e.g., from approximately 200 seconds using $\varepsilon = 0.1$ to 100 seconds using $\varepsilon = 0.2$). The average SU end-to-end delay increases with the exploration probability ε . Additionally, higher exploration probability causes instability of SUs' network performance (e.g., SU end-to-end delay has higher fluctuations using $\varepsilon = 0.4$). Also, a peak is observed when exploration probability is low using $\varepsilon = 0.1$ because a SU may exploit nonoptimal routes at most of the time since the Q-values are initialized to 0 values which encourages exploration.

This initial investigation motivates us to achieve a balanced tradeoff between exploitation and exploration in route selection.

5.2. Exploration Approaches. We present two traditional exploration approaches (i.e., ε -greedy and softmax) and a

variant of the exploration approach (i.e., dynamic softmax) in this section.

5.2.1. The ε -Greedy Approach. The ε -greedy approach performs exploration with a small probability ε (e.g., $\varepsilon = 0.1$) and exploitation with probability $1 - \varepsilon$ [5]. A drawback is that, during exploration, it selects nonoptimal routes with below-average performance in a random manner with equal probability, and if these routes are chosen most of the times, the SUs' network performance may degrade.

5.2.2. The Softmax Approach. The softmax approach chooses an exploration action based on Q-values [5]. Specifically, routes with lower Q-values (i.e., lower cost) are likely to be explored than routes with higher Q-values (i.e., higher cost), and this addresses the drawback of ε -greedy; specifically, it minimizes the detrimental effects of exploration to SUs' network performance while exploring nonoptimal routes.

Using softmax (or the Boltzmann distribution), SU node i chooses its next-hop SU neighbor node $a_t^i \in A$ with the following probability:

$$P(s_t^i, a_t^i) = \frac{e^{-Q_t^i(s_t^i, a_t^i)/M}}{\sum_{a \in A^i} e^{-Q_t^i(s_t^i, a)/M}}, \quad (3)$$

where A^i represents a set of SU node i 's neighbor nodes and M is the temperature that determines the level of exploration. Higher M value indicates higher possibility of exploration, whereas lower M value indicates higher possibility of exploitation.

5.2.3. The Dynamic Softmax Approach. In ε -greedy and softmax, the exploration probability is predefined; specifically, the exploration probability ε and temperature M are predefined. There are two shortcomings. Firstly, while the optimal ε and M values are dependent on the dynamicity and unpredictability of the network conditions, it may not be feasible to determine these values on the fly. Secondly, routing in CRNs involves a number of SU nodes, and it may not be feasible to determine the optimal ε and M values for each SU node, each of which may operate in distinctive channels.

We propose a simple and pragmatic exploration approach called dynamic softmax, which is based on the traditional softmax approach. Dynamic softmax regulates the exploration temperature M (see (3)) of a SU based on the dynamicity and unpredictability of the network conditions. In [18], a similar approach to regulate the exploration probability of a nontraditional exploration approach is applied in wireless sensor networks.

In dynamic softmax, temperature M is increased and decreased by a constant factor f based on the network conditions, so that more exploration is performed only when necessary (see Algorithm 1). Higher values of f may increase the convergence rate at the expense of higher fluctuations in SUs' network performance, whereas lower values of f may reduce the convergence rate; however, it achieves lower fluctuations (or higher stability) of SUs' network performance.

```

initialize  $M = [M_{\min}, M_{\max}]$ 
while (updating a new Q-value  $Q_{t+1}^i$ )
  if ( $Q_{t+1}^i > (Q_t^i + \vartheta)$  &&  $M < M_{\max}$ ) then  $M+ = f$ 
  else if ( $Q_{t+1}^i < (Q_t^i - \vartheta)$ ) then  $M = M$ 
  else if ( $(Q_t^i - \vartheta) \leq Q_{t+1}^i \leq (Q_t^i + \vartheta)$  &&  $M > M_{\min}$ ) then  $M- = f$ 
  end if
end

```

ALGORITHM 1: Dynamic softmax algorithm at SU node i .

TABLE 5: Simulation parameters and values for investigating the exploration approaches.

Category	Parameter	Value
SU	Traditional ε -greedy exploration probability, ε	{0.07, 0.14}
	Traditional softmax exploration temperature, M	{0.04, 0.05}
	Initial dynamic softmax temperature, M	0.05
	Dynamic softmax adjustment factor, f	0.01
	Dynamic softmax temperature range, $[M_{\min}, M_{\max}]$	[0.01, 0.1]
	Dynamic softmax Q-value threshold, ϑ	0.1
PU	Standard deviation of PUL, σ_{PUL}	{0.2, 0.8}
Channel	Mean PER, μ_{PER}	0
	Standard deviation of PER, σ_{PER}	0

The adjustment of temperature M is based on the trend of the Q-value of a route as follows.

- (i) When the Q-value increases (i.e., higher routing cost), temperature M is increased in order to encourage exploration of other routes as this may indicate the emergence of PUs' activities which have degraded the SUs' network performance.
- (ii) When the Q-value decreases (i.e., lower routing cost), temperature M is left unchanged as this may indicate a forthcoming convergence to an optimal route which provides greater stability to SUs' network performance.
- (iii) When the change of Q-value is less than a threshold ϑ , the temperature M is decreased in order to encourage exploitation as this indicates that there has been convergence to an optimal route which provides a stable network performance.

5.3. Comparison of ε -Greedy, Softmax, Dynamic Softmax, and Exploitation-Only Approaches. This section investigates the effects of various exploration approaches applied to WCRQ-routing on network performance. We compare the network performance achieved by WCRQ-routing using two traditional exploration approaches, namely, ε -greedy and softmax, an exploitation-only approach and a variant of the exploration approach which we propose, namely dynamic softmax. The exploitation-only approach exploits at all times and does not explore.

While Table 3 presents the default simulation parameters and values, Table 5 presents the specific simulation parameters and values for this investigation. Generally speaking, we

simplify the simulation values in order to focus on the effects of various exploration approaches on network performance in the presence of the dynamicity of the channel availability with respect to PUL, so we assume a noiseless channel with $\mu_{\text{PER}} = 0$ and $\sigma_{\text{PER}} = 0$. The simulation results are shown for two conditions, in which the standard deviation of PUL $\sigma_{\text{PUL}} \in \{0.2, 0.8\}$ indicates low and high levels of unpredictability of the channel availability, respectively. The exploration metrics of the traditional approaches are $\varepsilon \in \{0.07, 0.14\}$ and temperature $M \in \{0.04, 0.05\}$, and these are chosen empirically; specifically, these are the average optimal values estimated by running extensive simulations using different values of exploration metrics under different levels of channel unpredictability σ_{PUL} . Hence, we compare the dynamic softmax approach with the best possible network performance achieved by ε -greedy and softmax. For dynamic softmax (see Algorithm 1), the temperature range is $[M_{\min}, M_{\max}] = [0.01, 0.1]$ because network performance degrades significantly when $M > 0.1$. The initial temperature is set to an average value of $M = 0.05$. Finally, the adjustment factor is set to $f = 0.01$ and the Q-value threshold is set to $\vartheta = 0.1$, and these values are chosen empirically.

We present simulation results for the four performance metrics in the rest of this section.

5.3.1. SUs' Interference to PUs. When the standard deviation of PUL is low $\sigma_{\text{PUL}} = 0.2$, most next-hop node (or link) and channel pairs have very similar PU mean arrival rate μ_{PUL} ; however, dynamic softmax outperforms the traditional exploration approaches, specifically up to 39% compared to softmax and up to 22% compared to ε -greedy while achieving very similar network performance with the exploitation-only

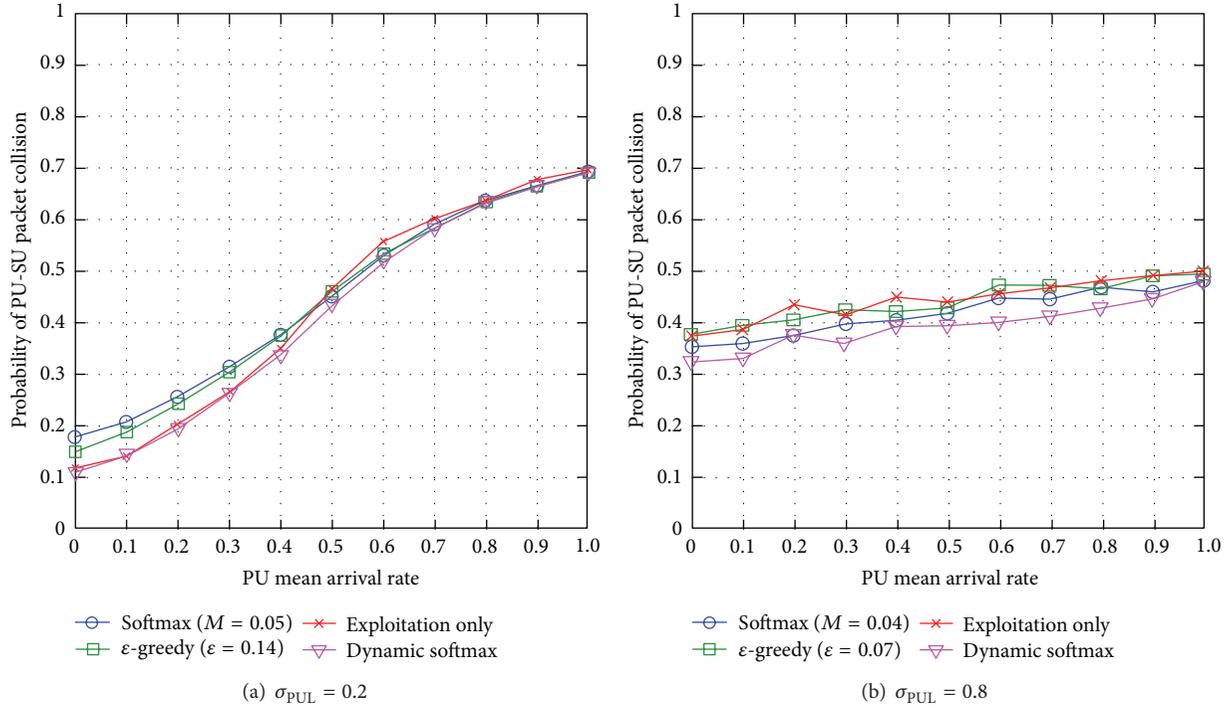


FIGURE 11: SUs' interference to PUs for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

approach (see Figure 11(a)). When the PU mean arrival rate μ_{PUL} becomes higher, all channels have high levels of PUs' activities and Q-values among the channels do not generally vary. So, there is lack of exploration and all approaches achieve very similar network performance. When $\sigma_{PUL} = 0.8$, the link and channel pairs have the greatest difference in the levels of PU mean arrival rate μ_{PUL} . Dynamic softmax outperforms the other exploration approaches, and the exploitation-only approach causes the highest SUs' interference to PUs (see Figure 11(b)). When the standard deviation of PUL σ_{PUL} becomes higher, all channels have different levels of PUs' activities, and Q-values among the channels generally vary. So, more explorations are necessary explaining why the exploitation-only approach causes the worst network performance with the highest SUs' interference to PUs. In general, softmax outperforms ϵ -greedy in most cases because softmax explores lesser below-average routes compared to ϵ -greedy.

Generally speaking, dynamic softmax achieves similar or better network performance in terms of SUs' interference to PUs compared to the traditional exploration approaches with optimal exploration metrics. This is because dynamic softmax learns the optimal exploration temperature dynamically based on the dynamicity and unpredictability levels of the network conditions. Additionally, the SUs' interference to PUs increases with the PU mean arrival rate μ_{PUL} and reduces with the standard deviation of PUL σ_{PUL} .

5.3.2. SU End-to-End Delay. When the standard deviation of PUL is low $\sigma_{PUL} = 0.2$, most link and channel pairs have very similar PU mean arrival rate μ_{PUL} . Dynamic softmax

outperforms the traditional exploration approaches, and up to 52% compared to exploitation-only which incurs the highest end-to-end delay (see Figure 12(a)). The exploitation-only approach exploits the best-possible route at all times and there is lack of load balancing among routes causing network congestion and increased SU end-to-end delay. Note that when there are low levels of PU mean arrival rate $\mu_{PUL} = [0.0, 0.4]$, softmax incurs the highest end-to-end delay because it performs the most unnecessary explorations. When $\sigma_{PUL} = 0.8$, the link and channel pairs have great difference in the levels of PU mean arrival rate μ_{PUL} . Dynamic softmax outperforms the other exploration approaches (see Figure 12(b)). However, when there are low levels of PU mean arrival rate $\mu_{PUL} = [0.0, 0.5]$, dynamic softmax incurs higher SU end-to-end delay because it performs more unnecessary explorations as a result of greater variations in Q-values, while ϵ -greedy explores routes randomly which increases load-balancing, and softmax chooses more above-average routes which increases network congestion.

Generally speaking, dynamic softmax reduces SU end-to-end delay compared to the traditional exploration approaches with optimal exploration metrics. Additionally, the SUs' interference to PUs increases with the PU mean arrival rate μ_{PUL} and reduces with the standard deviation of PUL σ_{PUL} .

5.3.3. SU Packet Loss. When the standard deviation of PUL is low $\sigma_{PUL} = 0.2$, dynamic softmax either achieves similar network performance or outperforms the traditional exploration approaches, specifically up to 57% compared to the exploitation-only approach which incurs the highest packet loss (see Figure 13(a)). The exploitation-only approach

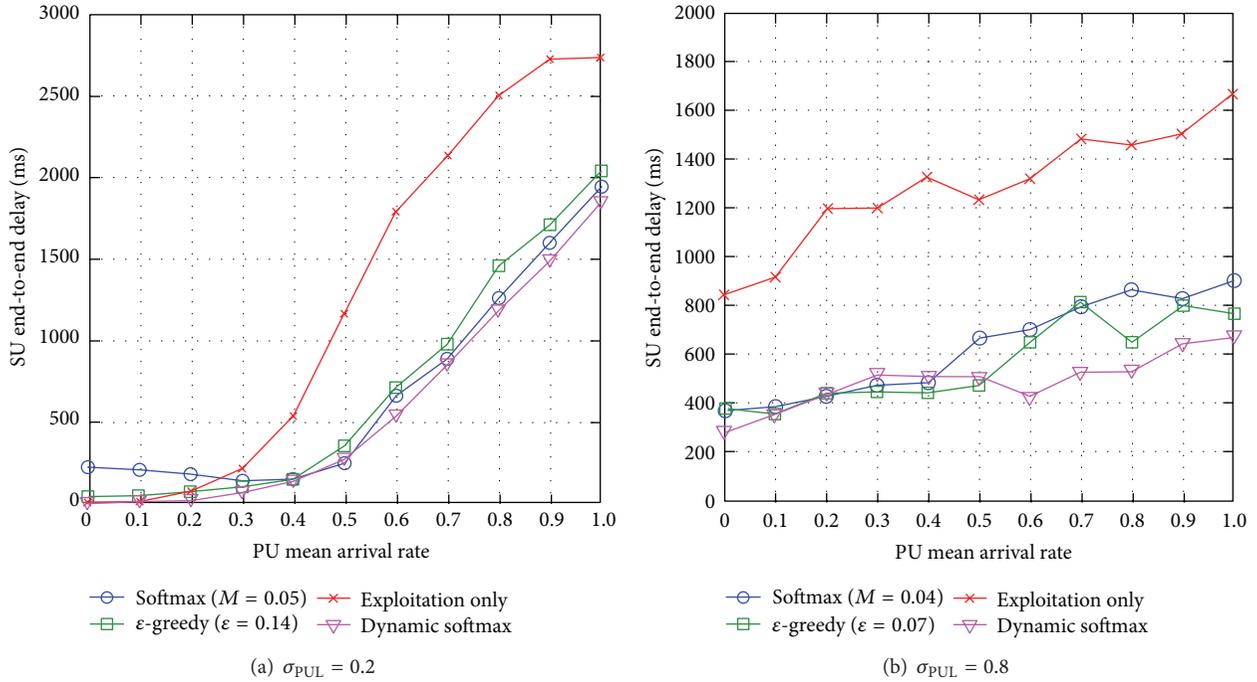


FIGURE 12: SU end-to-end delay for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

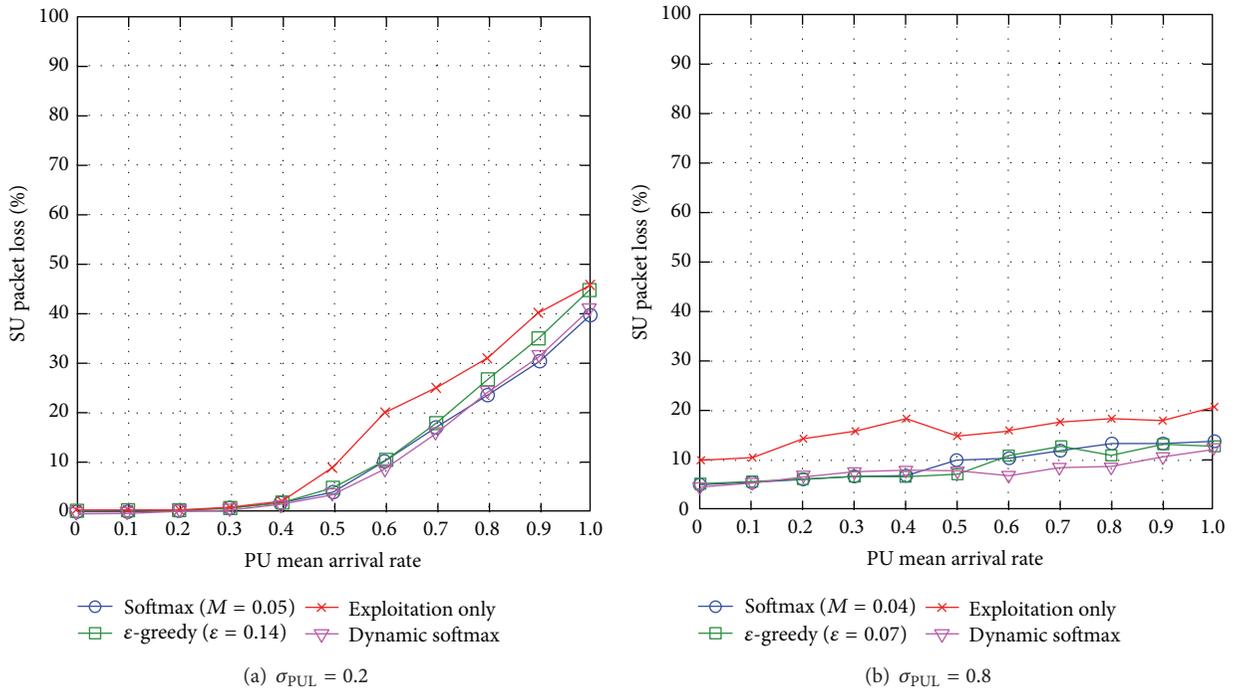


FIGURE 13: SU packet loss for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

exploits the best-possible route at all times and there is lack of load balancing among routes causing network congestion and increased SU packet loss. When $\sigma_{PUL} = 0.8$, dynamic softmax outperforms the other exploration approaches (see

Figure 13(b)). The ϵ -greedy approach explores routes randomly which increases load-balancing, and so it outperforms dynamic softmax at times, while softmax chooses more above-average routes causing greater network congestions.

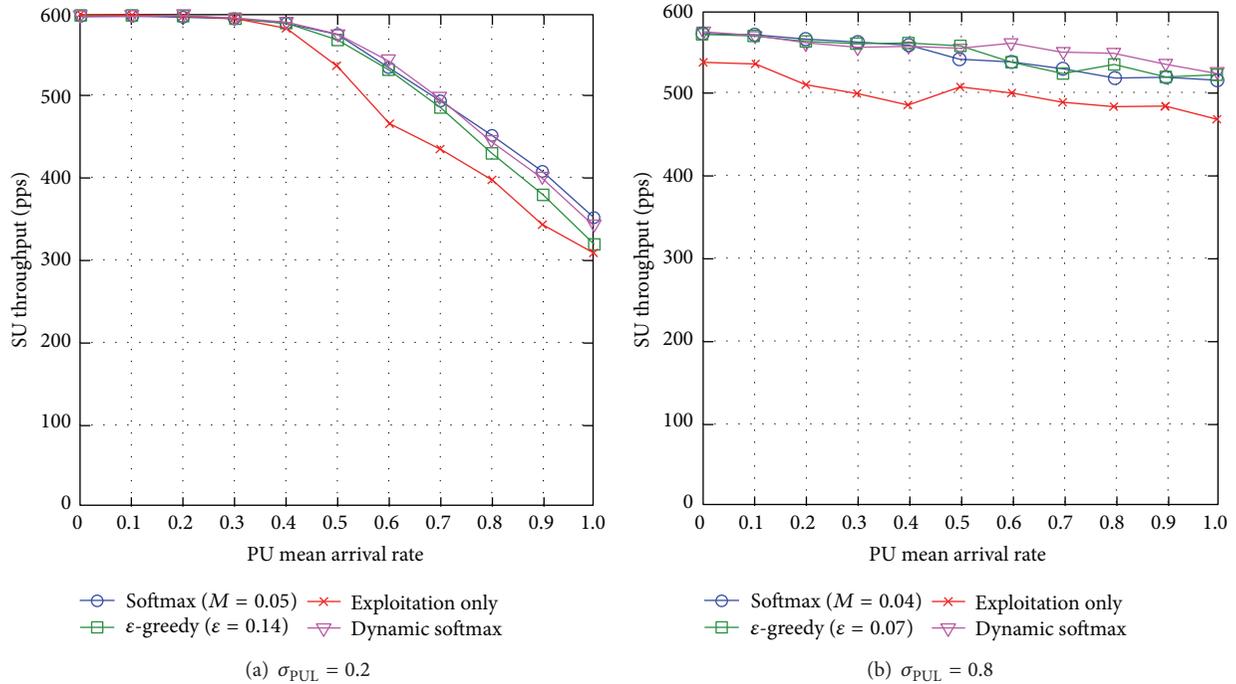


FIGURE 14: SU throughput for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

Generally speaking, dynamic softmax reduces SU packet loss compared to the traditional exploration approaches with optimal exploration metrics. Additionally, the SU packet loss increases with the PU mean arrival rate μ_{PUL} and reduces with the standard deviation of PUL σ_{PUL} .

5.3.4. SU Throughput. When the standard deviation of PUL is low $\sigma_{PUL} = 0.2$, dynamic softmax either achieves similar network performance or outperforms the traditional exploration approaches, specifically, up to 17% compared to the exploitation-only approach which incurs the highest packet loss (see Figure 14(a)). The exploitation-only approach exploits the best-possible route at all times and there is lack of load balancing among routes causing network congestion and reduced SU throughput. When $\sigma_{PUL} = 0.8$, similar trends are observed in network scenario of $\sigma_{PUL} = 0.8$ in the investigation of SU end-to-end delay (see Section 5.3.3).

Generally speaking, dynamic softmax increases SU throughput compared to the traditional exploration approaches with optimal exploration metrics. Additionally, the SU throughput reduces with the PU mean arrival rate μ_{PUL} and the standard deviation of PUL σ_{PUL} .

5.3.5. Section Summary. We summarize simulation outcomes for the comparison of ϵ -greedy, softmax, dynamic softmax, and exploitation-only approaches as follows.

- (i) Network performance degrades as the dynamicity of PUs' activities μ_{PUL} increases.
- (ii) Exploration approaches (i.e., ϵ -greedy, softmax, and dynamic softmax) achieve load-balancing among the available routes, and hence they improve SUs' (and

PUs' in some cases) network performance compared to the exploitation-only approach.

- (iii) Traditional exploration approaches, namely ϵ -greedy and softmax, outperform each other under different network conditions. For instance, when the unpredictability of the channel availability σ_{PUL} is low, softmax achieves better SUs' network performance than ϵ -greedy; however, when the unpredictability level of the channel availability σ_{PUL} becomes higher, ϵ -greedy achieves better SUs' network performance than softmax due to greater load balancing among routes, so there is lesser network congestion. Hence, neither ϵ -greedy nor softmax achieves the best-possible network performance under all network conditions.
- (iv) WCRQ-routing and CRQ-routing minimize SUs' interference to PUs with respect to PUL in the presence of dynamicity and unpredictability of the channel availability.
- (v) Dynamic softmax achieves the best-possible PUs' and SUs' network performances in most cases compared to other approaches because it learns the best-possible exploration temperature dynamically based on the dynamicity and unpredictability levels of the channel availability.

6. Enhancement of Learning Rate Adjustment

The learning rate α , which is used to regulate the speed of convergence to the optimal or near-optimal action, is another important parameter. A suitable learning rate α

value is essential. There have been some limited efforts to investigate the learning rate α in the domain of routing in wireless networks [19], and this investigation applies to routing in CRNs. We present an overview of learning rate, the traditional (called the Win-or-Learn-Fast Policy Hill Climbing approach or win-lose [6]) and our proposed (called the counterapproach) learning rate adjustment approaches to dynamically regulate the learning rate, as well as the simulation results for the four performance metrics, in the rest of this section.

6.1. An Overview of Learning Rate. Higher learning rate $0 \leq \alpha \leq 1$ indicates higher speed of learning and convergence rate, and it is more responsive to the dynamicity of the operating environment. Higher learning rate may cause fluctuation in Q-value because the Q-value is now more dependent on its recent estimates, which may be unstable, rather than its previous experience. On the other hand, lower learning rate may cause very low convergence rate because the Q-value is now more dependent on its previous experience rather than its recent estimates.

We present simulation results of a preliminary investigation to show the effects of learning rate α on the SUs' network performance, particularly SU end-to-end delay. Note that more details on simulation setup and parameters are presented in later section. Figure 15 shows that neither the lowest (i.e., $\alpha = 0.0001$) nor the highest (i.e., $\alpha = 1$) learning rate achieves the lowest SU end-to-end delay, and learning rate $\alpha = 0.0016$ achieves the lowest SU end-to-end delay. Hence, too low learning rate causes a SU to learn about the available routes slowly and so it does not adapt well to the changes in the PUs' mean arrival rate, while too high a learning rate causes a SU to learn very fast and so it changes its route more frequently causing the SU end-to-end delay to increase.

Hence, a suitable learning rate α value is essential to provide network performance enhancement. Traditionally, the learning rate is a predefined and fixed value which is not adaptive to the dynamicity and unpredictability levels of the operating environment, and so it does not provide the best network performance in a dynamic and unpredictable operating environment.

This initial investigation motivates us to search for a suitable learning rate.

6.2. Learning Rate Adjustment Approaches. We present a traditional exploration approach (i.e., the win-lose approach) and a variant of the exploration approach (i.e., the counterapproach), as well as a baseline approach (i.e., the random approach), in this section.

6.2.1. The Win-Lose Approach. In [20], a stochastic learning algorithm, namely, win-or-learn-fast policy hill climbing (or win-lose for simplicity) [6] is applied to regulate the learning rate dynamically based on the dynamicity of the operating environment in a wireless network. Given that higher Q-values provide network performance enhancement, the algorithm defines *winning* and *losing* as receiving higher

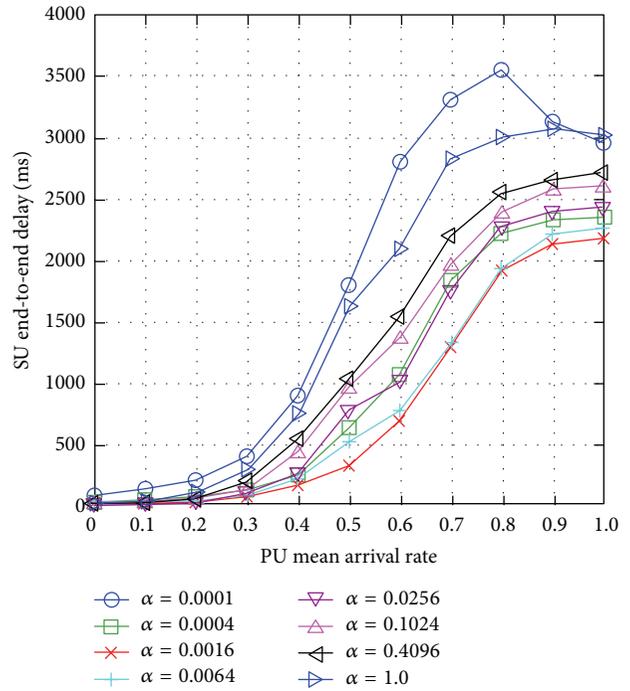


FIGURE 15: SU end-to-end delay for varying learning rate α .

and lower Q-values than its current Q-value, which is its expectation, respectively. When the algorithm is winning, the learning rate is set to a lower value, and vice-versa. The reason is that when a SU wins, it must be cautious in changing its routing policy and more time must be given to other SUs to adjust their own policies in favor of this winning. On the other hand, when a SU node loses, it must adapt faster to changes in the operating environment because its network performance (or rewards) is lower than its expectation.

6.2.2. The Counterapproach. The traditional win-lose approach (see Section 6.2.1) regulates the learning rate for each update of Q-value. Nevertheless, in a highly dynamic and unpredictable operating environment, Q-values may vary greatly and this causes frequent changes to learning rate. As a consequence, there are frequent changes to routing decision causing higher fluctuation in SUs' network performance.

The proposed counterapproach addresses the aforementioned issue by regulating the learning rate based on the historical Q-values. Specifically, for each Q-value (or state-action pair), it keeps track of a counter for winning c_t^p and another counter for losing c_t^n (see Algorithm 2). Subsequently, it calculates a ratio $o_t^n = c_t^p / c_t^n$, which represents the number of winning to the number of losing for a Q-value. When the ratio is $o_t^n > 1$, the learning rate is set to a lower value because this indicates a winning event, while the learning rate is set to a higher value when the ratio is $o_t^n < 1$ because this indicates a losing event, and when the ratio is $o_t^n = 1$, the learning rate is unchanged as it indicates a stable network performance. Similarly, when the newly received Q-value is similar to the previous value, the counter is not

```

initialize  $\alpha c_{t+1}^p \leftarrow [\alpha_{\min}, \alpha_{\max}]$ 
while (updating a new  $Q$ -value  $Q_{t+1}^i$ )
  if ( $Q_{t+1}^i < Q_t^i$ ) then  $c_{t+1}^p \leftarrow +$ 
  else  $c_{t+1}^n \leftarrow +$ 
  end if
  compute  $o_{t+1}^n = c_{t+1}^p / c_{t+1}^n$ 
  if ( $o_{t+1}^n < 1 \ \&\& \ \alpha < \alpha_{\max}$ ) then  $\alpha \leftarrow + f$ 
  else if ( $o_{t+1}^n > 1 \ \&\& \ \alpha > \alpha_{\min}$ ) then  $\alpha \leftarrow - f$ 
  end if
end

```

ALGORITHM 2: Counterapproach algorithm at SU node i .

updated. Note that we assume the learning rate α is increased and decreased by a small constant factor f in order to avoid fluctuations in SUs' network performance.

6.2.3. The Random Approach. The random approach, which does not apply any learning mechanism, regulates the learning rate in a round robin fashion without considering the dynamicity and unpredictability of the operating environment. This scheme serves as a baseline for comparison with win-lose and the counterapproach, and more importantly, it is used to show the effects of nonoptimal learning rates on network performance. This approach regulates the learning rate whenever it loses. Specifically, an agent decreases its learning rate α until the minimum learning rate limit α_{\min} is reached, and then it increases the learning rate α again until the maximum learning rate limit α_{\max} is reached, and this continues to loop in between the minimum α_{\min} and the maximum α_{\max} limits.

6.3. Comparison of Win-Lose and Counter and Random Approaches. This section investigates the effects of various learning rate adjustment approaches applied to WCRQ-routing on network performance. We compare the network performance achieved by WCRQ-routing using a traditional learning rate adjustment approach (i.e., win-lose), a variant of the win-lose (i.e., the counterapproach), a baseline (i.e., the random approach), and another baseline that provides the best empirical learning approach (which we may conveniently call *best*). The best empirical approach provides the best possible network performance, and the learning rate α is obtained by running extensive simulations under different levels of channel unpredictability.

While Table 3 presents the default simulation parameters and values, Table 6 presents the specific simulation parameters and values for this investigation. Generally speaking, we simplify the simulation values in order to focus on the effects of various learning rate adjustment approaches on network performance in the presence of the dynamicity of the channel availability with respect to PUL, so we assume a noiseless channel with $\mu_{\text{PER}} = 0$ and $\sigma_{\text{PER}} = 0$. The simulation results are shown for two conditions, in which the standard deviation

TABLE 6: Simulation parameters and values for investigating the learning rate adjustment approaches.

Category	Parameter	Value
SU	Best empirical learning rate, α	{0.0064, 0.004}
	Learning rate adjustment factor, f	0.001
	Learning rate range, $[\alpha_{\min}, \alpha_{\max}]$	[0.001, 0.1]
PU	Standard deviation of PUL, σ_{PUL}	{0.2, 0.8}
Channel	Mean PER, μ_{PER}	0
	Standard deviation of PER, σ_{PER}	0

of PUL $\sigma_{\text{PUL}} \in \{0.2, 0.8\}$ indicates low and high levels of the unpredictability of channel availability, respectively. The best empirical learning rate is set to $\alpha \in \{0.0064, 0.004\}$, which is estimated by running extensive simulations. The range of the learning rate is $[\alpha_{\min}, \alpha_{\max}] = [0.001, 0.1]$ because network performance degrades significantly when $\alpha > 0.1$. Finally, the learning rate adjustment factor is $f = 0.001$, which is estimated by running extensive simulations with the objective of enhancing SUs' network performance.

We present simulation results for the four performance metrics in the rest of this section.

6.3.1. SUs' Interference to PUs. When the standard deviation of PUL is low $\sigma_{\text{PUL}} = 0.2$, most next-hop node (or link) and channel pairs have very similar PU mean arrival rate μ_{PUL} . The effect of learning rate on network performance is minimal and so all the learning rate adjustment approaches achieve almost similar network performance (see Figure 16(a)). When $\sigma_{\text{PUL}} = 0.8$, the link and channel pairs have great difference in the levels of PU mean arrival rate μ_{PUL} , and the Q -values among the channels generally vary greatly. The counterapproach achieves almost similar SUs' interference to PUs compared to the best empirical approach and the lowest interference level compared to the win-lose and random approaches (see Figure 16(b)). This is because when σ_{PUL} is high, the counterapproach chooses a suitable learning rate that reduces fluctuations in Q -values while making routing decisions, while the win-lose and random approaches adjust the SU learning rate very fast resulting

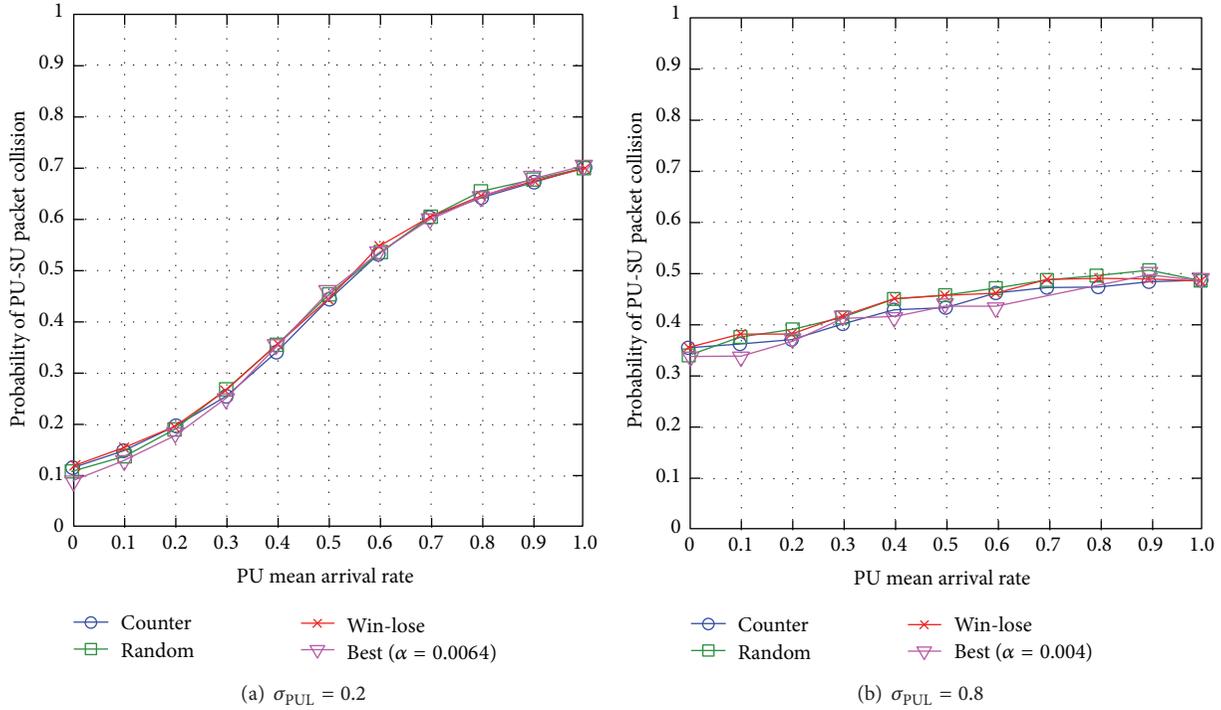


FIGURE 16: SUs’ interference to PUs for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

in more frequent changes to SU route decision, and hence higher SUs’ interference to PUs.

Generally speaking, the counterapproach reduces SUs’ interference to PUs. Additionally, the SUs’ interference to PUs increases with the PU mean arrival rate μ_{PUL} and reduces with the standard deviation of PUL σ_{PUL} .

6.3.2. SU End-to-End Delay. When the standard deviation of PUL is low $\sigma_{PUL} = 0.2$, most link and channel pairs have very similar PU mean arrival rate μ_{PUL} . The counterapproach achieves almost similar SU end-to-end delay compared to the best empirical approach and the lowest SU end-to-end delay compared to the win-lose and random approaches, specifically up to 11% compared to win-lose which incurs the highest packet loss (see Figure 17(a)). The random approach has the highest SU end-to-end delay, which is up to 12% higher compared to the best empirical learning rate approach, and this shows the effects of non-optimal learning rate on SU end-to-end delay. When $\sigma_{PUL} = 0.8$, the link and channel pairs have great difference in the levels of PU mean arrival rate μ_{PUL} , and the Q-values among the channels generally vary greatly. So, the learning rate has greater effects on SU end-to-end delay as a SU needs a suitable learning rate to learn about the available routes while minimizing fluctuations in Q-values. Similarly, the counterapproach achieves almost similar SU end-to-end delay to PUs compared to the best empirical approach, and the lowest SU end-to-end delay compared to the win-lose and random approaches, specifically up to 21% lower compared to win-lose (see Figure 17(b)). This is because when σ_{PUL} is high, the counterapproach

chooses a suitable learning rate that reduces fluctuations in Q-values while making routing decisions, while the win-lose and random approaches adjust the SU learning rate very fast resulting in more frequent changes to SU route decision, and hence higher SU end-to-end delay. When PU mean arrival rate is $\mu_{PUL} = 1$, the win-lose and random approaches reduce SU end-to-end delay since faster adjustment of learning helps to achieve load balancing among the available routes.

Generally speaking, the counterapproach reduces SU end-to-end delay. Additionally, the SU end-to-end delay increases with the PU mean arrival rate μ_{PUL} and reduces with the standard deviation of PUL σ_{PUL} .

6.3.3. SU Packet Loss. When the standard deviation of PUL is low $\sigma_{PUL} = 0.2$, the effect of learning rate is minimal and so all the learning rate adjustment approaches achieve almost similar network performance. When the PU mean arrival rate is $\mu_{PUL} = [0.8, 1.0]$, the win-lose and random approaches achieve lower packet loss compared to other approaches, specifically up to 8% lower compared to the counterapproach (see Figure 18(a)). This is because the win-lose and random approaches adjust the SU learning rate faster than the counterapproach leading to more frequent changes to routes, and as the routes become more congested, this helps to achieve load balancing among the available routes and reduces SU packet loss. When $\sigma_{PUL} = 0.8$, the learning rate has slightly greater effects on SU packet loss as a SU needs a suitable learning rate to learn about the available routes while minimizing fluctuations in Q-values. Similarly, the counterapproach achieves almost similar SU

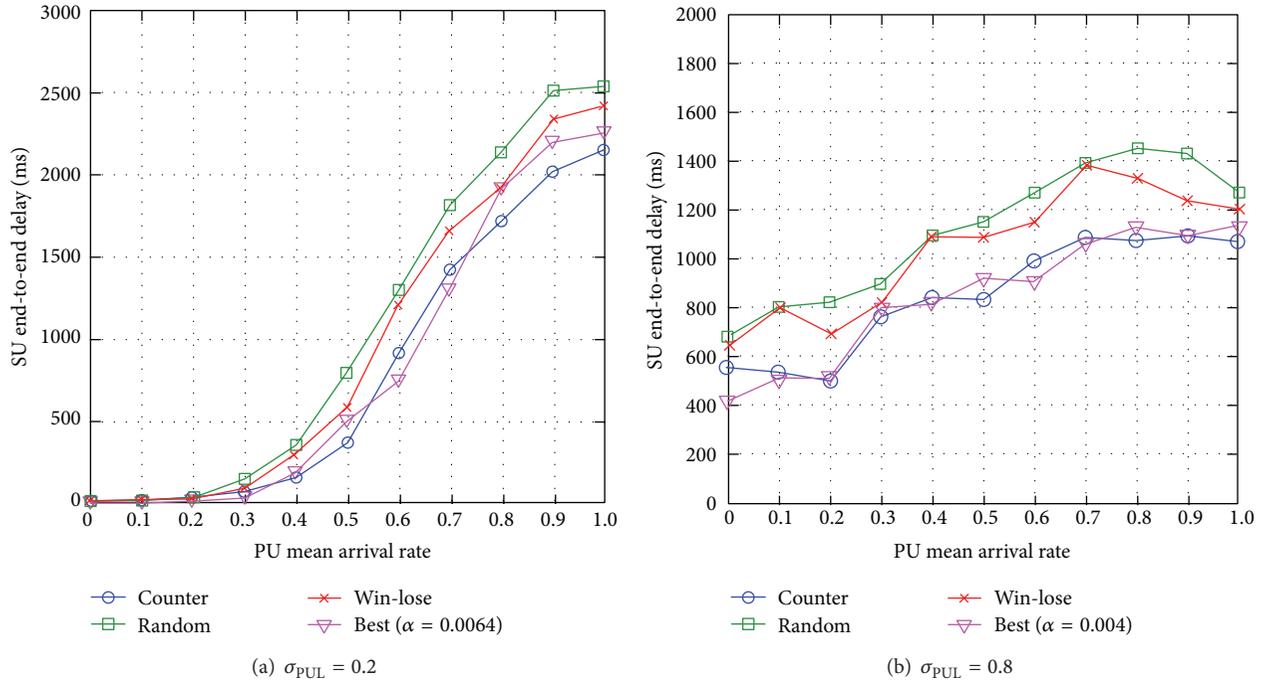


FIGURE 17: SU end-to-end delay for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

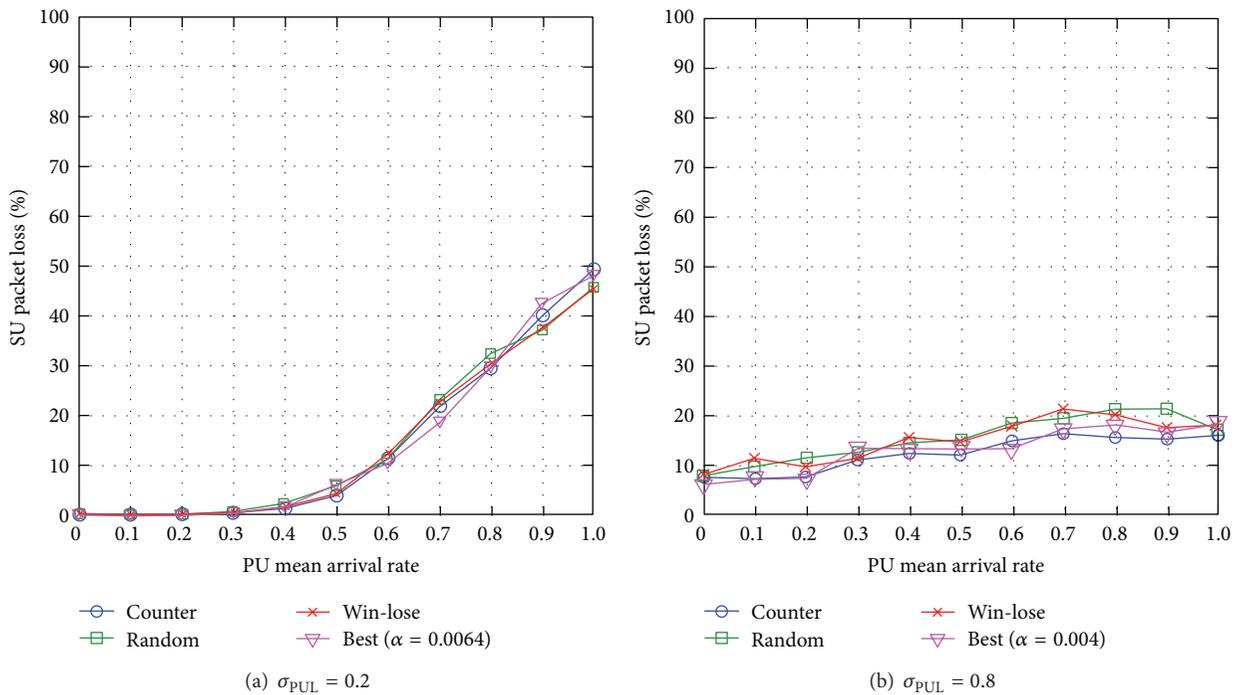


FIGURE 18: SU packet loss for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

packet loss compared to the best empirical approach, and the lowest SU packet loss compared to the win-lose and random approaches, specifically up to 23% lower compared to win-lose (see Figure 18(b)). This is because when σ_{PUL} is high, the

counterapproach chooses a suitable learning rate that reduces fluctuations in Q-values while making routing decisions.

Generally speaking, the counterapproach reduces SU packet loss. Additionally, the SU packet loss increases with

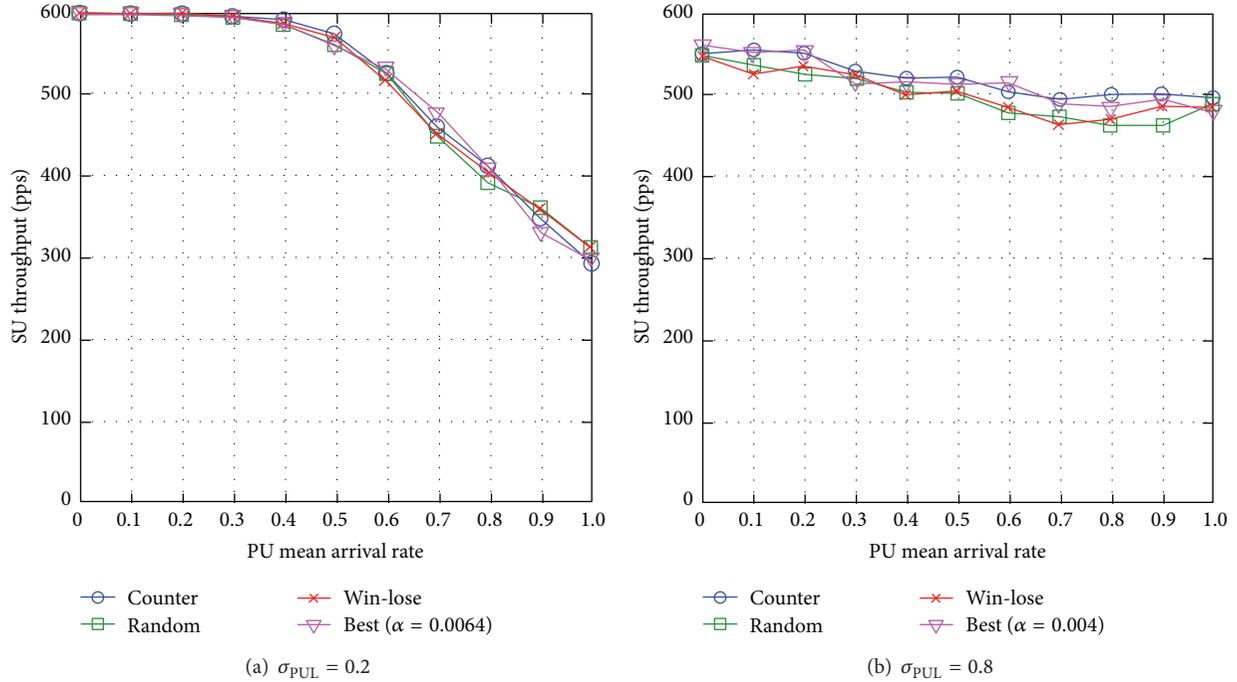


FIGURE 19: SU throughput for varying PU mean arrival rate μ_{PUL} for different levels of standard deviation of PUL σ_{PUL} .

the PU mean arrival rate μ_{PUL} and reduces with the standard deviation of PUL σ_{PUL} .

6.3.4. SU Throughput. When the standard deviation of PUL is low $\sigma_{PUL} = 0.2$, the effect of learning rate is minimal and so all the learning rate adjustment approaches achieve almost similar network performance (see Figure 19(a)), and the reasons are similar to those observed in the investigation of SU packet loss (see Section 6.3.3). When $\sigma_{PUL} = 0.8$, the counterapproach achieves almost similar SU throughput compared to the best empirical approach, and the highest SU throughput compared to the win-lose and random approaches, specifically up to 7% higher compared to win-lose (see Figure 19(b)), and the reasons are similar to those observed in the investigation of SU packet loss (see Section 6.3.3).

Generally speaking, the SU throughput decreases with the PU mean arrival rate μ_{PUL} and reduces with the standard deviation of PUL σ_{PUL} .

6.3.5. Section Summary. We summarize simulation outcomes for the comparison of win-lose, counter and random approaches as follows.

- (i) Network performance degrades as the dynamicity of PUs' activities μ_{PUL} increases.
- (ii) The effects of learning rate on SUs' network performance increase with the unpredictability of the PUs' activities σ_{PUL} .
- (iii) When the PU mean arrival rate is high (i.e., $\mu_{PUL} = [0.8, 1.0]$), where route congestion is higher, the win-lose and random approaches improve SUs' network

performance. This is because these approaches regulate the learning rate at higher speed, and this helps to achieve load balancing among the available routes, and subsequently enhances SUs' network performance.

- (iv) The counterapproach achieves almost similar SU network performance to the best empirical learning rate approach and outperforms the win-lose and random approaches in most cases, particularly when the unpredictability of the PUs' activities σ_{PUL} becomes higher. This is because the counterapproach chooses a suitable learning rate to reduce fluctuations in Q-values which start to vary at higher σ_{PUL} values.

7. Conclusions

Through simulation, this paper investigates the effects of reinforcement learning (RL) parameters on network performance for routing scheme in the presence of the dynamicity and unpredictability of the channel availability in cognitive radio ad hoc networks. We present WCRQ-routing that incorporates a weight factor ω in the reward representation to adjust the tradeoff between PUs' and SUs' network performances, as well as to further improve the overall network performance of SUs. Higher weight factor ω increases PUs' network performance, while lower ω increases SUs' network performance, and so a balanced value of ω helps to achieve the best SUs' network performance, particularly lower packet loss and higher throughput. The SUs' network performance can be further enhanced by regulating exploration probability and learning rate. We present a simple and pragmatic exploration approach called *dynamic softmax* to regulate the exploration

temperature. Dynamic softmax learns a near-optimal exploration temperature dynamically according to the dynamicity and unpredictability of the channel availability, and it achieves better network performance in most cases compared to the traditional exploration approaches, namely ϵ -greedy and softmax. We present a simple and pragmatic learning rate adjustment approach called the *control* approach to regulate the learning rate dynamically based on the historical Q-values. The counterapproach achieves better SUs' network performance compared to the traditional win-lose approach, and it achieves almost similar SUs' network performance to the best empirical learning rate approach by learning a near-optimal learning rate dynamically based on the dynamicity and unpredictability of the operating environment.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported by the Malaysian Ministry of Science, Technology and Innovation (MOSTI) under Science Fund 01-02-16-SF0027.

References

- [1] H. A. A. Al-Rawi and K.-L. A. Yau, "Routing in distributed cognitive radio networks: a survey," *Wireless Personal Communications*, vol. 69, no. 4, pp. 1983–2020, 2013.
- [2] H. A. A. Al-Rawi, M. A. Ng, and K.-L. A. Yau, "Application of reinforcement learning to routing in distributed wireless networks: a review," *Artificial Intelligence Review*, 2013.
- [3] P. Derakhshan-Barjoei, G. Dadashzadeh, F. Razzazi, and S. M. Razavizadeh, "Power and time slot allocation in cognitive relay networks using particle swarm optimization," *The Scientific World Journal*, vol. 2013, Article ID 424162, 9 pages, 2013.
- [4] J. Zhao and J. Yuan, "An improved centralized cognitive radio network spectrum allocation algorithm based on the allocation sequence," *International Journal of Distributed Sensor Networks*, vol. 2013, Article ID 875342, 13 pages, 2013.
- [5] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, Mass, USA, 1998.
- [6] M. Bowling and M. Veloso, "Multiagent learning using a variable learning rate," *Artificial Intelligence*, vol. 136, no. 2, pp. 215–250, 2002.
- [7] Q. Guan, F. R. Yu, S. Jiang, and G. Wei, "Prediction-based topology control and routing in cognitive radio mobile ad hoc networks," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 9, pp. 4443–4452, 2010.
- [8] K. R. Chowdhury and I. F. Akyildiz, "CRP: a routing protocol for cognitive radio ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 4, pp. 794–804, 2011.
- [9] Q. Zhu, Z. Yuan, J. B. Song, Z. Han, and T. Başar, "Dynamic interference minimization routing game for on-demand cognitive pilot channel," in *Proceedings of the 53rd IEEE Global Communications Conference (GLOBECOM '10)*, pp. 1–6, Miami, Fla, USA, December 2010.
- [10] M. Xie, W. Zhang, and K. Wong, "A geometric approach to improve spectrum efficiency for cognitive relay networks," *IEEE Transactions on Wireless Communications*, vol. 9, no. 1, pp. 268–281, 2010.
- [11] B. Xia, M. H. Wahab, Y. Yang, Z. Fan, and M. Sooriyabandara, "Reinforcement learning based spectrum-aware routing in multi-hop cognitive radio networks," in *Proceedings of the 4th International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM '09)*, pp. 1–5, Hannover, Germany, June 2009.
- [12] D. Oužeki and D. Jevtić, "Reinforcement learning as adaptive network routing of mobile agents," in *Proceedings of the 33rd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO '10)*, pp. 479–484, Opatija, Croatia, May 2010.
- [13] J. Dowling, E. Curran, R. Cunningham, and V. Cahill, "Using feedback in collaborative reinforcement learning to adaptively optimize MANET routing," *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, vol. 35, no. 3, pp. 360–372, 2005.
- [14] J. A. Boyan and M. L. Littman, "Packet routing in dynamically changing networks: a reinforcement learning approach," *Advances in Neural Information Processing Systems*, vol. 6, pp. 671–678, 1994.
- [15] Y. Zhang and M. Fromherz, "Constrained flooding: a robust and efficient routing framework for wireless sensor networks," in *Proceedings of 20th IEEE International Conference on Advanced Information Networking and Applications (AINA '20)*, IEEE, Vienna, Austria, 2006.
- [16] H. A. Al-Rawi, K.-L. A. Yau, H. Mohamad, N. Ramli, and W. Hashim, "A reinforcement learning-based routing scheme for cognitive radio ad hoc networks," in *Proceedings of the 7th IFIP Wireless and Mobile Networking Conference (WMNC '14)*, May 2014.
- [17] G. E. Box and M. E. Muller, "A note on the generation of random normal deviates," *The Annals of Mathematical Statistics (Institute of Mathematical Statistics)*, vol. 29, no. 2, pp. 610–611, 1958.
- [18] A. Förster and A. L. Murphys, "FROMS: feedback routing for optimizing multiple sinks in WSN with reinforcement learning," in *Proceedings of the International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP '07)*, pp. 371–376, IEEE, Melbourne, Australia, December 2007.
- [19] A. McAuley, K. Sinkar, L. Kant, C. Graff, and M. Patel, "Tuning of reinforcement learning parameters applied to OLSR using a cognitive network design tool," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '12)*, pp. 2786–2791, IEEE, Shanghai, China, April 2012.
- [20] P. Nurmi, "Reinforcement learning for routing in ad hoc networks," in *Proceedings of the 5th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt '07)*, pp. 1–8, Limassol, Cyprus, April 2007.

Research Article

Features Extraction of Flotation Froth Images and BP Neural Network Soft-Sensor Model of Concentrate Grade Optimized by Shuffled Cuckoo Searching Algorithm

Jie-sheng Wang,^{1,2} Shuang Han,¹ Na-na Shen,¹ and Shu-xia Li¹

¹ School of Electronic and Information Engineering, University of Science & Technology Liaoning, Anshan 114044, China

² National Financial Security and System Equipment Engineering Research Center, University of Science & Technology Liaoning, Anshan 114044, China

Correspondence should be addressed to Jie-sheng Wang; wang.jiesheng@126.com

Received 24 April 2014; Accepted 9 June 2014; Published 16 July 2014

Academic Editor: Xin-She Yang

Copyright © 2014 Jie-sheng Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

For meeting the forecasting target of key technology indicators in the flotation process, a BP neural network soft-sensor model based on features extraction of flotation froth images and optimized by shuffled cuckoo search algorithm is proposed. Based on the digital image processing technique, the color features in HSI color space, the visual features based on the gray level cooccurrence matrix, and the shape characteristics based on the geometric theory of flotation froth images are extracted, respectively, as the input variables of the proposed soft-sensor model. Then the isometric mapping method is used to reduce the input dimension, the network size, and learning time of BP neural network. Finally, a shuffled cuckoo search algorithm is adopted to optimize the BP neural network soft-sensor model. Simulation results show that the model has better generalization results and prediction accuracy.

1. Introduction

Flotation is known as froth flotation, and it is a physico-chemical reaction process. Flotation is the process which is based on the differences of the surface property of solid materials to separate useful minerals and gangue by means of the buoyancy of air bubbles from ore pulp by this method to improve the concentrate grade [1]. In the production process of flotation, concentrate grade and other economic and technical indicators are key control indicators of the production process. Process control indicators of domestic flotation process are mainly based on an experienced operator to observe the information (such as foam color, size, flow rate, and texture features) which is provided by the bubble state formed on the surface of the flotation tank and to adjust the flotation level and change agents system. Froth flotation method of manual observation has limitations of its space, time, and subjectivity; meanwhile this method cannot be organically combined with computer control systems to implement advanced control. Inference

estimate (soft sensor) technology can effectively solve the online estimation problems that the online measurement of the economic and technical indicators in the flotation process is difficult.

Digital image processing techniques are applied to froth feature extraction and key technical indicators of soft-sensor modeling in the flotation process by scholars at home and abroad and many achievements have been made [2–6]. Woodbum and many other scholars created a bubble dynamic model based on image processing through researching flotation foam structure and calculated the content of useful minerals in foam through this model [7]. Zhou et al. extracted color and size characteristics of the foam by using digital image processing method and established a recovery prediction model, which was proved to have good predictive results by analyzing foam and process image data [8]. Bartolacci et al. extracted visual characteristic parameter from the flotation RGB image by the use of MIA and GLCM methods, and using partial least squares method established predictive model of flotation foam concentrate

grade. The model is used in the case of an industrial mineral process and demonstrated the effectiveness of this method [9]. Zhou et al. extracted image feature such as bubbles' color, speed, and size as the model parameters and built predictive model using least squares support vector machine. The experimental results show that this method can effectively predict recovery [10]. Chun-hua et al. established a soft-sensor model of flotation pulp pH value based on sparse multicore least squares support vector machines through taking foam video image characteristics as auxiliary variables and using Schmidt orthogonalization theory reducing multicore matrix [11]. After that, they extracted image features by using color cooccurrence matrix (CCM), and this method was compared with the traditional GLCM extracting the image features method. Li et al. took the feature information of the image as the model input by extracting feature information of foam image, and using the sliding window updating and deviation compensation strategy, respectively, updated model parameters in real time and compensated output. This method achieved soft-sensor modeling of concentrate grade of flotation process [12]. Wang and Zhang proposed a kind of soft-sensor model of economic and technical index based on PCA and ANFIS, and combining PSO algorithm with LSM put forward a new learning process to optimize parameters of ANFIS [13]. Geng and Chai utilized least squares support vector machine to establish soft-sensor model of concentrate grade and tailing grade in the flotation process based on analyzing related influencing factors of concentrate grade and tailing grade of the flotation process technology indicators [14].

The established flotation process soft-sensor model above just used part feature information of the froth image in the flotation process, but much feature information of flotation froth image was not integrated, coordinated, and optimized. In this paper, a BP neural network soft-sensor model based on color feature parameters, visual feature parameters, and shape feature parameters of flotation froth images is proposed, and then a shuffled cuckoo search algorithm based on adaptive step is put forward to optimize BP neural network soft-sensor model parameters. Simulation results demonstrate the effectiveness of the proposed method. The paper is organized as follows. In Section 2, the technique flowchart of the flotation process is introduced. The feature extraction methods of flotation froth images are presented in Section 3. In Section 4, the soft-sensor modeling of flotation process is introduced. The simulation experiments and results analysis are introduced in details in Section 5. Finally, the conclusion illustrates the last part.

2. Technique Flowchart of Flotation Process

Flotation is a sorting mineral technology in the vapor-liquid-solid interface, which consists of roughing, selection, and scavenging process. Flotation is divided into positive flotation and reverse flotation. The so-called positive flotation is that the flotation froth above the flotation tank is concentrate, and the underflow slurry is tailing. Instead of reverse flotation process, the basic principle of positive flotation shows in

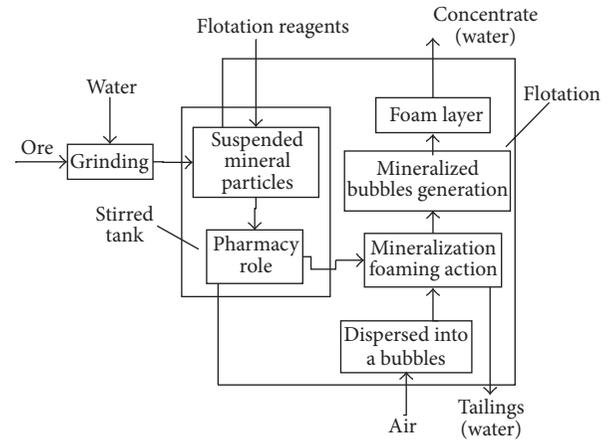


FIGURE 1: Basic technique diagram of flotation process.

Figure 1. Figure 2 shows a typical iron ore flotation process consisting of roughing, selection, and scavenging processes [15].

The input of the system is concentrate pulp under a fine sieve, which is the output of the early front beneficiation process, and the pulp density is about 38%, and concentrate grade is about 64%. The entrance pulp is pumped into the efficient stirred tank through a slurry pipeline. Meanwhile, flotation reagent which is according to a certain concentration ratio is pumped into efficient stirred tank by dosing pump, and pulp temperature reaches suitable temperature through heat treatment. Then the entrance pulp flows into the flotation tank, and if the dose is proper the flotation tank can output concentrate that its grade is 68.5%–69.5%. The output tailings of the flotation tank are fed into scavenger tank for further processing. A part of output of scavenger tank is pumped into an efficient stirred tank for reprocessing by the central mine return pump, and another part of output goes into the postflotation process, magnetic separator, dewatering tank, and other equipment for postprocessing. A part of results is discharged as the tailings, and another part is pumped into the efficient stirred tank for recycling.

Control objective of flotation process is to ensure that the concentrate grade and tailings recovery are controlled within a certain target range. So using the way of offline artificial test obtains grade value, and then, according to the grade value, operators adjust the flotation cell liquid level and the amount of flotation reagents. Because artificial test is 2 hours once; thus, when the process variables and boundary conditions of flotation process change, operators cannot adjust flotation cell liquid level and pharmaceutical dosage based on test values timely, which often results in that flotation concentrate grade and tailings recovery are too high or too low. So it has important industrial application values adopting soft-sensor modeling method to achieve real-time measurement of concentrate grade for optimal control of flotation process.

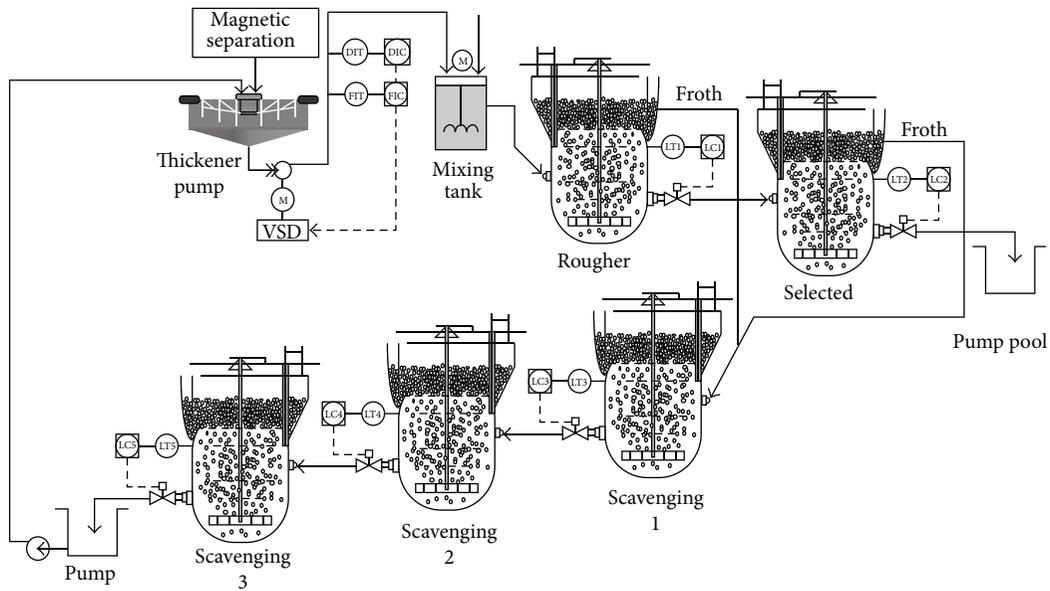


FIGURE 2: Technique flowchart of flotation process.

3. Feature Extraction of Flotation Froth Images

Bubble image contains important information which is related to the flotation process, and traditionally the flotation froth state is observed by experienced workers. Generally, flotation froth state can be described by bubble size, foam color (gray value), texture, and stability. However, due to the subjectivity of the operators, understanding of flotation froth state is different and bubble state cannot be observed carefully, which result in that flotation process control cannot reach the optimal state. Therefore, the use of digital image processing technology analyses and processes bubble image, which has important implications for predicting flotation performance indicators.

3.1. Digital Image Processing Technology. Digital image processing, also known as the computer image processing, refers to convert image signal into digital signal and use the computer to deal with the process; that is to say, it makes traditional analog image store as digital image by image sampling and quantification [16]. The digital image acquisition system comprises three basic units: the imaging system, the sampling system, and the quantizer. Sampling is a quantifying process of spatial coordinates actually, and quantization is a discrete process of the function value of the image. Sampling and quantizing system are collectively referred to as the digitization. Pixel is the basic element of a digital image, and each pixel has position coordinates with the integer row (high) and column (wide). In the sampling, if the number of horizontal pixels is m , the number of vertical pixels is n , and the total number of pixel is $m \times n$ pixels in the image. The number of different gray values is referred as gray level in the image. Generally an image has 256 gray levels. If the storage size of an image is $m \times n \times 8$, then the size of the

image data is got. A $m \times n$ digital image can be expressed as a matrix:

$$F = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,n-1) \\ f(1,0) & f(1,1) & \dots & f(1,n-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(m-1,0) & f(m-1,1) & \dots & f(m-1,n-1) \end{bmatrix} \quad (1)$$

Each value of matrix corresponds to a pixel of a digital image; digital image represented as matrix form has the advantage of analysis of bubble images, and thus we can obtain more information about the flotation process. In the computer, according to the number of colors and gray scale, images can be divided into four basic types: binary image, gray image, indexed image, and true color RGB image. True color RGB image refers to that each pixel value is divided into the R, G, and B, three primary color components, and each primary color component will directly determine the intensity of its color. RGB color model produces a variety of different colors by different degrees superposition of the three primary colors. This standard can cover all the colors the human eyes can perceive, and it is one of the widely used color system currently. RGB color space can be represented by a three-dimensional Cartesian coordinate system as shown in Figure 3.

Generally, flotation froth image is a true color RGB image; each pixel is specified by the three values in the image: red (R), green (G), and blue (B) of the three color components. This is represented by a three-dimensional array, namely, $m \times n \times 3$ array. Because the red, green, and blue components are, respectively, represented by 8-bit gray levels; that is to say, $(R, G, B) \in [0, \dots, 255]$, so the image can contain 2^{24} kinds of color theoretically. Since any color can be obtained by mixing red (R), green (G), and blue (B), the three primary colors,

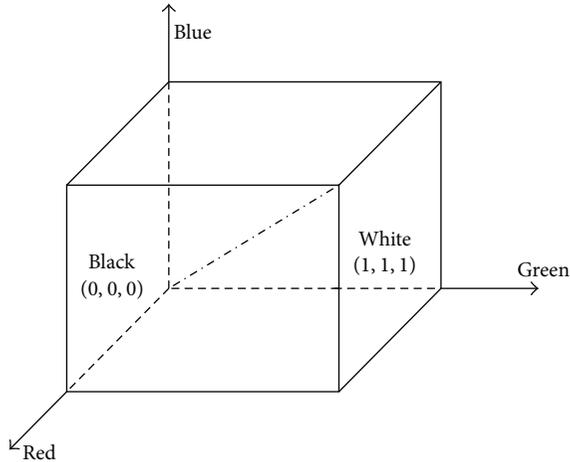


FIGURE 3: RGB color space model.

therefore the image can be represented by three-dimensional function of position coordinates:

$$f(x, y, z) = \{f_{\text{red}}(x, y, z), f_{\text{green}}(x, y, z), f_{\text{blue}}(x, y, z)\}, \quad (2)$$

where f represents the position point color of space coordinate (x, y, z) and f_{red} , f_{green} , and f_{blue} , respectively, represent component values of red, green, and blue of this position point. But a planar image is usually considered in study; since each point only includes two coordinate values on the plane, a planar image can be expressed by the two-dimensional function of the position coordinates:

$$f(x, y) = \{f_{\text{red}}(x, y), f_{\text{green}}(x, y), f_{\text{blue}}(x, y)\}, \quad (3)$$

where two-dimensional function $f(x, y)$ represents a pixel of an image, namely, each value in (1).

RGB color model is very simple, but since this model is uneven from the point of view of human perception and its three components have a larger correlation, we rarely apply RGB space model of images directly. HSI color space is based on the human visual system, and using hue, saturation (chroma), and brightness (intensity) describes color. RGB color space can be converted to the HSI space by transforming.

3.2. Image Feature Selection of Flotation Froth. Flotation froth image is got from a CCD camera mounted on a flotation cell. Then the computer image acquisition card converts the continuous analog signals into discrete digital signals, and discrete digital signals are sent into computer for flotation froth visual feature extraction. Typical flotation froth images are shown in Figure 4.

According to flotation technology and expertise, froth image will be divided into three categories as follows. (1) The size of bubbles is larger, and there are parts of large bubbles in the foam. The texture is lighter, but rough. The image complexity is small, and the color is hoary. The foam contains less SiO_2 , and the grade of ore concentrate is low.

(2) The bubble size is appropriate, uniform, and stable, the color is partial gray, the texture is finer, and image is more complex. At this time, flotation process is good, and iron concentrate grade meets the requirements. (3) The color of foam is darker, even partial black. The foam is finer, some foam is even difficult to distinguish, and the texture is very complex. At this time, the foam has a high SiO_2 content. Although iron concentrate grade of output is higher, the drug dosage is too big; therefore, this does not meet the economic requirements of companies.

In the flotation process, the foam layer and flotation performance indexes have important links [17, 18], such as color and froth size. It is important for predicting the performance indexes of the flotation process to extract feature information of flotation froth image. Feature extraction is a concept of computer vision and image processing. It refers to that using a computer extracts image information and determines whether each point of the image belongs to an image feature. The results of the feature extraction are to divide points of image into different subsets. These subsets are often isolated points, a continuous curve, or a continuous area. In the froth flotation process, flotation foam is discharged from flotation tank by the movement of a mechanical scraper. The moving speed of the scraper has a great influence on the mobility of the foam. So the measuring difficulty of the foam mobility is larger. Therefore, we can first consider the image color features and texture features. The shape is the outline of the object, and it does not change with the target color of images. Hence, the shape feature of images also has the advantages which the color and texture features do not have, and it is necessary to consider the shape feature of the image. Therefore, color feature, texture feature, and shape feature of froth flotation images are extracted in this paper.

3.3. Color Feature Extraction of Flotation Froth Images. Color is a very important visual feature for images, and it has certain stability. It is not sensitive for the changes of size, orientation, translation, and rotation of the image. The operator of flotation production process is also based on the closeness between flotation froth color and gray to determine the merits of the flotation process. System collected images are RGB true color images, which are indicated by red, green, and blue, but the three components often have close correlation. Because the sensitivity to brightness of the human visual is far stronger than the sensitivity to the color shades, the color information of Hue, Saturation and Intensity (HSI) model is similar to the human color visual perception. In order to dispose and identify color conveniently, the human visual system has often used HSI color space, which is more in accord with the human visual characteristic than RGB color space. In the image processing and computer vision, a lot of algorithms can be easily used in the HSI color space; meanwhile they can be treated separately and are independent of each other. Therefore, image analysis and processing workload can be greatly simplified in the HSI color space. HSI color space can be described using a cone-space model. Figure 5 is the HSI color space model.

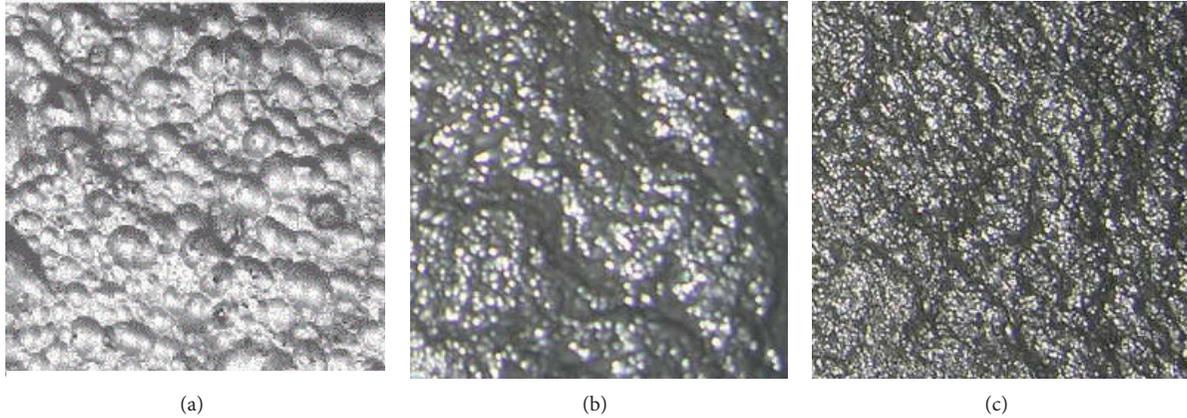


FIGURE 4: Typical iron ore flotation froth images.

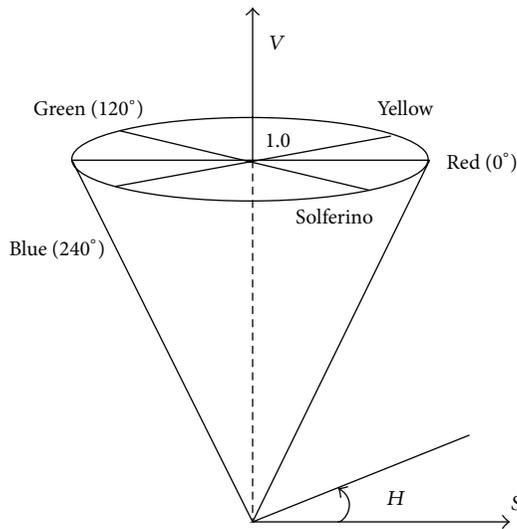


FIGURE 5: HSI color space model.

In HSI model, hue (H) can be measured from 0 to 360 degrees, which represents feels of human senses for different colors, such as red, green, and blue. It may also represent a range of colors, such as warm color and cool color; wherein the angle of pure red is 0, the angle of pure green is $2\pi/3$ and the angle of pure blue is $4\pi/3$. Saturation (S) is the distance between any point in the color space and I axis, which represents the depth of color or the degree of shade. Intensity (I) is mainly affected by the light source, which represents the light and dark degree of colors. In industrial applications, the range of S is $[0, 1]$, which changes from unsaturation to a full saturation (nonwhite); the range of I is $[0, 1]$, which corresponds to color from dark to bright. HSI color space and RGB color space are just different representations of the same physical quantity, so there is a conversion between them:

$$I = \frac{R + G + B}{3},$$

$$S = 1 - \frac{3}{R + G + B} [\min(R, G, B)],$$

$$H = \cos^{-1} \left[\frac{(R - G) + (R - B)}{2\sqrt{(R - G)^2 + (R - B)(G - B)}} \right]$$

$R \neq B \text{ or } G \neq B.$

(4)

Thus, the three characteristic parameters (hue, saturation, and brightness) extracted from images are as model inputs to predict the content of the concentrate grade.

3.4. Texture Feature Extraction of Flotation Froth Image. Image texture reflects the attribute of image itself. In general, texture is a pattern with a small shape and arranged regularly in a certain range of an image. It is an important characteristic for describing images. For the flotation industrial production process, flotation froth images appear to be more obvious texture features, and their texture statistical characteristics can reflect the flotation process conditions. Statistics method, structure method, and spectral method are three texture analysis methods. For different situations, we should adopt different analysis methods. When texture is subtle, for example, wood, meadows, forests, and so forth, statistical analysis method can be adopted. When texture is coarse and has a strong regularity, structural analysis method can be adopted; spectrum method refers to using the frequency characteristics of Fourier transform to describe the periodic texture images.

As can be seen in Figure 4, texture of flotation process froth images is relatively subtle, so we adopt statistical analysis method. In the statistical analysis method, the frequently used methods are histogram analysis method and GLCM method (GLCM). Although the histogram is relatively simple, intuitive, the histogram is a measure of similarity; that is to say, images may have different texture features even though they are similar. The texture is formed by the gray distribution recurring in the spatial position; therefore, there will be a certain gray relationship between two pixels

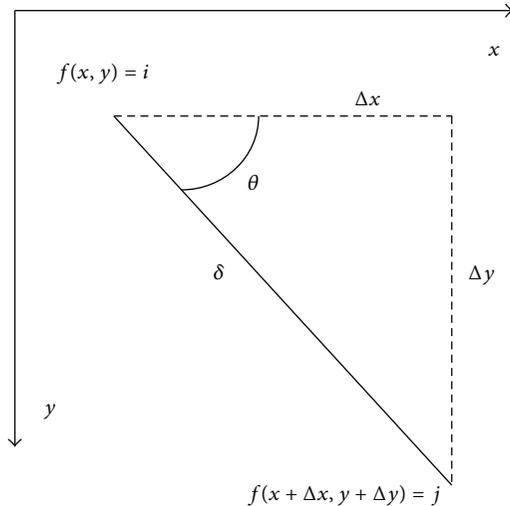


FIGURE 6: Grey-level cooccurrence matrix.

separated by a certain distance in the image space, namely, the gray spatial correlation properties in images. Gray level cooccurrence matrix (GLCM) is a common way describing the texture by studying the spatial correlation properties of gray, which is based on the second-order combination condition probability density function of estimation images. Cooccurrence matrix is defined using the joint probability density of pixels of two positions. It not only reflects the distribution characteristics of intensity but also reflects position distribution characteristics of pixels that are with the same or near intensity. It is the second-order statistical characteristics related image intensity changes, and it is also the basis for the definition of a set of texture features. Figure 6 is a GLCM schematic view, wherein i , j indicate the corresponding pixel gray values.

Supposing $f(x, y)$ is a two-dimensional digital image, GLCM means the simultaneous occurrence probability $P(i, j, \delta, \theta)$ of two pixels. They are the pixel with gray scale i from the image $f(x, y)$ and the pixel $(x + \Delta x, y + \Delta y)$ with gray scale j , declination θ and distance δ . The mathematical formula is expressed as:

$$\begin{aligned}
 &P(i, j, \delta, \theta) \\
 &= \{ [(x, y), (x + \Delta x, y + \Delta y)] \\
 &\quad | f(x, y) = i, f(x + \Delta x, y + \Delta y) = j; \\
 &\quad x = 0, 1, \dots, N_x - 1; y = 0, 1, \dots, N_y - 1 \}, \quad (5)
 \end{aligned}$$

where $i, j = 0, 1, \dots, L - 1$, x, y are the coordinates of pixels in the image, L is the image gray levels, and N_x, N_y are the number of rows and columns of the image, respectively. According to the above definition, the i th row and j th column elements of the composed gray cooccurrence matrix mean the appearing frequency of all pairs of pixels that with direction θ , separation δ , gray i and j , respectively.

In order to describe texture condition using the cooccurrence matrix intuitively, we do not directly use cooccurrence matrix generally but derive some parameters that can

reflect the matrix situation from the cooccurrence matrix. These parameters can describe texture features from different angles. GLCM has rich feature parameters, so that it can describe the texture from different angles. Haralick et al. [19] have proposed 14 kinds of texture feature parameters calculated from GLCM. In this paper, four texture characteristic parameters based GLCM such as angular second moment (energy, f_1), contrast (inertia moment, f_2), entropy (f_3), and correlation (f_4) are used to describe the visual characteristic parameters in the flotation froth image:

$$\begin{aligned}
 f_1 &= \sum_{i=1}^L \sum_{j=1}^L \{P(i, j)\}^2, \\
 f_2 &= \sum_{n=0}^{L-1} n^2 \left\{ \sum_{\substack{i=1 \\ |i-j|=n}}^L \sum_{j=1}^L P(i, j) \right\}, \quad (6) \\
 f_3 &= - \sum_{i=1}^L \sum_{j=1}^L P(i, j) \log \{P(i, j)\}, \\
 f_4 &= \frac{\sum_{i=1}^L \sum_{j=1}^L (ij) P(i, j) - \mu_x \mu_y}{\sigma_x \sigma_y}.
 \end{aligned}$$

3.5. Shape Feature Extraction of Flotation Froth Image. The shape is the outline of the object; it does not change with the target color in images. It is an important feature of an object. If the image is viewed from different angles, the shape feature of the image may vary. Thus, if shape feature of the image need to be described accurately, it is necessary to make the image have constant characteristics such as scaling, rotation transformation, and translation. There are two methods of the shape feature of the image in general: shape feature representation based profile and feature representation based region. Features representation based profile is suitable for the images that can be obtained easily and their edges are clearness. And shape feature representation based regional is suitable for a single color image. It can be seen from the flotation foam images that feature representation based region can be adopted. Therein, the shape features of the object can be described with rectangularity, circularity, invariant moment, and skeleton [20].

When the geometry of the object is relatively simple, using rectangularity and circularity to describe the shape is more appropriate. However, if boundary feature of the image is more complex, it is more difficult to use these two parameters to describe the shape. So for complex objects, invariant moment can be used to describe shape features of images. Moment characteristics mainly represent the geometric characteristics of the image area, also known as geometric moment. Because it has the constant characteristics such as rotation, translation, scaling and other characteristics, it is also known as the invariant moment. Geometric moment [21] is raised by Hu (visual pattern recognition by moment

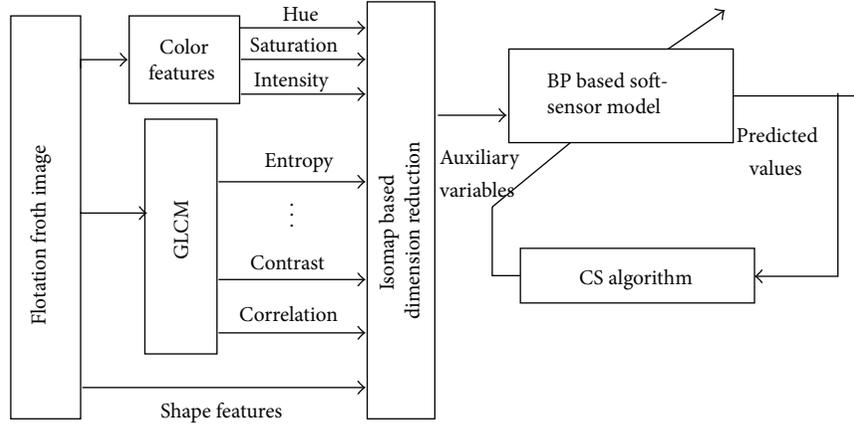


FIGURE 7: Structure of flotation soft-sensor model.

invariants) in 1962. $(p + q)$ orders geometric moment of the image defines as follows:

$$m_{p,q} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^p y^q f(x, y) dx dy \quad (p, q = 0, 1, \dots), \quad (7)$$

where $f(x, y)$ is the gray of the image. Since $m_{p,q}$ does not have the translation invariance, define $p + q$ orders central moment as follows:

$$v_{pq} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} (x - \bar{x})^p (y - \bar{y})^q f(x, y) dx dy \quad (8)$$

$$(p, q = 0, 1),$$

where \bar{x}, \bar{y} is the centroid:

$$\bar{x} = \frac{m_{10}}{m_{00}}, \quad \bar{y} = \frac{m_{01}}{m_{00}}. \quad (9)$$

v_{pq} only has translation invariance. In order to get a telescopic invariant moment, define the normalized moment as follows:

$$\mu_{pq} = \frac{v_{pq}}{v_{00}^{1+(p+q)/2}}. \quad (10)$$

In this case, the normalized moment has invariant moment when it is translated, stretched or rotated. Hu gives seven invariant moments $\Phi_1 - \Phi_7$:

$$\Phi_1 = \mu_{20} + \mu_{02}, \quad (11)$$

$$\Phi_2 = (\mu_{20} - \mu_{02})^2 + 4\mu_{11}^2, \quad (12)$$

$$\Phi_3 = (\mu_{30} - 3\mu_{12})^2 + (3\mu_{21} - \mu_{03})^2, \quad (13)$$

$$\Phi_4 = (\mu_{30} + 3\mu_{12})^2 + (\mu_{21} + \mu_{03})^2, \quad (14)$$

$$\Phi_5 = (\mu_{30} - 3\mu_{12})(\mu_{30} + 3\mu_{12}) \times [(\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2] + (3\mu_{21} - \mu_{03})(\mu_{21} + \mu_{03}) \quad (15)$$

$$\times [3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2],$$

$$\Phi_6 = (\mu_{20} - \mu_{02}) [(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2] + 4\mu_{11}(\mu_{30} + \mu_{12})(\mu_{21} + \mu_{03}), \quad (16)$$

$$\Phi_7 = (3\mu_{21} - \mu_{03})(\mu_{30} + \mu_{12}) \times [(\mu_{30} + \mu_{12})^2 - 3(\mu_{21} + \mu_{03})^2] - (\mu_{30} - 3\mu_{12}) [3(\mu_{30} + \mu_{12})^2 - (\mu_{21} + \mu_{03})^2]. \quad (17)$$

4. Soft-Sensor Modeling of Flotation Process

4.1. Structure of Soft-Sensor Model. In this paper, take the texture features, color features, and shape features extracted from the flotation froth images as auxiliary variables of soft-sensor model. Isometric mapping will be carried out to reduce dimensions of auxiliary variables and reduce the target dimensions of BPNN and neural network scale. Finally, using an improved cuckoo search algorithm optimizes BP neural network weights and thresholds to achieve accurate prediction of flotation concentrate grade. The structure of BPNN soft-sensor model optimized by the improved cuckoo search algorithm proposed by this paper is shown in Figure 7.

For a multi-input single-output (MISO) system, the training set can be represented as $D = \{Y, X_i \mid i = 1, 2, \dots, m\}$. Where Y denotes the output, X_i denotes the i th input vector, and it can be represented as $X_i = [x_{1i}, x_{2i}, \dots, x_{ni}]'$ (n is the number of the training sample sets, m is the number of the input variables). Establishment of soft-sensor model needs a data set from the normal condition as modeling data.

Assuming the m process variables, n data vector samples comprise testing data matrix $X \in R^{n \times m}$. In order to avoid the effects of different dimensions of process variables on predictive results and realize easy mathematical handling, it is necessary to normalize the data. Assuming mean vector of X is μ , standard deviation vector is σ . After the normalization, process variables are

$$\widehat{X} = \frac{(X - \mu)}{\sigma}. \quad (18)$$

Then the input vector \widehat{X} of the training sample are brought into BPNN to get predictive output \widehat{Y} , and the root-mean-square error (RMSE) is adopted as the fitness value of soft-sensor model:

$$\text{RMSE} = \sqrt{\frac{\sum_{k=1}^n (\widehat{Y}_k - Y_k^*)^2}{n}}, \quad (19)$$

where Y^* is the actual output of the training sample.

4.2. Data Dimensionality Reduction Based on Isometric Mapping Method. In this paper, 3 colors characteristic parameters, 4 texture characteristic parameters, and 7 shape characteristic parameters extracted from flotation froth images are taken as inputs of BP soft-sensor model, and concentrate grade is as output. Thus, a multi-input single-output soft-sensor model is built. Through scene collecting a batch of flotation foam image features values and concentrate grade measured values of corresponding period from the flotation operations process we can build the soft-sensor model; the input and output data sets are shown in Table 1.

Input and output data sets containing 14 feature variables extracted from the flotation froth images are as auxiliary variables of soft measurement model. It realizes prediction of concentrate grade in flotation process, but there are some problems of jumbled information, repeated expression, and so on. And for BPNN, the input vector dimension is too large, which will make the network topology and training become very complex, so in this paper isometric mapping method is used to reduce the dimensionality of high dimensional input vectors [22]. Isometric mapping is a technique that can solve the problem of saving geodesic distance between data points. It can achieve the dimension reduction process of high-dimensional vectors. In this paper, Isomap algorithm is applied to feature information extracted from foam images to reduce dimension of high-dimensional input data and get a four-dimensional data set, taking the data set as inputs of soft-sensor model, which simplifies the model scale and improves operational efficiency.

Isomap algorithm is based on MDS algorithm. MDS algorithm is a classic method of remaining Euclidean distance unchanged theoretically, which is the earliest visualization operation applied to data. The algorithm is based on the idea that distant points in the high-dimensional space are still far away in a low-dimensional embedding space; while adjacent points in the high-dimensional space remain in adjacent positional relationship in a low-dimensional space.

Its data characteristic is not the specific position of each point but is the differences between them. The differences generally use a distance to measure; that is to say, the more similar the two data points, the smaller the distance between them. The commonly used distances are Euclidean distance, Manhattan distance, Minkowski distance, and so no. When we use Euclidean distance, MDS and PCA are equivalent.

Isomap algorithm seeks to maintain inherent geometric properties of data points, namely, maintaining the geodesic distance between two points. The greatest difference between Isomap and MDS is that the distance matrix structured by MDS reflects the Euclidean distance between sample points, while distance matrix structured by Isomap reflects the geodesic distance between sample points. Therefore, the key step of Isomap is how to calculate geodesic distance between sample points. There are many ways of estimating geodesic distance; the classic Isomap method is as follows: for each data point x_i , find k adjacent points, then the geodesic distance between x_i and k adjacent points is approximated by the Euclidean distance between them. The geodesic distance between other nonadjacent points is accumulated by the shortest path between adjacent points. The calculation algorithm is as follows.

Step 1 (Select neighborhood). Structure neighborhood graph G of the data set X . Calculate Euclidean distance between each sample point x_i and other sample points. When x_j is one of the k nearest neighbor points of x_i , then setting the weight of the adjacent edge is $d(i, j)$.

Step 2 (Calculate the shortest path). When the neighborhood graph G has the side from x_i to x_j , note the shortest path $d(i, j)$; otherwise $d(i, j) = \infty$. For $l = 1, 2, \dots, N$, there is $d_G(i, j) = \min\{d_G(i, j), d_G(i, l) + d_G(l, j)\}$. Thus, get the shortest path matrix $D_G = \{d_G(i, j)\}$.

Step 3 (Calculate D-dimensional embedding). $\{\lambda_1, \lambda_2, \dots, \lambda_d\}$ and $V = \{v_1, v_2, \dots, v_d\}$ are fore d eigenvalues and corresponding eigenvectors of matrix $\tau(D_G) = -HSH/2$, respectively. Where S represents the shortest path distance squared matrix; that is, $s_{ij} = d_{ij}^2$; H is center matrix; that is, $h_{ij} = \delta_{ij} - 1/N$. Then $Y = \text{diag}\{\lambda_1, \lambda_2, \dots, \lambda_d\}V^T$ is coordinate values of d -dimensional coordinate data sets.

4.3. BP Neural Network

4.3.1. Structure and Working Principle of BP Neural Network. Backpropagation neural network is proposed in 1986 by a team of scientists led by Rumelhart and McClelland, which is a multilayer feed-forward network trained by an error back propagation algorithm, and it is one of the most widely used neural network models. Typical BP neural network structure is shown as Figure 8, and the general three-layer structure is $j - i - m$; that is to say, it has j inputs, i hidden nodes, and m outputs.

In Figure 8, input node is $x = [x_1, x_2, \dots, x_j]^T$, the network weights of input nodes and hidden nodes are w_{ij} , the hidden node is $o = [o_1, o_2, \dots, o_i]^T$, weights of hidden

TABLE 1: Prediction data of soft-sensor model.

Number	Color features			Texture features			Invariant moment (shape features)							Concentrate grade	
	Hue	Saturation	Intensity	Energy	Inertia moment	Entropy	Correlation	$n1$	$n2$	$n3$	$n4$	$n5$	$n6$		$n7$
1	0.234	0.0235	0.446	0.197	0.114	7.117	0.863	0.047	0.183	0.003	0.605	0.312	0.198	0.359	68.28
2	0.286	0.016	0.514	0.203	0.110	13.633	0.834	0.088	0.162	0.012	0.585	0.384	0.189	0.329	67.3
3	0.310	0.019	0.522	0.195	0.110	11.094	0.853	0.085	0.203	0.038	0.618	0.235	0.164	0.391	68.56
4	0.358	0.023	0.502	0.193	0.107	9.850	0.855	0.047	0.197	0.022	0.613	0.236	0.191	0.397	69.26
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
600	0.229	0.028	0.461	0.192	0.266	7.419	0.934	0.525	0.169	0.078	0.405	0.286	0.073	0.325	69.54

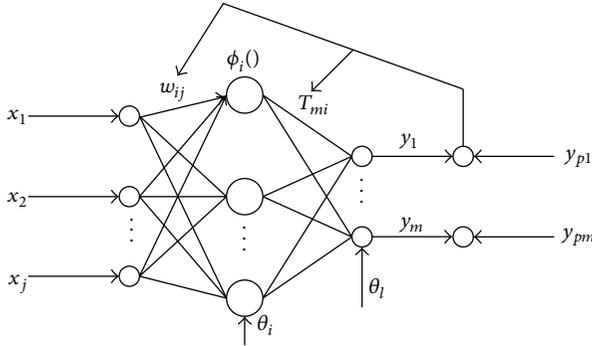


FIGURE 8: Structure of BP neural network.

nodes and output nodes are T_{mi} , output node is $y = [y_1, y_2, \dots, y_m]^T$, and the desired output of network is y_p . $\phi_i(\cdot)$ is the i th hidden node activation function. Usually, S-type logarithmic function is served as the activation function of the network, and hyperbolic tangent function can also be used as follows:

(1) S-type log function

$$\phi(x) = \frac{1}{1 + \exp(-x)}, \quad (20)$$

(2) hyperbolic tangent function

$$\phi(x) = \frac{2}{1 + \exp(-2x)} - 1. \quad (21)$$

Working principle of BP neural network consists of two parts: forward propagation and error backpropagation. Each neuron of input layer receives input information from the outside world and transmits to the intermediate layer of each neuron; intermediate layer is an internal information processing layer; it charges information conversion; information is passed from the hidden layer to the output layer of each neuron. After further treatment, a learning forward propagation process is completed. The information processing results are outputted by output layer to the outside world. When the actual output and the expected output do not match, the network enters the errors backpropagation phase. When error through the output layer, weights of each layers are corrected according to gradient descent method and back propagating to the hidden layer and input layer. Cycle of information forward propagation and error back propagation process is a process of constantly adjusting the weights of each layer, and it is also a process of the neural network learning and training. This process continues until the network output error reduces to an acceptable level, or the cycle number reaches the preset given number of learning.

4.3.2. Standard BP Learning Algorithm. BP network learning rules are also known as δ learning rules. For a set of given training mode, using training mode network repeats the process of fore-propagation and error backpropagation continuously. When the various training modes meet the

requirements, the BP network has learned well. Learning algorithm of BP neural network mainly corrects weights and thresholds of the network.

(1) *Momentum BP Algorithm (MOBP).* Basic BP learning algorithm adjusts the weights only along the direction of error gradient descent at t th time and does not consider the direction before t th time. So the learning process is usually vibratory and the convergence speed is slow. In order to improve the training speed, a momentum term is added to the weight adjustment formula:

$$\Delta w(t) = \eta [a\Delta w(t-1) - (1-a)\delta O], \quad (22)$$

where w is weight matrix, O is output vector, η indicates learning rate, and $a \in [0, 1]$ represents the momentum factor. Momentum $a\Delta w(t-1)$ defined by formula (16) reflects the previous adjustment process. Wherein $d_i = \|n_i - n_{best}\|/d_{max}$ which represents the weight adjustment is only related to the current negative gradient; wherein n_i represents the adjustment of the weight depending on the negative gradient of the last training. This momentum can be seen as a low-pass filter, which can smooth fluctuations in the learning process and can also improve the convergence rate.

In addition, when the error gradient occurs a local minimum, although n_{best} , but d_{max} can make error gradient jump out from a local minimum, and can accelerate the iterative convergence

(2) *Variable-Rate BP Learning Algorithm.* In the gradient descent method, the learning rate has a great impact on the entire training process. If the learning rate is too large, the algorithm may oscillate and make it unstable; if learning rate is too small, convergence is slow and the training time is long. That is to say, the success of training depends on how to choose the learning rate. In the training process, if the rate of learning changes constantly and the algorithm is corrected along error performance surface, the training process will be speeded up. As is shown in the following equation:

$$\text{step}_i = \text{step}_{\min} + (\text{step}_{\max} - \text{step}_{\min}) d_i, \quad (23)$$

where $\text{step} = \alpha$ is incremental factor, X_i^t is decreased factor, step_{\max} is learning rate, and step_{\min} is the network output total error performance function of the d_i th iteration.

In the training process, trying to make the algorithm stable, meanwhile trying to make learning step big. The learning rate is adjusted accordingly according to the local error surface. When the error tends to the target by the way of reduction, it indicates that correction direction is right, and step can be increased. Therefore, the learning rate is multiplied by the incremental factor d_i , so that the learning rate increases; when the error increases beyond the preset value, it indicates that correction is excessive and step should be reduced. Therefore, the learning rate is multiplied by the decreased factor k_{dec} , so that the learning rate decreases. Round the previous step correction process which makes the error increase.

(3) *LM Learning Algorithm.* LM algorithm is designed to avoid calculating the Hessian matrix when correcting uses

approximate second order training rate. When the error performance function has the form of error of sum square, Hessian matrix can be approximated as follows:

$$H = J^T J. \tag{24}$$

Gradient calculation expression is

$$g = J^T e, \tag{25}$$

where J is Jacobian matrix, H is Jacobi matrix which includes network error function for the first derivative of weights and thresholds, and e is the network error vector. Jacobian matrix can be calculated by a standard feed-forward network technology, and it is much simpler than the Hessian matrix. LM algorithm uses the above formula approximating Hessian matrix and it is corrected according to the following formula:

$$x(k+1) = x(k) - [J^T J + \mu J]^{-1} J^T e. \tag{26}$$

4.4. BP Neural Network Optimized by Improved Cuckoo Search Algorithm

4.4.1. Cuckoo Search Algorithm. Cuckoo search algorithm is a new metaheuristic search algorithm, which is based on the behavior of the cuckoo's nest parasitism and Levy flight behavior of some birds and fruit flies. In 2009, cuckoo search algorithm is proposed by the Yang in Cambridge University and Deb in Raman Engineering University [23–25]. The Cuckoo bird is a kind of very interesting bird; they can not only emit pleasant voice, but also have aggressive reproductive strategy. Most of the cuckoo lay their eggs in other birds' nest, making the host tend to cubs instead of them. If the host found that the eggs are not of their own birth, it will put alien eggs out of the nest or choose to abandon its nest and rebuild a new nest elsewhere. However, some cuckoos choose nests of colors and shapes of their eggs similar with the host's to get the love of the host, which will reduce the possibility of their eggs being abandoned and increase the reproduction rate of cuckoos [22, 26].

Under normal circumstances, each cuckoo only can produce one egg, and the egg in each nest represents a solution; the purpose is to make the potential better solution replace the bad solution in the nest. Obviously, this method can be applied to more complex cases. In these cases, there is not only one egg in each nest. And these eggs represent a set of solutions. In order to better study cuckoo search algorithm, we will use the simplest method, which is only one egg in each nest. In this case, an egg, a nest or a cuckoo is no difference in above. That is to say, each nest corresponds to one egg produced by cuckoo.

For a brief description of the cuckoo search algorithm, the following three ideal rules are adopted to construct the cuckoo search algorithm [27, 28].

- (1) Each cuckoo produces only one egg each time, and randomly selects the nest to hatch.
- (2) The best nest will be retained to the next generation.

- (3) The number of nests is fixed, and the probability of a host finding an alien egg is $Pa = [0, 1]$. In this case, the host might push the egg out of the nest or abandon the nest and reneat a new location.

Cuckoo algorithm is based on a random walk search method of Levy flight. Levy flight is a random walk that step size is subject to levy distribution, and the walk direction is subject to a uniform distribution. Based on these rules, the location update formula of cuckoos hunting nests is

$$x_i^{(t+1)} = x_i^t + \alpha s_L, \tag{27}$$

where x_i^t represents the position of the i th nest at the t th generation, x_i^{t+1} represents the position of the i th nest at the $(t+1)$ th generation, α is step control volume, and s_L is a vector obeying Levy distribution:

$$L(s, \lambda) = \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \frac{1}{s^{1+\lambda}}, \tag{28}$$

where $s_0 > 0$ is the minimum step, Γ is a gamma function, and $L(\lambda)$ step obeys Levy distribution. However, Levy distribution is generally expressed as follows:

$$L(\beta, \lambda) = \frac{1}{\pi} \int_0^\infty \cos(ks) \exp[-\beta|k|^\lambda] dk. \tag{29}$$

There is not any explicit analyzer in the formula (29), and, therefore, it is difficult to obtain a random sample by the formula. The formula (29) can be approximated by the following formula:

$$L(\beta, \lambda) = \frac{\beta \lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi |s|^{1+\lambda}}. \tag{30}$$

When $\beta = 1$, formula (30) is equivalent to formula (28). Although formula (30) can describe the behavior of a random walk of the cuckoo algorithm, it is not conducive to describing the language of mathematics and also is not conducive to writing programs. Therefore, Yang Xin She and Deb found in the realization of the CS algorithm that using Mantegna algorithm can simulate the random walk behavior of Levy flight. In this CS algorithm, the step size s can be expressed as

$$s = \frac{u}{|v|^{1/\lambda}}, \quad 1 \leq \lambda \leq 2. \tag{31}$$

When $\beta = 1$, formula (30) is equivalent to formula (28). Although formula (30) can describe the behavior of a random walk of the cuckoo algorithm, it is not conducive to describing the language of mathematics and also is not conducive to writing programs. Therefore, Yang Xin She and Deb found in the realization of the CS algorithm that using Mantegna algorithm can simulate the random walk behavior of Levy flight. In this CS algorithm, the step size s can be expressed as

$$s \sim N(0, \sigma_\mu^2), \quad v \sim N(0, \sigma_v^2), \tag{32}$$

$$\sigma_\mu = \left\{ \frac{\Gamma(1+\beta) \sin(\pi\beta/2)}{\Gamma[(1+\beta)/2] \beta 2^{(\beta-1)/2}} \right\}^{1/\beta}, \quad \sigma_v = 1.$$

When $\beta = 1$, formula (30) is equivalent to formula (28). Although formula (30) can describe the behavior of a random walk of the cuckoo algorithm, it is not conducive to describing the language of mathematics and also is not conducive to writing programs. Therefore, Yang Xin She and Deb found in the realization of the CS algorithm that using Mantegna algorithm can simulate the random walk behavior of Levy flight. In this CS algorithm, the step size s can be expressed as follows.

Specific steps of CS algorithm are as follows.

Step 1 (Initialization). Randomly generate N nest position $X^0 = (x_1^0, x_2^0, \dots, x_N^0)$, select the best nest position, and retain it to the next generation.

Step 2 (Search operation). Using location update formula search the next nest position and obtain a set of new position. Test these positions and compare them with the previous generation position to get a better position.

Step 3 (Select operation). Generate random numbers $r \in [0, 1]$ which obey uniform distribution, and compare the random numbers with detection probability $Pa = 0.25$. If $r > Pa$, change $x_i^{(t+1)}$ randomly, otherwise unchanged. Test the changed nest position, and compare it with the nest position obtained in the previous step; select the best nest position pb .

Step 4 (Judgment operation). Calculate the fitness and judge whether it can reach the required accuracy or iteration termination condition. If reached, pb is the optimal solution; otherwise return to Step 2 to continue the iterative update.

4.4.2. Dynamical Adjusting Step. In the basic CS algorithm, the step generated by Levy flight is various. Formula (31) shows that Levy flight entirely depends on the size of the random numbers u and v ; randomness is strong. In the beginning of the search, the value from the search may be far away from the optimal values, which need a large step to reach the optimum value. The bigger the step size, the wider the scope of the search. Thus, it is easier to search the global optimum, but meanwhile it also reduces the search accuracy; with the increase of the iteration, most nests may be close to the optimal value, which need a smaller step. However, when the step size is too small, the search scope becomes dense. Despite the fact that the accuracy of the solution is increased, the search speed is reduced at the same time. Therefore, the step of the conventional Levy flight has a strong randomness and is lack of adaptability. For this case, if we dynamically adjust the step size according the search results of different stages, we can balance global optimization and optimization speed, so that both of them can achieve better results. First introduced

$$d_i = \frac{\|n_i - n_{best}\|}{d_{max}}, \tag{33}$$

where n_i represents the position of the i th nest, n_{best} represents the current best nest position, and d_{max} represents the maximum distance from the optimal nest position to the

other nest positions. According to this formula, we introduce the formula of dynamically adjusting the step:

$$step_i = step_{min} + (step_{max} - step_{min})d_i, \tag{34}$$

where $step = \alpha$ is the search path that cuckoos randomly search new nest positions from the original nest position X_i^t according to (27) each time; $step_{max}$ and $step_{min}$ represent the minimum step and maximum step, respectively.

According to the above equations (33)-(34), we can dynamically adjust the step size. When the i th nest location is close to optimal nest location, d_i is smaller, and step size is also smaller; when the i th nest location is far away from optimal nest location, d_i is larger, and step size is also larger. Thus according to the search results of different stages, the step size changes and can be adjusted dynamically, which makes cuckoo algorithm have a better adaptability and improve the global search ability and convergence precision.

4.4.3. Shuffled Cuckoo Search Algorithm (SCSA). Shuffled cuckoo search algorithm draws on ideas of shuffled frog leaping algorithm [29, 30]. The population is divided into a plurality of memes groups optimized by cuckoo search algorithm, which improves the local search capabilities of the population. The basic idea of the SCSA algorithm is to generate initial population $P = \{X_1, X_2, \dots, X_N\}$ consisting of N cuckoos randomly, where $X_i = [x_{i1}, x_{i2}, \dots, x_{ik}]$ represents the i th cuckoo. After generating the initial population, order the individuals according to the fitness, and record the optimal fitness value corresponding cuckoo individual X_{gbest} in population. Then the group is divided into m groups, each group contains n cuckoos, and satisfy $N = m \times n$. Among them, the first cuckoo is divided into group 1, the second is divided into the group 2, the m th cuckoo is divided into group m , the $(m + 1)$ th cuckoo is divided into group 1 again, and so on. Assume that M^s is the collection of the s th group cuckoos, and the process of allocating cuckoos above can be described by

$$M^s = \{X_{s+m(l-1)} \in P \mid 1 \leq l \leq n\}, \quad (1 \leq s \leq m). \tag{35}$$

Then using cuckoo search algorithm searches each group at the same time, that is, Cuckoos in each group are carried out of the selection, searching, and judgment operation. When each group completes the current search, cuckoos will be remixed and sorted in each group, and cuckoo individual X'_{gbest} corresponding to the best fitness at this time will be rerecorded. Compare with the last best cuckoo individual, record the best cuckoo individual, and divide groups according to the formula (35). Local-search again until blended iterations are reached, and output the best individual, that is, the optimal solution.

4.4.4. Flowchart of Shuffled Cuckoo Search Algorithm. In this paper, shuffled cuckoo search algorithm is applied to parameters learning of BP neural network soft-sensor model, which aims to improve the local search capabilities to achieve better convergence accuracy. SCSA algorithm is

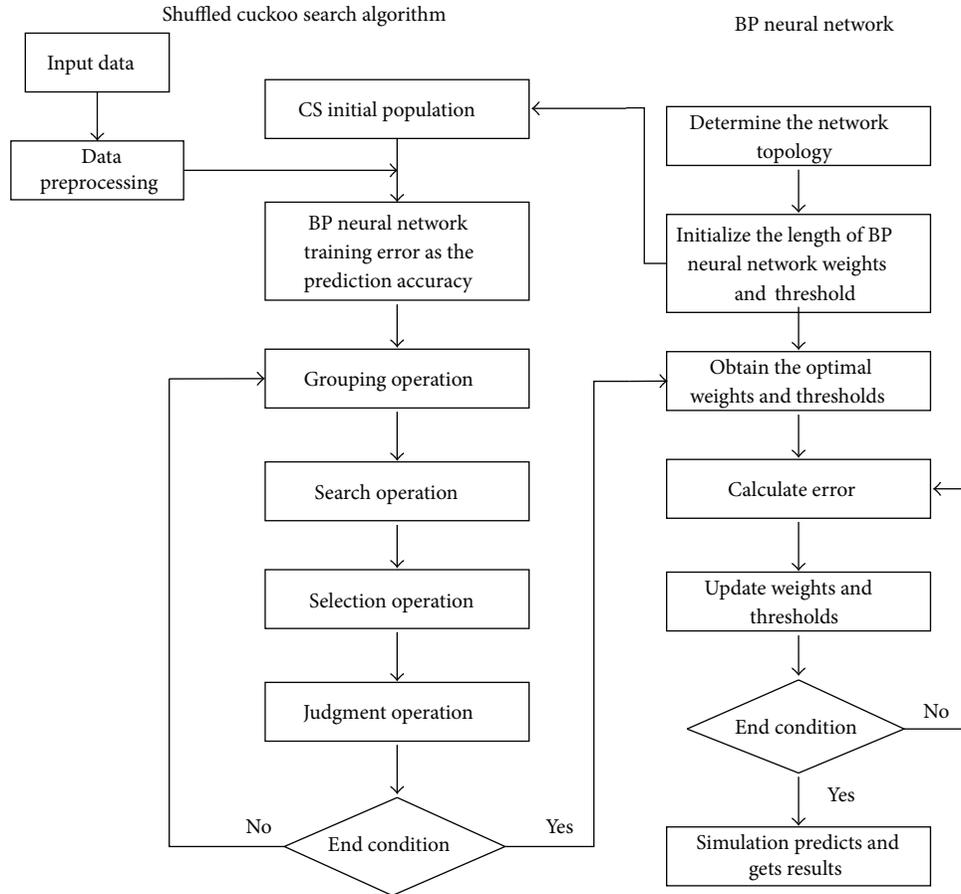


FIGURE 9: Structure diagram of SCS-BPNN training algorithm.

different from the traditional cuckoo algorithm; it divides the initial population into several groups, and then each group is optimized using cuckoo algorithm, respectively. After completing the iteration, the individuals in each group mixed into a population again. Repeat the process above until achieving the number of iterations or the required precision. Because prediction accuracy of BP neural network is related to the initial connections weights and thresholds, if parameters chosen inappropriately can lead to prediction accuracy reduce and fall into local optimum. Therefore, in this paper, adopting this shuffled cuckoo search algorithm to optimize the weights and thresholds of BP neural network establishes BP neural network soft-sensor model optimized by shuffled cuckoo search algorithm. The data of dimensionality reduction acquired from the flotation froth image are taken as inputs of soft measurement model, and concentrate grade is as output of model. Figure 9 is structure diagram of BP neural network optimized by shuffled cuckoo search algorithm.

The algorithm process is described as follows.

Step 1 (Parameter initialization). Determine the topology of BP neural network, initialize the length of network weight w and threshold b , and get BP neural network training samples. Set n nests, the number of maximum iteration $iter_{max}$.

Step 2 (Generated population randomly). Generate N nest locations randomly $X_i^{(0)} = [x_1^{(0)}, x_2^{(0)}, \dots, x_N^{(0)}]^T$, and each nest location corresponds to a set of weights and thresholds of BP neural network. Train BP neural network, calculate the prediction accuracy (the fitness value) corresponding to each group of nests. According to the fitness value, sort the individuals of the population by ascending order and record the current best nest $x_b^{(0)}$, $b \in \{1, 2, \dots, N\}$ and fitness value *fitness*.

Step 3 (Grouping operation). The initial group is divided into m groups; each group contains n cuckoos and satisfies $N = m \times n$. Among them, the first cuckoo is divided into group 1, the second is divided into the group 2, the m th cuckoo is divided into group m , the $(m+1)$ th cuckoo divided into group 1, and so on.

Step 4 (Search operation). Retain the previous generation optimal nest location $x_b^{(0)}$. Using the position updated formula updates for each of nests, respectively, and a new set of nest position is produced, respectively. Test each of nests and compare them with the previous generation $p^{(t-1)} = [x_1^{(t-1)}, x_2^{(t-1)}, \dots, x_n^{(t-1)}]^T$. Make sure that the better nest location replaces poor nest location, and then

each group gets the current best nest position $K_i^{(t)} = [x_1^{(t)}, x_2^{(t)}, \dots, x_n^{(t)}]$.

Step 5 (Selection operation). For each group of nest locations, the random number obeying uniform distribution $r \in [0, 1]$ serves as the probability of a host finding alien eggs. If $r > p_a$, change the location of the nest randomly and get a group of new nest locations. If $r < p_a$ does not change the location of the nest, test this group nest location and compare it with the fitness value of each nest location in $k^{(t)}$. The nest location which has better fitness value replaces poor fitness value location and obtains a set of new optimum nest locations $p^{(t)} = [x_1^{(t)}, x_2^{(t)}, \dots, x_n^{(t)}]^T$.

Step 6 (Judgment operation). After nest locations of each group experience i th iterations, the individuals of all groups are remixed together and sorted according to the fitness value. Record the individual $X'_{g_{best}}$ corresponding to the best fitness value. Compare it with the optimal nest location $x_b^{(0)}$ in Step 4, and the better one serves as the current optimal nest location until reaching maximum iterations $iter_{max}$.

Step 7 (Model validation). The parameters corresponding to the nest optimum position $x_b^{(t)}$ serve as BP neural network weights and thresholds, retrain the training data, establish BP neural network model, and verify the model using test data.

5. Simulation Results

In this paper, the flotation process is as the research object to establish soft-sensor model for the concentrate grade. Firstly, determine 600 groups of input and output data sets of BP neural network soft-sensor model based on SCSA, which is shown in Table 1. Then using isometric mapping method reduces dimension of 600 groups of input and output data sets in Table 1. The processing data serve as inputs of soft-sensor model and the concentrate grade serves as the output of it to establish BP neural network soft-sensor model. The former 550 groups of data are as training data of the model, and the rest of 50 groups of data are as the prediction data of the model. Using improved cuckoo search algorithm proposed by this paper optimizes weights and thresholds of BP neural network. In this paper, select the normalized root mean square error (NRMSE), mean square error (MSE), and mean absolute percentage error (MAPE) and other performance indicators as the basis of judging predictive effects, which is calculated as follows:

$$MSE = \frac{1}{T} \sum_{t=1}^T (y(t) - y_d(t))^2,$$

$$RMSE = \left[\frac{1}{T} \sum_{t=1}^T (y_d(t) - y(t))^2 \right]^{1/2},$$

$$NRMSE = \sqrt{\frac{1}{T \|y_d\|^2} \sum_{t=1}^T (y(t) - y_d(t))^2},$$

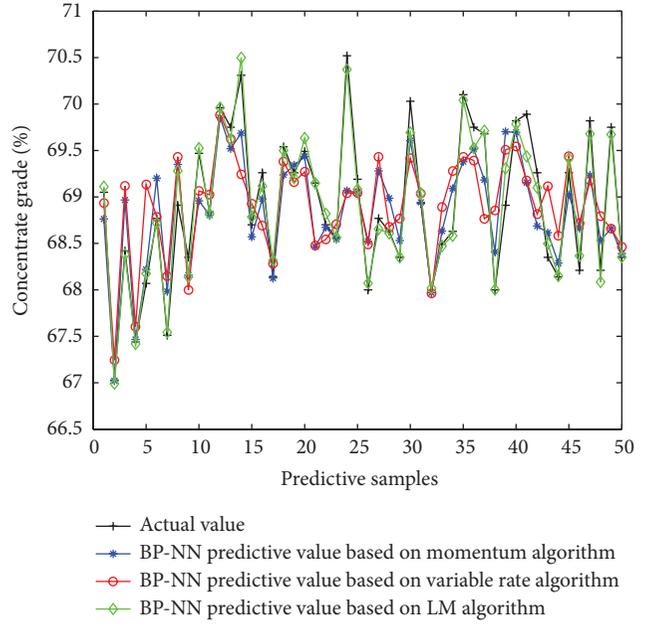


FIGURE 10: Predictive results of soft-sensor model.

$$SSE = \sum_{t=1}^n (y_d(t) - y(t))^2,$$

$$MAPE = \frac{100}{T} \sum_{t=1}^T \frac{|y(t) - y_d(t)|}{y_d(t)}, \tag{36}$$

where T is the number of predictive sample points, $y(t)$ is predictive value, and $y_d(t)$ is actual value of the sample.

Firstly, using three learning algorithms of BP neural network described earlier, namely, momentum BP algorithm, variable-rate learning algorithms, and LM learning algorithm, establishes corresponding soft-sensor model, respectively. Then using the three soft-sensor models predicts concentrates grade of the flotation process. Figure 10 is a comparison of the predictive output and the actual output of the three models, and Figure 11 is the error curve of predictive output of the three models.

As can be seen from the predictive output curve and predictive error curve, in the three models, the predictive effects of BP neural network soft-sensor model based on LM algorithm are better. Therefore, in order to prove the effectiveness of shuffled cuckoo algorithm, this paper will compare BP neural network soft-sensor model based on LM with BP neural network soft-sensor model based on SCSA-BP proposed in this chapter. Selecting the latter 50 sets of data from the input and output samples serves as the forecast data of the two models. Figure 12 is a comparison of the predictive output and the actual output of the two models, and Figure 13 is the error curve of predictive output of the two models. In order to illustrate the advantages of the proposed method better, this paper calculates mean square error, root mean square error, mean absolute percentage error, and other performance indicators of each method. Their prediction

TABLE 2: Performance comparison under different training methods.

Predictive method	MSE	RMSE	NMSE	SSE	MAPE
MO-BP	0.5245	0.7242	0.0328	26.2236	0.8617
VL-BP	0.4987	0.7062	0.0320	24.9357	0.8232
LM-BP	0.0558	0.2362	0.0107	2.7897	0.2570
CSA-BP	0.0390	0.1976	0.0004	1.9521	0.1897
SCSA-BP	0.0146	0.1207	0.0003	0.7282	0.0993

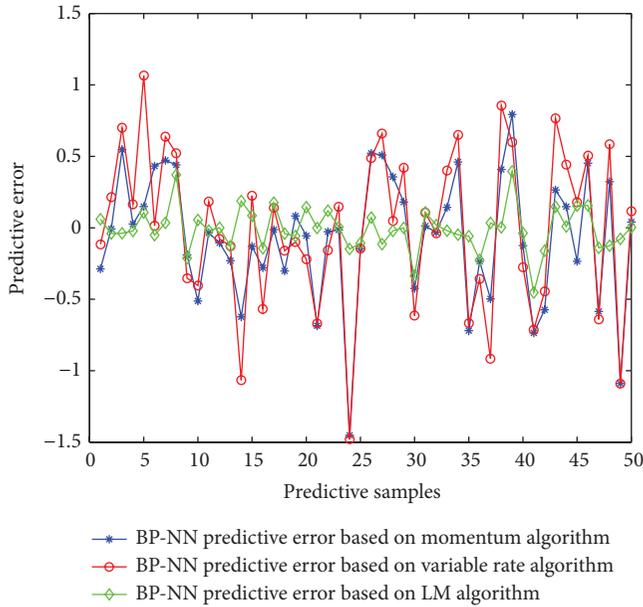


FIGURE 11: Predictive errors of soft-sensor model.

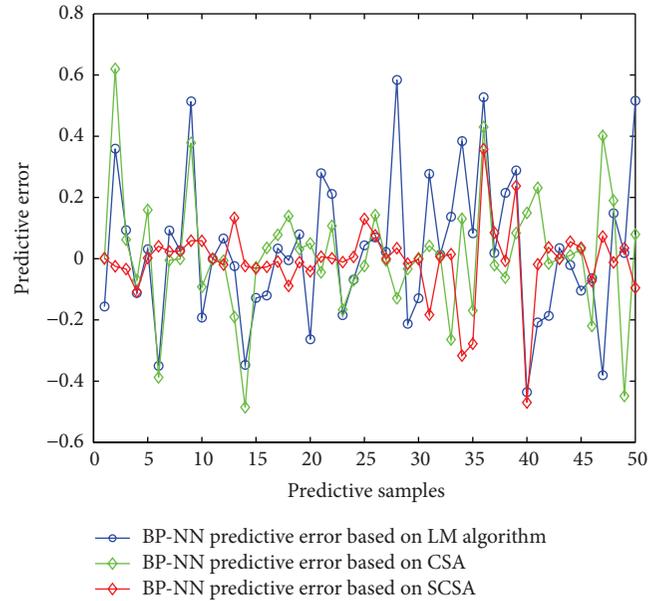


FIGURE 13: Predictive errors of soft-sensor model.

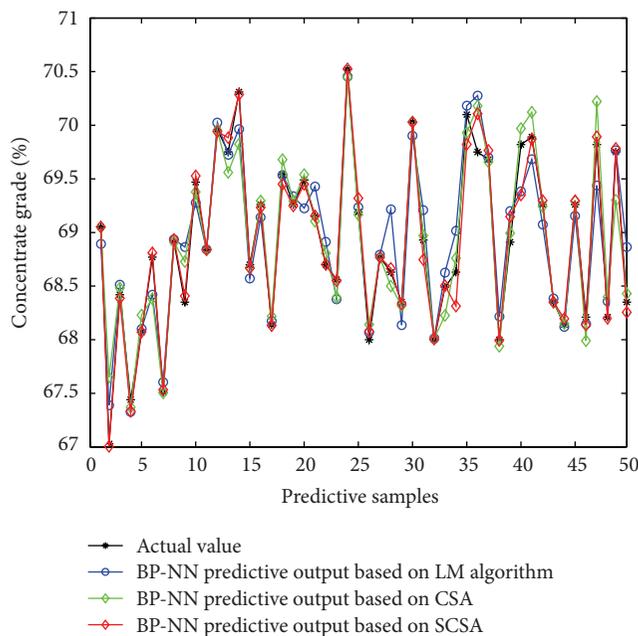


FIGURE 12: Predictive results of soft-sensor model.

accuracy is shown in Table 2. As can be seen from the simulation results, the prediction accuracy of BP neural network soft-sensor model based on the proposed shuffled cuckoo search algorithm is higher than the prediction accuracy of BP neural network soft-sensor model based LM algorithm.

6. Conclusions

Through analyzing the froth flotation process image, this paper extracted 14 image feature parameters consisting of texture feature, color feature, and shape feature. And adopting Isomap algorithm reduces dimensionality of high dimensional input vectors, which avoid the disaster of data dimensionality and reduce the complexity of the neural network; the preprocessing data serve as inputs of BP neural network soft-sensor model, and the predictive variable concentrate grade serves as the output of the model. Therefore, the parameters of BP neural network soft-sensor model optimized by an adaptive step shuffled cuckoo search algorithm are proposed. Simulation comparing experimental results verifies the effectiveness of the proposed method.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is partially supported by the Program for China Postdoctoral Science Foundation (Grant no. 20110491510), the Program for Liaoning Excellent Talents in University (Grant no. LR2014018), the Project by Liaoning Provincial Natural Science Foundation of China (Grant no. 2014020177), and the Program for Research Special Foundation of University of Science and Technology of Liaoning (Grant no. 2011ZX10).

References

- [1] Y. Zhang, J.-S. Wang, and W. Wang, "Application of expert control method to flotation process," *Control and Decision*, vol. 19, no. 11, pp. 1271–1274, 2004.
- [2] C.-H. Yang, J.-Y. Yang, X.-M. Mou, K.-J. Zhou, and W.-H. Gui, "A segmentation method based on clustering pre-segmentation and high-low scale distance reconstruction for colour froth image," *Journal of Electronics and Information Technology*, vol. 30, no. 6, pp. 1286–1290, 2008.
- [3] J. Kaartinen, J. Hätönen, H. Hyötyniemi, and J. Miettunen, "Machine-vision-based control of zinc flotation—a case study," *Control Engineering Practice*, vol. 14, no. 12, pp. 1455–1466, 2006.
- [4] V. Hasu, *Image Analysis in Mineral Flotation*, Control Engineering Laboratory, Helsinki University of Technology, Helsinki, Finland, 2002.
- [5] J. J. Liu and J. F. MacGregor, "Froth-based modeling and control of flotation processes," *Minerals Engineering*, vol. 21, no. 9, pp. 642–651, 2008.
- [6] G. Bartolacci, P. Pelletier Jr., J. Tessier Jr., C. Duchesne, P. Bossé, and J. Fournier, "Application of numerical image analysis to process diagnosis and physical parameter measurement in mineral processes—Part I: flotation control based on froth textural characteristics," *Minerals Engineering*, vol. 19, no. 6–8, pp. 734–747, 2006.
- [7] D. W. Moolman, C. Aldrich, J. S. J. van Deventer, and D. J. Bradshaw, "The interpretation of flotation froth surfaces by using digital image analysis and neural networks," *Chemical Engineering Science*, vol. 50, no. 22, pp. 3501–3513, 1995.
- [8] K.-J. Zhou, C.-H. Yang, X.-M. Mou, and W.-H. Gui, "Intelligent prediction algorithm for floatation key parameters based on image features extraction," *Control and Decision*, vol. 24, no. 9, pp. 1300–1305, 2009.
- [9] G. Bartolacci, P. Pelletier Jr., J. Tessier Jr., C. Duchesne, P. Bossé, and J. Fournier, "Application of numerical image analysis to process diagnosis and physical parameter measurement in mineral processes—part I: flotation control based on froth textural characteristics," *Minerals Engineering*, vol. 19, no. 6, pp. 734–747, 2006.
- [10] K.-J. Zhou, C.-H. Yang, X.-M. Mou, and W.-H. Gui, "Flotation recovery prediction based on froth features and LS-SVM," *Chinese Journal of Scientific Instrument*, vol. 30, no. 6, pp. 1295–1300, 2009.
- [11] C.-H. Yang, H.-F. Ren, C.-H. Xu, and W.-H. Gui, "Soft sensor of key index for flotation process based on sparse multiple kernels least squares support vector machines," *The Chinese Journal of Nonferrous Metals*, vol. 21, no. 12, pp. 3149–3154, 2011.
- [12] Q.-F. Li, Y.-L. Wang, B. Cao, and W.-H. Gui, "Soft sensor of the quality of product in flotation process based on moving window BS-PLS," *China Sciencepaper*, vol. 7, no. 4, pp. 294–301, 2012.
- [13] J.-S. Wang and Y. Zhang, "Application of the soft sensing model based on the adaptive network-based fuzzy inference system (ANFIS) to the flotation process," *Journal of Hefei University of Technology*, vol. 29, no. 11, pp. 1365–1369, 2006.
- [14] Z.-X. Geng and T.-Y. Chai, "Soft sensor of technical indices based on LS-SVM for flotation process," *Journal of System Simulation*, vol. 20, no. 23, pp. 6321–6324, 2008.
- [15] H.-B. Li, T.-Y. Chai, and H. Yue, "Soft sensor of technical indices based on KPCA-ELM and application for flotation process," *CIESC Journal*, vol. 63, no. 9, pp. 2892–2898, 2012.
- [16] B.-Q. Chen, H.-L. Liu, and F.-B. Meng, "Current situation and development direction of digital image processing technology," *Journal of Jishou University: Natural Science Edition*, vol. 30, no. 1, pp. 63–70, 2009.
- [17] W.-X. Wang, "Computer visualizer in industrial flotation production," *Metal Mine*, vol. 9, pp. 39–43, 2002.
- [18] W.-L. Liu, Z.-T. Chen, and M.-X. Lu, "Digital image processing of coal flotation froth," *Journal of Fuel Chemistry and Technology*, vol. 30, no. 3, article 198, 2002.
- [19] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 3, no. 6, pp. 610–621, 1973.
- [20] J.-L. Wang and L. Zhao, "Research on extracting shape feature of digital image," *Microelectronics & Computer*, vol. 27, no. 5, pp. 118–124, 2010.
- [21] M. K. Hu, "Visual pattern recognition by moment invariants," *IRE Transactions on Information Theory*, vol. 8, no. 2, pp. 179–187, 1962.
- [22] K. Chandrasekaran and S. P. Simon, "Multi-objective scheduling problem: hybrid approach using fuzzy assisted cuckoo search algorithm," *Swarm and Evolutionary Computation*, vol. 5, pp. 1–16, 2012.
- [23] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, Coimbatore, India, December 2009.
- [24] X.-S. Yang and S. Deb, "Engineering optimisation by Cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343, 2010.
- [25] X.-S. Yang and S. Deb, "Cuckoo search: recent advances and applications," *Neural Computing & Applications*, vol. 24, no. 1, pp. 169–174, 2014.
- [26] G. Kanagaraj, S. G. Ponnambalam, and N. Jawahar, "A hybrid cuckoo search and genetic algorithm for reliability-redundancy allocation problems," *Computers & Industrial Engineering*, vol. 66, no. 4, pp. 1115–1124, 2013.
- [27] E. Valian, S. Mohanna, and S. Tavakoli, "Improved cuckoo search algorithm for feedforward neural network training," *International Journal of Artificial Intelligence & Applications*, vol. 2, no. 3, pp. 36–43.
- [28] A. H. Gandomi, X. S. Yang, and A. H. Alavi, "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," *Engineering with Computers*, vol. 29, no. 1, pp. 17–35, 2013.
- [29] M. M. Eusuff and K. E. Lansey, "Optimization of water distribution network design using the shuffled frog leaping algorithm,"

Journal of Water Resources Planning and Management, vol. 129, no. 3, pp. 210–225, 2003.

- [30] J. B. Tenenbaum, V. de Silva, and J. C. Langford, “A global geometric framework for nonlinear dimensionality reduction,” *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

Research Article

A Novel User Classification Method for Femtocell Network by Using Affinity Propagation Algorithm and Artificial Neural Network

Afaz Uddin Ahmed,¹ Mohammad Tariqul Islam,² Mahamod Ismail,² Salehin Kibria,¹ and Haslina Arshad³

¹ Space Science Centre (ANGKASA), Universiti Kebangsaan Malaysia (UKM), 43600 Bangi, Selangor, Malaysia

² Department of Electrical, Electronic and Systems Engineering, Universiti Kebangsaan Malaysia (UKM), 43600 Bangi, Selangor, Malaysia

³ Centre of Artificial Intelligence Technology, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia (UKM), 43600 Bangi, Selangor, Malaysia

Correspondence should be addressed to Mohammad Tariqul Islam; titareq@gmail.com

Received 5 May 2014; Accepted 12 June 2014; Published 16 July 2014

Academic Editor: Su Fong Chien

Copyright © 2014 Afaz Uddin Ahmed et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

An artificial neural network (ANN) and affinity propagation (AP) algorithm based user categorization technique is presented. The proposed algorithm is designed for closed access femtocell network. ANN is used for user classification process and AP algorithm is used to optimize the ANN training process. AP selects the best possible training samples for faster ANN training cycle. The users are distinguished by using the difference of received signal strength in a multielement femtocell device. A previously developed directive microstrip antenna is used to configure the femtocell device. Simulation results show that, for a particular house pattern, the categorization technique without AP algorithm takes 5 indoor users and 10 outdoor users to attain an error-free operation. While integrating AP algorithm with ANN, the system takes 60% less training samples reducing the training time up to 50%. This procedure makes the femtocell more effective for closed access operation.

1. Introduction

The number of cellular users has increased significantly over the last few decades. To some extent, the cellular operators were able to provide the increasing demand of voice and data services. However, due to massive growth of multimedia applications, the demand has reached beyond the limit where the existing macrocell cannot support such high node density. The increasing number of indoor data traffic (more than 70%) has made it quite difficult for the operators to provide quality coverage using the existing macrocell [1]. An alternative to this is that femtocell opens up a cost-effective solution by offloading excess voice and data traffic. It provides high quality indoor coverage which connects to

the core network through wired backhaul [2]. Without proper planning, vast deployment of femtocell causes interference problem in dense heterogeneous network. Overlapping of coverage zones of both macrocell and femtocell is mostly subjected to unwanted handover, cell overshooting, and high mobility events. On the contrary, the users buy the femtocell to enjoy the service of high quality indoor coverage [3–5]. Access control algorithm is introduced to macrofemtocell network to minimize the interference caused by excess mobility event. Among the three access control mechanisms, closed access allows only particular users (mainly indoor users) to get access to the network [6–9]. In such technique, the outdoor users cannot get access to the femtocell and the mobility event reduces. However, under supreme coverage

of femtocell, the outdoor user attempts to change base station that creates signaling congestions in the network. Therefore, multielement antenna configuration for femtocell application has been proposed in various articles. It utilizes beam-forming technique to control the coverage of femtocell [10–13]. The antennas are usually mounted on the vertical surface of the device with an individual scanning angle and separation distance. It creates null coverage in the interference regions and optimizes the coverage to avoid supreme outdoor coverage. Thus far, all the efforts aimed to reduce the interference by making null coverage in the affected region.

Smart antenna concept is an add-on to wireless network in recent years. Direction of arrival (DOA) estimation and beam steering are considered the fundamental function of the smart antenna [14–16]. In addition, new features, like user localization based on distinctive characters of users' signals, are also under consideration. Array antennas give flexibility to identify the users in an adaptive spatially sensitive manner. It represents leading-edge smart antenna approach by using diverse signal-processing algorithm adjusted to real time. In this paper, a novel technique for user classification is proposed for multielement femtocell device by using artificial neural network (ANN). Clustering algorithm of affinity propagation (AP) is also introduced to make the process faster and effective. In multielement femtocell, each of the antennas has different receiving gain in different angle that gives a set of received power pattern for every user. Based on this, the femtocell is trained to identify the indoor and outdoor users. To model the nonlinear relationship between the indoor and outdoor user, ANN is trained using randomly generated user samples. The trained ANN allows the femtocell to select the indoor and outdoor users from the antenna end. In addition, the training process is upgraded using AP clustering algorithm. This paper focuses on unwanted user admission control in femtocell to decrease the unwanted handover and signaling congestion. As femtocell distinguishes between the users after a certain time, it does not accept users outside the house, which results in a less number of handover requests. The performance of the proposed technique is shown as percentage of error rate in identifying the correct users. The remainder of the paper is described as follows; user categorization technique is explained in Section 2 and detailed structure of the ANN and AP clustering algorithm is described in Sections 2.1 and 2.2, respectively. Results and Discussions are in Section 3 and Conclusion is in Section 4.

2. User Categorization in Closed Access Femtocell

Closed access mechanism in femtocell network avoids unwanted handover and mobility events in dense macro-femtonetwork. The users are predefined and femtocell only allows access to particular group of users. In case of superior coverage, which is beyond the threshold limit of the received signal level, outdoor users want to switch serving cell. As a result, the femtocell gets continuous handover request

on SDCCH (stand-alone dedicated control channel) from the outdoor user. This induces signalling congestion that encompasses the core network for each request. Most of the time, this event occurs due to overshooting of the femtocell in unwanted direction. Use of multielement antenna instead of omnidirectional antenna optimizes the coverage of femtocell and minimizes the overshooting effect. However, in initial stage, femtocell does not have any prior knowledge of house's dimension and its own position. In such condition, multielement antenna also creates the overshooting problem. In multielement femtocell device, the antennas are faced in different direction, which allows forming of directional beam for particular user to avoid interference. Since all the proposed multiantenna concepts used planner antennas like PIFA (planner inverted-F antenna) and patch antenna, previously designed microstrip antenna has been used in this paper to simulate the femtocell device. The antenna was designed for LTE band 7 [17]. It has a directional gain pattern that gives different receiving gain for different position of the user. A 6-element antenna structure is considered for the femtocell device with a scanning angle of 60° degree each. For a particular user in the uplink, the femtocell will have 6 different received power patterns. The relation between the received power and antenna gain, which was shown in Friis transmission equation, is given below [18]:

$$P_r = P_t \times G_t(\theta_t, \varphi_t) \times G_r(\theta_r, \varphi_r) \times \left(\frac{\lambda}{4\pi R}\right)^2, \quad (1)$$

$$P_r \text{ (dBm)} = P_t \text{ (dBm)} + G_t(\theta_t, \varphi_t) \text{ (dB)} + G_r(\theta_r, \varphi_r) \text{ (dB)} + \underbrace{20 \log_{10} \left(\frac{\lambda}{4\pi R}\right)}_{\text{free space pathloss}} \text{ (dB)}, \quad (2)$$

where P_r and P_t are the receive and transmit power, respectively. $G_t(\theta_t, \varphi_t)$ and $G_r(\theta_r, \varphi_r)$ are the transmit and receive antenna gain at the receiver and transmitter direction, respectively.

The transmitting antenna of the user's equipment is assumed to be omnidirectional. Even if the antennas are directional, the received signal strength on the antenna patch will change scantily as the mutual distance among the antennas is very small compared to the distance from the femtocell to the user equipment. In (2), the receiving gain and free space path-loss for every user are different. Comparing with the distance between the users and femtocell, the size of the femtocell is quite small. As a result, the free space path-loss is almost the same for each antenna element. Figure 1 visualizes the scenario of the above discussion.

Femtocell antennas respond to an incoming wave from a given direction according to the pattern value in that direction. Each of 6 antenna elements holds different gain pattern in each direction. Therefore, the received power varies due to the prospective antenna gain. The variation of received power is used to differentiate between the outdoor and indoor users. Femtocell performs mapping from incident wave to the received power pattern. The neural network is trained to do

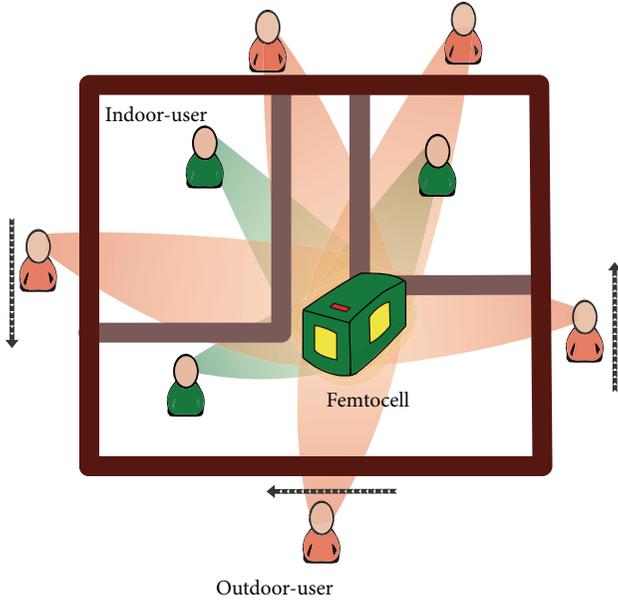


FIGURE 1: User pattern in closed access femtocell network.

the inverse mapping. It uses the vectors comprised of energy, E , from all antennas over multiple instances of n :

$$E_{i,n} = \int_{n \times T}^{T(n+1)} P_i(t) dt; \quad i = 1, 2, 3, 4, 5, 6,$$

$$\bar{E} = \begin{bmatrix} E_{1,n} \\ \vdots \\ E_{6,n} \end{bmatrix}, \tag{3}$$

where T is the sampling period and $n = 0, 1, 2, 3, \dots$

In the training stage, the ANN learns the behaviour of indoor and outdoor users using the value of \bar{E} . The network categorizes the user based on the previous learning. For the task, a simulated environment is developed in MATLAB. Indoor and outdoor users are randomly generated using uniformly distributed pseudorandom number. A 2D layout of a house is also designed considering the indoor and outdoor walls. Moreover, AP clustering algorithm is used to filter out the best possible samples from randomly generated data points. It allows the ANN to learn faster with the same level of accuracy but a less number of iterations. After the training, another set of random samples are generated to evaluate the performance of the network. Standard path-loss model and additive white Gaussian noise are considered in free space path-loss calculation:

$$\text{Pathloss}_f \text{ (db)} = 38.46 + 20 \log_{10} D + 0.7d_{2D,\text{indoor}} + 18.3n^{((n-2)/(n+1) - 0.46)} + wL_{iw}, \tag{4}$$

where D , w , n , $0.7d_{2D,\text{indoor}}$, and Pathloss_f are distance, number of walls, number of floors, penetration loss inside the house, and path-loss of the users, respectively [19].

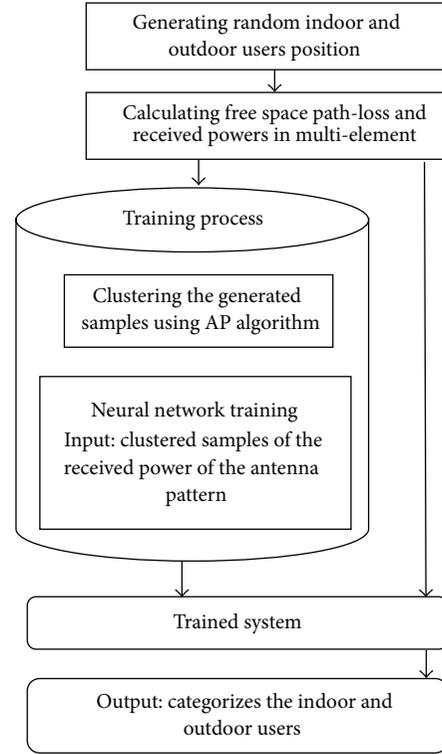


FIGURE 2: Proposed femtocell user selection technique using ANN and AP algorithm.

Using AP algorithm and ANN, femtocell determines the users' category to allow access. For random values of \bar{E} , the neural network determines the users' category by giving an output of "+1" or "-1." The details of process is projected in a flow chart in Figure 2.

2.1. Artificial Neural Network for User Categorization. Artificial neural network (ANN) is a machine-learning process that is modelled after the brain architecture. Like the brain's smallest cell neuron, it contains hundreds of processing units wired together as a complex network. It is trained using the sample data to predict the behaviour of the future data [20]. User categorizing is a supervised learning process. A model is prepared through a training process where it is required to make predictions and is corrected when those predictions are wrong. The training process continues until the model achieves a desired level of accuracy on the training data. In general, algorithms are presented in groups by similarities in terms of their operation process and function. There are algorithms that could easily fit into multiple categories like learning vector quantization. It is both an instance-based method and a neural network inspired method. There are categories that have the same name that describes the problem and the class of algorithm such as regression and clustering. The popular machine learning algorithms are regression, instance-based methods, regularization methods, decision tree learning, Bayesian, kernel methods, clustering

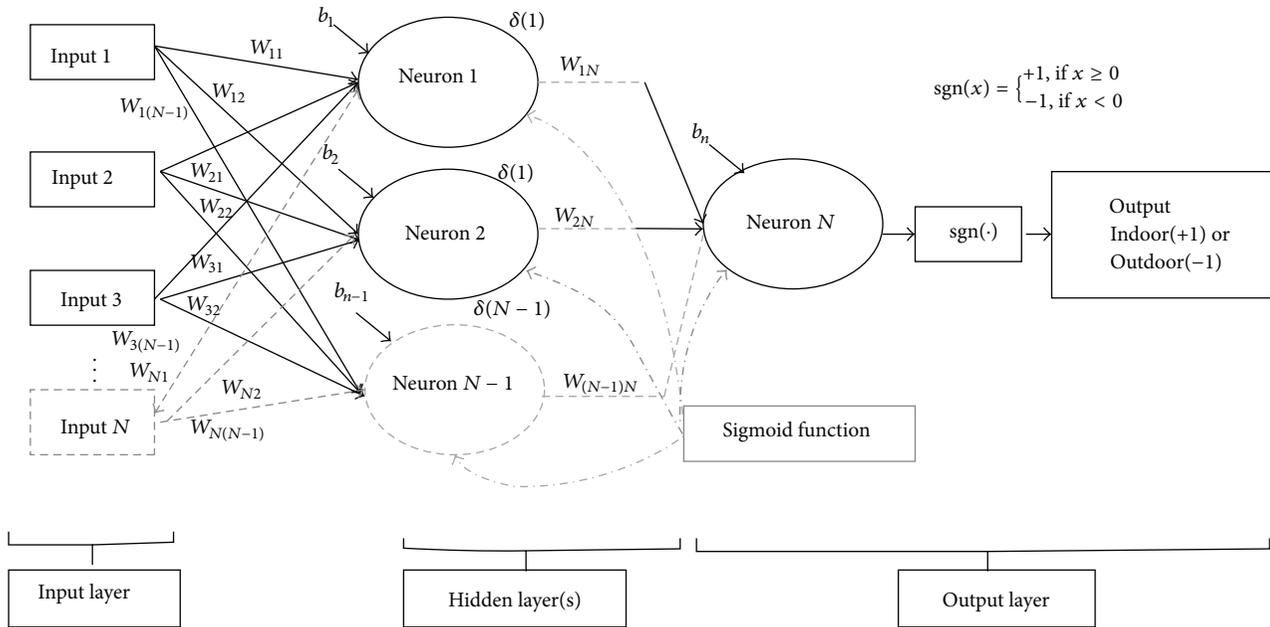


FIGURE 3: Structure of MLPFFBP in the proposed technique.

methods, association rule learning, deep learning, dimensionality reduction, ensemble methods, and artificial neural network [21]. However, in machine learning algorithms themselves, there is no perfect model, just a good enough model depending on how the application layout is designed. ANN has many attractive theoretic properties, specifically, the ability to detect nonpredefined relations such as nonlinear effects and/or interactions. These theoretic advantages come at the cost of reduced interpretability of the model output. Many authors have analysed the same data set, based on these factors, with both standard statistical methods and ANN [22–24].

In the proposed technique, multilayer perceptron feed forward backpropagation (MLPFFBP) neural network is used to categorize the users. MLPFFBP uses error backpropagation to adjust the weights of the neurons. There are two passes in the layers of the network: forward pass and backward pass. The network consists of three layers: input layer, output layer, and the hidden layer. The input layer is fed with initial data. The output layer gives the desired solution. In between there exists a series of hidden layers. The primary layer is connected with the input layer and the last layer is connected to the output layer. Each subsequent layer is connected with the previous layer. Based on the network design, each hidden layer consists of multiple numbers of neurons. The neurons use differentiable transfer function to generate the output. During the training period, the input and output values of the network are specified and based on these values and the hidden layer builds up a set of weights for the neurons [25].

The differentiable transfer function (*tansig*) used here is a sigmoid function. In multilayer sigmoid function, if the input vector is very large, the weight becomes so small to prevent the transfer function being saturated. Thus, the gradient will be very small and the neural network will

be very slow. On the contrary, higher number of training samples with higher number of neurons makes the network more accurate but such a process makes the network bulky and time-consuming. For this, preprocessing steps are added in-between the input layers and the hidden layers. The performance of the neural network is made more effective by using a preprocessing step in training sample selection. In this case, AP clustering algorithm is used to select the best-suited samples for the network training.

In Figure 3, $b_1, b_2, \dots, b_{n-1}, b_n$ and $w_{11}, w_{12} \dots w_{21}, w_{22} \dots w_{N(N-1)}, w_{NN}$ are the biases and the weights of the network nodes, respectively. Biases are also considered the primary weights that are initially put as 1. Moreover, “signum” function is used to compute the actual response of the perceptron. The final output from the last neuron passes through the “signum” function that gives the binary output.

The transfer function is

$$\varphi(v) = \frac{1}{1 + \exp(-v)}. \quad (5)$$

The signum function is

$$\text{sgn}(x) = \begin{cases} +1, & \text{if } x \geq 0 \\ -1, & \text{if } x < 0. \end{cases} \quad (6)$$

The weights are calculated as

$$w(n+1) = w(n) + \alpha * w(n-1) + \eta * \delta(n) * y, \quad (7)$$

$$\delta(n) = \varphi'(v) * (d - y),$$

where α , η , d , y , and δ are the mobility factor, the training parameter, the desired output, the real output, and the local gradient for the nodes of the network, respectively [26, 27].

After the training process of the network, the femtocell takes the 6-element antennas received power as input and gives the category of the users in the output.

2.2. Affinity Propagation Algorithm for Selecting the Best Samples. AP algorithm is a recent clustering algorithm proposed by Frey and Dueck [28]. It is widely accepted because of its high quality set of cluster samples. The proposed user classification in neural network is a supervised technique. The performance of the network is subjected to the nature and quantity of the training samples. Higher number of training samples led to precise values of the neurons' weight, but it makes the training process slower. Clustering of data set based on similarities is a vital step in data analysis problem. A common practice in both supervised and unsupervised learning is to cluster the data based on the similarities [29, 30]. Affinity propagation (AP) is the latest clustering algorithm that reduces the redundancy of the training data set. It accelerates the computing process of ANN by reducing the sample numbers.

Traditional clustering algorithms follow random selection of initial data subset as exemplars and refine it iteratively. AP takes an input set of pairwise similarities between the data points and finds the clusters based on maximum total similarities between the exemplars and the data points [31]. The real messages are exchanged between the data points until the finest set of exemplars and corresponding clusters progressively emerges. It has a better clustering performance than K -means, K -medians, fuzzy c -means, Hill combining (HC), and self-organizing map (SOM) algorithms [32, 33]. It is computationally efficient and simple to implement and customize. In AP algorithm, all the sample data points are considered a possible candidate to be the desired exemplars. Each step exchanges real-valued messages between them until a superior set of exemplar shows up. Messages are updated based on simple formulae that reflect on the sum-product or max-product. It updates the rules until the magnitude of the messages reflects on the current affinity for choosing another data point as its exemplar. Each data point is considered a node in the network. The process of the algorithm is described briefly below.

Input is a set of pairwise similarities as

$$\{s(i, k)\} = -\|x_i - x_k\|^2, \\ i \neq k \text{ (squared Euclidean distance),}$$

$$\text{where, } (i, k) \in \{1, \dots, N\}^2, \quad s(i, k) \in \mathfrak{R}. \quad (8)$$

Here $s(i, k) \in \mathfrak{R}$ indicates how well suited the data point k is as an exemplar for data point i .

For each data point k , a real number $s(k, k)$ represents the preference that is to be considered as an exemplar:

$$s(k, k) = \rho, \quad \forall k \in \{1, \dots, N\}. \quad (9)$$

Initialization: set availabilities to zero, for all i, k : $a(i, k) = 0$.

TABLE 1: System parameters.

System parameters	Value/range
Frequency	2.53 GHz
Number of training indoor users	10
Number of training outdoor users	15
Number of randomly placed users (after training)	20
Femtocell antenna height	1 m
User equipment height	1 m
Frequency	2.6 GHz
UE transmit power (fixed)	13 dBm
Indoor wall loss	5 dB
Outdoor wall loss	10 dB
Shadow fading std.	6 dB
White noise power density	-174 dBm/Hz
Number of neurons in hidden layer	10

Repeat responsibility and availability updates until convergence:

$$\forall i, k : r(i, k) = s(i, k) - \max [s(i, k') + a(i, k')],$$

$$\forall i, k : a(i, k)$$

$$= \begin{cases} \sum \max [0, r(i', k)], & \text{for } k = i \\ \min [0, r(k, k) \\ + \sum_{i': i' \notin \{i, k\}} \max [0, r(i', k)], & \text{for } k \neq i. \end{cases} \quad (10)$$

Output is assignments $\hat{c} = (\hat{c}_1, \dots, \hat{c}_N)$, where $\hat{c}_i = \arg \max_k [a(i, k) + r(i, k)]$. Here \hat{c}_i indexes the cluster's exemplar at which point i is assigned. If point i is a cluster with point k as the exemplar, then $\hat{c}_i = k$ and $\hat{c}_k = k$ [34].

3. Results and Discussions

A layout of functioning area is modelled with a femtocell in the middle of the house. Six-microstrip antennas are operating with 60° separation angle on the same axis inside the femtocell device. A previously designed microstrip antenna is used here to configure the directive gain pattern of each antenna element [17]. The house has indoor and outdoor walls that decrease the strength of the signal based on their thickness. Initially random indoor and outdoor users are generated and the received powers are measured. The dimension of the house is set to $7 \text{ m} \times 6 \text{ m}$. In Figure 4(a) the users and the house are plotted in a $20 \text{ m} \times 20 \text{ m}$ window. The radiation pattern of the microstrip antenna is shown in Figure 4(b).

To demonstrate the performance of the technique, ANN is initially trained without using AP clustering. Random samples are generated by varying the numbers of indoor and outdoor users. In the performance analysis stage, again, random samples are generated to categorize users using the previous experiences. The system parameters that have been used in the simulation are given in Table 1. In the model, the outdoor wall loss is considered higher than the indoor wall.

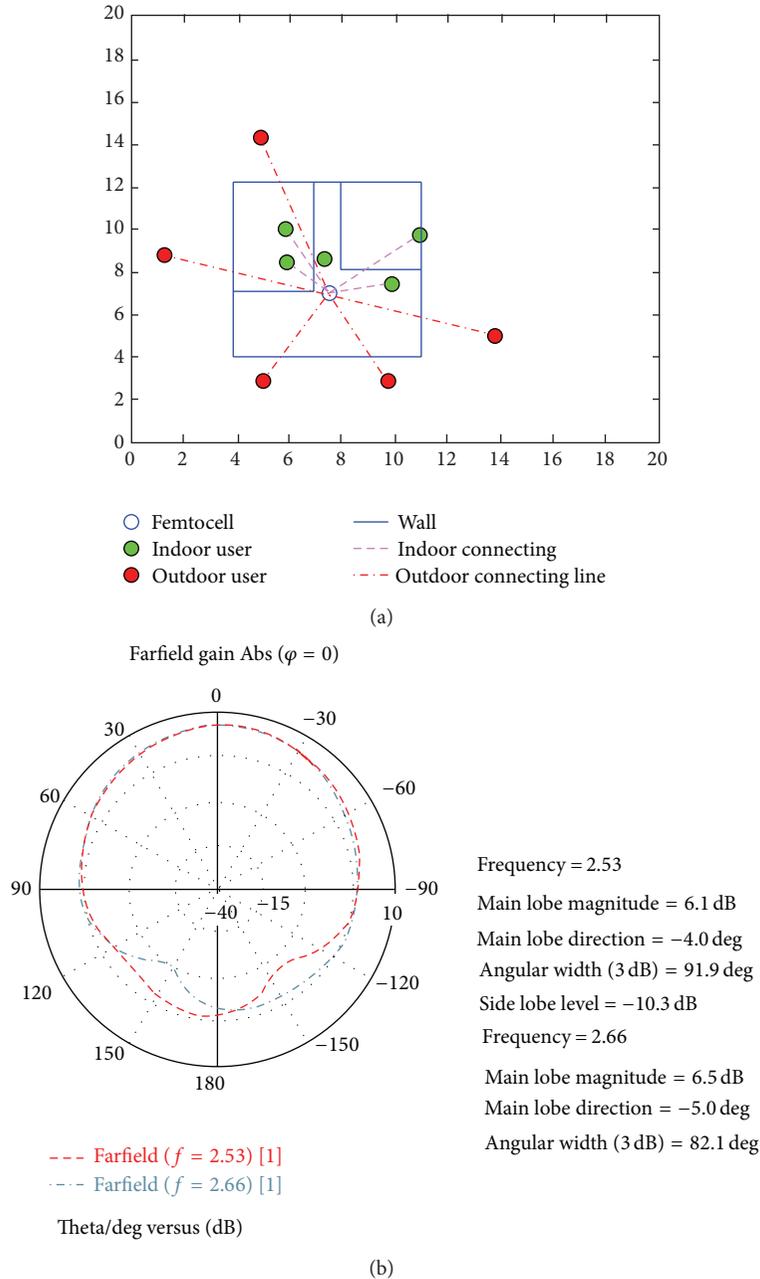


FIGURE 4: (a) Layout of the simulation environment. (b) Radiation pattern of the microstrip antenna at 2.53 GHz and 2.66 GHz.

One of the reasons is that usually the outdoor walls are thicker than the indoor wall with more concrete and steel materials for the foundation or shape. This increases the loss exponent of the outdoor walls. Another reason is that outdoor walls are more subjected to rust and moist from the environment that weakens the incoming signal [35].

3.1. Femtocell Network Performance with ANN. Figure 5(a) shows the training stage of the femtocell device. The red dots are the outdoor users and green dots are the indoor users. In Figure 5(b), random users are generated for the

femtocell to classify the indoor and outdoor users by using the learning experience. Femtocell only allows connection to the indoor users to be connected. The green connecting lines between the femtocell and the indoor users confirm the proper recognition of the users.

Figures 6(a) and 6(b) show the training state and performance validation state for a simulation with 10 indoor and 15 outdoor training samples. The minimum gradient of the ANN is set to 1×10^{-6} . In this particular iteration, ANN takes 38 epochs to train up and adjust the values of biases and weights to achieve the minimum gradient value. The “validation graph” shows a downward curve. It confirms that,

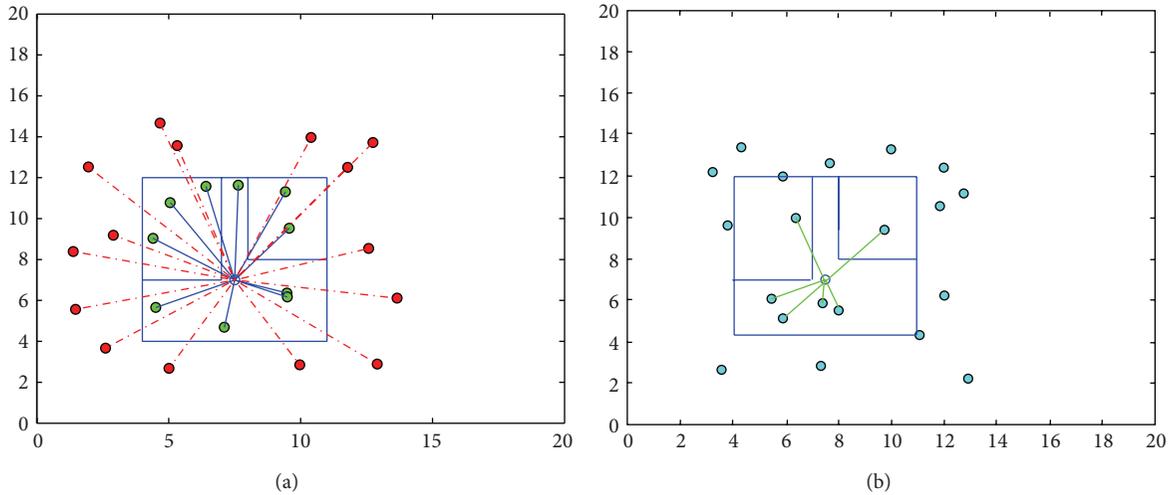


FIGURE 5: (a) Training and (b) testing of the femtocell device.

TABLE 2

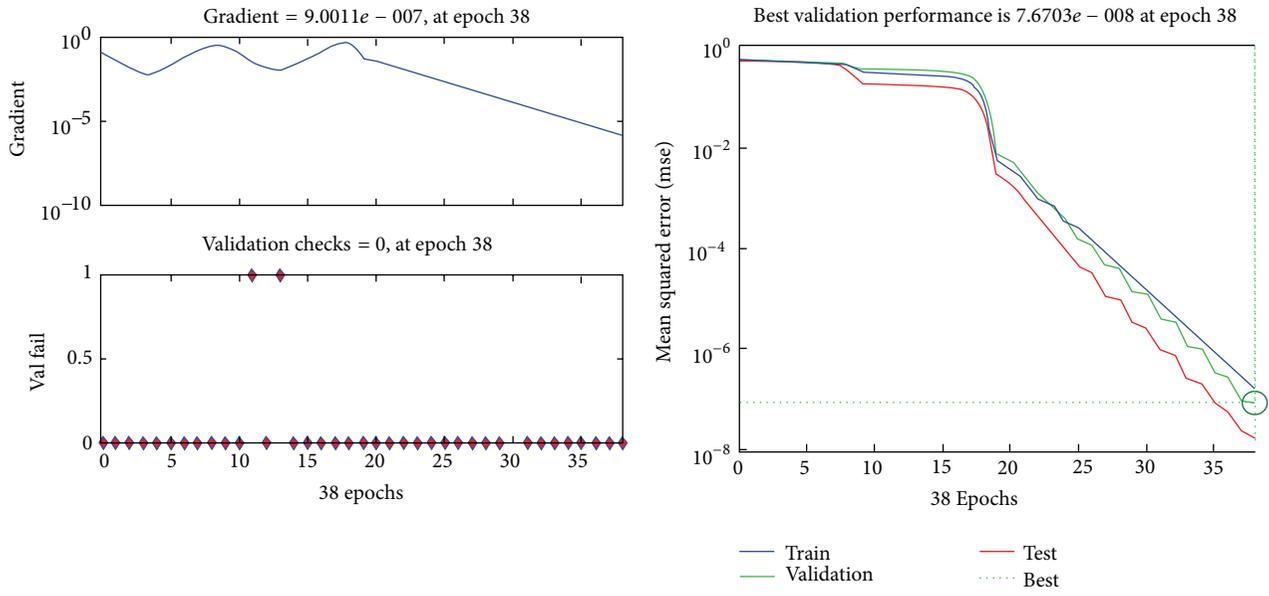
Number of samples		ANN training performance		AP + ANN training performance			Performance comparison after AP	
Indoor	Outdoor	Training time (sec.)	Number of epochs*	AP clustering time (sec.)	Total training time (sec.)	Number of epochs*	Training time decreases (%)	Number of epochs decreases (%)
5	10	1.5165	21	0.2885	0.7647	13	49.57	38.09
6	11	1.5655	21	0.3633	0.8257	14	47.25	33.33
7	12	1.6221	22	0.3921	0.9071	14	44.07	36.36
8	13	1.6416	23	0.4172	0.934	15	43.10	34.78
9	14	1.6443	24	0.4226	0.9504	16	42.20	33.33
10	15	1.6504	26	0.4244	0.9611	16	41.76	38.46
11	16	1.6541	26	0.439	0.9803	17	40.73	34.61
12	17	1.6669	27	0.4461	0.9848	18	40.92	33.33
13	18	1.6709	27	0.4556	0.9951	19	40.44	29.62
14	19	1.6801	28	0.4671	0.9951	19	40.77	32.14
15	20	1.6819	28	0.4702	0.9964	19	40.75	32.14
16	21	1.688	29	0.4704	0.9982	19	40.86	34.48
17	22	1.7322	30	0.4997	1.0017	21	42.17	30
18	23	1.7767	31	0.5073	1.0397	22	41.48	29.03
19	24	1.7966	32	0.5234	1.0695	22	40.47	31.25
20	25	1.7996	34	0.5484	1.0511	22	41.59	35.29

*The fraction values of the epochs are expressed by the nearest integer values.

after every epoch, the latest values of the weights and biases validate the previous training samples.

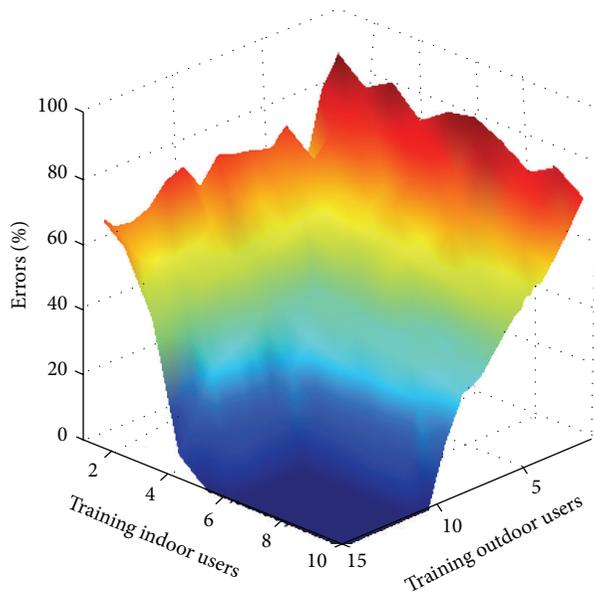
Figure 6(c) shows the performance of the femtocell in percentage of error for different number of outdoor and indoor training user samples. In every iteration, the network is tested using 20 random users to verify the performance. In both types of users, the error rate is quite high at the beginning. Due to lack of knowledge of the users' behaviour, the system cannot categorize the nature of the randomly created users. For the same number of indoor users, outdoor users' percentage of error rate is higher. This is because of the unpredictable nature of wireless signal propagation

from the outdoor users end. The outdoor walls, their shapes, and constructing materials also add more variations in the outdoor users signal strength due to absorption losses and diffraction loss. As a result, the ANN requires higher number of outdoor users training samples for categorizing the users. However, after 5 indoor and 10 outdoor user samples, the network reaches the perfection with error-free user detection. It shows that the performance of the indoor sample is better than the outdoor sample. In the indoor situation, the variation of the signal strength is limited to a certain bound. The effects of indoor free-space loss, refraction, diffraction, reflection, aperture-medium coupling loss, and absorption



(a)

(b)



(c)

FIGURE 6: (a) Training state and (b) performance of best validation. (c) Performance of femtocell for different number of samples.

are comparatively smaller which allows the system to verify any random users signal strength within a certain variation of received power strength. Nevertheless, the number of the sample users always depends on the geographical shape of the houses. The system requires higher number of indoor samples when the variation bounds overlap with the outdoor users variation bound. Such a case is studied below.

The proposed method is now tested in a more complex scenario. A “U” shaped house layout is designed to test the performance of the system. In this layout, indoor wall is ignored. Figures 7(a) and 7(b) show the training and testing process of the femtocell network. The challenging shape of

the house makes the user pattern more improvised than the previous one. In this case, the system requires higher number of indoor and outdoor training user samples to reach an error-free performance. Figure 7(c) shows the required number of indoor and outdoor users against the percentage of error occurrences in detecting the users’ category. Here the required number of users for both categories is above 25 users. The rest of the performance analysis of the process is done using the previous layout of the house.

3.2. Femtocell Network Performance with ANN and AP Clustering Algorithm. AP algorithm clusters the users into

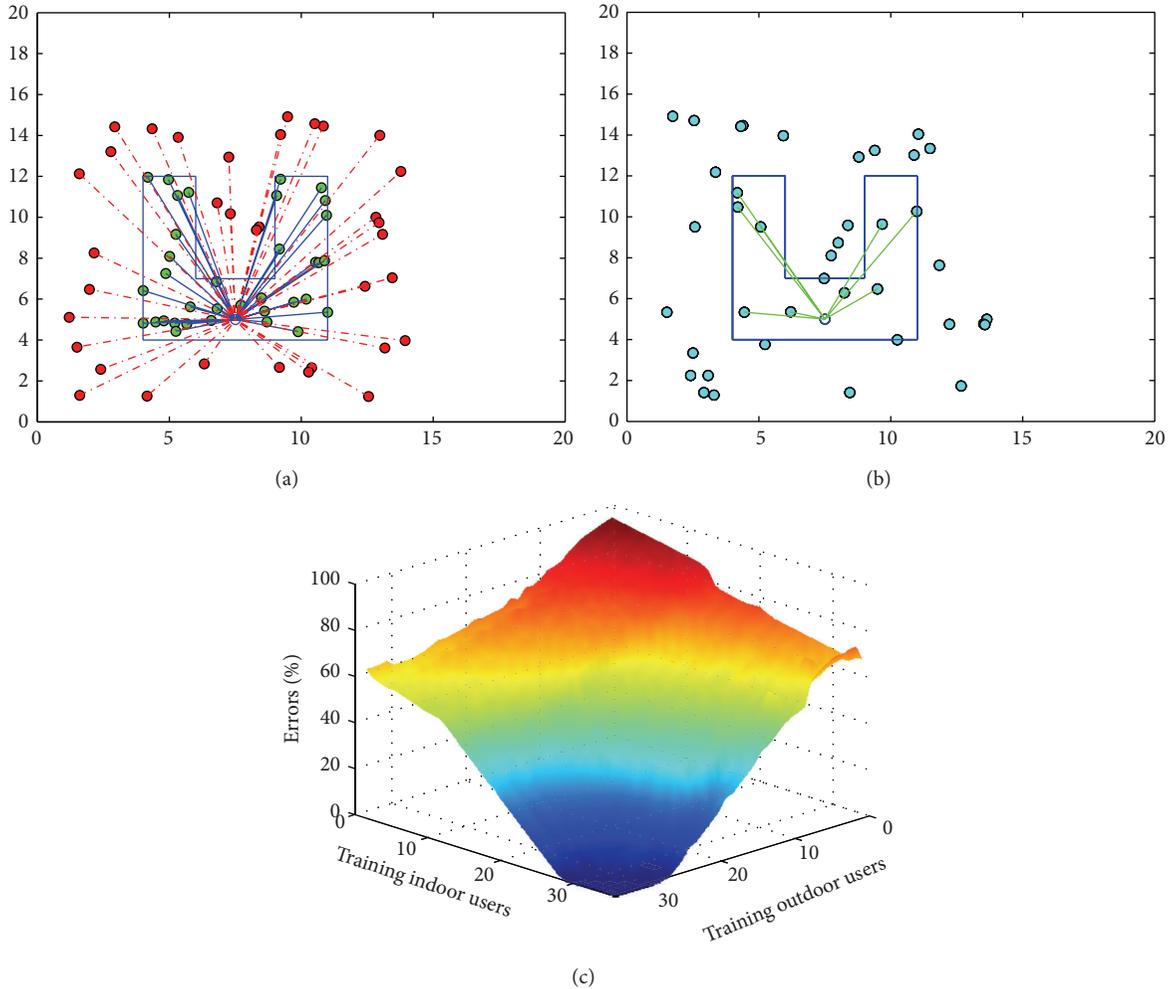


FIGURE 7: (a) Training and (b) testing process of the femtocell with “U” shaped house. (c) Performance of femtocell for different number of samples.

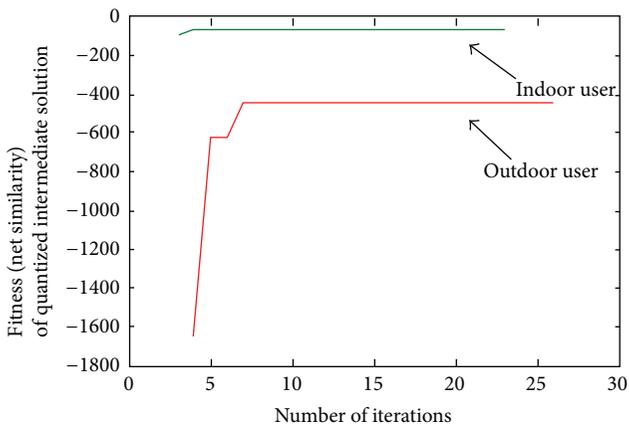


FIGURE 8: Fitness of AP clustering algorithm for indoor and outdoor users.

subgroups based on their power pattern and selects a representative from each subgroup. Unlike other clustering

methods, AP algorithm selects the clusters/subgroups based on the samples nature. If the nature of the sample varies immensely, the number of clusters gets higher. The clustering performance of the AP algorithm is presented in Figure 8 as a form of achieved fitness (net similarities) with respect to the iteration number. Both the outdoor and indoor users reach their best fitness before 8 iterations. However, a safe margin of 25 is kept to ensure the best fitness for both types of users.

Figure 9(a) shows the general ANN training process. During the training, the ANN adjusts the values of the weights and the biases of the network. In Figure 9(b), the AP algorithm clusters the users based on their similarities: power pattern. A representative has been chosen among the data points of a subgroup, which has most of similarities with the other data points of the subgroup. There might also exist subgroups with only one data point. Figure shows that, instead of training ANN with 15 outdoor users and 10 indoor users, the AP selects 3 outdoor users and 3 indoor users. Figures 9(c) and 9(d) show the performance of the network with and without AP algorithm. For a random simulation, both processes show the same accuracy.

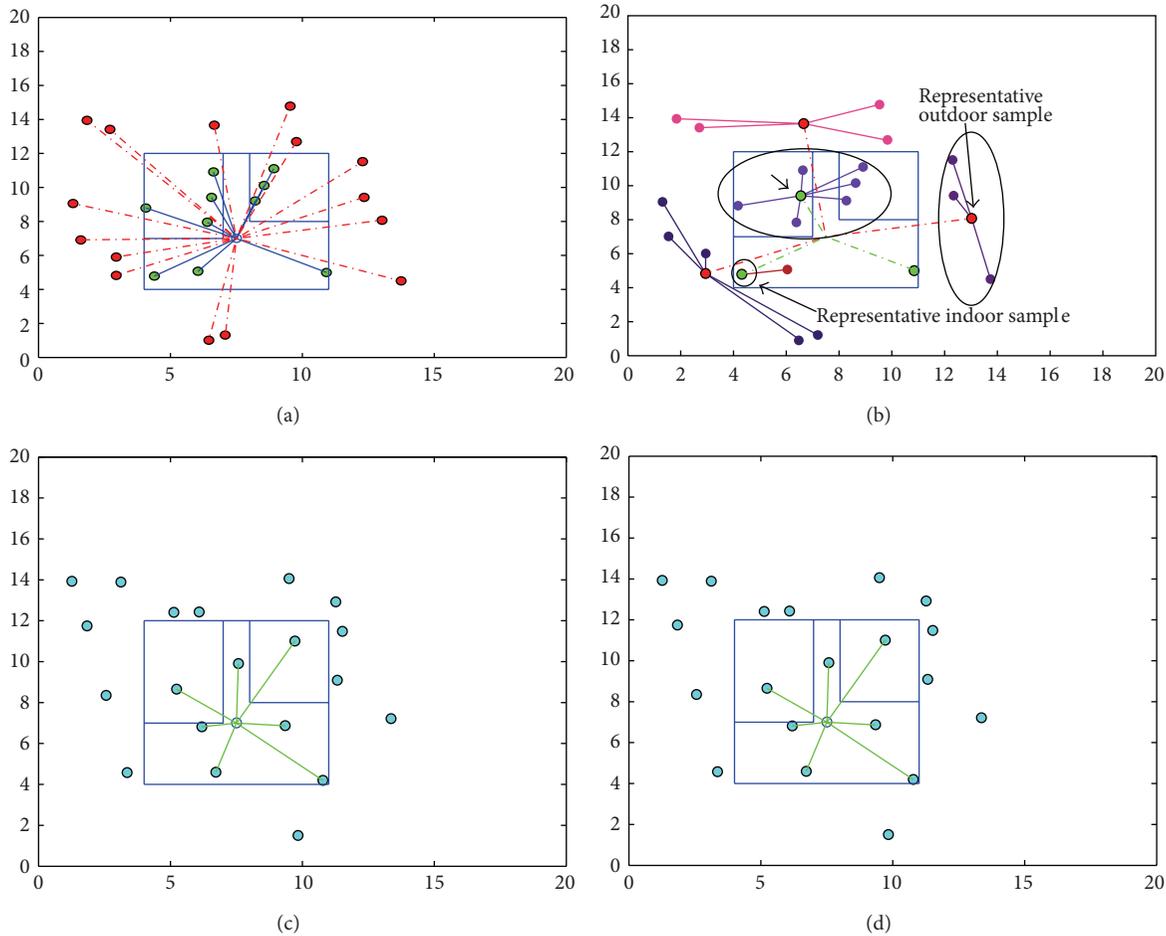


FIGURE 9: (a) Training with ANN. (b) Training with ANN+AP. (c) Performance of the network with ANN training. (d) Performance of the network with ANN+AP training.

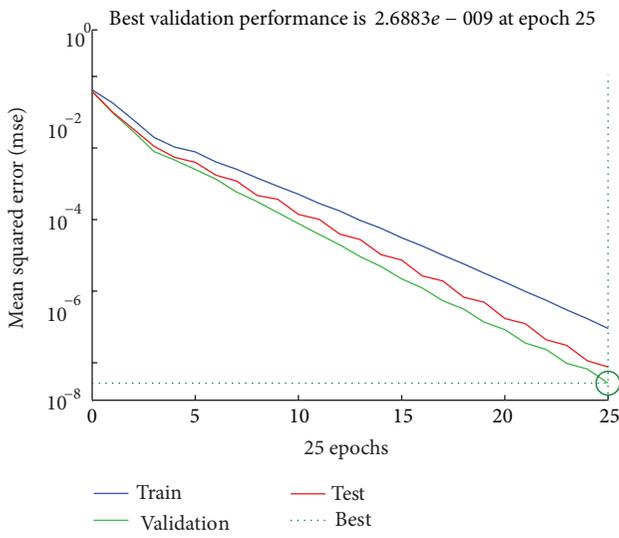
Results show that training the ANN in corporation with AP clustering requires less number of training samples. The process takes less number of epochs to reach the gradient's threshold value. For the above simulation, the ANN took 25 epochs while it took 12 epochs using AP clustered samples. The representative of the data points helps the ANN to explore all the possible variations of the characters of the users' power pattern and guide the network to balance the values of weights and the biases with a faster time interval. Figures 10(a) and 10(b) show the mean square rate (MSE) of the training process. Due to higher number of sample data points, the accuracy of the regular ANN training is more precise. However, in the training process with clustered data samples, the mean square error decreases drastically and gets to the desired value with less number of epochs. In Figures 10(c) and 10(d), the validation check shows a good fitness since the number of indoor and outdoor users is chosen from the error-free region achieved in the result in Figure 6(c).

The performance analysis of both processes is shown in Table 2. Randomly, 20 users have been generated every time to test the performance of the network. Each resultant data is

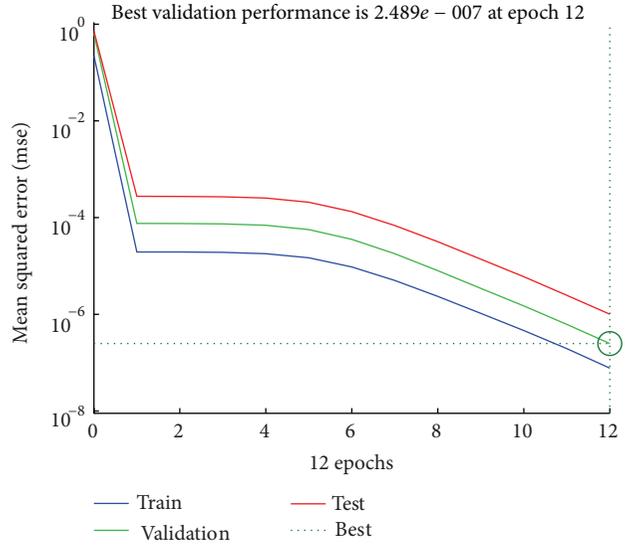
an average value of 1000 simulations. The AP+ANN training process takes around 75%–85% less time than the regular ANN training process; meanwhile AP clustering process takes some additional time which makes the total AP+ANN time around 50%–60% less than ANN regular training time. After AP algorithm implementation, the number of epochs also decreases down to 40%. The fraction values of the epochs in Table 2 are expressed by the nearest integer value.

3.3. AP Clustering Algorithm versus K-Means Clustering Algorithm and Fuzzy c-Means Clustering. To justify the selection of AP clustering algorithm over the traditional clustering algorithm, two popular algorithms, K-means and fuzzy c-means clustering, are compared with AP clustering in the ANN training process.

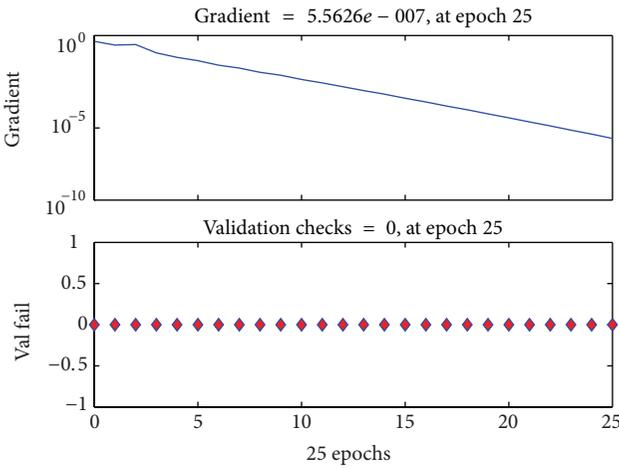
K-Means. K-means is one of the simplest unsupervised learning algorithms that solves the well-known clustering problems. It partitions the data set into k mutually exclusive clusters and returns the index of the cluster to which it



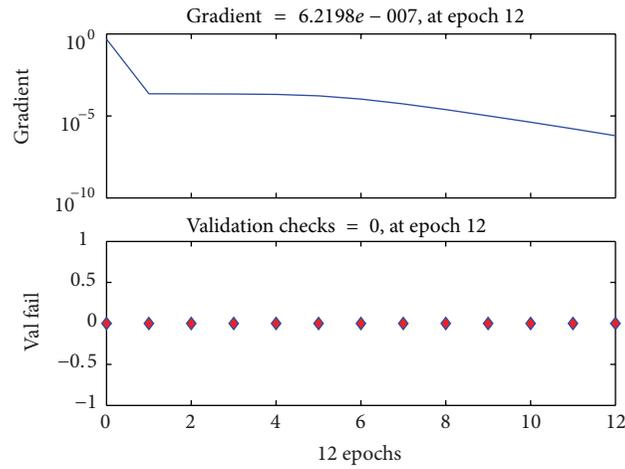
(a)



(b)



(c)



(d)

FIGURE 10: (a) Best validation performance of ANN. (b) Best validation performance of ANN+AP. (c) Training state of ANN. (d) Training state of ANN+AP femtocell network.

has assigned each observation. Unlike AP, *K*-means creates a single level of clusters and needs the number of clusters assigned before the execution. The algorithm breaks the data set into *k* different clusters. If it is unable to find *k* clusters, it breaks the data set into *k* - 1 clusters. Initially it takes *k* number of random observation data set, which is considered the seeds of the algorithm. Then, it assigns all the other observations to *k* seeds based on their proximity to the seeds. In general sense, the algorithm takes a set of objects *S* and an integer *k* and gives a partition of *S* into subsets S_1, \dots, S_k defined by *k* cluster centroid locations or centres [36].

Fuzzy c-Means. The central idea in fuzzy clustering is the nonunique partitioning of the data in a collection of clusters. Like *K*-means, fuzzy *c*-means creates a single level of clusters and needs the number of clusters assigned before the

execution. Cluster centres are randomly initialized and data point (x_i) assigned into clusters ($C_j, j = 1$ to k). Distance metric (Euclidean distance) calculate how far away a point is from a cluster centre. When all data points have been assigned to clusters, new cluster centres (centroids) are calculated. The process of calculating cluster memberships and recalculating cluster centres continues until the cluster centres no longer change from one cycle to the next [37, 38].

Figures 11(b), 11(c), and 11(d) illustrate the representative selection process of AP, *K*-means, and fuzzy *c*-means clustering algorithm in the functioning area. The green dots show the indoor representative points of the data set while the red dots represent the outdoor. In both *K*-means and fuzzy *c*-means, the centroid points are not user data sample; it is a point of each cluster that has a minimum value distance from

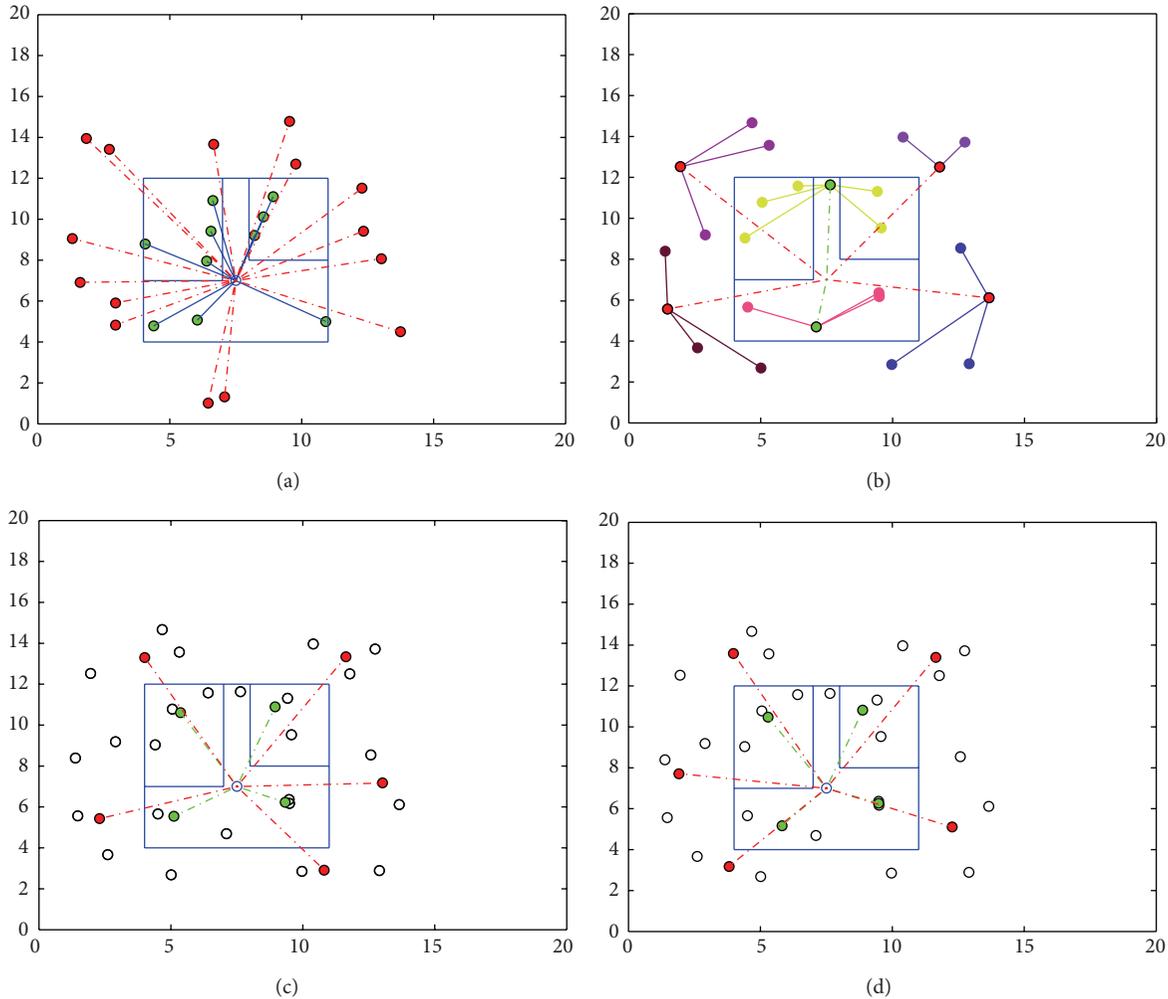


FIGURE 11: (a) Position of the sample indoor and outdoor users. (b) AP clustering of indoor and outdoor users based on signal strength. (c) K -means clustering with 9 clusters. (d) Fuzzy c -means clustering with 9 clusters.

each of the members of the clusters. In the case of K -means, it just executes the distance calculation, whereas fuzzy c -means needs to do a full inverse-distance weighting. To obtain the error-free performance in the ANN, K -means and fuzzy c -means require different number of clusters each time. A little comparison of the performance is shown in Table 3.

K -means minimizes the sum of distances from each data points to its cluster centroid. The process repeats until the sum of distances cannot be decreased further. This process takes more time than AP. On the other hand, K -means needs to do a distance calculation, whereas fuzzy c -means needs to do a full inverse-distance weighting. Fuzzy c -means thus performs slower than both clustering algorithms in this particular case. However, for higher number of data samples, the time increment is a little less than the AP clustering algorithm. Although the overall clustering time of AP algorithm is always less by a fair distance, the number of clusters has to be determined maintaining the same accuracy of the ANN output. Except AP algorithm, the challenge in the other clustering processes mostly lies in selecting the number

of clusters to perform an error-free training. On this note, AP algorithm is the best candidate in this process as it selects the number of clusters by itself analysing the samples in every simulation.

4. Conclusion

This paper proposed a novel technique to classify the users in closed access femtocell network by using ANN and AP clustering algorithm. The technique is developed using a multielement antenna femtocell device. The power pattern of each user is used to distinguish different level of users. A machine learning process is adopted by using ANN to inaugurate the user recognition feature in the femtocell. After using a certain number of user samples, the femtocell successfully recognizes the indoor and outdoor users. In the later part, AP clustering algorithm is included along with ANN to speed up the training process. Performance analysis shows that the femtocell takes less time to recognize user

TABLE 3

Number of samples		ANN + AP performance		ANN + K-means performance		ANN + fuzzy <i>c</i> -means clustering	
Indoor	Outdoor	Number of samples for error-free operation*	Clustering + training time (sec.)	Number of samples for error-free operation*	Clustering + training time (sec.)	Number of samples for error-free operation*	Clustering + training time (sec.)
5	10	6	0.7647	8	1.2516	8	1.3712
10	15	6	0.9611	9	1.3354	8	1.4157
15	20	7	0.9964	9	1.3847	9	1.4869
20	25	7	1.0511	9	1.4964	9	1.5738

*The fraction values of the epochs are expressed by the nearest integer values.

without compromising the accuracy. Finally, a comparison of AP clustering, *K*-means clustering, and fuzzy *c*-means is showed in the user classification process to justify the selection of AP clustering method. The result shows for same simulation that both *K*-means and fuzzy *c*-means consume more time and give less efficiency.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The authors would like to thank the Ministry of Higher Education Research Grant FRGS/1/2014/TK03/UKM/01/1 for sponsoring this work.

References

- [1] V. Chandrasekhar, J. G. Andrews, and A. Gatherer, "Femtocell networks: a survey," *IEEE Communications Magazine*, vol. 46, no. 9, pp. 59–67, 2008.
- [2] A. Rath, S. Hua, and S. S. Panwar, "FemtoHaul: using femtocells with relays to increase macrocell backhaul bandwidth," in *Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM '10)*, March 2010.
- [3] D. López-Pérez, A. Valcarce, G. de La Roche, and J. Zhang, "OFDMA femtocells: a roadmap on interference avoidance," *IEEE Communications Magazine*, vol. 47, no. 9, pp. 41–48, 2009.
- [4] H. Widiarti, S. Pyun, and D. Cho, "Interference mitigation based on femtocells grouping in low duty operation," in *Proceedings of the IEEE 72nd Vehicular Technology Conference Fall (VTC-Fall '10)*, pp. 1–5, September 2010.
- [5] A. U. Ahmed, M. T. Islam, M. Ismail, and M. Ghanbarisabagh, "Dynamic resource allocation in hybrid access femtocell network," *The Scientific World Journal*, vol. 2014, Article ID 539720, p. 7, 2014.
- [6] G. de La Roche, A. Valcarce, D. López-Pérez, and J. Zhang, "Access control mechanisms for femtocells," *IEEE Communications Magazine*, vol. 48, no. 1, pp. 33–39, 2010.
- [7] H. A. Mahmoud, I. Guvenc, and F. Watanabe, "Performance of open access femtocell networks with different cell-selection methods," in *Proceedings of the IEEE 71st Vehicular Technology Conference (VTC '10-Spring)*, pp. 1–5, Taipei, Taiwan, May 2010.
- [8] P. Xia, V. Chandrasekhar, and J. G. Andrews, "Open vs. closed access femtocells in the uplink," *IEEE Transactions on Wireless Communications*, vol. 9, no. 12, pp. 3798–3809, 2010.
- [9] W. Zheng, H. Zhang, X. Chu, and X. Wen, "Mobility robustness optimization in self-organizing LTE femtocell networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2013, article 27, no. 1, 2013.
- [10] H. Claussen and F. Pivitt, "Femtocell coverage optimization using switched multi-element antennas," in *Proceedings of the IEEE International Conference on Communications (ICC '09)*, pp. 1–6, Dresden, Germany, June 2009.
- [11] A. Cabedo, J. Anguera, C. Picher, M. Ribó, and C. Puente, "Multiband handset antenna combining a PIFA, slots, and ground plane modes," *IEEE Transactions on Antennas and Propagation*, vol. 57, no. 9, pp. 2526–2533, 2009.
- [12] A.-H. Tsai, L.-C. Wang, J.-H. Huang, and R.-B. Hwang, "High-capacity OFDMA femtocells by directional antennas and location awareness," *IEEE Systems Journal*, vol. 6, no. 2, pp. 329–340, 2012.
- [13] S. Al-Rubaye, A. Al-Dulaimi, and J. Cosmas, "Cognitive femtocell," *IEEE Vehicular Technology Magazine*, vol. 6, no. 1, pp. 44–51, 2011.
- [14] M. Agatonović, Z. Stanković, N. Dončova, L. Sit, B. Milanović, and T. Zwick, "Application of artificial neural networks for efficient high-resolution 2D DOA estimation," *Radioengineering*, vol. 21, p. 1179, 2012.
- [15] D. Inserra and A. M. Tonello, "A multiple antenna wireless testbed for the validation of DoA estimation algorithms," *AEU—International Journal of Electronics and Communications*, vol. 68, no. 1, pp. 10–18, 2014.
- [16] T. S. G. Basha, M. N. G. Prasad, and P. V. Sridevi, "Hybrid technique for beam forming in smart antenna with spatial diversity," *International Journal of Wireless and Mobile Computing*, vol. 5, no. 2, pp. 126–136, 2012.
- [17] A. U. Ahmed, M. T. Islam, R. Azim, M. Ismail, and M. F. Mansor, "Microstrip antenna design for femtocell coverage optimization," *International Journal of Antennas and Propagation*, vol. 2014, Article ID 480140, 8 pages, 2014.
- [18] S. Promwong and J.-I. Takada, "Free space link budget estimation scheme for ultra wideband impulse radio with imperfect antennas," *IEICE Electronic Express*, vol. 1, pp. 188–192, 2004.
- [19] A. U. Ahmed, M. T. Islam, and M. Ismail, "A review on femtocell and its diverse interference mitigation techniques in heterogeneous network," *Wireless Personal Communications*, pp. 1–22, 2014.
- [20] D. F. Specht, "Probabilistic neural networks," *Neural Networks*, vol. 3, no. 1, pp. 109–118, 1990.

- [21] J. Wang, P. Urriza, Y. Han, and D. Cabric, "Weighted centroid localization algorithm: theoretical analysis and distributed implementation," *IEEE Transactions on Wireless Communications*, vol. 10, no. 10, pp. 3403–3413, 2011.
- [22] J. Benedicto, S. Dinwiddy, G. Gatti, R. Lucas, and M. Lugert, *GALILEO: Satellite System Design*, European Space Agency, 2000.
- [23] W. G. Griswold, R. Boyer, S. W. Brown et al., *ActiveCampus: Sustaining Educational Communities through Mobile Technology*, Department of Computer Science and Engineering, University of California, San Diego, Calif, USA, 2002.
- [24] T. Ogawa, S. Yoshino, and M. Shimizu, "Location determination method for wireless systems based on learning vector quantization," *NTT Technical Review*, vol. 1, no. 9, pp. 27–36, 2003.
- [25] P. S. Roy and S. Chakraborty, "Design of C-slotted microstrip antenna using artificial neural network model," *International Journal for Research in Science & Advanced Technologies*, vol. 2, 2012.
- [26] L. Fausett, *Fundamentals of Neural Networks: Architectures, Algorithms, and Applications*, Prentice-Hall, New York, NY, USA, 1994.
- [27] M. T. Hagan, H. B. Demuth, and M. H. Beale, *Neural Network Design*, Pws, Boston, Mass, USA, 1996.
- [28] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.
- [29] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man and Cybernetics C: Applications and Reviews*, vol. 37, no. 6, pp. 1067–1080, 2007.
- [30] J. Zhao, Y. Zhang, and M. Ye, "Research on the received signal strength indication location algorithm for RFID system," in *Proceedings of the International Symposium on Communications and Information Technologies (ISCIT '06)*, pp. 881–885, Bangkok, Thailand, October 2006.
- [31] W.-S. Lai, M.-E. Chiang, S.-C. Lee, and T.-S. Lee, "Game theoretic distributed dynamic resource allocation with interference avoidance in cognitive femtocell networks," in *Proceedings of the IEEE Wireless Communications and Networking Conference (WCNC '13)*, pp. 3364–3369, Shanghai, China, April 2013.
- [32] D. Demb le and P. Kastner, "Fuzzy C-means method for clustering microarray data," *Bioinformatics*, vol. 19, no. 8, pp. 973–980, 2003.
- [33] T. Kohonen, E. Oja, O. Simula, A. Visa, and J. Kangas, "Engineering applications of the self-organizing map," *Proceedings of the IEEE*, vol. 84, no. 10, pp. 1358–1384, 1996.
- [34] J. Meinil , P. Ky sti, T. J ms , and L. Hentil , "WINNER II channel models," in *Radio Technologies and Concepts for IMT-Advanced*, pp. 39–92, 2009.
- [35] Y. Miura, Y. Oda, and T. Taga, "Outdoor-to-indoor propagation modelling with the identification of path passing through wall openings," in *Proceedings of the 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC '02)*, pp. 130–134, September 2002.
- [36] A. Ghosh, R. Ratasuk, W. Xiao et al., "Uplink control channel design for 3GPP LTE," in *Proceedings of the 18th Annual IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC '07)*, pp. 1–5, Athens, Ga, USA, September 2007.
- [37] N. R. Pal, K. Pal, J. M. Keller, and J. C. Bezdek, "A possibilistic fuzzy c-means clustering algorithm," *IEEE Transactions on Fuzzy Systems*, vol. 13, no. 4, pp. 517–530, 2005.
- [38] R. L. Cannon, J. V. Dave, and J. C. Bezdek, "Efficient implementation of the fuzzy c-means clustering algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 2, pp. 248–255, 1986.

Research Article

A Synchronous-Asynchronous Particle Swarm Optimisation Algorithm

**Nor Azlina Ab Aziz,^{1,2} Marizan Mubin,¹
Mohd Saberi Mohamad,³ and Kamarulzaman Ab Aziz²**

¹ Faculty of Engineering, University of Malaya, 50603 Kuala Lumpur, Malaysia

² Multimedia University, Jalan Ayer Keroh Lama, 75450 Bukit Beruang, Melaka, Malaysia

³ Faculty of Computing, Universiti Teknologi Malaysia, 81310 Johor Bahru, Malaysia

Correspondence should be addressed to Nor Azlina Ab Aziz; azlina.aziz@mmu.edu.my

Received 23 April 2014; Accepted 20 June 2014; Published 10 July 2014

Academic Editor: T. O. Ting

Copyright © 2014 Nor Azlina Ab Aziz et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

In the original particle swarm optimisation (PSO) algorithm, the particles' velocities and positions are updated after the whole swarm performance is evaluated. This algorithm is also known as synchronous PSO (S-PSO). The strength of this update method is in the exploitation of the information. Asynchronous update PSO (A-PSO) has been proposed as an alternative to S-PSO. A particle in A-PSO updates its velocity and position as soon as its own performance has been evaluated. Hence, particles are updated using partial information, leading to stronger exploration. In this paper, we attempt to improve PSO by merging both update methods to utilise the strengths of both methods. The proposed synchronous-asynchronous PSO (SA-PSO) algorithm divides the particles into smaller groups. The best member of a group and the swarm's best are chosen to lead the search. Members within a group are updated synchronously, while the groups themselves are asynchronously updated. Five well-known unimodal functions, four multimodal functions, and a real world optimisation problem are used to study the performance of SA-PSO, which is compared with the performances of S-PSO and A-PSO. The results are statistically analysed and show that the proposed SA-PSO has performed consistently well.

1. Introduction

Particle swarm optimisation (PSO) was introduced by Kennedy and Eberhart in 1995 [1]. It is a swarm-based stochastic optimisation algorithm that mimics the social behaviour of organisms such as birds and fishes. These organisms' success in looking for food source is achieved through individual effort as well as corporation with surrounding neighbours. In PSO, the individuals are represented by a swarm of agents called particles. The particles move within the search area to find the optimal solution by updating their velocity and position. These values are influenced by the experience of the particles and their social interactions. The PSO algorithm has been successfully applied in various fields, such as human tremor analysis for biomedical engineering [2, 3], electric power and voltage management [4], machine scheduling [5], robotics [6], and VLSI circuit design [7].

Since its introduction, PSO has undergone numerous evolutionary processes. Many variations of PSO have been proposed to improve the effectiveness of the algorithm. Some of the improvement involves introduction of a new parameter to the algorithm such as inertia weight [8] and constriction factor [9], while others focus on solving specific type of problems such as multiobjective optimization [10, 11], discrete optimization problems [12, 13], and dynamic optimization problems [14].

Here we focus on the effect of the particles' update sequence on the performance of PSO. In the original PSO, a particle's information on its neighbourhood's best found solution is updated after the performance of the whole swarm is evaluated. This version of PSO algorithm is known as synchronous PSO (S-PSO). The synchronous update in S-PSO provides the perfect information concerning the particles, thus allowing the swarm to choose a better neighbour and

exploit the information provided by this neighbour. However, this strategy could cause the particles to converge too fast.

Another variation of PSO, known as asynchronous PSO (A-PSO), has been discussed by Carlisle and Dozier [15]. In A-PSO, the best solutions are updated as soon as a particle's performance has been evaluated. Therefore, a particle's search is guided by the partial or imperfect information from its neighbourhood. This strategy leads to diversity in the swarm [16], wherein the particles updated at the beginning of an iteration use more information from the previous iteration while particles at the end of the iteration are updated based on the information from the current iteration [17]. In several studies [15, 16, 18], A-PSO has been claimed to perform better than S-PSO. Xue et al. [19] reported that asynchronous updates contribute to a shorter execution time. Imperfect information due to asynchronous updates causes the information of the current best found solution to be communicated to the particles more slowly, thus encouraging more exploration. However, a study conducted by Juan et al. [20] reported that S-PSO is better than A-PSO in terms of the quality of the solution and also the convergence speed. This is due to the stronger exploitation.

The synchronicity of the particles influences exploration and exploitation among the particles [17]. Exploration and exploitation play important roles in determining the quality of a solution. Exploration in asynchronous update ensures that the search space is thoroughly searched so that the area containing the best solution is discovered. However, exploitation in synchronous update helps to fine tune the search so that the best solution can be found. Hence, in this paper, we attempt to improve the PSO algorithm by merging both synchronous and asynchronous updates in the search process so that the advantages of both methods can be utilised. The proposed algorithm, which is named as the synchronous-asynchronous PSO (SA-PSO), divides the particles into smaller groups. These groups are updated asynchronously, while members within the same group are updated synchronously. After the performance of all the particles in a group is evaluated, the velocities and positions of the particles are updated using a combination of information from the current iteration of their own group and the groups updated before them, as well as the information from the previous iteration of the groups that have not yet been updated. The search for the optimal solution in SA-PSO is led by the groups' best members together with the swarm's best. This strategy is different from the original S-PSO and A-PSO, where the search is led by the particles' own experience together with the swarm's best.

The rest of the paper is organised as follows. The S-PSO and A-PSO algorithms are discussed in Section 2. The proposed SA-PSO algorithm is described in detail in Section 3. In Section 4, the performance of the SA-PSO algorithm is evaluated using ten benchmark functions comprising of five unimodal functions, four multimodal functions, and a real world optimisation problem. The results of the tests are presented and discussed in Section 5. Our conclusions are presented in Section 6.

2. Particle Swarm Optimisation

2.1. Synchronous PSO. In PSO, the search for the optimal solution is conducted by a swarm of P particles. At time t , the i th particle has a position, $x_i(t)$, and a velocity, $v_i(t)$. The position represents a solution suggested by the particle while velocity is the rate of change from the current position to the next position. At the beginning of the algorithm, these two values (position and velocity) are randomly initialised. In subsequent iterations, the search process is conducted by updating the position and velocity using the following equations:

$$v_i(t) = \omega v_i(t-1) + c_1 r_1 (pBest_i - x_i(t-1)) + c_2 r_2 (gBest - x_i(t-1)), \quad (1)$$

$$x_i(t) = v_i(t) + x_i(t-1). \quad (2)$$

To prevent the particles from venturing too far from the feasible region, the $v_i(t)$ value is clamped to $\pm V_{\max}$. If the value of V_{\max} is too large, then the exploration range is too wide. Conversely, if the value of V_{\max} is too small, then the particles will favour the local search [21]. In (1), c_1 and c_2 are the learning factors that control the effect of the cognitive and social influence on a particle. Typically, both c_1 and c_2 are set to 2 [22]. Two independent random numbers r_1 and r_2 in the range [0.0, 1.0] are incorporated into the velocity equation. These random terms provide stochastic behaviour to the particles, thus encouraging them to explore a wider area. Inertia weight, ω , which is a term added to improve the PSO's performance, controls the particles' momentum. When a good area is found, the particles can switch to fine tuning by manipulating ω [8]. To ensure convergence, a time decreasing inertia weight is more favourable than a fixed inertia weight [21]. This is because a large inertia weight at the beginning helps to find a good area through exploration and a small inertia weight towards the end—when typically a good area is already found—facilitates fine tuning. The small inertia weight at the end of the search reduces the global search activity [23].

An individual success in PSO is affected not only by the particle's own effort and experience but also by the information shared by its surrounding neighbours. The particle's experience is represented in (1) by $pBest_i$, which is the best position found so far by the i th particle. The neighbours' influence is represented by $gBest$, which is the best position found by the swarm up to the current iteration.

The particle's position, $x_i(t)$, is updated using (2), in which a particle's next search is launched from its previous position and the new search is influenced by the past search [24]. Typically, $x_i(t)$ is bounded to prevent the particles from searching in an infeasible region [25]. The quality of $x_i(t)$ is evaluated by a problem-dependent fitness function. Each of the particles is evaluated to determine its current fitness. If a new position with a better fitness than the current fitness of $gBest$ or $pBest_i$ or both is found, then the new position value will accordingly be saved as $gBest$ or $pBest_i$; otherwise the old best values will be adopted. This update process continues until the stopping criterion is met, when either the

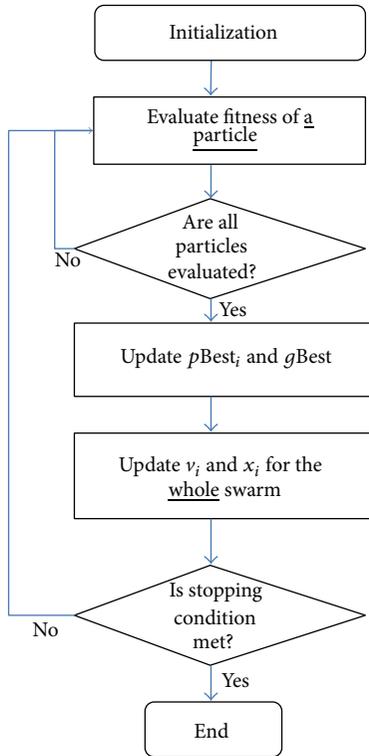


FIGURE 1: S-PSO flowchart.

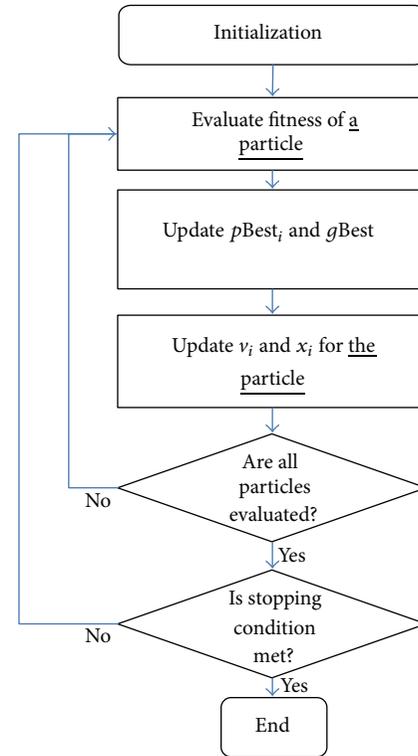


FIGURE 2: A-PSO flowchart.

maximum iteration limit, T , is achieved or the target solution is attained. Therefore, for a swarm with P number of particles, the maximum number of fitness evaluation in a run is $(P \times T)$.

The original PSO algorithm is shown in the flowchart of Figure 1. As shown in the algorithm, the particles' $pBest_i$ and $gBest$ updates are conducted after the fitness of all the particles has been evaluated. Therefore, this version of PSO is known as synchronous PSO (S-PSO). Because the $pBest_i$ and $gBest$ are updated after all the particles are evaluated, S-PSO ensures that all the particles receive perfect and complete information about their neighbourhood, leading to a better choice of $gBest$ and thus allowing the particles to exploit this information so that a better solution can be found. However, this possibly leads the particles in S-PSO to converge faster, resulting in a premature convergence.

2.2. Asynchronous PSO. In S-PSO, a particle must wait for the whole swarm to be evaluated before it can move to a new position and continue its search. Thus, the first evaluated particle is idle for the longest time, waiting for the whole swarm to be updated. An alternative to S-PSO is A-PSO, in which the particles are updated based on the current state of the swarm. A particle in A-PSO is updated as soon as its fitness is evaluated. The particle selects $gBest$ using a combination of information from the current and the previous iteration. This is different from S-PSO, in which all the particles use information from the same iteration. Consequently, in A-PSO, particles of the same iteration might use various values of $gBest$, as it is selected based on the available information during a particle's update process.

The flowchart in Figure 2 shows the A-PSO algorithm. The flow of A-PSO is different than S-PSO; however the fitness function is still called for P times per iteration, once for each particle. Therefore, the maximum number of fitness evaluation is $(P \times T)$. This is similar to S-PSO. The velocity and position are calculated using the same equations as S-PSO.

Other than the variety of information, the lack of synchronicity in A-PSO solves the issue of idle particles faced in S-PSO [26]. An asynchronous update also enables the update sequence of the particles to change dynamically or a particle to be updated more than once [26, 27]. The change in the update sequence offers different levels of available information among the particles, and such differences can prevent the particles from being trapped in local optima [17].

3. The Proposed Synchronous-Asynchronous PSO (SA-PSO)

In this paper, the PSO algorithm is improved by merging both update methods. The proposed algorithm, synchronous-asynchronous PSO (SA-PSO), divides the particles into smaller groups. In S-PSO and A-PSO, the particles learn from their own best experience, $pBest_i$ and $gBest$. However, in the proposed algorithm, instead of using their own experience, the particles learn from their group's performance.

The algorithm proposed is presented in the flowchart shown in Figure 3. The algorithm starts with initialisation of particles. The particles in SA-PSO are divided into C groups, each of which consists of N number of particles. Initially, C central particles, one for each group, are randomly

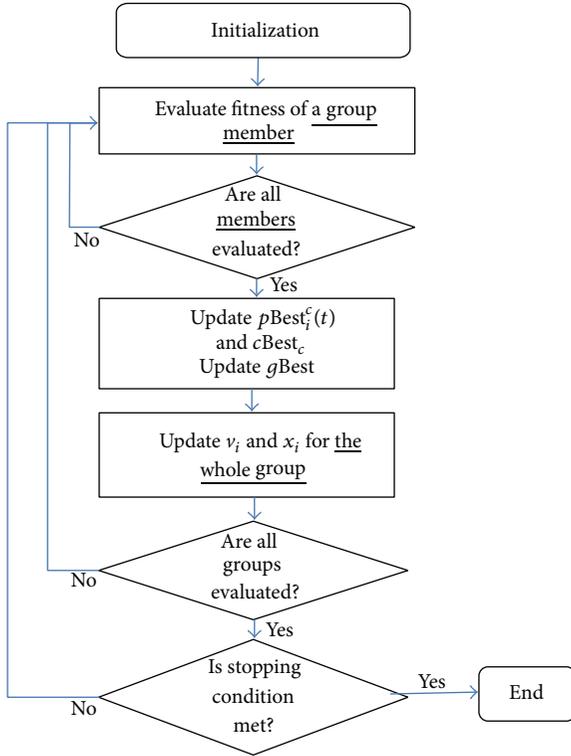


FIGURE 3: SA-PSO flowchart.

initialized within the search space. This is followed by random placement of $N - 1$ number of members for each group. The distances of members are within the radius of $\pm\Delta$ from the central particle of their respective groups. Therefore, Δ is the maximum distance of a particle from the central particle of its group. This parameter is only used once throughout the execution of the algorithm—during the initialisation phase. Group memberships remain fixed throughout the search process. The total number of particles, P , is $C \times N$ for the SA-PSO algorithm.

The groups are updated one by one; that is, asynchronous update is used across groups. The particles from the group that is being updated use three groups of information to update their velocity. The first group of information is the current information of the particles' group members; the particles use this information to try to match their group's best performer. The particles also use recent information from the groups that were updated earlier and information from the previous iteration for the groups to be updated later.

When a group is updated, the group members' velocity and position updates are performed after the whole group performance is evaluated. Therefore, the particles in a group are updated synchronously.

When a group evaluates the performance of its members, the fitness function is called for N times. One by one of the groups' members are updated in an iteration. Since there is C number of groups, hence the fitness function is called for $C \times N$ times, which is equivalent to P times per iteration. Therefore, although the particles in SA-PSO are divided into

TABLE 1: Parameters setting for S-PSO, A-PSO, and SA-PSO.

Parameter	Value
Number of runs for each experiment	500
Number of iterations	2000
Velocity clamping, V_{\max}	4
Range of inertia weight, ω	0.9–0.4
Learning factors	
c_1	2
c_2	2

groups, the maximum number of fitness evaluation per run is the same as S-PSO and A-PSO which is $(P \times T)$.

The velocity at time t of i th particle that belongs to c th group, $v_i^c(t)$, is updated using the following equation:

$$v_i^c(t) = \omega v_i^c(t-1) + c_1 r_1 (cBest_c - x_i^c(t-1)) + c_2 r_2 (gBest - x_i^c(t-1)). \quad (3)$$

Equation (3) shows that the information used to update the velocity are $cBest_c$ and $gBest$. $cBest_c$ is the best member of c th group, where c is $[1, C]$, and it is chosen among the particle's best of c th group, $pBest_i^c$. This value, together with the swarm's best, $gBest$, leads the particles' search in the SA-PSO algorithm. The $gBest$ is updated after all once a new $cBest_c$ outperforms $gBest$. Thus, $gBest$ is the best $cBest_c$. The communication among the groups in SA-PSO is conducted through the best performing member of the groups. The position of the particle, $x_i^c(t)$, is updated using

$$x_i^c(t) = v_i^c(t) + x_i^c(t-1). \quad (4)$$

The algorithm is ended when either the ideal fitness is achieved or maximum iteration is reached.

The SA-PSO algorithm takes advantage of both A-PSO and S-PSO algorithms. In A-PSO, the particles are updated using imperfect information, which contributes to the diversity and exploration. In S-PSO, the quality of the solution found is ensured by evaluating the performance of the whole swarm first. The S-PSO particles are then updated by exploiting this information. The asynchronous update characteristic of A-PSO is imitated by SA-PSO by updating the groups one after another. Hence, members of a group are updated using the information from mixed iterations. This strategy encourages exploration due to the imperfect information. However, the performance of all members of a group in SA-PSO is evaluated first before the velocity and position update process starts. This is the synchronous aspect of SA-PSO. It provides the complete information of the group and allows the members to exploit the available information.

4. Experiments

The proposed SA-PSO and the existing S-PSO and A-PSO were implemented using MATLAB. The parameter settings are summarised in Table 1. Each experiment was subjected to 500 runs. The initial velocity was set to random value subject

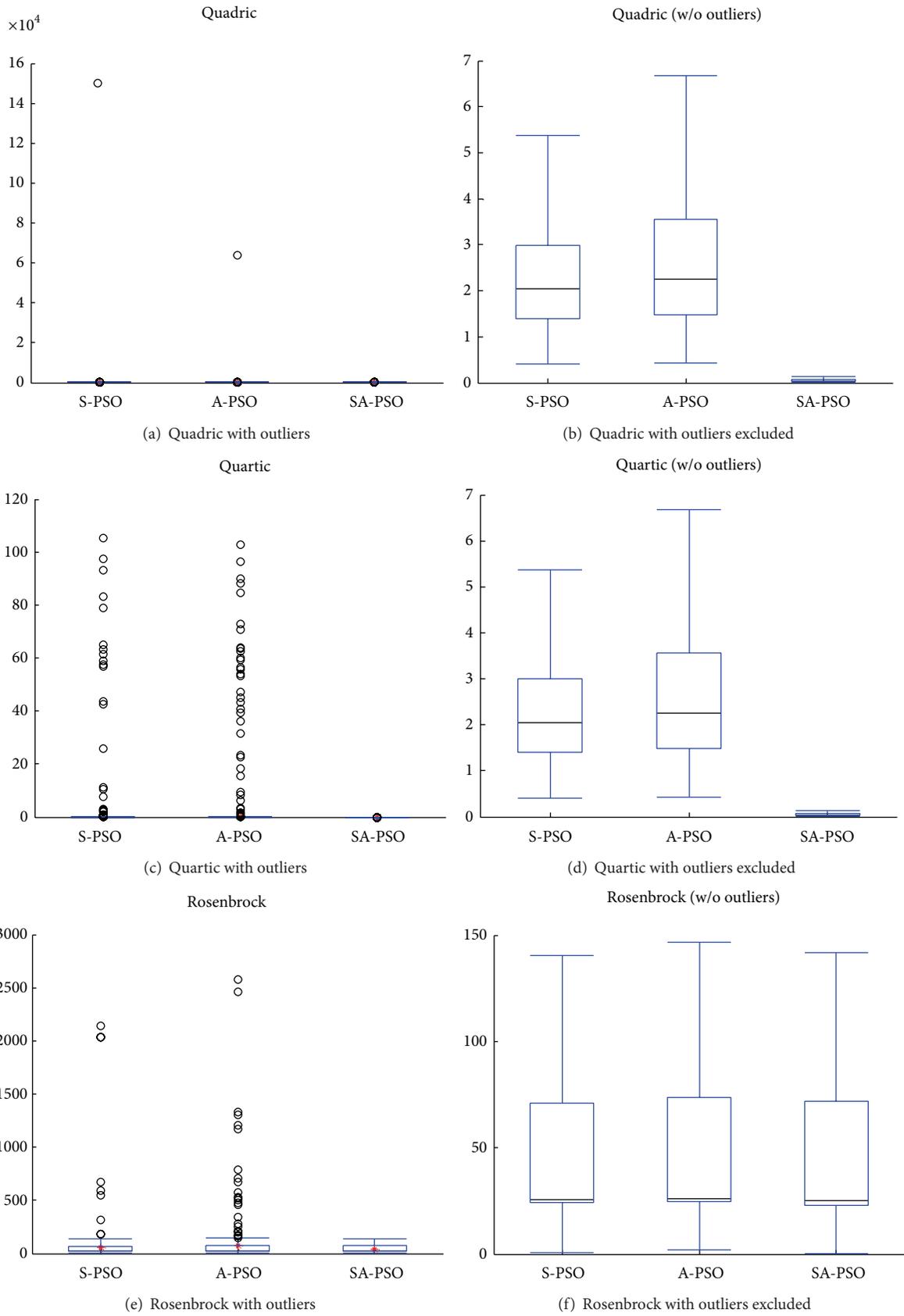


FIGURE 4: Continued.

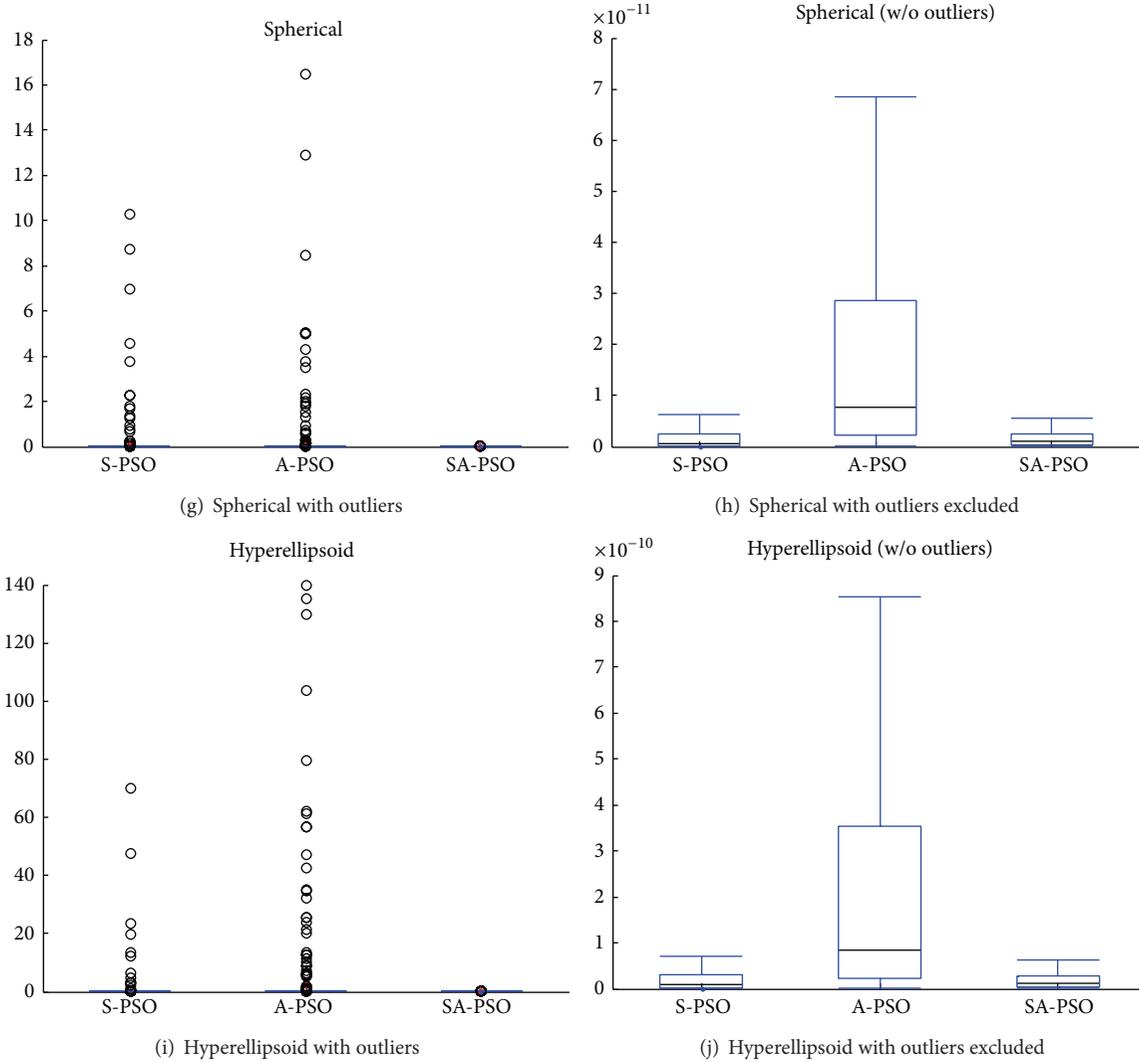


FIGURE 4: Results of experiments on unimodal functions.

to the velocity clamping range, $\pm V_{\max}$. The position of the particles was randomly initialised within the search space. A linear decreasing inertia weight ranging from 0.9 to 0.4 was employed to encourage fine tuning towards the end of the search. The cognitive and social learning factors were set to 2 which is a typical value for c_1 and c_2 . The search was terminated either due to the number of iterations reaching 2000 or the ideal solution being found. The maximum number of iteration is set to 2000 to limit the computational time taken. The final $gBest$ values were recorded. The setting for the additional parameters in SA-PSO is given in Table 2. Exclusively for SA-PSO, the members of the groups were randomly initialised with their distance to group centres, Δ . The group centres were randomly initialised within the boundary of the search space.

A group of benchmark test problems had been identified for assessing the performance of the proposed SA-PSO and the original S-PSO and A-PSO algorithms. The benchmark test problems consist of five unimodal functions, four multimodal functions, and one real world optimisation

TABLE 2: Parameters setting for the additional parameters in SA-PSO.

Parameter	Value
Number of groups, C	5
Group size (particles per group)	10
Initial distance to group centre, Δ	50% of the length of the search space

problem, namely, frequency-modulated (FM) sound wave synthesis which is taken from CEC2011 competition on testing evolutionary algorithms on real world optimisation problems [28]. These functions are given in Table 3. All functions used are minimisation functions with ideal fitness value of $f(x) = 0$. The dimension of the unimodal and multimodal problems, n , was set to 30. The search spaces for these problems are therefore high dimensional [29, 30].

TABLE 3: Test functions.

Function type	Function name	Equation
Unimodal	Quadric	$f_1(x) = \sum_{i=1}^n \left(\sum_{j=1}^i x_j \right)^2$ where $-100 \leq x_j \leq 100$
	Quartic	$f_2(x) = \sum_{i=1}^n i x_i^4$ where $-1.28 \leq x_i \leq 1.28$
	Rosenbrock	$f_3(x) = \sum_{i=1}^{n-1} \left[100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2 \right]$ where $-2.048 \leq x_i \leq 2.048$
	Spherical/De Jong's	$f_4(x) = \sum_{i=1}^n x_i^2$ where $-5.12 \leq x_i \leq 5.12$
	Hyperellipsoid	$f_5(x) = \sum_{i=1}^n i x_i^2$ where $-5.12 \leq x_i \leq 5.12$
Multimodal	Ackley	$f_6(x) = 20 + e - 20 \exp \left[-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2} \right] - \exp \left[\frac{1}{n} \sum_{i=1}^n \cos(2\pi x_i) \right]$ where $-32.768 \leq x_i \leq 32.768$
	Griewank	$f_7(x) = 1 + \frac{1}{4000} \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos \left(\frac{x_i}{\sqrt{i}} \right)$ where $-600 \leq x_i \leq 600$
	Rastrigin	$f_8(x) = 10n + \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i)]$ where $-5.12 \leq x_i \leq 5.12$
	Salomon	$f_9(x) = 1 - \cos \left(2\pi \sqrt{\sum_{i=1}^n x_i^2} \right) + 0.1 \sqrt{\sum_{i=1}^n x_i^2}$ where $-600 \leq x_i \leq 600$
Real world problem	FM sound wave	$y(t) = x_1 \sin(x_2 t\theta + x_3 \sin(x_4 t\theta + x_5 \sin(x_6 t\theta)))$ $y_0(t) = (1) \sin((5) t\theta) + (-1.5) \sin((4.8) t\theta) + (2.0) \sin((4.9) t\theta))$ $f_{10}(x) = \sum_{t=0}^{100} (y(t) - y_0(t))^2$ where $\theta = \frac{2\pi}{100}$ and $-6.4 \leq x_i \leq 6.35$

Note that the FM sound wave problem is a six-dimensional problem.

The solutions found by the algorithms tested are presented here using boxplot. A boxplot shows the quality and also the consistency of an algorithm's performance. The size of the box shows the magnitude of the variance of the results; thus a smaller box suggests a consistent performance of the algorithm. Because the benchmark functions used in this study are minimisation problems, a lower boxplot is desirable as it indicates better quality of the solutions found.

The algorithms are compared using a nonparametric test due to the nature of the solutions found, where they are not normally distributed. The test chosen is the Friedman test with significance level $\alpha = 0.05$. This test is suitable for comparison of more than two algorithms [31]. The algorithms are first ranked based on their average performance for each benchmark function. The average rank is then used to calculate the Friedman statistic value. According to the test, if the statistic value is lesser than the critical value, the algorithms tested are identical to each other; otherwise,

significant differences exist. If a significant difference is found, the algorithms are then compared using a post hoc procedure. The chosen post hoc procedure here is the Holm procedure. It is able to pinpoint the algorithms that are not identical to each other, a result that cannot be detected by the Friedman test.

5. Results and Discussion

5.1. SA-PSO versus S-PSO and A-PSO. The boxplots in Figure 4 show the quality of the results for unimodal test functions using the three algorithms. The results obtained by S-PSO and A-PSO algorithms contain multiple outliers. These out-of-norm observations are caused by the stochastic behaviour of the algorithms. The proposed SA-PSO exhibits no outliers for the unimodal test functions. The particles in SA-PSO are led by two particles with good experience, $gBest$ and $cBest_c$, instead of $gBest$ only like S-PSO and A-PSO. Learning from $cBest_c$ of each group reduces the effect of the stochastic behaviour in SA-PSO.

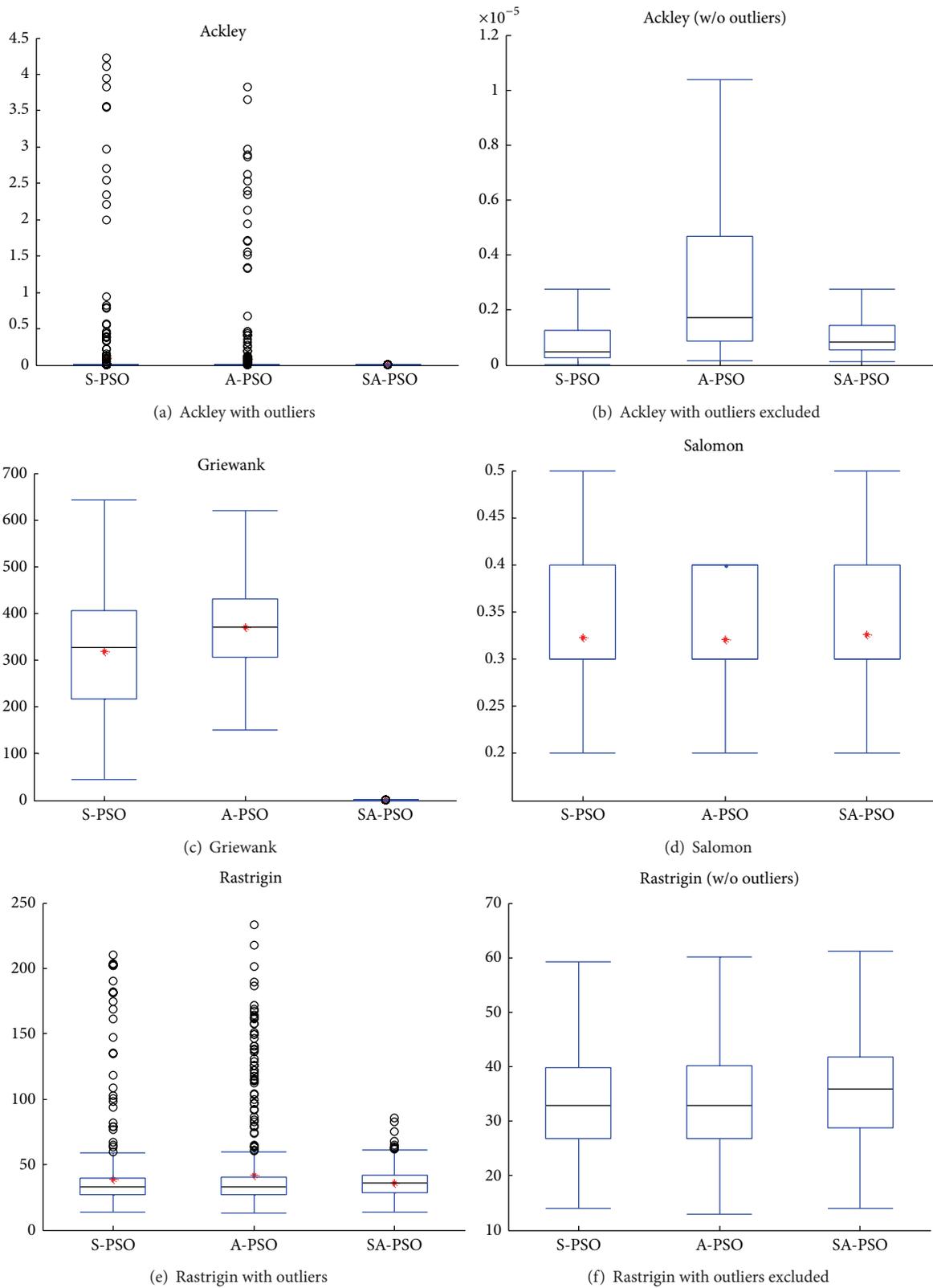


FIGURE 5: Results of experiments on multimodal functions.

TABLE 4: Friedman test on the results of the experiments.

		S-PSO	A-PSO	SA-PSO
Quadric	Mean	305.4320	131.4246	0.0537
	Friedman rank	3	2	1
Quartic	Mean	1.9524	3.0793	0.0000
	Friedman rank	2	3	1
Rosenbrock	Mean	58.7646	71.5899	37.9161
	Friedman rank	2	3	1
Spherical	Mean	0.0963	0.1628	0.0000
	Friedman rank	2	3	1
Hyperellipsoid	Mean	0.4151	2.5037	0.0000
	Friedman rank	2	3	1
Ackley	Mean	0.0941	0.0898	0.0000
	Friedman rank	3	2	1
Griewank	Mean	317.8628	371.7447	0.0071
	Friedman rank	2	3	1
Rastrigin	Mean	38.6035	42.2694	36.1207
	Friedman rank	2	3	1
Salomon	Mean	0.3227	0.3211	0.3263
	Friedman rank	2	1	3
FM sound wave	Mean	5.7751	5.4484	5.7402
	Friedman rank	3	1	2
Average Friedman rank		2.3	2.4	1.3

TABLE 5: Holm procedure on the results of the experiments.

Dataset	z	P	Holm
A-PSO versus SA-PSO	2.4597	0.0139	0.0167
S-PSO versus SA-PSO	2.2361	0.0253	0.0250
S-PSO versus A-PSO	0.2236	0.8231	0.0500

TABLE 6: Experimental setup for size of groups.

Number of particles	Size of groups
20	4
25	5
30	6
35	7
40	8
45	9
50	10

The presence of the outliers makes it difficult to observe the variance of the results through the box plot. Therefore, the outliers are trimmed in the boxplots of Figures 4(b), 4(d), 4(f), 4(h), and 4(j). The benchmark functions tested here are minimisation functions; hence, a lower boxplot indicates better quality of the algorithm. It can be observed from the figure that SA-PSO continuously gives good performance in all the unimodal functions tested. The sizes of the boxplots show that the SA-PSO algorithm provides a more consistent performance with smaller variance.

TABLE 7: Experimental setup for number of groups.

Number of particles	Number of groups
20	4
25	5
30	6
35	7
40	8
45	9
50	10

The results of the test on multimodal problems are shown in the boxplots in Figure 5. S-PSO and A-PSO have outliers for Ackley and Rastrigin while SA-PSO only has outliers in the results of Rastrigin. The Rastrigin function has a nonprotruding minima, which complicates the convergence [32]. However, SA-PSO has fewer outliers compared to S-PSO and A-PSO. This observation once again proves the efficiency of learning from two good particles, $gBest_c$ and $cBest_c$.

Similar to the boxplots for unimodal test functions, the boxplots, after trimming of the outliers, show that the variance of the solutions found by SA-PSO is small. The variance proves the consistency of SA-PSO's performance. SA-PSO found much better results for the Griewank function compared to the other two algorithms.

The three algorithms tested have similar performance for the FM sound wave parameter estimation problem as shown in Figure 6. However, from the distribution of the solution in the boxplot, it could be seen that SA-PSO and A-PSO have

TABLE 8: Average results on the experiments involving the size of group.

Size of groups	Quadric	Quartic	Rosenbrock	Spherical	Hyperellipsoid	Ackley	Griewank	Rastrigin	Salomon
4	1.3459	5.183E - 11	45.3448	4.285E - 08	4.805E - 07	0.0184	0.0081	51.6422	0.3847
5	0.7224	1.289E - 12	41.0048	0.445	2.275E - 08	0.0089	0.0074	46.2869	0.3663
6	0.3932	1.925E - 13	41.5781	4.594E - 10	4.621E - 09	1.3387E - 05	0.007	43.886	0.3505
7	0.223	1.09E - 14	38.6543	7.226E - 11	7.833E - 10	6.9718E - 06	0.0072	41.6152	0.3396
8	0.1357	1.485E - 15	38.4704	1.637E - 11	2.455E - 10	3.5115E - 06	0.0068	39.9117	0.3315
9	0.0896	2.63E - 16	39.4469	6.376E - 12	8.392E - 11	1.9843E - 06	0.0073	38.2884	0.3297
10	0.0537	4.735E - 17	37.9161	4.086E - 12	2.704E - 11	1.1777E - 06	0.0071	36.1207	0.3263

TABLE 9: Average results on the experiments involving the number of groups.

Number of groups	Quadric	Quartic	Rosenbrock	Spherical	Hyperellipsoid	Ackley	Griewank	Rastrigin	Salomon
4	1.1854	0.262	45.3952	2.864E - 06	4.349E - 07	0.0167	0.0073	51.9996	0.3941
5	0.7224	1.289E - 12	41.0048	0.445	2.275E - 08	0.0089	0.0074	46.2869	0.3663
6	0.462	1.381E - 13	39.7459	9.038E - 10	9.866E - 09	0.0027	0.0073	43.3588	0.3453
7	0.3886	9.477E - 14	39.66	1.205E - 10	1.79E - 09	9.4418E - 06	0.0071	40.1133	0.3297
8	0.3013	2.671E - 14	40.0051	5.109E - 11	7.606E - 10	6.1644E - 06	0.0075	38.2584	0.3193
9	0.25	5.154E - 15	40.682	0.1889	3.668E - 10	4.2811E - 06	0.0067	36.3674	0.3141
10	0.2111	3.448E - 15	37.3187	1.824E - 11	2.134E - 10	3.1415E - 06	0.0068	35.7875	0.3093

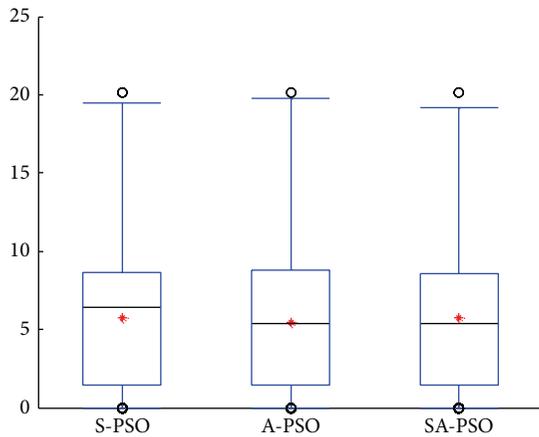


FIGURE 6: Results of experiments on parameter estimation for FM sound wave.

slightly better performance than S-PSO as more solutions found are at the lower part of the box.

In Table 4, the Friedman test is conducted to analyse whether significant differences exist between the algorithms. The performances of the algorithms for all test functions are ranked based on their mean value. The means used here are calculated inclusive of the outliers because the outliers are genuine outliers that are neither measurement nor clerical errors and are therefore valid solutions. The means are shown in the boxplots (before trimming of outliers) using the * symbol. According to the Friedman test, SA-PSO ranked the best among the three algorithms. The Friedman statistic value shows that significant differences exist between the algorithms. Therefore, the Holm procedure is conducted, and the three algorithms are compared against each other. The results in Table 5 show that there is significant difference

between SA-PSO and the A-PSO algorithm. The Holm procedure also shows that the performance of SA-PSO is on a par with S-PSO.

5.2. *Effect of SA-PSO Parameters.* The number of particles can influence the size and the number of groups. To study the effect of these parameters, the number of particles is varied from 20 to 50. Only test functions one to nine are used here as they have similar dimension. There are 7 experiments conducted each for size of the groups and number of groups as listed in Tables 6 and 7. In the experiments for the size of the group, the number of groups is fixed at 5 and the size of the groups is increased from 4 to 10 members. The effect of the number of groups is studied using groups of 5 members; the number of groups is increased from 4, 5, 6, 7, 8, 9, and 10.

The average results for the effect of size of groups and number of groups are presented in Tables 8 and 9. Generally the results show that, similar to the original PSO algorithm, the number of particles affects the performance of SA-PSO. A higher number of particles, that is, bigger groups or higher number of groups, contributes to a better performance. However, the effect is also influenced by the test function. This can be observed in Figure 7, for quadric and Ackley functions, the effect is more obvious compared to other functions.

Friedman test is performed on the experimental results in Tables 8 and 9. The test is conducted to study the effect of number of group and group's size on SA-PSO's performance. The average rank is presented in Table 10.

The result of Friedman test shows that significant difference exists in the SA-PSO performance for different number of groups. Hence, Holm procedure is conducted and its statistical values are tabulated in Table 11. The result of the Holm procedure shows that significant differences

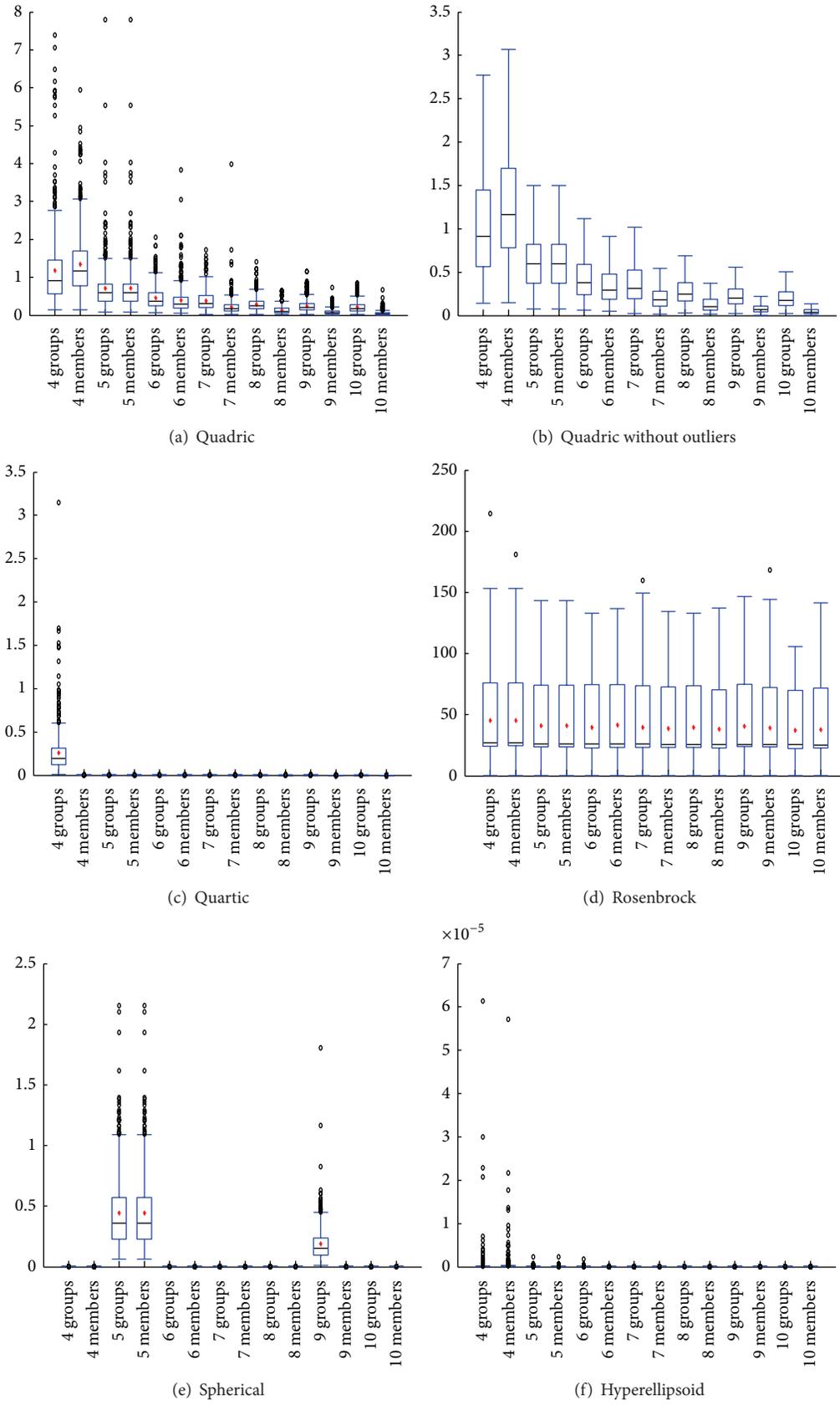


FIGURE 7: Continued.

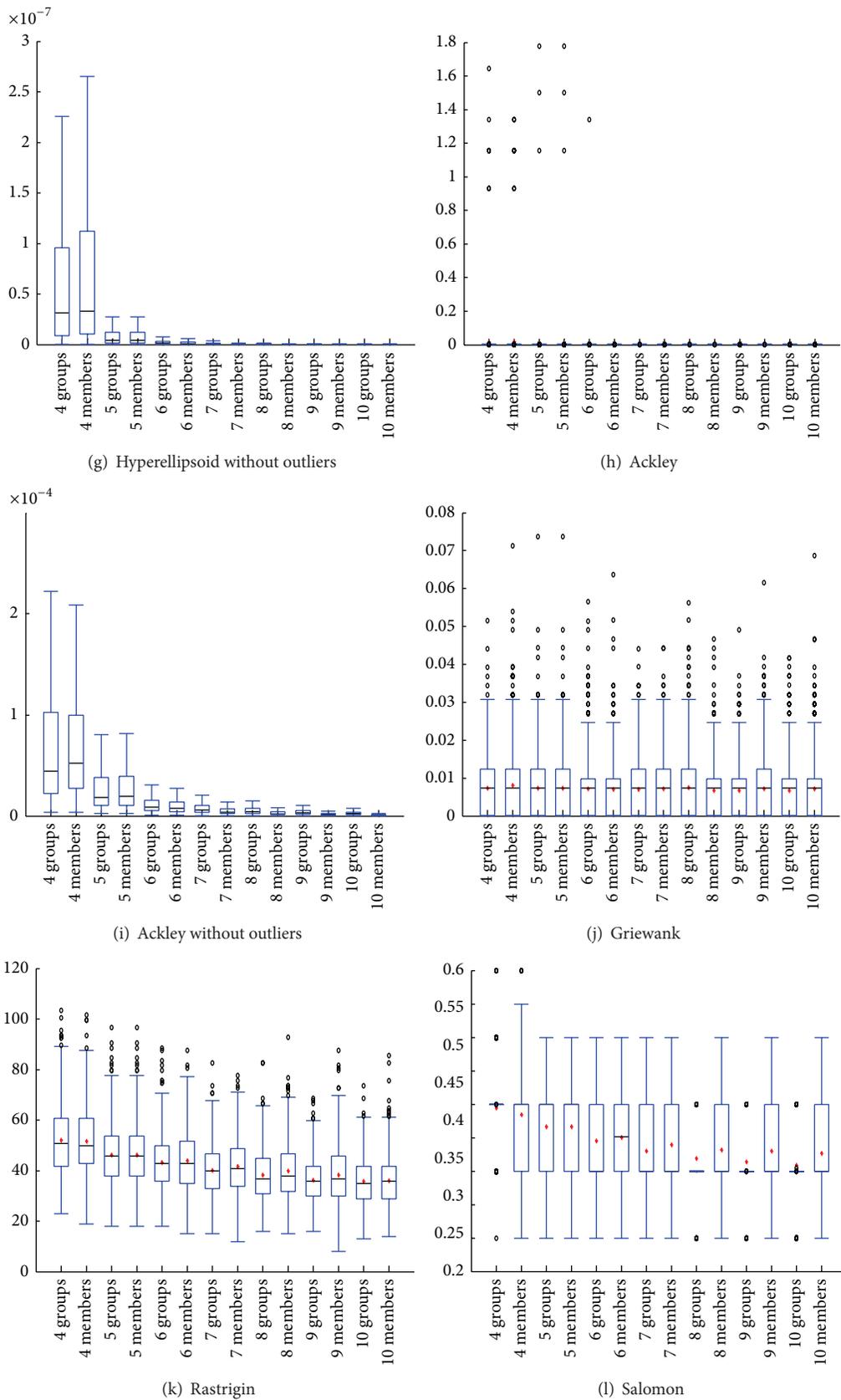


FIGURE 7: Effect of number of groups and group size.

TABLE 10: Friedman test on the effect of number of groups and group size.

Number of groups	4	5	6	7	8	9	20
Average Friedman rank	6.50	6.11	4.61	3.56	3.44	2.67	1.11
Size of groups	4	5	6	7	8	9	20
Average Friedman rank	6.89	6.00	4.78	3.89	2.67	2.56	1.22

TABLE 11: Holm procedure on the effect of number of groups.

Dataset	<i>P</i>	<i>z</i>	Holm
4 groups versus 10 groups	0.0000	5.2918	0.0024
5 groups versus 10 groups	0.0000	4.9099	0.0025
4 groups versus 9 groups	0.0002	3.7643	0.0026
6 groups versus 10 groups	0.0006	3.4369	0.0028
5 groups versus 9 groups	0.0007	3.3824	0.0029
4 groups versus 8 groups	0.0027	3.0005	0.0031
4 groups versus 7 groups	0.0038	2.8914	0.0033
5 groups versus 8 groups	0.0088	2.6186	0.0036
5 groups versus 7 groups	0.0121	2.5095	0.0038
7 groups versus 10 groups	0.0164	2.4004	0.0042
8 groups versus 10 groups	0.0219	2.2913	0.0045
6 groups versus 9 groups	0.0562	1.9094	0.0050
4 groups versus 6 groups	0.0636	1.8549	0.0056
9 groups versus 10 groups	0.1266	1.5275	0.0063
5 groups versus 6 groups	0.1408	1.4730	0.0071
6 groups versus 8 groups	0.2519	1.1456	0.0083
6 groups versus 7 groups	0.3000	1.0365	0.0100
7 groups versus 9 groups	0.3827	0.8729	0.0125
8 groups versus 9 groups	0.4450	0.7638	0.0167
4 groups versus 5 groups	0.7025	0.3819	0.0250
7 groups versus 8 groups	0.9131	0.1091	0.0500

exist between SA-PSO implementations if the populations in each implementation consist of unequal number of groups and the difference in the number of groups is greater than three.

The Friedman test performed on the effect of the group size shows that the SA-PSO implemented with groups of different sizes are significantly different. This observation is further studied using Holm procedure as in Table 12. The outcome of Holm procedure reveals that significant difference exists between two implementations of SA-PSO algorithm if the difference in the group size is greater than three particles.

Δ is a new parameter introduced in SA-PSO. It represents the maximum distance of a particle to its group head during the initialisation stage of the algorithm. The value of Δ determines the distribution of the particles within the search space. A small Δ will result in close groups, while a large Δ will result in groups with a bigger radius. The effect of Δ is tested here, and the test parameters are listed in Table 13. For each of the test functions, the Δ value is set to 1%, 5%,

TABLE 12: Holm procedure on the effect of group size.

Dataset	<i>P</i>	<i>z</i>	Holm
4 members versus 10 members	0.0000	5.5646	0.0024
5 members versus 10 members	0.0000	4.6917	0.0025
4 members versus 9 members	0.0000	4.2552	0.0026
4 members versus 8 members	0.0000	4.1461	0.0028
6 members versus 10 members	0.0005	3.4915	0.0029
5 members versus 9 members	0.0007	3.3824	0.0031
5 members versus 8 members	0.0011	3.2733	0.0033
4 members versus 7 members	0.0032	2.9459	0.0036
7 members versus 10 members	0.0088	2.6186	0.0038
6 members versus 9 members	0.0291	2.1822	0.0042
4 members versus 6 members	0.0382	2.0731	0.0045
5 members versus 7 members	0.0382	2.0731	0.0050
6 members versus 8 members	0.0382	2.0731	0.0056
8 members versus 10 members	0.1561	1.4184	0.0063
9 members versus 10 members	0.1904	1.3093	0.0071
7 members versus 9 members	0.1904	1.3093	0.0083
5 members versus 6 members	0.2301	1.2002	0.0100
7 members versus 8 members	0.2301	1.2002	0.0125
4 members versus 5 members	0.3827	0.8729	0.0167
6 members versus 7 members	0.3827	0.8729	0.0250
8 members versus 9 members	0.9131	0.1091	0.0500

TABLE 13: Test parameters for experiment on the effect of Δ .

Parameter	Value
Number of runs for each experiment	500
Number of iterations	2000
Velocity clamping, V_{max}	4
Range of inertia weight, ω	0.9–0.4
Learning factors	
c_1	2
c_2	2
Number of groups	5
Group's size	6

10%, 50%, and 100% of the length of the search space. The average performance for different values of Δ is listed in Table 14.

The Friedman statistic shows that using different Δ values makes no significant difference to SA-PSO, thus showing that the performance of SA-PSO is not greatly affected by the choice of Δ . This result is confirmed by boxplots in Figure 8 where the sizes of the box in most of the test functions are similar to each other.

6. Conclusion

A synchronous-asynchronous PSO algorithm (SA-PSO) is proposed in this paper. The particles in this algorithm are

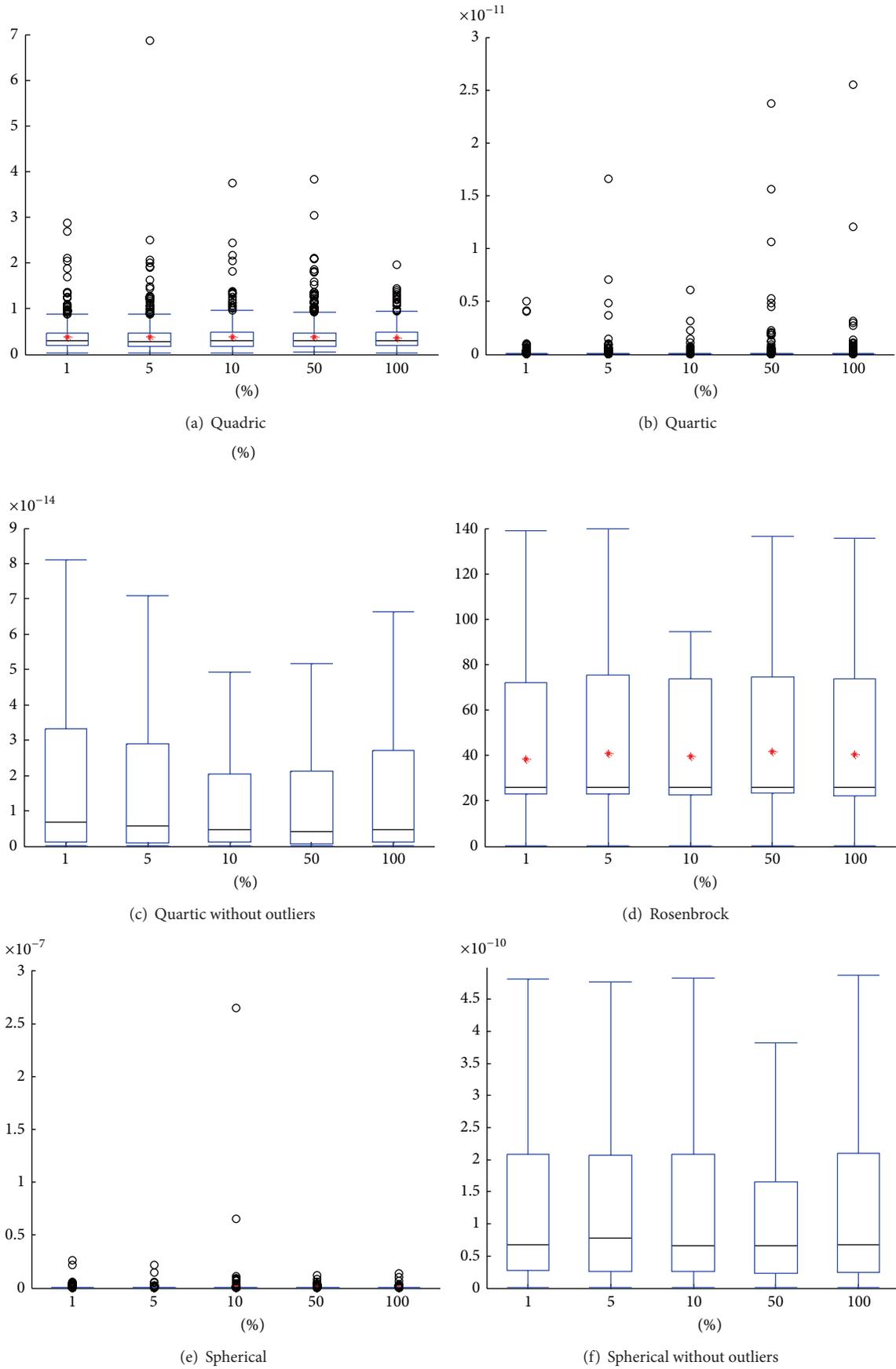


FIGURE 8: Continued.

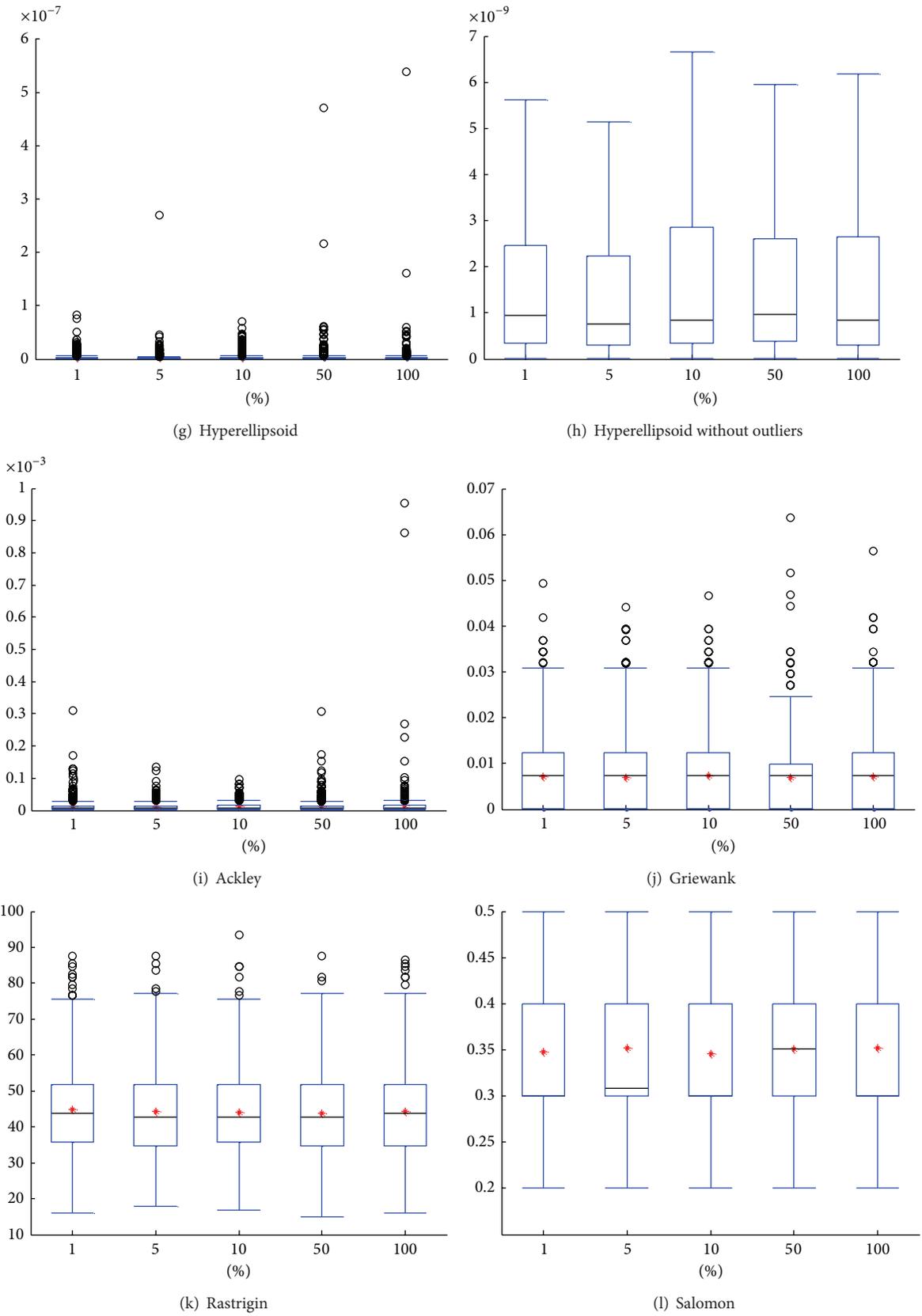


FIGURE 8: Effect of Δ .

TABLE 14: Friedman test on the effect of Δ .

		1%	5%	10%	50%	100%
f_1	Mean	0.3812	0.3944	0.3839	0.3932	0.3738
	Friedman rank	2	5	3	4	1
f_2	Mean	$0.0732 * e - 12$	$0.1061 * e - 12$	$0.0612 * e - 12$	$0.1925 * e - 12$	$0.1401 * e - 12$
	Friedman rank	2	3	1	5	4
f_3	Mean	38.4925	40.8531	39.8487	41.5781	40.5706
	Friedman rank	1	4	2	5	3
f_4	Mean	$0.3428 * e - 09$	$0.2885 * e - 09$	$0.9245 * e - 09$	$0.2542 * e - 09$	$0.2786 * e - 09$
	Friedman rank	4	3	5	1	2
f_5	Mean	$0.2956 * e - 08$	$0.2979 * e - 08$	$0.3365 * e - 08$	$0.4385 * e - 08$	$0.4271 * e - 08$
	Friedman rank	1	2	3	5	4
f_6	Mean	$0.1400 * e - 04$	$0.1180 * e - 04$	$0.1210 * e - 04$	$0.1339 * e - 04$	$0.1753 * e - 04$
	Friedman rank	4	1	2	3	5
f_7	Mean	0.0072	0.0069	0.0073	0.0070	0.0073
	Friedman rank	3	1	4.5	2	4.5
f_8	Mean	44.8242	44.4277	44.0441	43.8860	44.2639
	Friedman rank	5	4	2	1	3
f_9	Mean	0.3479	0.3515	0.3457	0.3505	0.3521
	Friedman rank	2	4	1	3	5
Average Friedman rank		2.67	3	2.61	3.22	3.5

updated in groups; the groups are updated asynchronously—one by one—while particles within a group are updated synchronously. A group's search is led by the group's best performer, $cBest_c$, and the best member of the swarm, $gBest$. The algorithm benefits from good exploitation and fine tuning provided by synchronous update while also taking advantage of the exploration by the asynchronous update. Learning from $cBest_c$ also contributes to the good performance of the SA-PSO algorithm. Overall, the performance of the algorithm proposed is better and more consistent than the original S-PSO and A-PSO.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research is funded by the Department of Higher Education of Malaysia under the Fundamental Research Grant Scheme (FRGS/1/2012/TK06/MMU/03/7), the UM-UMRG Scheme (031-2013), Ministry of Higher Education of Malaysia/High Impact Research Grant (UM-D000016-16001), and UM-Postgraduate Research Grant (PG097-2013A). The authors also would like to acknowledge the anonymous reviewers for their valuable comments and insights.

References

- [1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks (ICNN '95)*, vol. 4, pp. 1942–1948, Perth, Western Australia, November-December 1995.
- [2] R. C. Eberhart and X. Hu, "Human tremor analysis using particle swarm optimization," *Proceedings of the Congress on Evolutionary Computation (CEC '99)*, pp. 1927–1930, 1999, Cat. no. 99TH8406.
- [3] Z. Ibrahim, N. K. Khalid, J. A. A. Mukred et al., "A DNA sequence design for DNA computation based on binary vector evaluated particle swarm optimization," *International Journal of Unconventional Computing*, vol. 8, no. 2, pp. 119–137, 2012.
- [4] J. Hazra and A. K. Sinha, "Congestion management using multiobjective particle swarm optimization," *IEEE Transactions on Power Systems*, vol. 22, no. 4, pp. 1726–1734, 2007.
- [5] M. F. Tasgetiren, Y. Liang, M. Sevkli, and G. Gencyilmaz, "Particle swarm optimization algorithm for single machine total weighted tardiness problem," in *Proceedings of the Congress on Evolutionary Computation (CEC '04)*, pp. 1412–1419, June 2004.
- [6] A. Adam, A. F. Zainal Abidin, Z. Ibrahim, A. R. Husain, Z. Md Yusof, and I. Ibrahim, "A particle swarm optimization approach to Robotic Drill route optimization," in *Proceedings of the 4th International Conference on Mathematical Modelling and Computer Simulation (AMS '10)*, pp. 60–64, May 2010.
- [7] M. N. Ayob, Z. M. Yusof, A. Adam et al., "A particle swarm optimization approach for routing in VLSI," in *Proceedings of the 2nd International Conference on Computational Intelligence, Communication Systems and Networks (CICSyN '10)*, pp. 49–53, Liverpool, UK, July 2010.
- [8] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation and IEEE World Congress on Computational Intelligence*, (Cat. No.98TH8360), pp. 69–73, Anchorage, Alaska, USA, May 1998.
- [9] M. Clerc and J. Kennedy, "The particle swarm-explosion, stability, and convergence in a multidimensional complex space,"

- IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 58–73, 2002.
- [10] M. Reyes-Sierra and C. A. Coello Coello, “Multi-objective particle swarm optimizers: a survey of the state-of-the-art,” *International Journal of Computational Intelligence Research*, vol. 2, no. 3, pp. 287–308, 2006.
- [11] K. S. Lim, Z. Ibrahim, S. Buyamin et al., “Improving vector evaluated particle swarm optimisation by incorporating non-dominated solutions,” *The Scientific World Journal*, vol. 2013, Article ID 510763, 19 pages, 2013.
- [12] J. Kennedy and R. C. Eberhart, “Discrete binary version of the particle swarm algorithm,” in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, Computational Cybernetics and Simulation*, vol. 5, pp. 4104–4108, Orlando, Fla, USA, October 1997.
- [13] M. S. Mohamad, S. Omatu, S. Deris, M. Yoshioka, A. Abdullah, and Z. Ibrahim, “An enhancement of binary particle swarm optimization for gene selection in classifying cancer classes,” *Algorithms for Molecular Biology*, vol. 8, article 15, 2013.
- [14] J. Rada-Vilela, M. Zhang, and W. Seah, “A performance study on the effects of noise and evaporation in particle swarm optimization,” in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '12)*, pp. 1–8, June 2012.
- [15] A. Carlisle and G. Dozier, “An Off-The-Shelf PSO,” in *Proceedings of the Workshop on Particle Swarm Optimization*, 2001.
- [16] L. Mussi, S. Cagnoni, and F. Daolio, “Empirical assessment of the effects of update synchronization in particle swarm optimization,” in *Proceeding of the AIIA Workshop on Complexity, Evolution and Emergent Intelligence*, pp. 1–10, 2009.
- [17] J. Rada-Vilela, M. Zhang, and W. Seah, “A performance study on synchronicity and neighborhood size in particle swarm optimization,” *Soft Computing*, vol. 17, no. 6, pp. 1019–1030, 2013.
- [18] B. Jiang, N. Wang, and X. He, “Asynchronous particle swarm optimizer with relearning strategy,” in *Proceedings of the 37th Annual Conference of the IEEE Industrial Electronics Society (IECON '11)*, pp. 2341–2346, November 2011.
- [19] S. Xue, J. Zhang, and J. Zeng, “Parallel asynchronous control strategy for target search with swarm robots,” *International Journal of Bio-Inspired Computation*, vol. 1, no. 3, pp. 151–163, 2009.
- [20] R. Juan, M. Zhang, and W. Seah, “A performance study on synchronous and asynchronous updates in Particle Swarm Optimization,” in *Proceedings of the 13th Annual Genetic and Evolutionary Computation Conference (GECCO '11)*, pp. 21–28, July 2011.
- [21] Y. Shi and R. Eberhart, “Parameter selection in particle swarm optimization,” in *Evolutionary Programming VII*, vol. 1447 of *Lecture Notes in Computer Science*, pp. 591–600, Springer, New York, NY, USA, 1998.
- [22] Y. Kennedy, J. Eberhart, and R. Shi, *Swarm Intelligence*, Morgan Kaufmann, Boston, Mass, USA, 2001.
- [23] Y. Shi and R. C. Eberhart, “Empirical study of particle swarm optimization,” in *Proceedings of the Congress on Evolutionary Computation (CEC '99)*, pp. 1945–1950, 1999, Cat. no. 99TH8406.
- [24] J. Kennedy, “Why does it need velocity?” in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '95)*, pp. 38–44, 2005.
- [25] R. C. Eberhart and Y. Shi, “Comparing inertia weights and constriction factors in particle swarm optimization,” in *Proceedings of the Congress on Evolutionary Computation (CEC '00)*, vol. 1 of (Cat. No.00TH8512), pp. 84–88, July 2000.
- [26] J. Rada-Vilela, M. Zhang, and W. Seah, “Random asynchronous PSO,” in *Proceedings of the 5th International Conference on Automation, Robotics and Applications (ICARA '11)*, pp. 220–225, Wellington, New Zealand, December 2011.
- [27] L. Dioşan and M. Oltean, “Evolving the structure of the particle swarm optimization algorithms,” in *Proceedings of the 6th European Conference on Evolutionary Computation in Combinatorial Optimization (EvoCOP '06)*, vol. 3906 of *Lecture Notes in Computer Science*, pp. 25–36, Springer, 2006.
- [28] S. Das and P. N. Suganthan, *Problem Definitions and Evaluation Criteria for CEC 2011 Competition on Testing Evolutionary Algorithms on Real World Optimization Problems*, 2011.
- [29] T. Hatanaka, T. Korenaga, and N. Kondo, *Search Performance Improvement for PSO in High Dimensional Space*, 2007.
- [30] T. Hendtlass, “Particle swarm optimisation and high dimensional problem spaces,” in *Proceedings of the Eleventh conference on Congress on Evolutionary Computation (CEC '09)*, pp. 1988–1994, IEEE Press, Piscataway, NJ, USA, May 2009.
- [31] J. Derrac, S. García, D. Molina, and F. Herrera, “A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms,” *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.
- [32] J. Dieterich and B. Hartke, “Empirical review of standard benchmark functions using evolutionary global optimization,” In press, <http://arxiv.org/abs/1207.4318>.

Research Article

Gait Signal Analysis with Similarity Measure

Sanghyuk Lee¹ and Seungsoo Shin²

¹ Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China

² Department of Information Security, Tongmyong University, Sinseonno, Nam-gu, Busan 608-711, Republic of Korea

Correspondence should be addressed to Seungsoo Shin; shinss@tu.ac.kr

Received 25 April 2014; Accepted 10 June 2014; Published 7 July 2014

Academic Editor: T. O. Ting

Copyright © 2014 S. Lee and S. Shin. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Human gait decision was carried out with the help of similarity measure design. Gait signal was selected through hardware implementation including all in one sensor, control unit, and notebook with connector. Each gait signal was considered as high dimensional data. Therefore, high dimensional data analysis was considered via heuristic technique such as the similarity measure. Each human pattern such as walking, sitting, standing, and stepping up was obtained through experiment. By the results of the analysis, we also identified the overlapped and nonoverlapped data relation, and similarity measure analysis was also illustrated, and comparison with conventional similarity measure was also carried out. Hence, nonoverlapped data similarity analysis provided the clue to solve the similarity of high dimensional data. Considered high dimensional data analysis was designed with consideration of neighborhood information. Proposed similarity measure was applied to identify the behavior patterns of different persons, and different behaviours of the same person. Obtained analysis can be extended to organize health monitoring system for specially elderly persons.

1. Introduction

Analysis on the human gait signal has been studied steadily by numerous researchers [1–3]. The research on the gait signal applies to the field of healthcare development, security system, and another related area. Research methodology is based on how to classify the signal and develop pattern recognition algorithm for the comparable data sets. This algorithm applies the processing of the different signals of the same person and same action signals from multiple persons. Developing methodology for gait discrimination is challenge. However human gait signal has high dimensional characteristics, hence analyzing and designing explicit classifying formula is needed [3].

Generally, human gait signals consist of walking, sitting, standing, stepping up and down, and other usual behavior. Such a usual behavior would be done in house life, then it is quite easy for us to identify when we watch them in real situation. In order to develop a more massive monitoring system and healthcare system to analyze and identify behavior signal of each person, decision and classifying system for

the gait signal is required. More specifically, decision whether he/she is doing in normal activities or not can be applied to the design of health care system. Therefore, obtained research output can be applied to the identification, healthcare, and other related fields. Additionally, more detail checking result even for the healthy people such as athletes can provide useful information whether he/she has suffered from other problems compared to the previous behavior.

To discriminate between different patterns, rational measure obtained from a statistical approach or heuristic approach is needed. By the point of statistical method, signal autocorrelation and cross correlation knowledge are useful because such formula provide us how much the signals are related with each other by the numeric value. Also, it is rather convenient to calculate due to the conventional software such as Matlab toolbox. However, it is not easy for high dimensional data to construct high dimensional correlation/covariance matrices. For heuristic approach, it needs preliminary processing for the signal, such as data redefinition and measure design based on the human thinking. Even the realization of measure based on heuristic

idea is considered, ordinary measure for discrimination has to be considered such as distance. Fortunately, similarity measure design for the vague data has become a more interesting research topic; hence, numerous researchers have been focused on the similarity measure, entropy design problem for fuzzy set, and intuitionistic fuzzy set [4–8].

Similarity measure [9–12] provides useful knowledge to the clustering, and pattern recognition for data sets [13]. However, most of the conventional results were not included in high dimensional data. Human gait signal represents high dimensional data characteristics. So similarity measure design problem for high dimensional data are also needed to deal with the human gait signal. Similarity measure research has rather long history; square error clustering algorithm has been used from the late 1960s [14]. And it was modified to create the cluster program [15]. Naturally, similarity measure topic has been moved to many areas such as statistics [16], machine learning [17, 18], pattern recognition [19], and image processing. Extended research on high-dimensional data can be applied to the security business including fingerprint and iris identification, image processing enhancement, and even big data application recently.

Then, distance between vectors can be organized by norms such as 1-norm, Euclidean-norm, and so forth. Similarity measure is also designed explicitly with the distance norm. Similarity measure design problem for high-dimension needs more considerate approach. Conventionally, the similarity measure has been designed based on the distance measure between two considered data, that is, distance measure was considered information distance between two membership functions. In the similarity measure design with distance measure, measure structure should be related to the same support of the universe of discourse [14, 15]. Additionally, similarity measure consideration on overlapped or nonoverlapped data is needed because many cases of high dimensional data are represented nonoverlapped data structure. With conventional similarity measure, nonoverlapped data analysis is not possible. Hence, similarity measure design for nonoverlapped data should be followed. In order to design similarity measure on nonoverlapped data, neighbor data information was considered. Data has to be affected from the adjacent information, so similarity measure on nonoverlapped data was designed. Inside of literature, artificial data was given to compare with conventional similarity measure; calculation result was also illustrated.

In the following section, preliminary results on the similarity measure on overlapped and nonoverlapped data were introduced. Proposed similarity measure was proved and applied to overlapped and nonoverlapped artificial data. In Section 3, gait signal acquisition system was considered with sensor and data acquisition Gait signal which was also illustrated with different behaviors. High dimensional similarity measure was proposed and proved in Section 4. Similarity measure design for high dimensional data was also discussed by way of norm structure. Similarity calculation results for different behavior and individuals were also shown. Finally, conclusions are followed in Section 5.

2. Similarity Measure Based Distance Property

In order to design similarity measure explicitly, usual measure such as Hamming distance was commonly used as distance measure between sets A and B as follows:

$$d(A, B) = \frac{1}{n} \sum_{i=1}^n |\mu_A(x_i) - \mu_B(x_i)|, \quad (1)$$

where $X = \{x_1, x_2, \dots, x_n\}$, $\mu_A(x_i)$ and $\mu_B(x_i)$ are fuzzy membership function of fuzzy sets A and B at x_i , and $|k|$ was the absolute value of k . The membership function of $A \in F(X)$ is represented by $A = \{(x, \mu_A(x)) \mid x \in X, 0 \leq \mu_A(x) \leq 1\}$, X is total set, and $F(X)$ is the class of all fuzzy sets of X . Similarity measure definition was defined with the help of distance measure [14]. There are numerous similarity measures satisfying the following definition.

Definition 1 (see [14]). A real function $s: F^2 \rightarrow R^+$ is called a similarity measure if s has the following properties:

- (S1) $s(A, B) = s(B, A)$, $A, B \in F(X)$;
- (S2) $s(D, D^C) = 0$, $D \in P(X)$;
- (S3) $s(C, C) = \max_{A, B \in F} s(A, B)$, $C \in F(X)$;
- (S4) $A, B, C \in F(X)$, if $A \subset B \subset C$, then $s(A, B) \geq s(A, C)$ and $s(B, C) \geq s(A, C)$;

where $R^+ = [0, \infty)$, $P(X)$ is the class of ordinary sets of X and D^C is the complement set of D . By this definition, numerous similarity measures could be derived. In the following theorem, similarity measures based on distance measure is illustrated.

Theorem 2. For any set $A, B \in F(X)$, if d satisfies Hamming distance measure, then

$$s(A, B) = d((A \cap B), [0]_X) + d((A \cup B), [1]_X) \quad (2)$$

is the similarity measure between sets A and B .

Proof. Commutativity of (S1) is clear from (9) itself; that is, $s(A, B) = s(B, A)$.

For (S2),

$$\begin{aligned} s(D, D^C) &= d((D \cap D^C), [0]_X) + d((D \cup D^C), [1]_X) \\ &= d([0]_X, [0]_X) + d([1]_X, [1]_X) = 0 \end{aligned} \quad (3)$$

is obtained because of $(D \cap D^C) = [0]_X$ and $(D \cup D^C) = [1]_X$, where $[0]_X$ and $[1]_X$ denote zero and one over the whole universe of discourse of X . Hence, (S2) was satisfied.

(S3) is also easy to prove as follows:

$$\begin{aligned} s(C, C) &= d((C \cap C), [0]_X) + d((C \cup C), [1]_X) \\ &= d(C, [0]_X) + d(C, [1]_X) = 1. \end{aligned} \quad (4)$$

It is natural that $s(C, C)$ satisfied maximal value. Finally,

$$\begin{aligned} d((A \cap B), [0]_X) &= d((A \cap C), [0]_X), \\ d((A \cup C), [1]_X) &< d((A \cup B), [1]_X) \end{aligned} \quad (5)$$

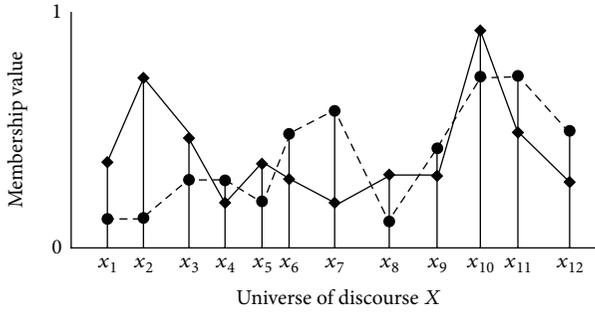


FIGURE 1: Overlapped data distribution.

guarantees $s(A, C) < s(A, B)$, and

$$\begin{aligned} d((B \cap C), [0]_X) &= d((A \cap C), [0]_X), \\ d((A \cup C), [1]_X) &< d((B \cup C), [1]_X) \end{aligned} \tag{6}$$

also guarantees $s(A, C) < s(A, B)$; therefore, triangular equality is obvious by the definition, and hence (S4) is also satisfied. \square

Besides Theorem 2, numerous similarity measures are possible. Another similarity measure is illustrated in Theorem 3, and its proof is also found in the previous result [15, 16].

Theorem 3. For any set $A, B \in F(X)$, if d satisfies Hamming distance measure, then

$$s(A, B) = 1 - d((A \cap B), (A \cup B)), \tag{7}$$

$$s(A, B) = 1 - d(A, A \cap B) - d(B, A \cap B), \tag{8}$$

$$s(A, B) = 2 - d((A \cap B), [1]_X) - d((A \cup B), [0]_X) \tag{9}$$

are the similarity measure between sets A and B .

Proof. Proofs are easy to be derived, and it was found in previous results [15, 16]. \square

Besides similarity measures of (7) to (9), other similarity measures are also illustrated in previous results [15–17]. With similarity measure in Theorems 2 and 3, it is only possible to compute the similarity measure for overlapped data (Figure 1). Following data distributions of diamonds (\blacklozenge) and circles (\bullet) illustrates nonoverlapped data; it is appropriate to design similarity measure for data in Figure 2. Consider the nonoverlapped data distribution of diamonds (\blacklozenge) and circles (\bullet), the similarity measure of (7) to (9) cannot provide the appropriate solution for the nonoverlapped distribution. Two data pairs that constitute different distributions are considered in Figure 2. Twelve data with six diamonds (\blacklozenge) and six circles (\bullet) are illustrated with different combination in Figures 2(a) and 2(b). Similarity degree between circles and diamonds must be different between Figures 2(a) and 2(b) because of different distribution. For example, (7) represents

$$s(A, B) = 1 - d((A \cap B), (A \cup B)). \tag{10}$$

From (7), $d((A \cap B), (A \cup B))$ provides distance between $(A \cap B)$ and $(A \cup B)$. By the following definitions:

$$A \cap B = \min(A, B), \quad A \cup B = \max(A, B). \tag{11}$$

Nonoverlapped data satisfies $A \cap B = \min(A, B) = 0$, and $A \cup B$ is defined as A or B . Hence, $s(A, B) = 1 - (1/N) \sum \max(A, B)$ shows similarity measure, where N is the total number of data sets A and B . From this property,

$$\begin{aligned} s(A, B) &= 1 - \frac{1}{N} \sum \max(A, B) \\ &= 1 - \frac{1}{12} \sum (0.5 + 0.8 + 0.6 + 0.4 + 0.5 + 0.6 + 0.7 \\ &\quad + 0.4 + 0.5 + 1 + 0.8 + 0.6) = 0.38, \end{aligned} \tag{12}$$

and the same result is obtained for Figures 2(a) and 2(b). Hence, similarity measures (2) to (9) are not proper for the nonoverlapped data distribution. Two different data in Figure 2(a) are less discriminate than Figure 2(b). It means that similarity measure of Figure 2(a) has a higher value than Figure 2(b). Similar results are also obtained by the calculation of similarity measures (8) and (9).

Hence, it is required to design similarity measure for nonoverlapping data distribution. Consider the following similarity measure for nonoverlapped data such as Figures 2(a) and 2(b).

Theorem 4. For singletons or discrete data $a, b \in P(X)$, if d satisfied Hamming distance measure, then

$$s(a, b) = 1 - |s_a - s_b| \tag{13}$$

is a similarity measure between singletons a and b . In (13), s_a and s_b satisfy $d((a \cap \mathbf{R}), [1]_X)$ and $d((b \cap \mathbf{R}), [1]_X)$, respectively. Where \mathbf{R} is whole data distribution including a and b .

Proof. (S1) and (S2) are clear. (S3) is also clear from definition as follows:

$$\begin{aligned} s(C, C) &= 1 - |s_C - s_C| \\ &= 1 - |d((C \cap \mathbf{R}), [1]_X) - d((C \cap \mathbf{R}), [1]_X)| = 1. \end{aligned} \tag{14}$$

Finally, (S4) $A, B, C \in F(X)$, if $A < B < C$, then

$$\begin{aligned} s(A, B) &= 1 - |d((A \cap \mathbf{R}), [1]_X) - d((B \cap \mathbf{R}), [1]_X)| \\ &\geq 1 - |d((A \cap \mathbf{R}), [1]_X) - d((C \cap \mathbf{R}), [1]_X)| \tag{15} \\ &= s(A, C), \end{aligned}$$

because $d((B \cap \mathbf{R}), [1]_X) > d((C \cap \mathbf{R}), [1]_X)$ is satisfied. Similarly, $s(B, C) \geq s(A, C)$ is also satisfied. \square

Similarity measure (13) is also designed with the distance measure such as Hamming distance. As noted before, conventional measures were not proper for nonoverlapping

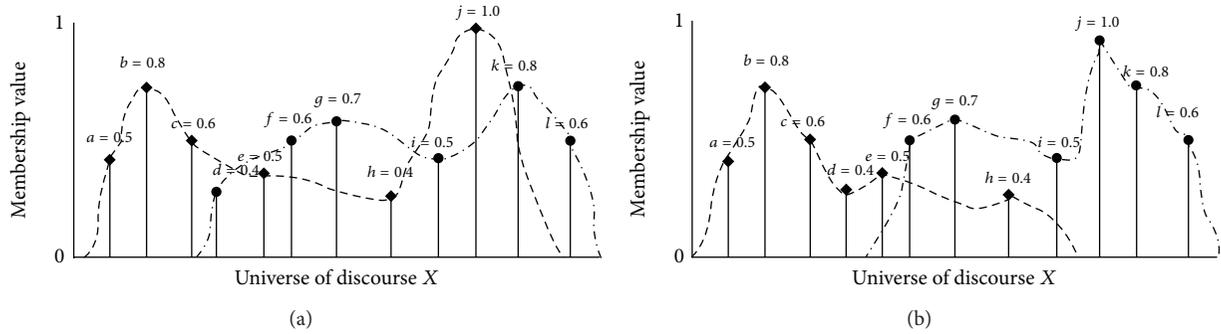


FIGURE 2: (a) Data distribution between circle and diamond. (b) Data distribution between circle and diamond.

continuous data distributions; this property is verified by the similarity measure calculation of Figures 2(a) and 2(b).

Next, calculate the similarity measure between circle and diamond with (13).

For Figure 2(a),

$$\begin{aligned}
 s(\blacklozenge, \bullet) &= 1 - |d((\blacklozenge \cap \mathbf{R}), [1]_X) - d((\bullet \cap \mathbf{R}), [1]_X)| \\
 &= 1 - 1 \times (6 |((1 - 0.5) + (1 - 0.8) + (1 - 0.6) \\
 &\quad + (1 - 0.5) + (1 - 0.4) + (1 - 1)) \\
 &\quad - ((1 - 0.4) + (1 - 0.6) + (1 - 0.7) \\
 &\quad + (1 - 0.5) + (1 - 0.8) + (1 - 0.6))|)^{-1} \\
 &= 1 - \frac{1}{6 |2.3 - 2.4|} = 0.983
 \end{aligned}
 \tag{16}$$

is satisfied.

For the calculation of Figure 2(b),

$$\begin{aligned}
 s(\blacklozenge, \bullet) &= 1 - |d((\blacklozenge \cap \mathbf{R}), [1]_X) - d((\bullet \cap \mathbf{R}), [1]_X)| \\
 &= 1 - 1 \times (6 |((1 - 0.5) + (1 - 0.8) + (1 - 0.6) \\
 &\quad + (1 - 0.4) + (1 - 0.5) + (1 - 0.4)) \\
 &\quad - ((1 - 0.6) + (1 - 0.7) + (1 - 0.5) \\
 &\quad + (1 - 1) + (1 - 0.8) + (1 - 0.6))|)^{-1} \\
 &= 1 - \frac{1}{6 |2.8 - 1.8|} = 0.833.
 \end{aligned}
 \tag{17}$$

Calculation result shows that the proposed similarity measure is possible to evaluate the degree of similarity for nonoverlapped distributions. By comparison with Figure 2, distribution between diamond and circle in Figure 2(a) shows more similar.

3. Human Behavior Signal Analysis and Experiments

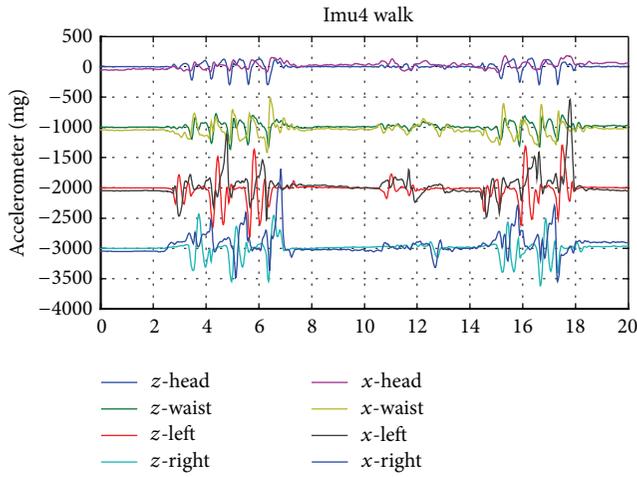
Gait signals are collected with experiment unit; acquisition system (Figure 3) system is composed with all in one sensor in



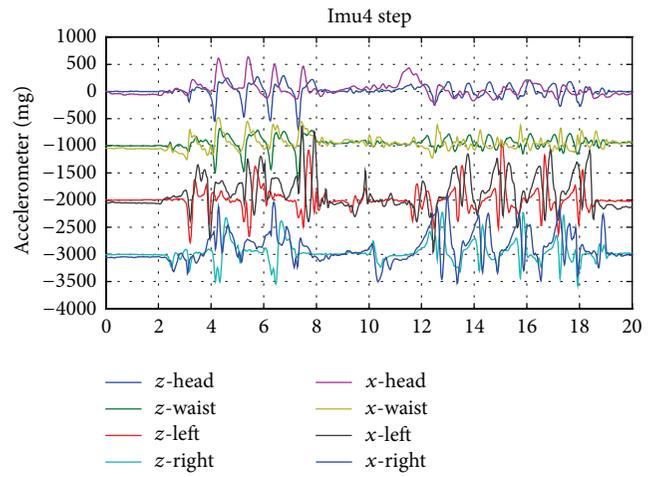
FIGURE 3: Data acquisition experiment.

which accelerator, magnetic, and Gyro sensor, mobile station, and connector are integrated. Signal acquisition experiment was done as shown in the following figures.

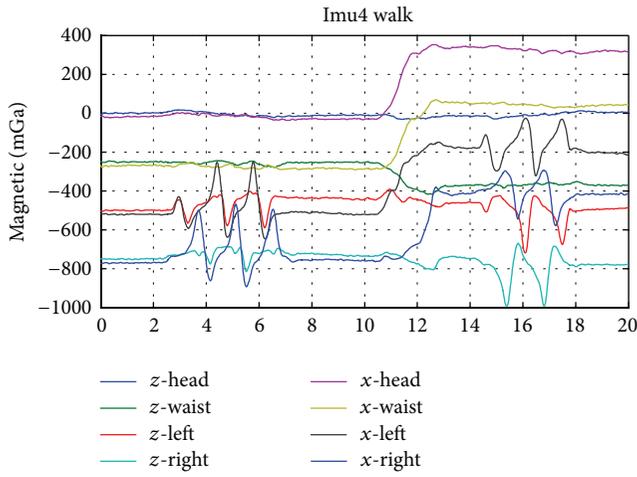
Gait patterns are composed of walking, step up and step down for 20 persons. For each behavior, signals are measured with all in one sensor which integrated with three sensors (accelerator, magnetic, and Gyro sensors); each sensor represents three dimension direct signals. Four sensors are attached to waist, two legs, and head. Example of obtained gait signals is illustrated in the following Figure 4. Among numerous cases, walking and stair up signals are illustrated with acceleration sensor in Figures 4(a) and 4(b), magnetic sensor in Figures 4(c) and 4(d), and Gyro sensor in Figures 4(e) and 4(f), respectively. Full signal was illustrated in Figure 4(f) for stair up with Gyro sensor; we can notice 12 signals for x-y-z direction, and it shows almost the same pattern for similar gait. Hence, x-z direction signals are considered in each figure. Due to the fact that signal patterns are almost the same and numerous quantity, we collect two directional signals. From the top, the first two signals represent z-x signals at head, and next ones are waist and left and right leg signals, respectively. We also carried out preprocessing to make synchronize signals and obtained gait signals that are illustrated in Figure 4.



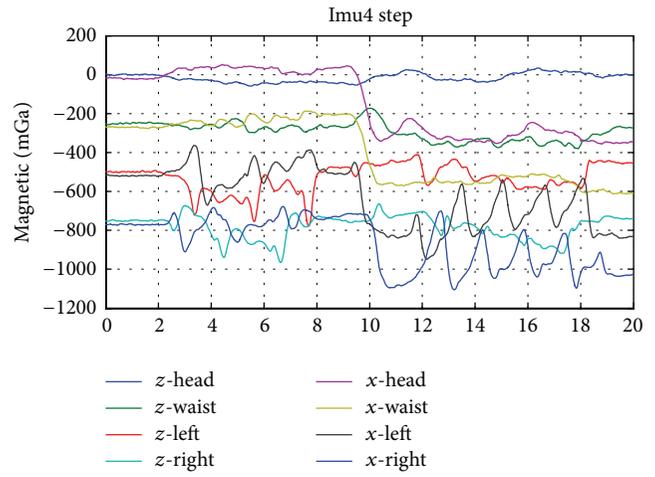
(a) Walking with acceleration sensor



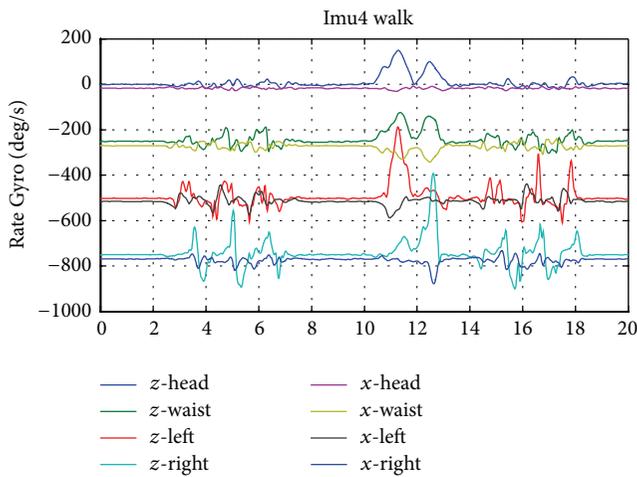
(b) Stair up with acceleration sensor



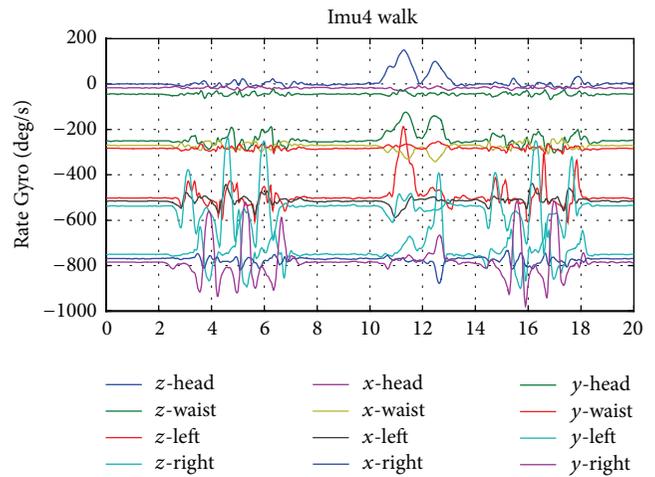
(c) Walking with magnetic sensor



(d) Stair up with magnetic sensor



(e) Walking with Gyro sensor



(f) Stair up with Gyro sensor

FIGURE 4: Gait signal with all in one sensor.

We get the signals from the control unit, and the signal is processed in a note book. Signal characteristics were considered peak value and magnitude distance between each gait signal. Next, by the application of the similarity measure, we get the calculation of each action such as walking, step up, and so on.

4. Numerical Decision Calculation

4.1. High Dimensional Analysis. Research on big data analysis has been emphasized by research outcomes recently [7, 8, 13, 19]. Big data examples are illustrated as follows.

- (i) Biomedical data such as DNA sequence or Electroencephalography (EEG) data. It contains not only high dimension but also large number of channel data.
- (ii) Recommendation systems and target marketing are important applications in the e-commerce area. Sets of customers/clients' information analysis help to predict their action to purchase based on customers' interest. It also includes a huge amount of data and high dimensional structure.
- (iii) Industry application such as EV station scheduling problem needs geometrical information, city size, population, traffic flow, and others. Hence, number of station and station size constitute huge data and high dimension.

Data might be expressed as high dimensional structure such as

$$D_i = (d_{i1}, d_{i2}, d_{i3}, \dots, d_{ij}), \quad (18)$$

where $i = 1, \dots, n$, $j = 1, \dots, m$,

where n and m denote the number of data and dimension, respectively.

Direct data comparison is applicable to overlapped data with norm definition including Euclidean norm such as

$$\begin{aligned} \mathbf{d}_1(D_i, D_j) &= \sum_{k=1}^m |d_{ik} - d_{jk}|, \quad \mathcal{L}_1 \text{ norm,} \\ \mathbf{d}_2(D_i, D_j) &= \sqrt{\sum_{k=1}^m (d_{ik} - d_{jk})^2}, \quad \text{Euclidean-norm,} \\ \mathbf{d}_p(D_i, D_j) &= \left(\sum_{k=1}^m |d_{ik} - d_{jk}|^p \right)^{1/p}, \quad \mathcal{L}_p \text{ norm.} \end{aligned} \quad (19)$$

Information distributions show various configurations, and hence it needs consider various types of distance measure to complete discriminative measure. Furthermore analysis of similarity and relation between different information should be considered carefully when it represents high-dimensional data. Specifically, m dimension represents the independent number of characteristics or attributes.

TABLE 1: Similarity measure comparison between patterns.

Similarity measure	Values
$s(W_i, SU_i)$	0.344
$s(W_i, SD_i)$	0.278
$s(SU_i, SD_i)$	0.550

TABLE 2: Average similarity measure between individuals.

Similarity measure	Values
$s(PW_i, PW_j)$	0.624
$s(PSD_i, PSD_j)$	0.643
$s(PSU_i, PSU_j)$	0.712

4.2. Illustrative Example. Hence, analysis and comparison with each attribute provide explicit importance of each data. Similarity measure provides analysis between patterns, such as

$$s(D_i, D_j), \quad \forall i, j = 1, \dots, n. \quad (20)$$

And comparison with different patterns for the same person is carrying out between walking, step up, or step down. Gait signal related with his/her movement was gathered to analyze their different patterns and different persons. Each signal constitutes walking, stair up, and stair down with 20 persons' gaits were gathered. Hence, personal information can be represented by multidimensional information such as

$$P_i = (W_i, SU_i, SD_i), \quad \text{where } i = 1, \dots, 20. \quad (21)$$

And among 20 personal data, 3 different behaviors were also expressed. Considering the data, it is obvious that data is overlapped. Hence, it is clear that similarity measures of (2) and (7)–(9) provide similarity calculation for overlapped data.

Similarity measure between person to person is expressed as

$$s(P_i, P_j) = 1 - d((P_i \cap P_j), (P_i \cup P_j)). \quad (22)$$

Also different action from the same person as follows:

$$s(W_i, SU_i) = 1 - d((W_i \cap SU_i), (W_i \cup SU_i)). \quad (23)$$

Normalized similarity calculation results are illustrated in Table 1.

Results illustrate that the stair up and down shows higher similarity than the others. However, even similarity calculation result is higher than others; it is not much close to one, it just satisfies 0.55. Due to different directions stair up/down should have basic limit to close maximum similarity.

Table 2 shows the average similarity between different individuals. Results show that the stair up is the closest even with a different gait. Naturally, walking pattern represents the least similar. In Table 3, walking similarity between different

TABLE 3: Similarity measure comparison between individuals (walking).

Similarity measure	1	2	3	4	5	6	7	...	18	19	20		
1	1	0.511	0.621	0.420	0.462	0.581	0.617		0.581	0.470	0.623		
2	0.511	1	0.592	0.590	0.490	0.632	0.543		0.603	0.619	0.577		
3	0.621	0.592	1	0.510	0.611	0.640	0.599		0.710	0.566	0.582		
4	0.420	0.590	0.510	1	0.701	0.653	0.589	...	0.612	0.629	0.585		
5	0.462	0.490	0.611	0.701	1	0.599	0.603		0.489	0.581	0.620		
6	0.581	0.632	0.640	0.653	0.599	1	0.576		0.499	0.545	0.629		
7	0.617	0.543	0.599	0.589	0.603	0.576	1		0.710	0.627	0.634		
⋮				⋮				⋮	0.590	0.611	0.589		
									0.429	0.570	0.630		
									0.559	0.628	0.549		
18	0.581	0.603	0.710	0.612	0.489	0.499	0.710	0.590	0.429	0.559	1	0.345	0.626
19	0.470	0.619	0.566	0.629	0.581	0.545	0.627	0.611	0.570	0.628	0.345	1	0.556
20	0.623	0.577	0.582	0.585	0.620	0.629	0.634	0.589	0.630	0.549	0.626	0.556	1

individuals is illustrated; symmetric results of 20 persons are listed in Table 3.

Similar results for stair up and down are obtained.

5. Conclusions

Gait signal identification was carried out through similarity measure design. Gait signal was obtained via data acquisition system including mobile station, all in one sensor attached to the head, waist, and two legs. In order to discriminate the gait signal with respect to different behaviors and individuals, similarity measure design was considered. Similarity measure was considered with the distance measure. For data distribution, overlapped and nonoverlapped distribution were considered, and similarity measure was applied to calculate the similarity. However, the conventional similarity measure was shown that it was not available to calculate the similarity on non-overlapped data. To overcome such a drawback, the similarity measure was considered with data information of neighbor. Closeness between neighbor data provides a measure of similarity among data sets; hence, the similarity measure was calculated. Calculation proposed two different artificial data, and the proposed similarity measure was useful to identify nonoverlapped data distribution. It is meaningful that similarity measure design can be extended to high dimensional data processing because gait signal was considered as a high dimensional data. With data acquisition system, 20 person gait signals were collected through experiments. Different gaits, walking, stair up, and stair down signals were obtained, and similarity measure was applied. By calculation, similarity between stair up and stair down showed higher similarity than others. Individual similarity for a different gait signal was also obtained.

Gait signal analysis can be used for behavior decision system development; it is also naturally extended to health care system, especially to elderly people. Additionally, it is also useful for athlete to provide useful information if he/she

is suffering from different actions compared to previous behavior.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported in part by a Grant from the Research Development Fund of XJTU (RDF 11-02-03).

References

- [1] D. H. Fisher, "Knowledge acquisition via incremental conceptual clustering," *Machine Learning*, vol. 2, no. 2, pp. 139–172, 1987.
- [2] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, 1988.
- [3] F. Murtagh, "A survey of recent advances in hierarchical clustering algorithms," *Computer Journal*, vol. 26, no. 4, pp. 354–359, 1983.
- [4] R. S. Michalski and R. E. Stepp, "Learning from observation: conceptual clustering," in *Machine Learning: An Artificial Intelligence Approach*, pp. 331–363, Springer, Berlin, Germany, 1983.
- [5] H. P. Friedman and J. Rubin, "On some invariant criteria for grouping data," *Journal of the American Statistical Association*, vol. 62, no. 320, pp. 1159–1178, 1967.
- [6] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, New York, NY, USA, 1990.
- [7] "Advancing Discovery in Science and Engineering. Computing Community Consortium," Spring 2011.
- [8] Advancing Personalized Education, *Computing Community Consortium*, 2011.
- [9] S. Lee and T. O. Ting, "The evaluation of data uncertainty and entropy analysis for multiple events," in *Advances in Swarm Intelligence*, pp. 175–182, Springer, New York, NY, USA, 2012.

- [10] S. Park, S. Lee, S. Lee, and T. O. Ting, "Design similarity measure and application to fault detection of lateral directional mode flight system," in *Advances in Swarm Intelligence*, vol. 7332 of *Lecture Notes in Computer Science*, pp. 183–191, Springer, New York, NY, USA, 2012.
- [11] S. Lee and T. O. Ting, "Uncertainty evaluation via fuzzy entropy for multiple facts," *International Journal of Electronic Commerce*, vol. 4, no. 2, pp. 345–354, 2013.
- [12] S. Lee, W. He, and T. O. Ting, "Study on similarity measure for overlapped and non-overlapped data," in *Proceedings of the International Conference on Information Science and Technology (ICIST '13)*, pp. 48–53, March 2013.
- [13] "Smart Health and Wellbeing," Computing Community Consortium, 2011.
- [14] X. C. Liu, "Entropy, distance measure and similarity measure of fuzzy sets and their relations," *Fuzzy Sets and Systems*, vol. 52, no. 3, pp. 305–318, 1992.
- [15] S. Lee, W. Pedrycz, and G. Sohn, "Design of similarity and dissimilarity measures for fuzzy sets on the basis of distance measure," *International Journal of Fuzzy Systems*, vol. 11, no. 2, pp. 67–72, 2009.
- [16] S. Lee, K. H. Ryu, and G. Sohn, "Study on entropy and similarity measure for fuzzy set," *IEICE Transactions on Information and Systems*, vol. 92, no. 9, pp. 1783–1786, 2009.
- [17] S. H. Lee, S. J. Kim, and N. Y. Jang, "Design of fuzzy entropy for non convex membership function," in *Advanced Intelligent Computing Theories and Applications. With Aspects of Contemporary Intelligent Computing Techniques*, vol. 15 of *Communications in Computer and Information Science*, pp. 55–60, 2008.
- [18] Y. Cheng and G. Church, "Biclustering of expression data," in *Proceedings of the 8th International Conference on Intelligent System for Molecular Biology*, 2000.
- [19] D. M. West, *Big Data for Education: Data Mining, Data Analytics, and Web Dashboards*, Governance Studies at Brookings, Washington, DC, USA, 2012.

Research Article

An Improved Ant Colony Optimization Approach for Optimization of Process Planning

JinFeng Wang, XiaoLiang Fan, and Haimin Ding

School of Energy, Power and Mechanical Engineering, North China Electric Power University, Baoding 071003, China

Correspondence should be addressed to JinFeng Wang; wml803@163.com

Received 25 April 2014; Accepted 9 June 2014; Published 6 July 2014

Academic Editor: Xin-She Yang

Copyright © 2014 JinFeng Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Computer-aided process planning (CAPP) is an important interface between computer-aided design (CAD) and computer-aided manufacturing (CAM) in computer-integrated manufacturing environments (CIMs). In this paper, process planning problem is described based on a weighted graph, and an ant colony optimization (ACO) approach is improved to deal with it effectively. The weighted graph consists of nodes, directed arcs, and undirected arcs, which denote operations, precedence constraints among operation, and the possible visited path among operations, respectively. Ant colony goes through the necessary nodes on the graph to achieve the optimal solution with the objective of minimizing total production costs (TPCs). A pheromone updating strategy proposed in this paper is incorporated in the standard ACO, which includes Global Update Rule and Local Update Rule. A simple method by controlling the repeated number of the same process plans is designed to avoid the local convergence. A case has been carried out to study the influence of various parameters of ACO on the system performance. Extensive comparative experiments have been carried out to validate the feasibility and efficiency of the proposed approach.

1. Introduction

Process planning for prismatic parts is a very complex and difficult process. For a prismatic part with complex structures and numerous features, process planning involves selecting machining operations for every feature and sequencing them considering precedence constraints, choosing available manufacturing resources, determining setup plans, and machining parameters, and so forth. In CAPP systems, these activities can be carried out simultaneously to achieve an optimal plan, thus the manufacturing efficiency could be largely increased or the production cost could be decreased. So, process planning problem is well known as a combinatorial optimization problem with constraints. With the advance of computer technology, some artificial intelligence (AI) techniques are used to solve combinatorial optimization problem. For example, some bioinspired algorithms are applied in complex decision-making process of solve combinatorial optimization problem [1–3].

In this paper, an improved ant colony optimization (ACO) approach is proposed to deal with process planning problem based on a weight graph. The weighted graph

consists of nodes, directed arcs, and undirected arcs, which denote operations, precedence constraints among operation, and the possible visited path among operations, respectively. Ant colony goes through the operation nodes on the graph along the directed/undirected arcs. The heuristic information of operation nodes and pheromone amount on the arcs will guide ant colony to achieve the optimal nodes set and arc set, which represents the optimal solution with the objective of minimizing total production costs (TPCs). Some efforts have been adopted to improve the efficiency of the approach.

2. Previous Related Works

In the past two decades, CAPP has received much attention [4–7]. Many optimization approaches have been developed and widely applied for solving process planning problem, such as knowledge-based reasoning approach [8, 9], genetic algorithm (GA) [1, 5, 10], artificial neural networks (ANN) [11], graph manipulation [7, 12], tabu search approach (TS) [6, 13], simulated annealing algorithm (SA) [13, 14], artificial immune system (AIS) [15], particle swarm optimization (PSO) [16, 17], and ant colony optimization (ACO) [18, 19].

Zhang et al. [5] constructed a novel computer-aided process planning model consisting of operation selecting and operation sequencing. A GA is proposed for process planning based on the model considering a job shop manufacturing environment. GA is used to select machining resources and sequence operations simultaneously. Ma et al. [14] modeled the constraints of process planning problems in a concurrent manner. Precedence relationships among all the operations are used to generate the entire solution space with multiple planning tasks. Based on the proposed model, an algorithm based on simulated annealing (SA) is proposed to search for the optimal solution. Li et al. [6] consider the process planning problem as a constraint-based optimization problem and propose a tabu search-based algorithm to solve it. In the proposed algorithm, costs of the machines and tools, machine changes, tool changes, setups, and penalty against good manufacturing practices are taken as the evaluation criteria. Precedence constraints between features and the corresponding operations are defined and classified according to their effects on the plan feasibility and processing quality. Chan et al. [15] model the machine tool selection and operation allocation of flexible manufacturing systems and solve process problem by a fuzzy goal—programming approach based on artificial immune systems. Guo et al. [16] proposed a PSO approach for process planning problem. The initial process plans randomly generated are encoded into particles of the PSO algorithm. To avoid falling into local optimal and improve moving efficient of the particles, several new operators have been developed. Penalty strategy is used considering the evaluation of infeasible particles. Krishna and Mallikarjuna Rao [18] proposed a novel approach to apply the ant colony algorithm as a global search technique for process planning problem by considering various feasibility constraints.

Recently, to improve the quality of results and efficiency of the search, many hybrid approaches are developed for process planning problem, for example, GA + SA [13], graph manipulation + GA [7], and local search algorithm + PSO [20]. Li et al. [6] developed a hybrid genetic algorithm and a simulated annealing approach for optimizing process plans for prismatic parts. They modeled the process planning as a combinatorial optimization problem with constraints. The evaluation criterion was the combination of machine costs, cutting tool costs, machine change costs, tool change, and setup costs. Ding et al. [20] proposed a hybrid approach to incorporate a genetic algorithm, neural network, and analytical hierarchical process (AHP) for process planning problem. A globally optimized fitness function is defined including the evaluation of manufacturing rules using AHP, calculation of cost and time, and determination of relative weights using neural network techniques. Huang et al. [7] model the process planning problem as a combinatorial optimization problem with constraints and developed a hybrid graph and genetic algorithm (GA) approach. In the approach, graph theory accompanied with matrix theory is embedded into the main frame of GA. The precedence constraints between operations are formulated in an operation precedence graph (OPG). An improved GA was applied to solve process planning problem based on the operation precedence graph (OPG).

Wang et al. [21] proposed an optimization approach based on particle swarm optimization (PSO) to solve the process planning problem and introduced a novel solution representation scheme for the application of PSO. In the hybrid approach, two kinds of local search algorithms are incorporated and interweaved with PSO evolution to improve the best solution in each generation.

Although significant improvements have been achieved for process planning problem, there still remains potential for further improvement [22]. For example, optimization approach needs to be improved to be more efficient, and a more reasonable constraint modeling and handling mechanism needs to be developed; also, some practical manufacturing environment should be considered, and the approach should provide the multiple alternative optimal plans. Especially, some bioinspired algorithms are improved to solve the complicated combination optimization problem [23, 24]. The attempt to use these algorithms to solve process planning problem should be performed to explore the more excellent results.

3. Graph-Based Process Planning Problem

In CAPP, a part is generally described by manufacturing features, which are geometric forms having machining meanings, such as holes, slots, and bosses. In the process planning for the part, the manufacturing features will be recognized by analyzing the geometrical and topological information of the part, which include position, dimensions, tolerance, and surface finish. A feature may be mapped to one or several sets of operation types (OPTs) [5]. An OPT refers to an operation without any attachment of machine (M), tool (T), and tool approach direction (TAD).

For an operation, there are a set of M s, T s, and TADs under which the operation can be executed. As a result, for a part, the process plan is a set of operations, which is represented as follows:

$$PP = \{OP_1, OP_2, \dots, OP_i\}, \quad (1)$$

where OP_i is the i th operation of the part, which is defined as follows:

$$OP_i = \{OPT_{i1}, OPT_{i2}, \dots, OPT_{ij}, \dots, OPT_{in}\}, \quad (2)$$

where OPT_{ij} is the j th alternative operation of the i th operation of the part, which is defined as follows:

$$OPT_{ij} = \{M_{ij}, T_{ij}, TAD_{ij}\}, \quad (3)$$

where M_{ij} , T_{ij} , and TAD_{ij} are the index of the machine, tool, and TAD respectively, by which the alternative operation OPT_{ij} is executed.

In process planning for a part, two tasks have to be done, namely, selecting operation OP_i for each feature of the part and sequencing operations. And, also, they must be carried out simultaneously to achieve an optimal process plan against a predetermined criterion. Due to the geometrical and manufacturing constraints between manufacturing features, operation sequencing must take into account the precedence

TABLE 1: Operation selection for the example part.

Features	Operations	Machines	Tools	TADs	Description
F1	Drilling (OP ₁)	M1	T1	-Y	M1: Vertical milling center
		M2	T1		M2: Drill press
F2	Milling (OP ₂)	M1	T2	+Z	T1: Drill
			T4	-Z	T2: End mill
			T5	-Z	T3 Drill
F3	Milling (OP ₃)	M1	T4	+Y	T4: Chamfer cutter
			T5	+Y	T5: Ball-nose cutter
			T6		T6: T-slot cutter
F4	Drilling (OP ₄)	M1	T3	-Z	
		M2			
F5	Milling (OP ₅)	M1	T2	+Z	
			T6	+Y	

TABLE 2: Precedence constraints for the example part.

ID	Features	Operations	Precedence constraints description
1	F1	OP ₁	CO ₁ is prior to CO ₂
2	F2	OP ₂	CO ₂ is prior to CO ₄
3	F4	OP ₄	CO ₄ is prior to CO ₃
4	F5	OP ₅	CO ₅ is prior to CO ₃

constraints between operations, which must be satisfied by the final operations sequence. Many constraints and rules have been proposed and summarized [1, 5, 7]. In general, these precedence constraints are as follows [19]:

- (1) primary surfaces prior to secondary surfaces,
- (2) planes prior to its associated features,
- (3) rough machining operation prior to finish machining operation,
- (4) datum surfaces prior to its associated features, and
- (5) some good manufacturing practice.

To construct process plan with the ACO approach, the process planning problem has to be visualized and represented by a weighted graph. The weighted graph is denoted as $D = (O, C, U)$, where O is a set of nodes, C is a set of directed arcs, and U is a set of undirected arcs. The nodes of O stand for all of the operations OP_i . C corresponds to the precedence constraints between the operations of the parts. U represents the set of arcs connecting all possible combinations of the nodes. Both C and U represent possible paths for ants travelling from one node to another. The ants are basically free to travel along the paths unless there is a precedence constraint specified by C . In this paper, an example part is used to illustrate the weighted graph [2].

The alternative operations for the example in Figure 1 are listed in Table 1. The precedence constraints for the example are listed in Table 2.

Figure 2 is the weighted graph for the example part. The set of nodes includes five nodes, $O_1, O_2, O_3, O_4,$ and O_5 , which represent the operations $OP_1, OP_2, OP_3, OP_4,$ and OP_5 . The

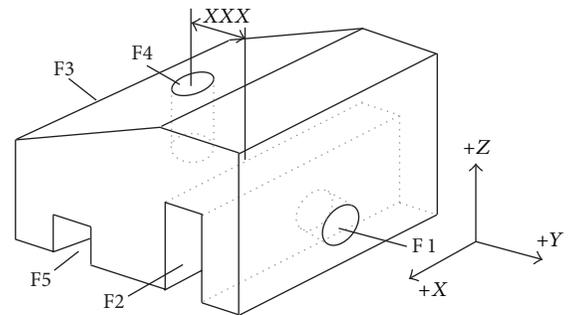


FIGURE 1: An example part [5].

set of directed arcs includes four arcs, $C_1, C_2, C_3,$ and C_4 , which represent the precedence constraints 1, 2, 3, and 4. The set of undirected arcs includes six arcs, $U_1, U_2, U_3, U_4, U_5,$ and U_6 .

While applying the ACO in the process planning by the weighted graph, the ant colony will be placed on the initial node visited by the ant colony first. The initial node determines which operation can be executed first. For the weighted graph in Figure 2, the nodes O_1 and O_5 are likely to be selected as the initial source node, since operations OP_1 and OP_5 have no precedence operations. To facilitate the execution of ACO in process planning, a dummy node O_b acting as the initial node is added to connect the possibly executed operations first in the weighted graph. The initial node O_b is used to connect the nodes O_1 and O_5 .

4. Process Plan Evaluation Criterion

Lots of process planning evaluation criteria have been proposed in the past literatures. The criterion of minimum production cost is generally used. The production cost evaluating process plans comprise five factors: machine processing cost (MC), tool processing cost (TC), machine change cost (MCC), tool change cost (TCC), and set-up change cost (SCC) [6, 8, 9, 12, 13]. The calculation procedures of these cost factors are described in detail below:

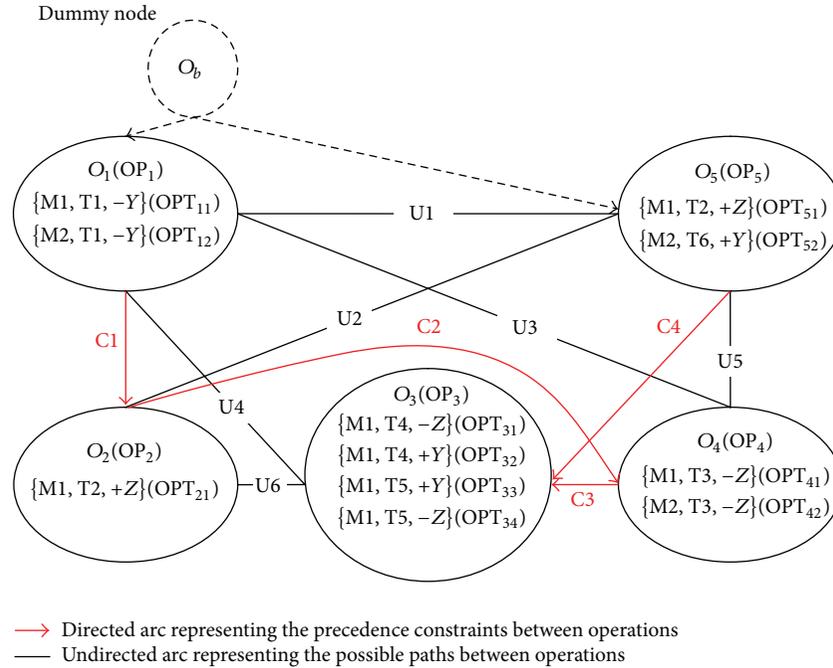


FIGURE 2: Weighted graph for the example part.

(1) total machine cost (TMC):

$$TMC = \sum_{i=1}^n MC_i, \quad (4)$$

where n is the total number of operations and MC_i is the machine cost of the i th machine for an operation, a constant for a specific machine;

(2) total tool cost (TTC):

$$TTC = \sum_{i=1}^n TC_i, \quad (5)$$

where TC_i is the tool cost of the i th tool for an operation, a constant for a specific machine;

(3) total setup cost (TSC):

$$TSC = SCC * NS, \quad (6)$$

where SCC is the setup cost and NS is the number of setups, which can be calculated by

$$NS = NSC + 1, \quad (7)$$

$$NSC = \sum_{i=1}^{n-1} \Omega_2 (\Omega_1 (M_{i+1}, M_i), \Omega_1 (TAD_{i+1}, TAD_{ii})),$$

where TAD_i is the i th TAD;

(4) total machine change cost (TMCC):

$$TMCC = MCC * NMC, \quad (8)$$

where MCC is machine change cost and NMC is number of machine change, which can be calculated by

$$NMC = \sum_{i=1}^{n-1} \Omega_1 (M_{i+1}, M_i), \quad (9)$$

$$\Omega_1 (x, y) = \begin{cases} 1 & x \neq y \\ 0 & x = y, \end{cases} \quad (10)$$

where M_i is the machine for the i th operation;

(5) total tool change cost (TTCC):

$$TTCC = TCC * NTC, \quad (11)$$

where TCC is the tool change cost and NTC is the number of tool change, which can be calculated by (10) and

$$NTC = \sum_{i=1}^{n-1} \Omega_2 (\Omega_1 (M_{i+1}, M_i), \Omega_1 (T_{i+1}, T_i)), \quad (12)$$

$$\Omega_2 (x, y) = \begin{cases} 0 & x = y = 0 \\ 1 & \text{otherwise,} \end{cases} \quad (13)$$

where T_i is the i th tool.

The definition of machine change, tool change, and setup change has been illustrated in detail [6, 9]. In this paper, the combination of TWC, TTC, TMCC, TTCC, and TSCC will be used as the objective of process planning problem, which

can be defined as total production cost (TPC) and calculated by

$$TPC = w_1 * TMC + w_2 * TTC + w_3 * TMCC + w_4 * TTCC + w_5 * TSCC. \tag{14}$$

In (14), $w_1, w_2, w_3, w_4,$ and w_5 are weights of TMC, TTC, TMCC, TTCC, and TSCC, respectively, the value of which is limited in $\{0, 1\}$. These weights can be assigned referring to the active situations, which provides the flexibility to customize the optimization objective function according to various situations [13].

5. The Proposed ACO Algorithm

The proposed ACO algorithm basically generates solutions by standard ACO procedures [2]. As described in Section 3, the directed graphs are used to represent the process planning problem [19, 25]. The approach in this paper is to solve the process planning problems using the ACO algorithm which corresponds to finding a path of the directed graph, where all necessary nodes have to be visited to complete the process plan, so that the objective of process planning is minimized. The explanations for symbols used are listed in the Symbols section.

5.1. Heuristic Information. To choose the next visiting node, the ant k is guided by the heuristic information η_{uv} on the node and the pheromone amount τ_{uv} on the arc linking the source node u and possible destination node v . η_{uv} is defined simply by a greedy heuristic

$$\eta_{uv} = \frac{E}{PC}, \tag{15}$$

where E is a positive constant, and it can be set by trial and error. Therefore, the nodes with the smaller processing cost have the higher heuristic information amount and these nodes have more attraction for the ant k . PC is the processing cost of the selected node operation and it is calculated as follows:

$$PC = w_1 * MC_i + w_2 * TC_i. \tag{16}$$

MC_i is illustrated in (4). TC_i is illustrated in (5). w_1 and w_2 are illustrated in (14).

5.2. Selection Probability. The heuristic information and the pheromone amount constructed a probability of moving from a node to another node for an ant. The more the pheromone amount on the arcs and the heuristic information on the nodes, the higher the selective probability. For the ant k , the selective probability p_{uv}^k from the source node u to the destination node v can be given as follows:

$$p_{uv}^k = \begin{cases} \frac{[\tau_{uv}^k]^\alpha [\eta_{uv}]^\beta}{\sum_{w \in S_k} [\tau_{uw}^k]^\alpha [\eta_{uw}]^\beta} & v \in S_k \\ 0 & v \notin S_k \end{cases} \tag{17}$$

where α and β denote the weighting parameters controlling the relative importance of the pheromone amount and the heuristic information, respectively. S_k is the set of nodes allowed at the next step for the ant k .

In order to adjust the convergence speed of the algorithm, a simple pheromone updating strategy is proposed in the standard ACO, which includes two pheromone updating rules. Local Update Ruler for the elite process plan is incorporated into Global Update Ruler. Three types of process planning solutions are specified at different stages of the algorithm so as to incorporate the pheromone updating strategy. Iteration best process plan PP_i denotes the best process plan generated in the current iteration by the total number of ants K , whose TPC is L_i . Restart best process plan PP_r denotes the best process plan generated since the last restart of the algorithm, whose TPC is L_r . Algorithm best process plan PP_b denotes the best process plan generated since the start of the algorithm, whose TPC is L_b . L_{avg} is the average TPC since the restart of the algorithm and is calculated as follows:

$$L_{avg} = \frac{\sum_{i=1}^{R_{ite}} L_i}{R_{ite}}. \tag{18}$$

5.3. Global Update Ruler. The pheromone level is initially set at τ_0 on every arc. Pheromone intensity on the arcs is dynamically updated after ant colony has completed process plans. The amount of pheromone deposited on the arcs by an ant k is proportional to respective L_k . The process plans with smaller L_k will accumulate a greater amount of pheromone on their corresponding arcs. To avoid unlimited accumulation of the pheromone, the pheromone also evaporates at every round of iterations. The pheromone amount τ_{uv} can be given as follows:

$$\tau_{uv}^k = (1 - \rho) * \tau_{uv}^k + \Delta\tau_{uv}^k, \tag{19}$$

where ρ is an evaporation coefficient of the pheromone on the arc linking the source node u and possible destination node v . $\Delta\tau_{uv}^k$ is the quantity of the pheromone increments on the arc (u, v) generated by the ant k after each iteration. Also, it can be given as

$$\Delta\tau_{uv}^k = \begin{cases} \frac{Q}{L_k} & \text{if } L_k \leq L_{avg} \text{ ant } k \text{ passes the arc } (u, v) \\ 0 & \text{otherwise,} \end{cases} \tag{20}$$

where Q is a positive constant. L_k is the TPC by the ant k .

5.4. Local Update Rule. Local Update Rule is introduced so that the elite process plan solutions are used to update the pheromone on the arcs again, which will accelerate the convergence of the algorithm to the optimal process plan. The iteration best process plan PP_i is first identified from all the ant process plans PP_k . If the L_i is smaller than that of L_r , L_r is replaced by L_i . Similarly, if L_r is smaller than that of L_b , L_b is replaced by L_r . Local Update Rule is used to update the pheromone intensity on the arcs again while update of L_r and

TABLE 3: Features, operations, and machining information for Part 1.

Features	Feature descriptions	Operations	TADs	Machines	Tools	Remarks
F1	Two replicated holes	Drilling (OP ₁)	+Z, -Z	M1, M2, M3	T1	M1(10):Drill press
F2	A chamfer	Milling (OP ₂)	-X, +Y, -Y, -Z	M2, M3	T8	M2(35):Vertical milling
F3	A slot	Milling (OP ₃)	+Y	M2, M3	T5, T6	M3(60):Vertical CNC milling
F4	A slot	Milling (OP ₄)	+Y	M2	T5, T6	T1(3):Drill 1
F5	A step	Milling (OP ₅)	+Y, -Z	M2, M3	T5, T6	T2(3):Drill 2
F6	Two replicated holes	Drilling (OP ₆)	+Z, -Z	M1, M2, M3	T2	T3(8):Reamer
F7	Four replicated holes	Drilling (OP ₇)	+Z, -Z	M1, M2, M3	T1	T4(15):Boring tool
F8	A slot	Milling (OP ₈)	+X	M2, M3	T5, T6	T5(10):Milling cutter 1
F9	Two replicated holes	Drilling (OP ₉)	-Z	M1, M2, M3	T1	T6(15):Milling cutter 2
F10	A slot	Milling (OP ₁₀)	-Y	M2, M3	T5, T6	T7(10):Chamfer tool
F11	A slot	Milling (OP ₁₁)	-Y	M2, M3	T5, T7	T8(10):Slot cutter
F12	Two replicated holes	Drilling (OP ₁₂)	+Z, -Z	M1, M2, M3	T1	MCC = 300
F13	A step	Milling (OP ₁₃)	-X, -Y	M2, M3	T5, T6	SCC = 120
F14	Two replicated holes	Drilling (OP ₁₄)	-Y	M1, M2, M3	T1	TCC = 15

L_b occurs. The pheromone amount τ_{uv} can be calculated as (19), and $\Delta\tau_{uv}^k$ will be calculated as follows:

$$\Delta\tau_{uv}^k = \begin{cases} \frac{Q}{L_k} & \text{if } (L_k \leq L_r \text{ or } L_k \leq L_b) \text{ ant } k \text{ passes the arc } (u, v) \\ 0 & \text{otherwise,} \end{cases} \quad (21)$$

where Q is a positive constant. L_k is the TPC by the ant k .

5.5. Termination. If all of the ants almost constructed the same process plan repeatedly at the early stage of the ACO algorithm, the algorithm would fall into the local convergence, which leads to failure in the exploration of new paths for the subsequent iteration. This is derived from an extraordinary accumulation of pheromone placed on the same set of arcs visited by the ants. Once the algorithm has fallen into the local convergence, the output of process planning would not be the optimal result, even far from the optimal results. To void the local convergence, the parameter of M_{rpt} controlling the repeated number of the same process plan is set in advance. When the adjacent two process plans are completely the same, the variable of N_{rpt} will increase by 1; otherwise N_{rpt} will be reset to be 0. When N_{rpt} reaches M_{rpt} , it means that no improvement on the solutions is made in the recent iterations. The ants may have converged to local optimal results. If the two events of $N_{\text{rpt}} = M_{\text{rpt}}$ and $N_{\text{ite}} < M_{\text{ite}}$ are satisfied simultaneously, it is considered that the local convergence occurs and the algorithm will be restarted. For the case where the algorithm is restarted, all the pheromones are reset to their initial value τ_0 , R_{ite} is reset to be 0, and L_r is reinitialized. In this case, the ants are able to escape from one particular solution to other possible paths and hence the search space will be increased. If the only event of $N_{\text{ite}} = M_{\text{ite}}$ is satisfied, the resulting process plan will be output and algorithm will be terminated.

TABLE 4: Precedence constraints for Part 1.

Constraints	Descriptions	Hard or soft
Tool interactions	OP ₁ should be prior to OP ₂ .	Hard
Datum interactions	OP ₆ should be prior to OP ₇ . OP ₁₀ should be prior to OP ₁₁ . OP ₁₃ should be prior to OP ₁₄ .	Hard
Thin-wall interactions	OP ₉ should be prior to OP ₈ . OP ₁₂ should be prior to OP ₁₀ .	Soft
Material removal interactions	OP ₈ should be prior to OP ₉ . OP ₁₀ should be prior to OP ₁₂ . OP ₁₃ should be prior to OP ₁₄ . OP ₃ should be prior to OP ₄ .	Soft

6. Experiments and Results

Two experiments have been conducted to illustrate and validate the feasibility and efficiency of the proposed approach. In the first experiment and the crucial parameters of the approach are determined. The second experiment is used to compare this approach with typical ACO, TS, GA, and SA methods.

Two prismatic parts are used for the case experiments. The first prismatic part (Part 1) used by Zhang et al. [5] is illustrated in Figure 3. It consists of 14 STEP-defined manufacturing features and 14 machining operations. The machining information and precedence constraints are given in Tables 3 and 4. The second prismatic part (Part 2) used by Li et al. [13] is illustrated in Figure 4. The machining information and precedence constraints are given in Tables 5 and 6.

6.1. Simulation Experiments. When ACO is applied in process planning, those parameters including K , ρ , α , β , E , Q , τ_0 , M_{ite} , and M_{rpt} have to be adjusted according the situation to achieve the optimal process plan. A lot of preliminary

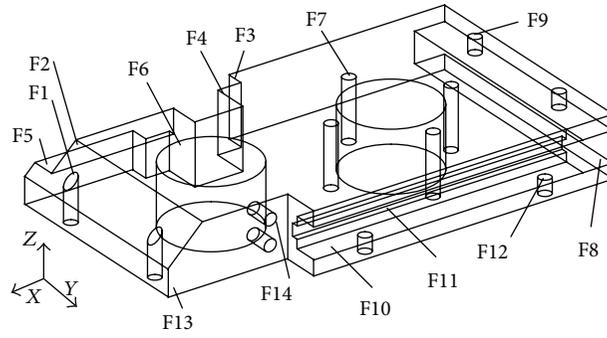


FIGURE 3: A sample part with 14 features and 14 operations: Part 1.

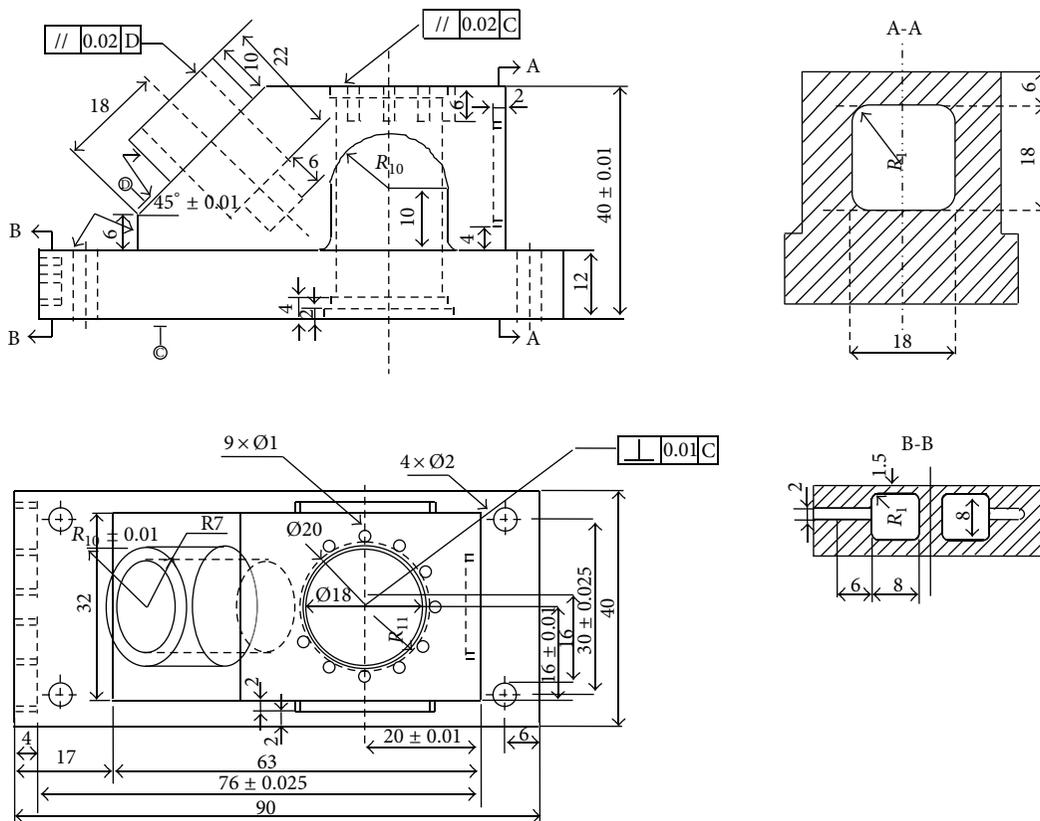
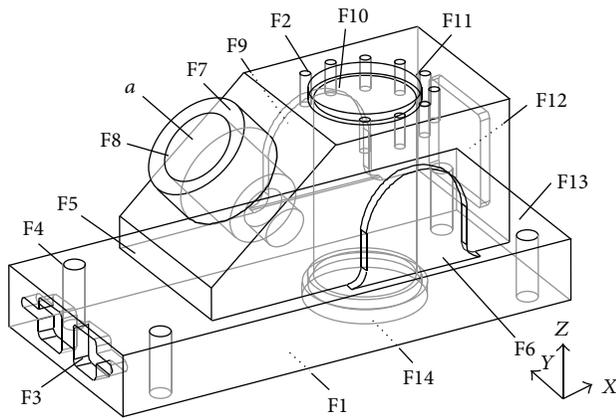


FIGURE 4: A sample part with 14 features and 20 operations: Part 2.

TABLE 5: Features, operations, and machining information for Part 2.

Features	Feature descriptions	Operations	TADs	Machines	Tools	Remarks
F1	Planar surface	Milling (OP ₁)	+Z	M2, M3	T6, T7, T8	M1(10): Drilling press
F2	Planar surface	Milling (OP ₂)	-Z	M2, M3	T6, T7, T8	M2(40): Three-axis vertical milling machine
F3	Two replicated pockets	Milling(OP ₃)	+X	M2, M3	T6, T7, T8	
F4	Four replicated holes	Drilling(OP ₄)	+Z, -Z	M1, M2, M3	T2	M3(100): CNC 3-axis vertical milling machine
F5	A step	Milling (OP ₅)	+X, -Z	M2, M3	T6, T7	
F6	A protrusion (rib)	Milling (OP ₆)	+Y, -Z	M2, M3	T7, T8	M4(60): Boring machine
F7	A boss	Milling (OP ₇)	-a	M2, M3	T7, T8	T1(7): Drill 1
F8	A compound hole	Drilling (OP ₈)	-a	M1, M2, M3	T2, T3, T4	T2(5): Drill 2
		Reaming (OP ₉)		M1, M2, M3	T9	T3(3): Drill 3
		Boring (OP ₁₀)		M2, M3	T10	T4(8): Drill 4
F9	A protrusion (rib)	Milling (OP ₁₁)	-Y, -Z	M2, M3	T7, T8	T5(7): Tapping tool
F10	A compound hole	Drilling (OP ₁₂)	-Z	M1, M2, M3	T2, T3, T4	T6(10): Mill 1
		Reaming (OP ₁₃)		M1, M2, M3	T9	T7(15): Mill 2
		Boring (OP ₁₄)		M3, M4	T10	T8(30): Mill 3
F11	Nine replicated holes	Drilling (OP ₁₅)	-Z	M1, M2, M3	T1	T9(15): Ream
		Tapping (OP ₁₆)		M1, M2, M3	T5	T10(20): Boring tool
F12	A pocket	Milling (OP ₁₇)	-X	M2, M3	T7, T8	MCC = 160
F13	A step	Milling (OP ₁₈)	-X, -Z	M2, M3	T6, T7	SCC = 100
F14	A compound hole	Reaming (OP ₁₉)	+Z	M1, M2, M3	T9	TCC = 20
		Boring (OP ₂₀)		M3, M4	T10	

TABLE 6: Precedence constraints for Part 2.

Features	Operation	Precedence constraints description	Hard or soft
F1	Milling (OP ₁)	F1 (OP ₁) is the datum and supporting face for the part; hence it is machined prior to all features and operations.	Hard
F2	Milling (OP ₂)	F2 (OP ₂) is prior to F10 (OP ₁₂ , OP ₁₃ , and OP ₁₄) and F11 (OP ₁₅ , OP ₁₆) for the material removal interactions.	Hard
F5	Milling (OP ₅)	F5(OP ₅) is prior to F4 (OP ₄) and F7(OP ₇) for the datum interactions	Hard
F6	Milling (OP ₆)	F6 (OP ₆) is prior to F10 (OP ₁₂ , OP ₁₃ , and OP ₁₄) for the datum interaction.	Hard
F7	Milling (OP ₇)	F7 (OP ₇) is prior to F8 (OP ₈ , OP ₉ , and OP ₁₀) for the datum interactions.	Hard
F8	Drilling (OP ₈)	OP ₈ is prior to OP ₉ and OP ₁₀ ; OP ₉ is prior to OP ₁₀ for the fixed order of machining operations.	Hard
	Reaming (OP ₉)		
	Boring (OP ₁₀)		
F9	Milling (OP ₁₁)	F9 (OP ₁₁) is prior to F10 (OP ₁₂ , OP ₁₃ , and OP ₁₄) for the datum interaction	Hard
F10	Drilling (OP ₁₂)	OP ₁₂ is prior to OP ₁₃ and OP ₁₄ ; OP ₁₃ is prior to OP ₁₄ for the fixed order of machining operations. F10 (OP ₁₂ , OP ₁₃ , and OP ₁₄) is prior to F11 (OP ₁₅ , OP ₁₆), and OP ₁₂ of F10 is prior to F14 (OP ₁₉ , OP ₂₀) for the datum interaction.	Hard
	Reaming (OP ₁₃)		
	Boring (OP ₁₄)		
F11	Drilling (OP ₁₅) Tapping (OP ₁₆)	OP ₁₅ is prior to OP ₁₆ for the fixed order of operations.	Hard
F13	Milling (OP ₁₈)	F13 (OP ₁₈) is prior to F4 (OP ₄) and F12 (OP ₁₇) for the material removal interaction.	Soft
F14	Reaming (OP ₁₉) Boring (OP ₂₀)	OP ₁₉ is prior to OP ₂₀ for the fixed order of machining operations.	Hard

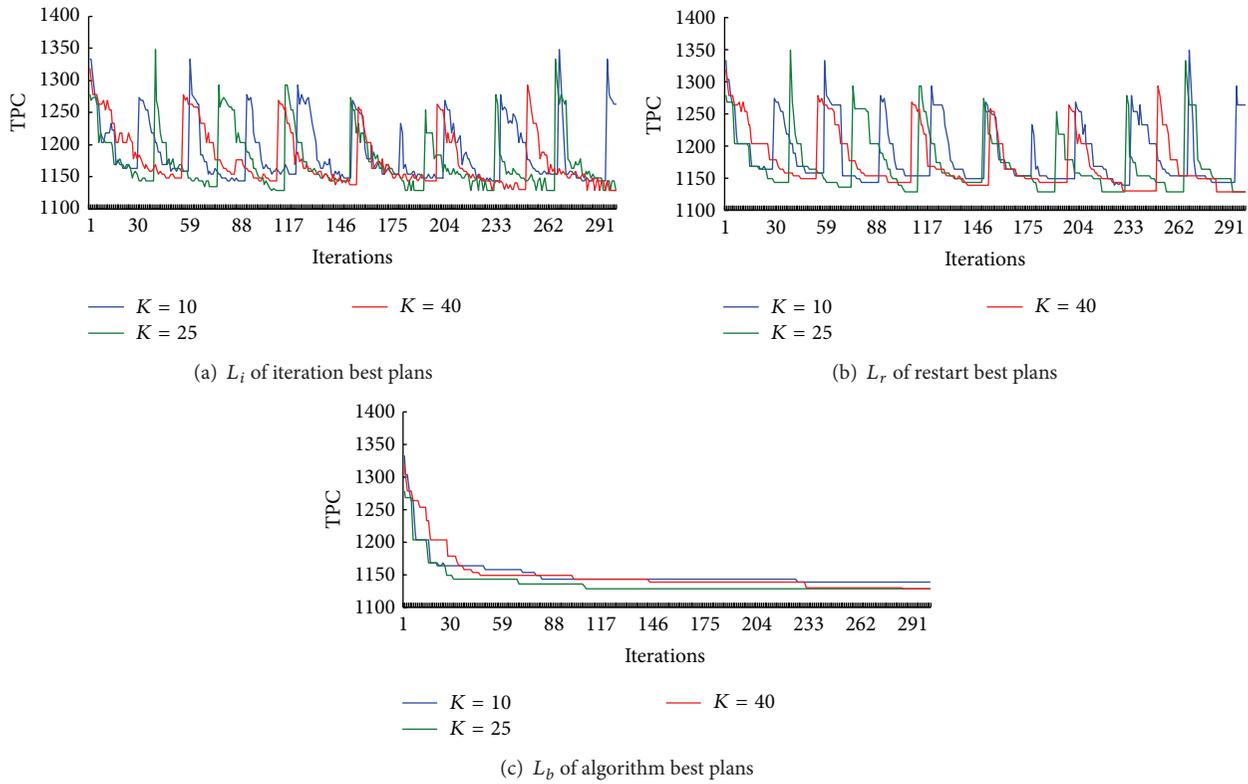


FIGURE 5: Determination of numbers of ants K .

experiments are dominated to test the effect of various parameters. In each experiment, one parameter is changed and the other parameters are fixed, and the effect of the changed parameter on the algorithm properties was analyzed at different levels. The process planning problem for Part 1 is used to illustrate how the crucial parameters are determined. It is assumed that all the machines and tools are available; namely, w_1-w_5 in (14) and (16) are set as 1.

Those parameters may be analyzed from three aspects, namely, initial parameters of ACO (K , ρ , α , β , and τ_0), problem data (M_{ite} and M_{rpt}), and problem size (E , Q). Firstly, the positive constants E and Q in (15), (20), and (21) are determined according to problem size, which includes average PC of each operation node and average TPC of process plan appearing in the previous paper. For Part 1, while the initial pheromone intensity on all the arcs τ_0 is set to 1, E and Q are fixed at 50 and 2000, respectively. Secondly, since an updating strategy including two updating rules is applied in the proposed ACO, the maximum iteration number M_{ite} just needs to guarantee the algorithm convergence. The maximum repeat number M_{rpt} affects the performance of ACO and optimization result.

Thirdly, initial parameters of ACO algorithm affect the performance of process planning using ACO. The effect is described and illustrated by an analysis of the application of the proposed ACO algorithm in the process planning problem for Part 1. Number of ants K has important effect on the convergence speed. If K is too small, searching randomness of ACO will increase and the computation time will be long. If K is too large, the optimization rate

will become very slow. Generally, value of K is considered according to the problem size. In the case of problems with $\rho = 0.75$, $\alpha = 1$, $\beta = 1$, $\tau_0 = 1$, $E = 50$, $Q = 2000$, $M_{ite} = 300$, and $M_{rpt} = 5$, 10 trials were separately conducted by varying the values of $K \in \{10, 25, 40\}$. The average results of the experiment are summarized in Figure 5.

All the hills and troughs on the TPC of L_i and L_r in Figures 5(a) and 5(b) denote the restart of the algorithm. They indicate that the local convergence avoidance mechanism takes effect to direct the ants from one solution region to another. Figure 5(b) shows that there are 10, 7, and 5 restarts corresponding to $K = 10$, $K = 25$, and $K = 40$ within the 300 iterations. Figure 5(c) shows that the compared results under $K = 10$, $K = 25$, and $K = 40$. Accordingly, K was determined as 25.

A suitable ρ can ensure good computational efficiency and algorithm stability. In the case of problems with $K = 25$, $\alpha = 1$, $\beta = 1$, $\tau_0 = 1$, $E = 50$, $Q = 2000$, $M_{ite} = 300$, and $M_{rpt} = 5$, 10 trials were separately conducted by varying the values of $\rho \in \{0.25, 0.5, 0.75\}$. The average results of L_b achieved by the algorithm best process PP_b are summarized in Figure 6.

In the case of problems with $K = 25$, $\rho = 0.75$, $\tau_0 = 1$, $E = 50$, $Q = 2000$, $M_{ite} = 300$, and $M_{rpt} = 5$, 10 trials were separately conducted by varying the values of $\alpha \in \{0.1, 1, 5\}$ and $\beta \in \{0.1, 1, 5\}$. The average results of L_r achieved by the restart best process PP_r are summarized in Table 7.

50 trials were separately conducted to evaluate the performance of the proposed approach. The results show that these parameters have a good performance at values $K = 25$,

TABLE 7: Determination of varying combinations of α and β .

	$\alpha = 0.1$			$\alpha = 1$			$\alpha = 5$		
	$\beta = 0.1$	$\beta = 1$	$\beta = 5$	$\beta = 0.1$	$\beta = 1$	$\beta = 5$	$\beta = 0.1$	$\beta = 1$	$\beta = 5$
Mean	1137.1	1134.4	1132.6	1136.1	1129.1	1132.8	1336.6	1129.9	1136.9
Maximum	1150.5	1147	1143.5	1148.5	1137	1145	1149	1141.5	1149
Minimum	1131	1128	1128	1128	1128	1128	1128	1128	1129.5

TABLE 8: One of the best process plans for Part 1.

Operation	6	1	7	9	12	5	3	4	8	10	11	13	14	2
Machine	2	2	2	2	2	2	2	2	2	2	2	2	2	2
Tool	2	1	1	1	1	5	5	5	5	5	5	5	1	8
TAD	-Z	-Z	-Z	-Z	-Z	-Z	+Y	+Y	+X	-Y	-Y	-Y	-Y	-Y

NMC = 0, NTC = 4, NSC = 3, TMCC = 0, TTCC = 60, TSCC = 480, TMC = 490, TTC = 98, and TPC = 1128

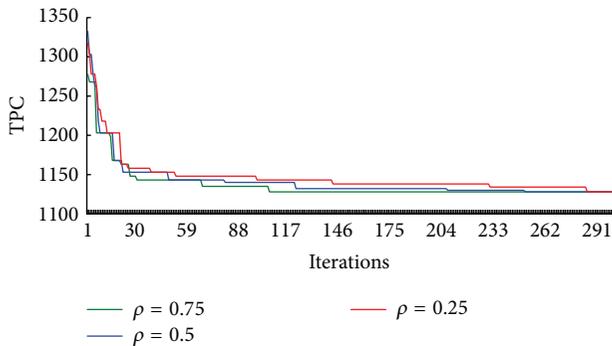


FIGURE 6: Determination of pheromone evaporation rate ρ .

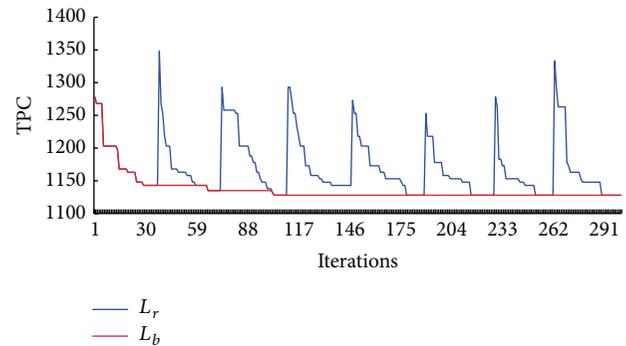


FIGURE 7: Simulation results of L_r and L_b corresponding to one of the best process plans.

$\rho = 0.75, \alpha = 1, \beta = 1, \tau_0 = 1, E = 50, Q = 2000, M_{ite} = 300,$ and $M_{rpt} = 5,$ under which one of the best process plans is shown in Table 8 and the corresponding simulation results of L_r and L_b are in Figure 7.

Figure 7 shows that there are 7 restarts within the 300 iterations. When iterations are between 34 and 38, the ants repeatedly generate plans without any further improvement on the TPC under 1143. The local convergence avoidance mechanism is triggered to renew all the pheromone trails and bring the ants to other search regions. Therefore, the first restart occurs on the iteration 39, on which the ants are released to construct process plans with a TPC of 1348. Although it is larger than 1143, the ACO algorithm is able to bring the ants to the better solutions again.

The above experiments are based on Part 1. To ensure that these parameters are applicable in other situations, the extensive comparative experiments for Part 2 will use the same chosen parameters.

6.2. Extensive Comparative Experiments. Three conditions are used to test proposed algorithm for the sample parts [6, 13].

- (1) All machines and tools are available, and w_1-w_5 in (11) and (13) are set as 1.
- (2) All machines and tools are available, and $w_2 = w_5 = 0; w_1 = w_3 = w_4 = 1.$

- (3) Machine M2 and tool T7 are down, and $w_2 = w_5 = 0; w_1 = w_3 = w_4 = 1.$

Under condition (1), condition (2), and condition (3), 10 trials were separately conducted to evaluate the proposed algorithm's performance for Part 2. Experimental observation has shown that $K = 40, \rho = 0.75, \alpha = 2, \beta = 1, \tau_0 = 1, E = 100, Q = 3000, M_{ite} = 300,$ and $M_{rpt} = 5$ are the best choices of these parameters. Under condition (1), one of the best operation sequences is shown in Table 9. Under condition (2), one of the best operation sequences is shown in Table 10. Under condition (3), one of the best operation sequences using proposed algorithm is shown in Table 11.

In Table 12, the TPCs generated by the proposed ACO are compared with those of GA and SA approach by Li et al. [13], TS by Li et al. [6], and the ACO by Liu et al. [19]

The comparing results show that the proposed algorithm is better than the other algorithms. Under condition (1), a lower TPC (2435.0) has been found using the improved ACO approach, and the mean TPC (2456.1) is better than the costs of other four algorithms. Under condition (2), a lower TPC (1970.0) has been found using the improved ACO approach. Under condition (3), the minimum TPC (2580) is the same as the TS [6]. The mean TPC generated by proposed approach is better than the other algorithms under the three conditions.

TABLE 9: One of the best process plans for Part 2 under condition (1).

Operation	1	2	6	11	18	12	13	19	17	3	5	7	8	9	10	20	14	4	15	16
Machine	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	4	4	1	1	1
Tool	7	7	7	7	7	3	9	9	7	7	7	7	3	9	10	10	10	2	1	5
TAD	+Z	-Z	-Z	-Z	-Z	-Z	-Z	+Z	-X	+X	+X	-a	-a	-a	-a	+Z	-Z	-Z	-Z	-Z

NMC = 2, NTC = 10, NSC = 8, TMCC = 320, TTCC = 200, TSCC = 900, TMC = 750, TTC = 265, and TPC = 2435

TABLE 10: One of the best process plans for Part 2 under condition (2).

Operation	1	2	18	11	6	12	13	19	17	3	5	7	8	9	10	20	14	4	15	16
Machine	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	4	4	1	1	1
Tool	7	7	7	7	7	3	9	9	7	7	7	7	7	3	9	10	10	2	1	5
TAD	+Z	-Z	-Z	-Z	-Z	-Z	-Z	+Z	-X	+X	+X	-a	-a	-a	-a	+Z	-Z	-Z	-Z	-Z

NMC = 2, NSC = 8, TMCC = 320, TSCC = 900, TMC = 750, and TPC = 1970

TABLE II: One of the best process plans for Part 2 under condition (3).

Operation	1	6	2	5	11	12	13	14	18	17	7	8	9	10	19	20	3	4	15	16
Machine	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	1	1	1
Tool	6	6	6	6	8	2	9	10	6	8	8	2	9	10	9	10	6	2	1	5
TAD	+Z	-Z	-X	-X	-a	-a	-a	-a	+Z	+Z	+X	-Z	-Z	-Z						

NMC = 1, NSC = 6, TMCC = 160, TSCC = 700, TMC = 1730, and TPC = 2590

TABLE 12: Results compared to other algorithms for Part 2.

Condition	Proposed approach	ACO	TS	SA	GA
(1)					
Mean	2456.1	2490.0	2609.6	2668.5	2796.0
Maximum	2527.0	2500.0	2690.0	2829.0	2885.0
Minimum	2435.0	2450.0	2527.0	2535.0	2667.0
(2)					
Mean	2115.4	2117.0	2208.0	2287.0	2370.0
Maximum	2380.0	2120.0	2390.0	2380.0	2580.0
Minimum	1970.0	2090.0	2120.0	2120.0	2220.0
(3)					
Mean	2600	2600.0	2630.0	2630.0	2705.0
Maximum	2740.0	2600.0	2740.0	2740.0	2840.0
Minimum	2580.0	2600.0	2580.0	2590.0	2600.0

7. Conclusions

An improved ACO approach is developed to solve the process planning optimization problem for prismatic parts. The approach is characterized by the following aspects.

- (1) A weighted graph is used to represent process planning problem. The graph includes nodes set, directed arcs set, and undirected arcs set, which denote operations, precedence constraints between the operations, and possible visited path connecting the nodes, respectively.
- (2) A pheromone updating strategy proposed in the proposed ACO is incorporated in the standard ACO, which includes Global Update Rule and Local Update Rule. A simple method by controlling the repeated number of the same process plan is designed to avoid the local convergence.

In a further study, a deep discussion of selecting the ACO approach parameters is conducted. In addition, the multi-objective optimization will be incorporated into the ACO approach for handling the multiobjective process planning problem.

Symbols

K : Number of ants
 k : Index of ant, $k \in [1, K]$
 u : Source node
 v : Destination node
 τ_{uv} : Pheromone
 η_{uv} : Heuristic information
 α : Relative weight of pheromone τ_{uv}
 β : Relative weight of heuristic information η_{uv}
 ρ : Pheromone evaporation rate
 E : Algorithm constant to determine η_{uv}
 Q : Algorithm constant to determine $\Delta\tau$
 τ_0 : Initial value of pheromone
 PP_k : Process plan achieved by ant k
 L_k : TPC achieved by ant k
 S_k : Set of nodes allowed by ant k

PP_b : Up-to-now best process plan
 L_b : Up-to-now best TPC
 PP_i : Iteration best process plan
 L_i : Iteration best TPC
 PP_r : Restart best process plan
 L_r : Restart best TPC
 L_{avg} : Average value of TPC since the latest restart
 M_{rpt} : Maximum number of repeats
 M_{ite} : Maximum number of iterations
 N_{rpt} : Number of repeats
 N_{ite} : Number of iterations
 R_{ite} : Number of iterations since the latest restart.

Conflict of Interests

All of authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported by the Fundamental Research Funds for the Central Universities (2014ZD37).

References

- [1] J. Vancza and A. Markus, "Genetic algorithms in process planning," *Computers in Industry*, vol. 17, no. 2-3, pp. 181-194, 1991.
- [2] M. Dorigo, V. Maniezzo, and A. Coloni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 26, no. 1, pp. 29-41, 1996.
- [3] P. R. Srivastava, A. Varshney, P. Nama, and X. Yang, "Software test effort estimation: a model based on cuckoo search," *International Journal of Bio-Inspired Computation*, vol. 4, no. 5, pp. 278-285, 2012.
- [4] L. Altıng and H. C. Zhang, "Computer aided process planning: the state-of-the-art survey," *International Journal of Production Research*, vol. 27, no. 4, pp. 553-585, 1989.
- [5] F. Zhang, Y. F. Zhang, and A. Y. C. Nee, "Using genetic algorithms in process planning for job shop machining," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 4, pp. 278-289, 1997.
- [6] W. D. Li, S. K. Ong, and A. Y. C. Nee, "Optimization of process plans using a constraint-based tabu search approach," *International Journal of Production Research*, vol. 42, no. 10, pp. 1955-1985, 2004.
- [7] W. Huang, Y. Hu, and L. Cai, "An effective hybrid graph and genetic algorithm approach to process planning optimization for prismatic parts," *The International Journal of Advanced Manufacturing Technology*, vol. 62, no. 9-12, pp. 1219-1232, 2012.
- [8] J. M. Usher and G. C. Sharma, "Intelligent reasoning in the generation of alternative sequences for feature-based process planning," *Intelligent Automation and Soft Computing*, vol. 3, no. 3, pp. 207-220, 1997.
- [9] Y. Tseng and C. Liu, "Concurrent analysis of machining sequences and fixturing set-ups for minimizing set-up changes

- for machining mill-turn parts," *International Journal of Production Research*, vol. 39, no. 18, pp. 4197–4214, 2001.
- [10] Z. W. Bo, L. Z. Hua, and Z. G. Yu, "Optimization of process route by Genetic Algorithms," *Robotics and Computer-Integrated Manufacturing*, vol. 22, no. 2, pp. 180–188, 2006.
- [11] S. Sette, L. Boullart, and L. van Langenhove, "Optimising a production process by a neural network/genetic algorithm approach," *Engineering Applications of Artificial Intelligence*, vol. 9, no. 6, pp. 681–689, 1996.
- [12] H. Zhang and E. Lin, "A hybrid-graph approach for automated setup planning in CAPP," *Robotics and Computer-Integrated Manufacturing*, vol. 15, no. 1, pp. 89–100, 1999.
- [13] W. D. Li, S. K. Ong, and A. Y. C. Nee, "Hybrid genetic algorithm and simulated annealing approach for the optimization of process plans for prismatic parts," *International Journal of Production Research*, vol. 40, no. 8, pp. 1899–1922, 2002.
- [14] G. H. Ma, Y. F. Zhang, and A. Y. C. Nee, "A simulated annealing-based optimization algorithm for process planning," *International Journal of Production Research*, vol. 38, no. 12, pp. 2671–2687, 2000.
- [15] F. T. S. Chan, R. Swarnkar, and M. K. Tiwari, "Fuzzy goal-programming model with an artificial immune system (AIS) approach for a machine tool selection and operation allocation problem in a flexible manufacturing system," *International Journal of Production Research*, vol. 43, no. 19, pp. 4147–4163, 2005.
- [16] Y. W. Guo, A. R. Mileham, G. W. Owen, and W. D. Li, "Operation sequencing optimization using a particle swarm optimization approach," *Proceedings of the Institution of Mechanical Engineers B: Journal of Engineering Manufacture*, vol. 220, no. 12, pp. 1945–1958, 2006.
- [17] X. Li, L. Gao, and X. Wen, "Application of an efficient modified particle swarm optimization algorithm for process planning," *International Journal of Advanced Manufacturing Technology*, vol. 67, no. 5–8, pp. 1355–1369, 2013.
- [18] A. G. Krishna and K. Mallikarjuna Rao, "Optimisation of operations sequence in CAPP using an ant colony algorithm," *International Journal of Advanced Manufacturing Technology*, vol. 29, no. 1-2, pp. 159–164, 2006.
- [19] X. Liu, H. Yi, and Z. Ni, "Application of ant colony optimization algorithm in process planning optimization," *Journal of Intelligent Manufacturing*, vol. 24, no. 1, pp. 1–13, 2013.
- [20] L. Ding, Y. Yue, K. Ahmet, M. Jackson, and R. Parkin, "Global optimization of a feature-based process sequence using GA and ANN techniques," *International Journal of Production Research*, vol. 43, no. 15, pp. 3247–3272, 2005.
- [21] Y. F. Wang, Y. F. Zhang, and J. Y. H. Fuh, "A hybrid particle swarm based method for process planning optimisation," *International Journal of Production Research*, vol. 50, no. 1, pp. 277–292, 2012.
- [22] X. Xu, L. Wang, and S. T. Newman, "Computer-aided process planning: a critical review of recent developments and future trends," *International Journal of Computer Integrated Manufacturing*, vol. 24, no. 1, pp. 1–31, 2011.
- [23] Y. Xin-She and H. Kingshi, "Bat algorithm: literature review and applications," *International Journal of Bio-Inspired Computation*, vol. 5, no. 3, pp. 141–149, 2013.
- [24] I. Fister, X. Yang, J. Brest, and I. Fister Jr., "Modified firefly algorithm using quaternion representation," *Expert Systems with Applications*, vol. 40, no. 18, pp. 7220–7230, 2013.
- [25] Y. K. Kim, K. Park, and J. Ko, "A symbiotic evolutionary algorithm for the integration of process planning and job shop scheduling," *Computers & Operations Research*, vol. 30, no. 8, pp. 1151–1171, 2003.

Research Article

Integrated Model of Multiple Kernel Learning and Differential Evolution for EUR/USD Trading

Shangkun Deng and Akito Sakurai

Graduate School of Science and Technology, Keio University, 3-14-1 Hiyoshi, Kohoku-ku, Yokohama 223-8522, Japan

Correspondence should be addressed to Shangkun Deng; dsk8672@gmail.com

Received 29 March 2014; Accepted 16 June 2014; Published 6 July 2014

Academic Editor: Xin-She Yang

Copyright © 2014 S. Deng and A. Sakurai. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Currency trading is an important area for individual investors, government policy decisions, and organization investments. In this study, we propose a hybrid approach referred to as MKL-DE, which combines multiple kernel learning (MKL) with differential evolution (DE) for trading a currency pair. MKL is used to learn a model that predicts changes in the target currency pair, whereas DE is used to generate the buy and sell signals for the target currency pair based on the relative strength index (RSI), while it is also combined with MKL as a trading signal. The new hybrid implementation is applied to EUR/USD trading, which is the most traded foreign exchange (FX) currency pair. MKL is essential for utilizing information from multiple information sources and DE is essential for formulating a trading rule based on a mixture of discrete structures and continuous parameters. Initially, the prediction model optimized by MKL predicts the returns based on a technical indicator called the moving average convergence and divergence. Next, a combined trading signal is optimized by DE using the inputs from the prediction model and technical indicator RSI obtained from multiple timeframes. The experimental results showed that trading using the prediction learned by MKL yielded consistent profits.

1. Introduction

The foreign exchange (FX) market is considered to be the largest financial market in the world. In the last few decades, currency trading has received considerable attention from researchers, individual investors, international trade companies, and government organizations. However, there is a problem with predicting directional change in the FX because it is affected by many factors, including financial policy, market mood, or even natural disasters such as earthquakes.

In general, researchers use technical indicators as features of the raw stock prices or FX rates. A technical indicator of stock prices or FX rates is a function that returns a value for given prices over a given length of time in the past. These technical indicators might provide traders with guidance on whether a currency pair is oversold or overbought, or whether a trend will continue or halt. Moving average (MA) [1] is the best-known technical indicator and it is also the basis of many other trend-following or overbought/oversold indicators. The MA is inherently a follower rather than a

leader, but it reflects the underlying trend in many cases. Many well-known advanced technical indicators are based on the MA, such as the MACD [2], RSI [3], BIAS ratio [4], and Bollinger Bands [5]. In general, the MACD is used to capture a trend while the RSI, BIAS ratio, and Bollinger Bands are used to provide an early warning of an overbought or oversold currency pair. Traders can follow the trend if it continues but they should also be cautious not to miss overbought or oversold signals related to the target trading stocks or currency pairs.

Previous researchers have used technical indicators such as some MA based methods to identify trends or used technical indicators such as the RSI, William %R, or BIAS ratio to determine whether a target currency pair has been overbought or oversold. For example, Jaruszewicz and Mańdziuk [6] applied technical analysis to predict the Japanese NIKKEI index and so they claimed that the technical indicators are useful in a short time as a day for time horizon. Deng et al. [7] used several technical indicators such as RSI, BIAS ratio, and William %R to generate trading rules by calculating

a linear combination of three technical indicators and a stock price change rate predicted value. Wei et al. [8] used several technical indicators such as RSI, MA, and William %R and their values calculated from historical prices were used as conditional features. Chong and Ng [9] predicted the London Stock Exchange based on technical indicators such as the MACD and RSI to generate trading rules, such as “a buy signal is triggered when the RSI crosses the center line (50) from below, while a sell signal is triggered when the RSI crosses the center line (50) from above,” and they found that trading strategy based on RSI or MACD obtained better return than buy-and-hold strategy. Comparing with the previous research of Jaruszewicz and Mańdziuk [6] and Chong and Ng [9], in our proposed method, we used a technical indicator to predict the directional change and used a technical indicator to find overbought/oversold conditions and then to combine a directional change signal with a trade signal from an overbought/oversold indicator, which may provide more reliable trading signal.

In recent years, machine learning techniques have been used increasingly as alternative methods to help investors or researchers forecast directional changes in stock prices or FX rates. The most popular and useful methods are support vector machines (SVMs) and genetic algorithms (GAs). Researchers often apply SVMs to predict directional changes or GAs to generate trading rules based on combinations of trading parameters. For example, Kamruzzaman et al. [10] used a SVM based model to predict FX rates. Shioda et al. [11] used a SVM for monitoring to predict the high volatility of FX rates. Other researchers have used GAs to generate trading rules. For example, Chang Chien and Chen [12] used a GA based model to generate rules for stock trading by mining associative classification rules. Deng and Sakurai [13] used GA to generate trading rules based on a technical indicator for FX trading. Hirabayashi et al. [14] used a GA to generate rules for FX intraday trading by mining features from several technical indicators. Esfahanipour and Mousavi [15] used a GA to generate risk-adjusted rules for trading.

In addition to GAs, differential evolution (DE) was proposed by Storn and Price [16] and it is a population based stochastic search, which functions as an efficient global optimizer in continuous search domains. DE has been applied successfully in various fields. For example, Worasuchep [17] used DE for forecasting the stock exchange index of Thailand. Takahama et al. [18] used DE to optimize neural networks for predicting stock prices. Peralta et al. [19] compared DE and GA for time series prediction and showed that the performance of DE was better than GA if more than 150 generations were generated.

In addition to SVMs, in the last decade, many researchers have used the multiple kernel learning (MKL) [20, 21] to address the problem of selecting suitable kernels for different feature sets. This technique mitigates the risk of erroneous kernel selection to some degree by taking a set of kernels, deriving a weight for each kernel, and making better predictions based on the weighted sum of the kernels. One of the major advantages of MKL is that it can combine different kernels for various input features. Many researchers have applied MKL in their research fields. For example, MKL was

used by Joutou and Yanai [22] for food image recognition. Foresti et al. [23] used MK regression for wind speed prediction and their results outperformed those of several conventional methods. Recently, researchers have used MKL for predicting the FX rate, crude oil prices, and stock prices. For example, Deng et al. [24] used MKL to fuse the information from stock time series and social network service for stock price prediction. Deng and Sakurai [25] used MKL for prediction and trading on crude oil markets. Fletcher et al. [26] used MKL for predicting the FX market from the limit order book. Luss and D’Aspremont [27] employed MKL for predicting abnormal returns based on the news using text classification. Yeh et al. [28] used MKL to predict stock prices on the Taiwan stock market and they showed that MKL was better than SVM for evaluating performances. Deng et al. [7] used MKL to predict short-term foreign exchange rate, and the prediction results of MKL based method are much better than conventional methods, in terms of root mean square (RMSE). The difference between the method proposed in Deng et al. [7] and this study is that the proposed method in this study uses one MKL to predict upward trend and uses another MKL for prediction of downward trend, while the method in Deng et al. [7] is used MKL to predict the change rates of FX rate. The reason for using one MKL to predict upward trend and using another MKL to predict downward trend is that our classification is a three-classification problem (upward trend, downward trend, and unknown). In addition, this study uses one technical indicator (RSI) but from three different timeframes, while Deng et al. [7] used three technical indicators but from one timeframe. Deng et al. [7] used multiple technical indicators because of the differences between different technical indicators since they may provide different trading signals, while this research used multiple timeframes of one technical indicator since different timeframes of the same technical indicator may provide different trading signals. In addition to using individual method, several researchers have used hybrid models for trading stocks or FX rate prediction. For example, Huang and Wu [29] used SVM and GA integrated model for predicting a stock index. Huang [30] combined SVM with GA to produce a stock selection model. The better performances of the hybrid SVM-GA model than individual method (SVM or GA), the superiority of DE to GA [19], and superiority of MKL to SVM [22, 26, 28] inspired us to try a new hybrid model which combines MKL and DE. It is logically expected that a MKL-DE will perform better than the previous methods.

In the present study, we use a hybrid method based on MKL and DE for prediction and to generate the trading rules for trading currency rates. In addition, we noticed that some researchers focused on extreme returns or abnormal movements of stock prices. For example, Beneish et al. [31] used contextual fundamental analysis for stock prediction and they focused only on extreme returns, that is, returns above a threshold. Luss and D’Aspremont [27] used MKL and they focused on abnormal movements, which were movements above a threshold. Inspired by their research, in this study, we use MKL to generate signals for upward trends, downward trends, and no trend. The directional

change predictor performs learning to predict the direction of price movements. The direction of movement is classified as an upward trend, a downward trend, or a probabilistic fluctuation. Thus, we simply set a threshold for the absolute values of changes, below which we consider the change to be a fluctuation.

In addition to trends, traders also consider the possibility of overbought or oversold conditions for the target currency pair. For example, if a trader predicts an upward trend but the target currency pair is overbought, that is, at a high level, it will be risky to continue following the trend. We could use a technical indicator as a tool to determine the degree to which the FX pair is oversold or overbought, before generating trading actions (buy, sell, or no trade) based on the overbought or oversold signal. In this study, we define the overbought or oversold signals based on a RSI (refer to Section 2.1.3).

Our trading time horizon is 1 hour, which means that we assess overbought or oversold signals based only on 1-hour time frame data. Clearly, it is possible that the judgment would be different if we made assessments using a longer or shorter timeframe. For example, Figure 1 shows the EUR/USD rate and its RSI values for 1-hour and 2-hour timeframes (i.e., 1-hour RSI and 2-hour RSI values). Note that at the eighth point (10:00:00, May 5, 2011) in Figure 1, the 1-hour RSI value is approximately 73.90, which provides us with a sell signal because the currency pair is overbought, whereas the 2-hour RSI value is approximately 43.98, which tells us that the currency is not overbought. The rate increased further from the eighth to the ninth point (11:00:00, May 5, 2011). In addition, the 1-hour RSI value is approximately 78.32 at the ninth point and the 2-hour RSI value is approximately 71.71, which suggests that both values provide overbought signals so it is highly probable that the rate will decrease from the ninth point onwards. This example shows that if we use the RSI to generate trading rules, we must assess the overbought or oversold conditions not only for the target timeframe, but also for relatively longer and shorter timeframes. For example, the features of the RSI from a relatively shorter timeframe (i.e., 30 minutes in this study) and a relatively longer timeframe (i.e., 2 hours) were used in this study as suitable signals for trading a target currency pair.

In the present study, we use the MACD indicator of two currency pairs as features, rather than only the target currency pair, and the RSI indicator from two different timeframes of the target trading currency pair, rather than the target timeframe.

According to the 2010 Triennial Survey (the share of trading volume), the most heavily traded currency pairs were: EUR/USD 28%, USD/JPY 14%, and GBP/USD 9%. The EUR/USD is the most traded currency pair in the world, so this is used as our target trading currency pair. JPY and GBP are the two most highly exchanged currencies with both USD and EUR, so we also employ GBP/USD and USD/JPY as supplementary information for predicting our target currency pair.

Evaluations of the experimental results should be based on the return-risk ratio as well as the return and the average return, because most investors prefer to obtain stable returns,

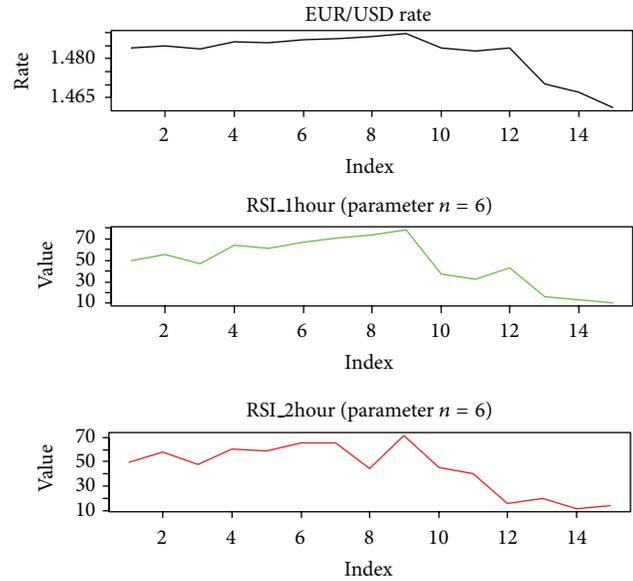


FIGURE 1: Example showing the relative strength index values from multiple timeframes.

rather than high returns with high volatility, that is, high risk. Therefore, the Sharpe ratio [32] is used as an evaluation measure to adjust the risk, in addition to the average return.

In summary, this study makes three main innovations, as follows: (1) to predict directional changes of EUR/USD, we set thresholds on the magnitude of the FX rate changes to distinguish upward trend or downward trend from random fluctuations to predict the return, whereas only a few studies employed this process. (2) To generate a trade signal, we fuse information from multiple currency pairs other than only the target currency pair and we combined multiple RSIs from multiple timeframes other than only the target trading timeframe, whereas many previous researchers have considered only the target trading currency pair with a target trading timeframe. (3) The hybrid model combined an upward trend/downward trend signal with the multiple RSI signal, and the hybrid model yielded greater profits. Proposed model outperformed the baseline and other methods based on the results of return and the return-risk ratio.

The remainder of this paper is organized as follows. Section 2 describes the background of this research. Section 3 explains the structure of the proposed method. Section 4 describes the experimental design. Section 5 presents the experimental results and provides a discussion. Section 6 concludes the paper.

2. Background

2.1. Technical Indicators. Technical indicators are broadly classified into two types: trend indicators and oscillator indicators. The best-known trend indicator is the MA, which is the basis of most other indicators. Next, we introduce the three technical indicators used in this study: MA, MACD as a trend indicator, and RSI as an overbought/oversold indicator.

2.1.1. Simple MA and Exponential MA. The MA is a technique for smoothing out short-term fluctuations, which can be obtained by calculating the mean value of the prices over the past n -periods. The MA is used to understand the present trend, which is why it is a so-called trend-following index. There are several types of MA, depending on how past prices are weighted.

The simple MA (SMA) is a simple mean value with identical weights for past prices:

$$SMA_n(t) = \frac{\sum_{k=t-n+1}^t P(k)}{n}, \quad (1)$$

where n is the period length and $P(k)$ is the foreign exchange rate or some other value under consideration.

Another type of MA, the exponential MA (EMA), is the mean of the underlying data, which is generally the price of a stock or foreign exchange rate for a given time period n , where larger weights are attributed to narrower changes. The difference between the EMA and the SMA is that the EMA is concerned more with the nearest movements, which may have greater effects on future changes than older changes. The EMA is calculated as follows:

$$EMA_n(t) = P(t) * a + (1 - a) * EMA_n(t - 1), \quad (2)$$

where $EMA_n(t)$ is the EMA of the rate at time t and $a = 2/(n+1)$, which is commonly used for the n -period EMA.

2.1.2. MACD. The MACD is used to predict trends in time series data and it provides two indicators: the MACD value and the MACD signal. In general, the MACD value is the difference between the 12-period and 26-period EMAs, as follows:

$$MACD_{\text{value}}(t) = EMA_{12}(t) - EMA_{26}(t). \quad (3)$$

The MACD signal is equal to the 9-period EMA of the MACD value, as follows:

$$MACD_{\text{signal}}(t) = EMA_9(MACD_{\text{value}}(t)). \quad (4)$$

The MACD parameters (12, 26, and 9) can be adjusted to meet the needs of traders. In our study, we simply use the default MACD parameters given above because they are used widely throughout the world.

2.1.3. RSI. In general, traders use the RSI as a momentum oscillator to compare the magnitude of recent gains with the magnitude of recent losses. If we let $P(t)$ represent the closing price on day t , then we can calculate the gain or loss in period t as follows:

$$G_t = \begin{cases} P(t) - P(t-1) & \text{if } P(t) > P(t-1) \\ 0 & \text{otherwise,} \end{cases} \quad (5)$$

$$L_t = \begin{cases} P(t) - P(t-1) & \text{if } P(t) < P(t-1) \\ 0 & \text{otherwise.} \end{cases}$$

Next, the n -period average gain ($AG(t)$) is calculated as

$$AG(t) = \frac{n-1}{n} \times AG(t-1) + \frac{1}{n} \times G_t, \quad (6)$$

and the n -period average loss ($AL(t)$) is calculated as

$$AL(t) = \frac{n-1}{n} \times AL(t-1) + \frac{1}{n} \times L_t. \quad (7)$$

Thus, the n -period RSI at time point t is calculated as

$$RSI_n(t) = \frac{AG(t)}{AG(t) + AL(t)} \times 100. \quad (8)$$

Traditionally, a RSI value higher than 70 indicates that the currency has been overbought, whereas a value below 30 indicates that the currency pair has been oversold. Thus, the RSI provides alarm signals for investors to close the current position or to open a new position to buy when the currency is oversold and to sell when it is overbought. The parameters used for the overbought and oversold levels can be set up by traders. In the present study, we use DE to optimize the RSI parameter.

2.2. SVM and MKL. A SVM is an optimal hyperplane used to separate two classes or a nonlinear separating surface optimized using a nonlinear mapping from the original input space into a high-dimensional feature space to search for an optimally separating hyperplane in the feature space. The latter solves classification problems that cannot be linearly separated in the input space. We designate a hyperplane as optimal if it has a maximal margin, where the margin is the minimal distance from the separating hyperplane to the closest data points, which are called the support vectors.

The concept used to map the data from the original feature space to a high-dimensional feature space is called a kernel method. Finding the optimal hyperplane is formalized as follows:

$$\begin{aligned} \min \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \zeta_i \\ \text{s.t.} \quad & y_i (\langle w \cdot x_i \rangle + b) \geq 1 - \zeta_i, \\ & \zeta_i \geq 0, \quad \forall i = 1, 2, \dots, n, \end{aligned} \quad (9)$$

where w is the vector of the parameters that define the optimal decision hyperplane $\langle w \cdot x_i \rangle + b = 0$ and b represents the bias. $(1/2)\|w\|^2$ is considered to be a regularization term, which controls the generalization capacities of the classifier. The second term $C \sum_{i=1}^n \zeta_i$ is the empirical risk (error). C is sometimes referred to as the soft margin parameter and it determines the tradeoff between the empirical risk and the regularization term. Increasing the value of C gives greater importance to empirical risk relative to the regularization term. Positive slack variables ζ_i allow classification errors.

To extend SVM, MKL uses multiple kernels to map the input space to a higher-dimensional feature space by combining different kernels to obtain a better separation function. In MKL, the kernels are combined linearly and the

weight of each kernel reflects its importance. The kernels can be different kernels or the same kernels with different parameters. Each kernel in the combination may account for a different feature or a different set of features. The use of multiple kernels can enhance the performance of the model.

Suppose k_m ($m = 1, \dots, M$) are M positive definite kernels on the same input space. Finding the optimal decision surface is formalized as follows:

$$\begin{aligned} \min_{w,b,\zeta} \quad & \frac{1}{2} \sum_{m=1}^M \frac{1}{d_m} \|F_m\|_{H_m}^2 + C \sum_{i=1}^N \zeta_i \sum_{i=1}^n X_i^2, \\ \text{s.t.} \quad & y_i \left(\sum_{m=1}^M \langle F_m, \Phi_m(x_i) \rangle + b \right) \geq 1 - \zeta_i, \\ & \zeta_i \geq 0, \quad \forall i = 1, 2, \dots, n, \\ & \sum_{m=1}^M d_m = 1, \quad d_m \geq 0, \end{aligned} \quad (10)$$

where Φ is a possibly nonlinear mapping from the input space to a feature space, F_m is the separation function, $\|\cdot\|$ is a norm, $\langle \cdot, \cdot \rangle$ is the inner product, C is used to control the generalization capacities of the classifier, which is selected by crossvalidation, and d_m are the optimized weights.

In our study, the optimized weights d_m directly represent the ranked relevance of each feature used in the prediction process. We employ MKL to learn the coefficients and parameter of the subkernels. We used the multiple kernel learning toolbox SHOGUN [21] in our experiments.

In our MKL based models, similarity is measured based on the instances of EUR/USD, instances of USD/JPY, and instances of GBP/USD. We construct three similarity matrices for each data source. These three derived similarity matrices are also taken as three subkernels of MKL and the weights of $d_{m,\text{EURUSD}}$, $d_{m,\text{GBPUSD}}$, and $d_{m,\text{USDJPY}}$ are learnt for the subkernels:

$$\begin{aligned} k(\vec{x}_i, \vec{x}_j) = & d_{m,\text{EURUSD}} k_{\text{EURUSD}}(\vec{x}_i^{(1)}, \vec{x}_j^{(1)}) \\ & + d_{m,\text{GBPUSD}} k_{\text{GBPUSD}}(\vec{x}_i^{(2)}, \vec{x}_j^{(2)}) \\ & + d_{m,\text{USDJPY}} k_{\text{USDJPY}}(\vec{x}_i^{(3)}, \vec{x}_j^{(3)}), \end{aligned} \quad (11)$$

where \vec{x}_i , $i = 1, 2, \dots, n$, are training samples, $d_{m,\text{EURUSD}}$, $d_{m,\text{GBPUSD}}$, and $d_{m,\text{USDJPY}} \geq 0$, and $d_{m,\text{EURUSD}} + d_{m,\text{GBPUSD}} + d_{m,\text{USDJPY}} = 1$. $x^{(1)}$ are EUR/USD instances, $x^{(2)}$ are GBP/USD instances, and $x^{(3)}$ are USD/JPY instances. In this study, k is the RBF (radial basis function) kernel for SVM and MKL. For other types of information sources or subkernel combinations, similar distance based similarity matrices and kernel functions can be constructed, which are easily imported into our multikernel based learning framework.

2.3. DE. The DE method proposed by Storn and Price [16] is a population based stochastic search approach, which can be used as an efficient global optimizer in a continuous search

domain. Like other evolutionary algorithms, DE also has a population with the size N_p and D -dimensional parameter vectors (D is the number of parameters present in an objective function). Two other parameters used in DE are the scaling factor F and the crossover rate C_r .

2.3.1. Population Structure. The current population, represented by P_x , comprises the vectors $x_i^{(G)}$, which have already been found to be acceptable, either as initial points or based on comparisons with other vectors, as follows:

$$\begin{aligned} P_x^{(G)} &= (x_i^{(G)}) \quad i = 0, 1, \dots, N_p - 1, \quad G = 0, 1, \dots, g_{\max}, \\ x_i^{(G)} &= (x_{i,j}^{(G)}) \quad j = 0, 1, \dots, D - 1. \end{aligned} \quad (12)$$

After initialization, DE mutates randomly selected vectors to produce an intermediary population $P_v^{(G)}$ of N_p mutant vectors $V_i^{(G)}$. Consider

$$\begin{aligned} P_v^{(G)} &= (V_i^{(G)}) \quad i = 0, 1, \dots, N_p - 1, \quad G = 0, 1, \dots, g_{\max}, \\ V_i^{(G)} &= (V_{i,j}^{(G)}) \quad j = 0, 1, \dots, D - 1. \end{aligned} \quad (13)$$

Each vector in the current population is recombined with a mutant to produce a trial population P_u of N_p trial vectors $u_i^{(G)}$. Consider

$$\begin{aligned} P_u^{(G)} &= (u_i^{(G)}) \quad i = 0, 1, \dots, N_p - 1, \quad G = 0, 1, \dots, g_{\max}, \\ u_i^{(G)} &= (u_{i,j}^{(G)}) \quad j = 0, 1, \dots, D - 1. \end{aligned} \quad (14)$$

2.3.2. Initialization. Before the population can be initialized, the upper and lower bounds of each parameter must be specified. They can be collected into two D -dimensional initialization vectors, x_U and x_L . After the initialization bounds have been specified, a random number generator assigns each element of every vector with a value from the prescribed range. For example, the initial value ($G = 0$) of the j th parameter of the i th vector is

$$\begin{aligned} P^{(0)} &= x_{i,j}^{(0)} = x_{j,L} + \text{rand}_j[0, 1] \cdot (x_{j,U} - x_{j,L}) \\ i &= 0, 1, \dots, N_p - 1; \quad j = 0, 1, \dots, D - 1, \end{aligned} \quad (15)$$

where $\text{rand}_j[0, 1]$ is a random number, which is generated uniformly between 0 and 1.

2.3.3. Mutation. After initialization, DE mutates and recombines the population to produce a population of N_p trial vectors. A mutant vector is produced according to the following formulation:

$$\begin{aligned} V_{i,j}^{(G)} &= x_{r1,j}^{(G-1)} + F \cdot (x_{r2,j}^{(G-1)} - x_{r3,j}^{(G-1)}) \\ i &= 0, 1, \dots, N_p - 1; \quad j = 0, 1, \dots, D - 1. \end{aligned} \quad (16)$$

The scale factor F is a positive real number, which controls the rate of population evolution. There is no upper limit to F , but effective values are seldom greater than 1. $r1$, $r2$, and $r3$ refer to three randomly selected indices from the population.

2.3.4. Crossover. DE also employs uniform crossover. Sometimes referred to as discrete recombination, crossover builds trial vectors from elements that have been copied from two different vectors. In particular, DE crosses each vector with a mutant vector:

$$u_{i,j}^{(G)} = \begin{cases} v_{i,j}^{(G)} & \text{if } (\text{rand}_{i,j}^{(G)} \leq C_r \text{ or } j = j_{\text{rand}}) \\ x_{i,j}^{(G-1)} & \text{otherwise,} \end{cases} \quad (17)$$

where the crossover probability $C_r \in [0, 1]$ is a user-defined value, which controls the fraction of elements that are copied from the mutant. To determine the source that contributes, a given uniform crossover compares C_r to a uniform random number $\text{rand}_{i,j}^{(G)}$ between 0 and 1. If the random number is less than or equal to C_r , the trial element is inherited from the mutant $V_{i,j}^{(G)}$; otherwise the element is copied from the vector $x_{i,j}^{(G-1)}$. In addition, the trial element with the randomly selected index j_{rand} is taken from the mutant to ensure that the trial vector does not duplicate $x_i^{(G)}$.

2.3.5. Selection. If the trial vector $u_i^{(G)}$ has an equal or lower objective function value than that of its target vector $x_i^{(G)}$, it replaces the target vector in the next generation; otherwise the target retains its place in the population for at least one more generation:

$$x_i^{(G+1)} = \begin{cases} u_i^{(G)} & \text{if } f(u_i^{(G)}) \leq f(x_i^{(G)}) \\ x_i^{(G)} & \text{otherwise.} \end{cases} \quad (18)$$

2.3.6. Stopping Criteria. After the new population is generated, the processes of mutation, recombination, and selection are repeated until the optimum is obtained, or a user-defined termination criterion, such as the number of generations, is reached at a preset maximum g_{max} .

2.4. Evaluation Measures. In the present study, we performed simulated trading using test samples based on the trading signals generated by MKL prediction and the multiple RSI signal, and we evaluated the return (gain or loss) obtained with the proposed model and other models. In general, a high return is inevitably accompanied by the potential for high risk. Therefore, investors desire a method that decreases risk while not decreasing the profits greatly, which results in a trade-off relationship. The Sharpe ratio, named after William Forsyth Sharpe, is a measure of the excess return per unit of risk in an investment asset or a trading strategy, which is defined as follows:

$$S = \frac{E[R - R_f]}{\sigma} = \frac{E[R - R_f]}{\sqrt{\text{var}[R - R_f]}}, \quad (19)$$

where R is the asset return, R_f is the return on a benchmark asset (usually a very low risk return such as a three-month US treasury bill), σ is the standard deviation of the asset return, and $E[R - R_f]$ is the expected value of the excess of the asset return relative to the benchmark asset return [32]. In our experiments, we used the Sharpe ratio as an evaluation measure to assess the return-risk ratio performance of our proposed method with other methods.

3. Proposed Method

3.1. Structure of the Proposed Method. Figure 2 shows the structure of the proposed method. First, the proposed method uses a MKL framework to predict directional changes in the currency rate based on the MACD of three currency pairs. The RSI signals are generated using multiple timeframe features of EUR/USD by considering the MKL trading signals. Finally, the MKL signal and RSI signal are combined to produce a final decision, that is, the trading signal.

The prediction and trading target currency pair in this study is EUR/USD. We selected it as our target due to the fact that the euro and US dollar are the two most traded currencies in the world, representing the world's two largest economies. Therefore, to better predict the changes in EUR/USD is considered to contribute much to the investors and international companies. In addition to EUR/USD data itself, since the two most traded currencies with USD and EUR in FX market are JPY and GBP, USD/JPY and GBP/USD are used for EUR/USD prediction. These three currency pairs share almost 50% of the FX market; other currencies such as AUD (Australian dollars), CAD (Canada dollars), and CHF (Swiss Franc) are also important currencies but since their shares in FX market are relatively small, we did not consider them in the structure of the proposed method.

The trading time interval is selected to be one hour in this study, which is also selected by Hirabayashi et al. [14]. To find overbought/oversold indicator values other than target 1-hour horizon data and to select some reasonable longer and shorter time horizons data are important. Since the trading time interval is one hour, 30-minute and 2-hour time horizon data are considered to be useful. Too high frequency time horizon data (such as minute data) or too low frequency time horizon data (such as daily data) are considered to have small impact if we fix the trading time interval to be one hour.

In this proposed method, we use MKL to predict directional changes and DE to find overbought/oversold information from RSI indicator. Although the predicted directional change can be used for simulated trading, in our preliminary experiments, the accumulated profits based on just the MKL predictions were not good enough (refer to Section 5.1); the same was true for accumulated profits based on using just DE and RSI indicator. Considering that the prediction and the technical indicators might have complementary components, we propose to combine them to get the trading signal. Therefore, we combine MKL and DE in the proposed method.

3.2. MKL Input and Output. For MKL, the input features are derived from three different sources: EUR/USD, GBP/USD,

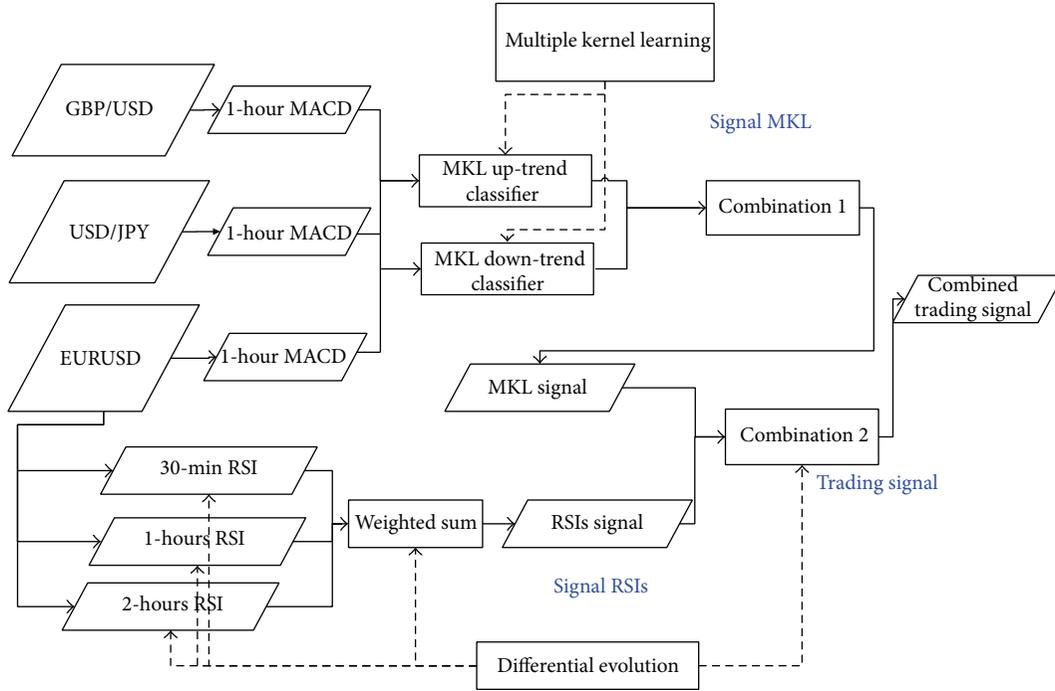


FIGURE 2: Structure of the proposed method.

TABLE 1: Features for each kernel.

No.	Feature
1	MACD-value at time t
2	MACD-signal at time t
3	MACD-value at time $(t - 1)$
4	MACD-signal at time $(t - 1)$
5	MACD-value at time $(t - 2)$
6	MACD-signal at time $(t - 2)$
7	MACD-value at time $(t - 3)$
8	MACD-signal at time $(t - 3)$
9	MACD-value at time $(t - 4)$
10	MACD-signal at time $(t - 4)$
11	MACD-value at time $(t - 5)$
12	MACD-signal at time $(t - 5)$
13	MACD-value at time $(t - 6)$
14	MACD-signal at time $(t - 6)$
15	MACD-value at time $(t - 7)$
16	MACD-signal at time $(t - 7)$

and USD/JPY. We transform the rates to MACD signals and values. For each kernel, the inputs are the MACD values and MACD signals for eight consecutive periods, which are shown in Table 1.

Using MKL, we construct two classifiers to output the MKL-up labels and the MKL-down labels (MKL-up refers to an upward trend classifier learned by MKL, while MKL-down refers to a downward trend classifier learned by MKL). We want to predict directional changes in a currency with an insensitive interval, where the changes from -0.1% to 0.1%

are not considered upward or downward. Thus, we set two threshold values, that is, 0.1% and -0.1% , which we refer to as the uptrend threshold value and the downtrend value, respectively, to label the training and testing samples. The rules for the MKL-up trend and MKL-down trend classifiers are shown in Table 2.

Based on the predictions of these two MKL classifiers, we obtain a combined MKL signal based on the rules, which are shown in Table 3. The combined MKL trading signal is one of the inputs for DE that needs to be combined with the multiple RSI signal.

3.3. *Combined Trading Signal Based on the Combined MKL and Multiple RSI Signals.* The multiple RSI signal value $Value_{RSIs}$ is the combined value of three timeframe RSI values:

$$Value_{RSIs} = \sum_{i=1}^3 w_i e_i, \quad (20)$$

where w_i are the weights of the three RSIs and e_i is the value of the RSI indicator. Note that the value of the RSI indicator is expressed as a ratio and we use $RSI/100$ from (8). The weights w_i of each RSI are learned by DE.

We compare the RSI values in (20) with the buy/sell threshold to determine the multiple RSI signal. The signal and the condition that need to be satisfied before the signal can be issued are shown in Table 4.

$Signal_{trading}$ is a signal used for making decisions based on both the combined MKL signal and the multiple RSI signal. Table 5 shows how the combined MKL and multiple RSI signal are combined to obtain the trading signal. If we decide to take a position (buy or sell), the position is retained

TABLE 2: Output labels for MKL up-trend and down-trend classifiers.

MKL classifier	MKL-trend signal	Conditions
MKL-up trend	MKL-up = +1	If the actual change rate is greater than the upward trend threshold value
	MKL-up = -1	If the actual change rate is less than the upward trend threshold value
MKL-down trend	MKL-down = +1	If the actual change rate is less than the downward trend threshold value
	MKL-down = -1	If the actual change rate is greater than the downward trend threshold value

TABLE 3: Conditions for issuing the MKL signal.

No	Combined MKL signal (Signal _{MKL})	Conditions
1	No trade	MKL-up = 1 and MKL-down = 1
2	No trade	MKL-up = -1 and MKL-down = -1
3	Buy	MKL-up = 1 and MKL-down = -1
4	Sell	MKL-up = -1 and MKL-down = 1

TABLE 4: Conditions that need to be satisfied before issuing the RSI signal.

No	Multiple RSI signal (Signal _{RSIs})	Conditions
1	Buy	Value _{RSIs} < buy threshold
2	Sell	Value _{RSIs} > sell threshold
3	No trade	otherwise

TABLE 5: Conditions that need to be satisfied before issuing the trading signal.

Trading signal (Signal _{trading})	Conditions	
	Combined MKL signal (Signal _{MKL})	Multiple RSI signal (Signal _{RSIs})
Buy	Buy	No trade
Sell	Sell	No trade
No trade	No trade	No trade
Sell	Any (buy, sell, or no trade)	Sell
Buy	Any (buy, sell, or no trade)	Buy

for 1 hour; that is, we check the conditions every hour. If the trading signal (buy or sell) is the same as that 1 hour before, we do not trade and we wait for 1 hour. The data we use are 1-hour EUR/USD (we used 30 min data to calculate the 30 min RSI value, and 1-hour data to calculate the 1-hour RSI value and the 2-hour RSI value).

3.4. DE Parameter Design. The DE parameter vectors shown in Table 6 are used to construct the multiple RSI signals. The representations of the parameter vectors are as follows.

- (1) The first three groups represent the parameters for each RSI (three RSIs in total). The values range from 3 to 10 (integer type).
- (2) Numbers 4 to 5 are used to decide the times to buy, sell, and close positions. The values range from 0 to 2 (floating point number type).

TABLE 6: DE parameter vector design.

No	Value	Description
1	3 to 10	parameter for 1-hour RSI
2	3 to 10	parameter for 2-hour RSI
3	3 to 10	parameter for 30-min RSI
4	0 to 2	buy threshold
5	0 to 2	sell threshold
6	0 to 1	weight value for 1-hour RSI
7	0 to 1	weight value for 2-hour RSI
8	0 to 1	weight value for 30-min RSI

- (3) Numbers 6 to 8 are the weights used to linearly combine signals, which are described in (20) in Section 3.3. The values range from 0 to 1 (floating point number type).

The population size is set to 200 and the maximum number of generations is set to 200 during the DE training step. The accumulated return obtained in the training step is selected as the objective function.

4. Experiment Design

The exchange rates used in this study were obtained from ICAP. The ICAP data was used in our previous study [13] for trading on EUR/USD. The ICAP data use the GMT +1 hour time zone (GMT +2 hour in summer) and they do cover the exchange rate in weekend. A list of best offers, best bids, and dealt prices for every second are comprised in the ICAP data. We transformed them into 30 min and 1-hour timeframes. We used exchange rate data for three currency pairs from ICAP data: EUR/USD, GBP/USD, and USD/JPY. We separate the overall data into three datasets and each dataset covered the period from January 3 to December 30 in each year, with a total of about 6200 observations (hourly data). The three datasets used for training and testing are shown in Table 7.

The data include the “open, high, low, and close” rates during each time interval (30 min and 1 hour). The data were divided into three disjoint datasets that covered consecutive periods, the details of which are shown in Table 8. Next, we divided each dataset into a training period and a testing period. The MKL training period covered 3000 observations (around 6 months) and the testing period covered 3000 observations (around 6 months). The MKL-DE training step covered 1500 trading hours and the MKL-DE testing step covered 1500 trading hours. Details of the length of each period are shown in Table 8.

TABLE 7: Three datasets used for training and testing.

Dataset	MKL training	MKL testing	MKL-DE training	MKL-DE testing
Dataset 1 (2008)	Jan. to Jun.	Jul. to Dec.	Jul. to Sep.	Oct. to Dec.
Dataset 2 (2009)	Jan. to Jun.	Jul. to Dec.	Jul. to Sep.	Oct. to Dec.
Dataset 3 (2010)	Jan. to Jun.	Jul. to Dec.	Jul. to Sep.	Oct. to Dec.

TABLE 8: Trading and testing periods for MKL and DE.

Period	Process	Length of period
1	MKL learning	3000 trading hours (around 6 months)
2	MKL testing (prediction)	3000 trading hours (around 6 months)
2-1	MKL-DE training	1500 trading hours (around 3 months)
2-2	MKL-DE testing (trading)	1500 trading hours (around 3 months)

Foreign exchange market is often and suddenly affected by economic events such as a bank rate decision or even unpredictable affair such as a big earthquake. Therefore, in a trading in the experiments, our initial investment is A US dollars. For each transaction (long or short), we fix the trading amount to be $A/2$ US dollars with a trading leverage ratio of 2 to 1. That is, although we did margin transaction, the trading in our experiments is conducted with very low leverage (or with a very high margin level), which ensures the safety of our transaction order even though there is a big shock in FX market.

Table 9 shows a list of the methods tested, including baseline methods, proposed methods, and intermediate methods. “Buy and hold” and “sell and hold” were selected as baseline methods because they are simple and well known, while they are the best methods for obtaining zero profit on average if the market is efficient and stationary. The trading rule they used was to buy or sell at the start of the testing period and to close the position at the end of the testing period. The other methods used for comparison comprising the simplest methods and our proposed methods. SVM-s used a kernelized linear model for exchange rates where the inputs were the exchange rates of only one currency pair with SVM as a learning method. SVM-m was the same as SVM-s but it utilized the features of three currency pairs. MKL-m was the same as SVM-m but the model was a multiple kernelized linear model that uses MKL. MKL-m-t and MKL-m-t-DE were the same as MKL-m but the prediction was changed to a three-classification problem from a two-classification problem. The trading rule used by SVM-s, SVM-m, and MKL-m was to buy a currency pair when the prediction was positive, to sell when negative, and “no trade” when the prediction was 0. The trading rule for MKL-m-t was based on $Signal_{MKL}$. The trading rule used by MKL-m-t-DE, our proposed method, was based on $Signal_{trading}$ where the parameters were optimized using MKL and DE (see Table 5). DE-only was based on $Signal_{RSI}$; that is, it relied only on multiple RSI signals. The DE algorithm includes random numbers, so we conducted 10 experiments with different seeds for MKL-m-t-DE and DE-only. In the list of methods tested, since GA based method are well-known methods in

the previous literatures [12–14], GA-s and GA-m which are implemented by Deng and Sakurai [13] are considered as benchmark methods, and we conducted 10 experiments with different seeds for GA-s and GA-m. “Buy and hold” and “sell and hold” are well-known baseline methods which are also used as baseline methods by Chong and Ng [9]; SVM-s, SVM-m, MKL-m, MKL-m-t, DE-only, and MKL-m-t-DE are implemented by us.

5. Experimental Results and Discussion

5.1. *Returns with the Three Datasets.* Table 10 shows the returns with the methods tested, where the returns were measured in proportion to the initial investment (the entries in the first three columns for MKL-m-t-DE, DE-only, GA-s, and GA-m are the average returns from 10 independent experiments with their standard deviations). First, we found that during the testing period (three months) for each dataset, our proposed method yielded good average returns (about 6.73%, 4.71%, and 3.52%). In addition, our proposed method obtained the best average return (4.98%) among all the methods tested.

Next, we focused on the baseline methods: “buy and hold” and “sell and hold.” We found that “buy and hold” yielded losses with all three testing datasets while “sell and hold” yielded better returns than the other methods except MKL-m-t-DE during the three testing periods. The “sell and hold” strategy yielded profits during the testing periods because EUR had declined against USD due to the European sovereign debt crisis [33], which occurred in the Eurozone after a big rise in EUR against USD from 2005 until the first half of 2008. We could not forecast the decline or surge before this period, so we could not decide whether “buy and hold” was better than “sell and hold” and we could not conclude that these two naive strategies performed well.

In addition, we compared the results with SVM-s and SVM-m. Table 10 shows that these SVM based methods yielded losses during all three testing periods. SVM-m used more information (the features of three FX pairs) than SVM-s (the features of EUR/USD only) in dataset 2 (2009), but the

TABLE 9: List of the methods tested.

Method	Description
GA-s	Trade based on the trading rules optimized by GA, with one RSI input
GA-m	Trade based on the trading rules optimization by GA, with three RSI input
Buy and hold	Buy and hold until the end point of a period
Sell and hold	Sell and hold until the end point of a period
SVM-s	Trade based on SVM prediction, with one FX pair input
SVM-m	Trade based on SVM prediction, with three FX pairs input
MKL-m	Trade based on MKL prediction, with three FX pairs input
MKL-m-t	Trade based on $\text{Signal}_{\text{MKL}}$
DE-only	Trade based on $\text{Signal}_{\text{RSIs}}$ (parameters are optimized by DE)
MKL-m-t-DE	Trade based on $\text{Signal}_{\text{trading}}$

TABLE 10: Returns with the methods tested (The numbers right to \pm is the standard deviation).

Method	Dataset 1 (2008)	Dataset 2 (2009)	Dataset 3 (2010)	Average returns
GA-s	0.0068 \pm 0.0230	-0.0454 \pm 0.0143	-0.0284 \pm 0.0569	-0.0223
GA-m	0.0098 \pm 0.0991	-0.0326 \pm 0.0286	0.0087 \pm 0.0241	-0.0046
Buy and hold	-0.0510	-0.0426	-0.0229	-0.0388
Sell and hold	0.0510	0.0426	0.0229	0.0388
SVM-s	-0.2039	-0.0225	-0.0559	-0.0941
SVM-m	-0.0397	-0.0324	-0.0299	-0.0340
MKL-m	-0.1932	-0.0103	0.0479	-0.0518
MKL-m-t	0.0216	0.0150	0.0048	0.0138
DE-only	0.0035 \pm 0.0991	-0.0318 \pm 0.0541	0.0082 \pm 0.0131	-0.0201
MKL-m-t-DE	0.0673 \pm 0.0343	0.0471 \pm 0.0362	0.0352 \pm 0.0215	0.0498

return with SVM-m (-3.2%) was not better than that with SVM-s (-2.2%).

Moreover, we compared the results of proposed method with that of GA-s and GA-m. Table 10 shows that GA-s yielded positive return on average during 2008, while yielded losses on average during 2009 and 2010. GA-m yielded positive return in 2008 and 2010, but it yielded losses on average during 2009 and the average return of three data sets is about -0.004, which is much worse than the results of our proposed method. In addition, the average return results of GA-m for the three data sets are better than those of GA-s, which agrees with the conclusion in Deng and Sakurai [13] that the return results improved when using information of RSI indicator from multiple timeframes.

Based on the average returns, we found that MKL-m-t performed better than MKL-m, which indicated that the returns were improved by neglecting small predicted changes such as fluctuations in the MKL-m method. DE-only used DE alone to generate the trading rules based on multiple RSI values, but it yielded losses on average. MKL-m-t-DE performed the best of the four methods (MKL-m, MKL-m-t, MKL-m-t-DE, and DE-only), which indicates that the integration of multiple RSI signals could improve the trading performance.

5.2. Sharpe Ratios. In addition to the returns, the Sharpe ratio was used to evaluate the performance of our proposed method and other methods. We used the one-year treasury

rate as the risk-free asset to calculate the Sharpe ratio. The one-year treasury rate ranged from 1.7% to 4.3% between 2008 and 2010. Next, we calculated the average risk-free returns from 2008 to 2010 and the average risk-free return for each testing period (three months in each year) was about 0.75%. Table 11 shows the average returns, standard deviations, and Sharpe ratios with each method (for the methods "MKL-m-t-DE" and "DE-only," "average return" results are the averages of all the returns obtained from 10 experiments for all the testing periods with all the datasets, while the "standard deviation" is the standard deviation of these returns).

A higher Sharpe ratio indicates a higher return or lower volatility. From Table 11, we found that for the methods "GA-s," "GA-m," "buy and hold," "SVM-s," "SVM-m," "MKL-m," and "DE-only," their Sharpe ratio values are negative, which indicates that their average return is less than the free-risk asset. There are three methods that obtained positive Sharpe ratio value: "sell and hold," "MKL-m-t," and our proposed method "MKL-m-t-DE." It is clear that our proposed method had a significantly higher Sharpe ratio (2.6111) than the other two methods during the testing periods. The Sharpe ratio results indicate that the proposed method is the best method when evaluated by return-risk ratio.

6. Conclusion and Future Work

In this study, we developed a hybrid method based on MKL and DE for EUR/USD trading. In the first step of our

TABLE 11: Sharpe ratios for the baseline, benchmark, and proposed methods.

Method	Average return	Standard deviation	Sharpe ratio
GA-s	-0.0223	0.0242	-0.5025
GA-m	-0.0046	0.0266	-1.1177
Buy and Hold	-0.0388	0.0144	-3.2152
Sell and Hold	0.0388	0.0144	2.1736
SVM-s	-0.0941	0.0965	-1.0528
SVM-m	-0.0340	0.0050	-8.3000
MKL-m	-0.0518	0.1258	-0.4713
MKL-m-t	0.0138	0.0084	0.7500
DE-only	-0.0201	0.0219	-1.2602
MKL-m-t-DE	0.0498	0.0162	2.6111

approach, we used MKL to predict the directional change in the currency rate (with an insensitive interval) to provide a combined MKL signal. In the second step, DE combined the combined MKL signal with the multiple RSI signal to generate a trading signal. The experimental results showed that MKL-m-t yielded profits with the three testing datasets (about 1.38% on average), while integration of the multiple RSI signal improved the trading profits (about 4.98% on average). In addition, the proposed method yielded the best Sharpe ratio (about 2.61) compared with all the models tested, which indicates that our proposed method outperformed other methods in terms of the return-risk ratio, as well as the returns.

However, there are still some unaddressed questions and some research directions for future work. For example, how to find the best insensitive interval (-0.1% to 0.1% in this study) is still an open question in this study: a too large insensitive interval could decrease the number trading times too much so that the trading profit also decreases, while a too small insensitive interval cannot filter the unknown movements well the trading profit decreases. For future work, one may combine MKL with GA to use GA to search the best parameters for insensitive interval in MKL automatically in order to solve the unaddressed problems. In addition, other than RSI, some other famous overbought/oversold indicators, such as BIAS and William %R, could be also implemented to improve the trading ability.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research was partially supported by the "Graduate School Doctoral Student Grant-in-Aid Program 2012" of Keio University, Japan. In addition, the authors wish to thank ICAP for making the data available for this research.

References

- [1] Online material 1, "Moving average," http://en.wikipedia.org/wiki/Moving_average.
- [2] Online material 2, "MACD, Wikipedia," <http://en.wikipedia.org/wiki/MACD>.
- [3] Online material 3, "RSI," Wikipedia, http://en.wikipedia.org/wiki/Relative_Strength_Index.
- [4] Online material 5, "BIAS ratio," Wikipedia, http://en.wikipedia.org/wiki/Bias_ratio_%28finance%29.
- [5] Online material 6, "Bollinger Bands," Wikipedia, http://en.wikipedia.org/wiki/Bollinger_Bands.
- [6] M. Jaruszewicz and J. Mańdziuk, "One day prediction of NIKKEI index considering information from other stock markets," in *Proceedings of the 7th International Conference on Artificial Intelligence and Soft Computing (ICAISC '04)*, pp. 1130–1135, Springer, Berlin, Germany, June 2004.
- [7] S. Deng, K. Yoshiyama, T. Mitsubuchi, and A. Sakurai, "Hybrid method of multiple kernel learning and genetic algorithm for forecasting short-term foreign exchange rates," *Computational Economics*, pp. 1–41, 2013.
- [8] L. Y. Wei, T. L. Chen, and T. H. Ho, "A hybrid model based on adaptive-network-based fuzzy inference system to forecast Taiwan stock market," *Expert Systems with Applications*, vol. 38, no. 11, pp. 13625–13631, 2011.
- [9] T. T.-L. Chong and W.-K. Ng, "Technical analysis and the London stock exchange: testing the MACD and RSI rules using the FT30," *Applied Economics Letters*, vol. 15, no. 14, pp. 1111–1114, 2008.
- [10] J. Kamruzzaman, R. A. Sarker, and I. Ahmad, "SVM based models for predicting foreign currency exchange rates," in *Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM '03)*, pp. 557–560, Melbourne, Fla, USA, November 2003.
- [11] K. Shioda, S. Deng, and A. Sakurai, "Prediction of foreign exchange market states with support vector machine," in *Proceedings of the 10th International Conference on Machine Learning and Applications (ICMLA '11)*, vol. 1, pp. 327–332, Honolulu, Hawaii, USA, December 2011.
- [12] Y. Chang Chien and Y. Chen, "Mining associative classification rules with stock trading data-A GA-based method," *Knowledge-Based Systems*, vol. 23, no. 6, pp. 605–614, 2010.

- [13] S. Deng and A. Sakurai, "Foreign exchange trading rules using a single technical indicator from multiple timeframes," in *Proceedings of the 27th International Conference on Advanced Information Networking and Applications Workshops (WAINA '13)*, pp. 207–212, IEEE, Barcelona, Spain, March 2013.
- [14] A. Hirabayashi, C. Aranha, and H. Iba, "Optimization of the trading rule in foreign exchange using genetic algorithm," in *Proceedings of the 11th Annual Genetic and Evolutionary Computation Conference (GECCO '09)*, pp. 1529–1536, Montreal, Canada, July 2009.
- [15] A. Esfahanipour and S. Mousavi, "A genetic programming model to generate risk-adjusted technical trading rules in stock markets," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8438–8445, 2011.
- [16] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
- [17] C. Worasuchee, "A new self adaptive differential evolution: its application in forecasting the index of stock exchange of Thailand," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 1918–1925, Singapore, September 2007.
- [18] T. Takahama, S. Sakai, A. Hara, and N. Iwane, "Predicting stock price using neural networks optimized by differential evolution with degeneration," *International Journal of Innovative Computing, Information and Control*, vol. 5, no. 12, pp. 5021–5031, 2009.
- [19] J. Peralta, X. Li, G. Gutierrez, and A. Sanchis, "Time series forecasting by evolving artificial neural networks using genetic algorithms and differential evolution," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN '10)*, pp. 1–8, IEEE, 2010.
- [20] F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan, "Multiple kernel learning, conic duality, and the SMO algorithm," in *Proceedings of the 21st International Conference on Machine Learning (ICML '04)*, pp. 41–48, ACM, Alberta, Canada, July 2004.
- [21] S. Sonnenburg, G. Rätsch, S. Henschel et al., "The SHOGUN machine learning toolbox," *The Journal of Machine Learning Research*, vol. 11, pp. 1799–1802, 2010.
- [22] T. Joutou and K. Yanai, "A food image recognition system with multiple kernel learning," in *Proceedings of the 16th IEEE International Conference on Image Processing (ICIP '09)*, pp. 285–288, IEEE, November 2009.
- [23] L. Foresti, D. Tuia, A. Pozdnoukhov, and M. Kanevski, "Multiple kernel learning of environmental data. Case study: analysis and mapping of wind fields," in *Artificial Neural Networks—ICANN 2009*, vol. 5769 of *Lecture Notes in Computer Science*, pp. 933–943, 2009.
- [24] S. Deng, T. Mitsubuchi, and A. Sakurai, "Stock price change rate prediction by utilizing social network activities," *The Scientific World Journal*, vol. 2014, Article ID 861641, 14 pages, 2014.
- [25] S. Deng and A. Sakurai, "Crude oil spot price forecasting based on multiple crude oil markets and timeframes," *Energies*, vol. 7, no. 5, pp. 2761–2779, 2014.
- [26] T. Fletcher, Z. Hussain, and J. Shawe-Taylor, "Multiple kernel learning on the limit order book," *Journal of Machine Learning Research-Proceedings Track*, vol. 11, pp. 167–174, 2010.
- [27] R. Luss and A. D'Aspremont, "Predicting abnormal returns from news using text classification," *Quantitative Finance*, pp. 1–14, 2012.
- [28] C. Y. Yeh, C. W. Huang, and S. J. Lee, "A multiple-kernel support vector regression approach for stock market price forecasting," *Expert Systems with Applications*, vol. 38, no. 3, pp. 2177–2186, 2011.
- [29] S. C. Huang and T. K. Wu, "Integrating GA-based time-scale feature extractions with SVMs for stock index forecasting," *Expert Systems with Applications*, vol. 35, no. 4, pp. 2080–2088, 2008.
- [30] C.-F. Huang, "A hybrid stock selection model using genetic algorithms and support vector regression," *Applied Soft Computing*, vol. 12, no. 2, pp. 807–818, 2012.
- [31] M. D. Beneish, C. M. Lee, and R. L. Tarpley, "Contextual fundamental analysis through the prediction of extreme returns," *Review of Accounting Studies*, vol. 6, no. 2-3, pp. 165–189, 2001.
- [32] W. F. Sharpe, *The Sharpe Ratio. Streetwise-The Best of the Journal of Portfolio Management*, University Press Princeton, Princeton, NJ, USA, 1998.
- [33] Online material 4, "European sovereign debt," Wikipedia, http://en.wikipedia.org/wiki/European_sovereign-debt_crisis.

Research Article

A Distributed Parallel Genetic Algorithm of Placement Strategy for Virtual Machines Deployment on Cloud Platform

Yu-Shuang Dong,¹ Gao-Chao Xu,^{1,2} and Xiao-Dong Fu¹

¹ College of Computer Science and Technology, Jilin University, Changchun 130012, China

² Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, China

Correspondence should be addressed to Yu-Shuang Dong; yushuangdong@gmail.com

Received 10 March 2014; Revised 17 June 2014; Accepted 17 June 2014; Published 3 July 2014

Academic Editor: Su Fong Chien

Copyright © 2014 Yu-Shuang Dong et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The cloud platform provides various services to users. More and more cloud centers provide infrastructure as the main way of operating. To improve the utilization rate of the cloud center and to decrease the operating cost, the cloud center provides services according to requirements of users by sharding the resources with virtualization. Considering both QoS for users and cost saving for cloud computing providers, we try to maximize performance and minimize energy cost as well. In this paper, we propose a distributed parallel genetic algorithm (DPGA) of placement strategy for virtual machines deployment on cloud platform. It executes the genetic algorithm parallelly and distributedly on several selected physical hosts in the first stage. Then it continues to execute the genetic algorithm of the second stage with solutions obtained from the first stage as the initial population. The solution calculated by the genetic algorithm of the second stage is the optimal one of the proposed approach. The experimental results show that the proposed placement strategy of VM deployment can ensure QoS for users and it is more effective and more energy efficient than other placement strategies on the cloud platform.

1. Introduction

Cloud computing is at the forefront of information technology. The internal system of cloud computing can be seen as a collection of a set of services [1], including infrastructure layer (IaaS), platform layer (PaaS), and application layer (SaaS). With the development of cloud computing, more and more cloud centers provide IaaS as the main way of operating. In order to improve the utilization rate of cloud center and to decrease the operating costs, virtualization technology has been applied to the cloud computing [2–4]. It provides services as required to users by sharding the resources with virtualization. But the distribution of virtual machines (VMs) will become sparser on cloud center with creating and closing the VMs. The placement problem of VMs has attracted more and more attention and became a research hotspot in cloud computing area quickly. It can be regarded as packing problem and has been proved as a NP-completeness problem [5].

Most of early researches were focused on increasing resources utilization rate in considering the system performance. With the increase of cloud center scale, energy saving has attracted significant attention in both industry and academia area. In order to reduce operating costs by saving energy, the concept of green cloud is proposed. Most researches are focused on VMs consolidation with living migration technology to reduce energy costs. If we take the energy costs into consideration as a parameter in the VMs deployment process, it can effectively reduce live migration frequency for decreasing the energy costs in maintenance of cloud center.

Genetic algorithm has been appreciated by academic circles as a solution of the VMs placement problem because of its speediness and adaptability advantages. Furthermore, parallel genetic algorithm can be used to solve the relatively complex problems. Even so, the genetic algorithm probably terminates before it gets a good enough solution in the case that there are a large number of servers in cloud platform and

users need to deploy a certain number of VMs. The traditional parallel genetic algorithm is executed on single physical host, but Amdahl's law [6] showed that the performance of parallel program executed on single physical host is not much better than serial program. The researches [7, 8] showed that we can get a better performance of parallel program by enlarging the scale of problem. Therefore, we propose a new distributed parallel genetic algorithm (DPGA) of placement strategy which is executed on several physical hosts for the large-scale VMs deployment problem. This algorithm can get a better and more accurate solution by increasing the iterative times. Comparing with the deployment process, the time cost of deployment strategy is relatively less. Therefore, we did not take the time cost in consideration in DPGA. We define the initial population of the DPGA as initial total population and the initial population of algorithm executing on each selected host as initial subpopulation. We assign the performance per watt as fitness value. In order to ensure the coverage of the solution space, we choose initial subpopulation from solution space dispersedly and averagely. It executes the first stage genetic algorithm on several selected physical hosts to choose initial subpopulation and get several solutions. Then it collects the solutions calculated by the first stage of DPGA and puts them into the second stage as initial population. Finally, we get a relatively satisfied solution from the second stage of DPGA.

2. Relevant Work

The proposed question refers to finding the target hosts to place the VMs. In this paper, relevant knowledge of DVFS will be used in the standard for evaluating the solution in considering of minimizing energy costs and ensuring performance as well. This subject has not been widely studied in the field related to placement strategy for VMs deployment. However, many researches are focused on the placement of applications and services in the cloud environment [9–15], and many researchers have been working on data placement in the cloud center [16–19].

There are also some researches focused on the similar problems. von Laszewski et al. have presented a scheduling algorithm to allocate virtual machines in a DVFS-enabled cluster [20]. The proposed algorithm was focused on scheduling virtual machines in a compute cluster to reduce power consumption via the technique of DVFS (dynamic voltage and frequency scaling). It dynamically adjusts the CPU frequencies and voltages of the compute nodes in a cluster without degrading the virtual machine performance beyond unacceptable levels. Recent studies have revealed that the network elements consume 10–20% of the total power in the data center. VMPlanner [21] optimized both virtual machine placement and traffic flow routing so as to turn off as many unneeded network elements as possible for network power reduction in the virtualization-based data centers. It took the advantage of the flexibility provided by dynamic VM migration and programmable flow-based routing to optimize network power consumption while satisfying network traffic demands. Ge et al. have presented distributed

performance-directed DVS (dynamic voltage scaling) scheduling strategies for use in scalable power-aware HPC (high-performance computing) clusters [22]. It uses DVS technology in high-performance microprocessors to reduce power consumption during parallel application runs in the case that peak CPU performance is not necessary due to load imbalance, communication delays, and so forth. VMPACS [23] is a multiobjective ant colony system algorithm for the virtual machine placement problem. The purpose of VMPACS is to efficiently obtain a nondominated Pareto set that simultaneously minimizes total resource wastage and power consumption. MILP [24] proposed a holistic approach for a large-scale cloud system where the cloud services are provisioned by several data centers interconnected over the backbone network. It is a mixed integer linear programming formulation that aims at virtualizing the backbone topology and placing the VMs in inter- and intradata centers with the objective of jointly optimized network delay and energy saving. OVMP [25] is an optimal virtual machine placement algorithm to provision the resources offered by multiple cloud providers. It is based on an IaaS model which leverages virtualization technologies that can minimize the total cost of resource in each plan for hosting virtual machines in a multiple cloud provider environment under future demand and price uncertainty. The tradeoff between the advance reservation of resources and the allocation of on-demand resources is adjusted to be optimal. It makes a decision based on the optimal solution of stochastic integer programming (SIP) to rent resources from cloud providers. Jing Tai Piao proposed a network-aware virtual machine placement and migration approach for data intensive applications in cloud computing environments to minimize the data transfer time consumption [26]. It places the VMs on physical machines with consideration of the network conditions between the physical machines and the data storage. It also considers the scenario in which instable network condition changed the data access behaviors and deteriorated the application performance. It migrates the VM to other physical machines in order to deal with this scenario.

3. Distributed Parallel Genetic Algorithm of VMs Placement

There are w physical hosts in the cloud platform and users need n VMs with $(h_0, h_1, h_2, \dots, h_{n-1})$ Hz CPU and $(m_0, m_1, m_2, \dots, m_{n-1})$ M RAM. We assume that the physical hosts in cloud center are DVFS [27] enabled and the cloud center can satisfy requirements of users; namely, w is big enough for n . We assume that $w \gg n$ and the physical hosts are in the same network environment. Solution space is recorded as follows: $P = (P_0, P_1, P_2, \dots, P_{w-1})$. We need to find n physical hosts to satisfy the requirements of users to place the VMs. The solution vector is as follows: $S = (S_0, S_1, S_2, \dots, S_{n-1})$. Remaining available CPU resource of physical host P_i is as follows: $AF_i = (1 - U_i) \times F_i$. U_i is the CPU utilization rate of P_i . The parameter F_i is the CPU frequency of P_i . Remaining available memory resource of physical host P_i is as follows: $AM_i = TM_i - UM_i - RM$. TM_i is the total

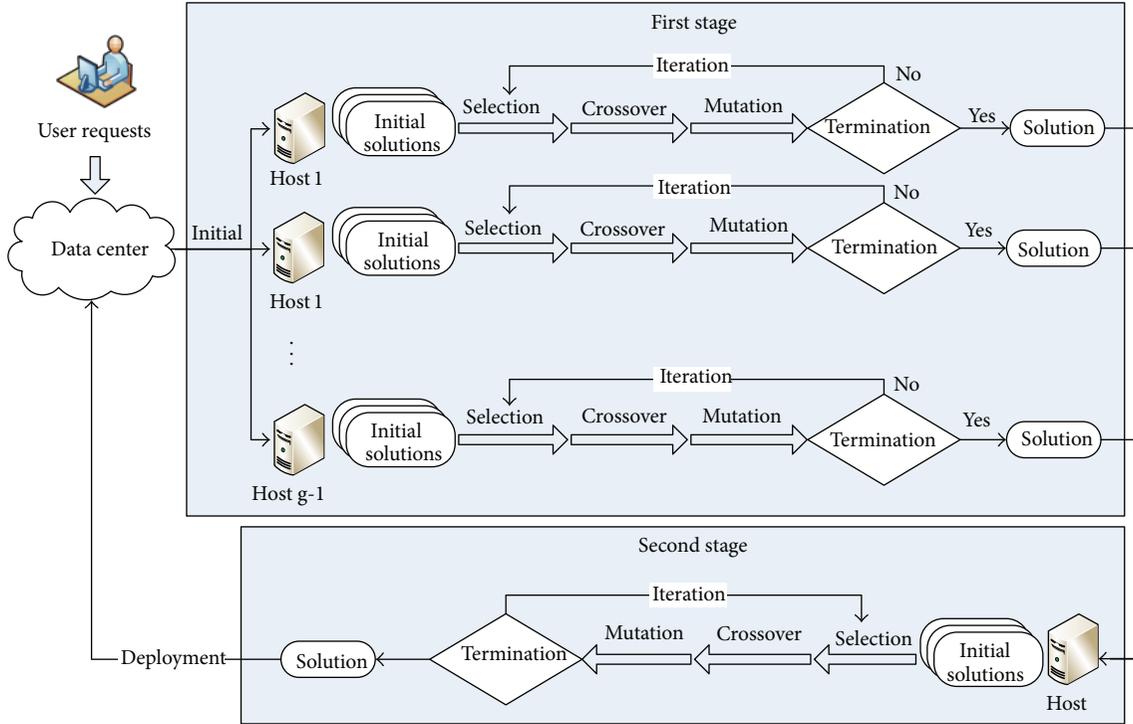


FIGURE 1: Distributed parallel genetic algorithm of VMs placement.

memory size of P_i . UM_i is the used memory size of P_i . RM is the reserved memory size of the system. P_i can be a member of solutions only if $AF_i > h$ and $AM_i > m$.

From the view of users, cloud center should select the physical hosts with more remaining resources to load the VMs with the objective of improving the QoS. From the view of cloud operators, cloud center should improve the utilization rates of resources and decrease the energy costs that aim at reducing the operating costs. Taken together, we assign the performance per watt to evaluation standard, namely, maximizing performance as well as minimizing energy costs. As shown in Figure 1, the idea of DPGA is divided into two stages. In the first stage, genetic algorithm is executed in parallel on g selected physical hosts. We select initial populations dispersedly and averagely by a certain step size in solution space for these physical hosts. Selection process chooses the solution vectors according to the probability which is proportional to the fitness value. Then the algorithm crosses the selected solution vectors and mutates the crossed solution vectors in the direction conducive to the fitness value. After crossover and mutation process, the algorithm iterates the first stage until it meets the iterative terminal conditions. In the second stage, the algorithm collects the solutions obtained from each selected physical host in the first stage, and then it executes the genetic algorithm again as in the first stage with collected solutions as initial population.

3.1. Initial Population Generation in the First Stage. Instead of random way, we assign the initial population for higher coverage rate in the initial population generating process.

Initial vector set w/n as jump volume among the vector members. We select g initial vectors as initial solution vectors and set $w/n/g$ as jump volume among the initial solutions. We set $w/n/g/g$ as jump volume among the initial solutions of the selected physical hosts. To ensure the algorithm executing correctly, in this paper, we assume that $w/n/g/g > 1$. In physical host x ($0 \leq x < g$), the vector member Q_{xyz} ($0 \leq z < n$) of initial solution vector S_{xy} ($0 \leq y < g$) is as follows:

$$Q_{xyz} = \begin{cases} P_{x \times w/n/g/g + y \times w/n/g + z \times w/n} \\ x \times w/n/g/g + y \times w/n/g + z \times w/n \leq w, \\ P_{x \times w/n/g/g + y \times w/n/g + z \times w/n - w} \\ x \times w/n/g/g + y \times w/n/g + z \times w/n > w. \end{cases} \quad (1)$$

Number x is the serial number of the physical host which is selected to execute the first stage algorithm. Number y is the solution vector serial number of the physical host x . Number z is the vector member serial number of the solution vector y .

For instance, we set $w = 1000$, $n = 10$, and $g = 4$. The initial population is showed in Table 1.

3.2. Fitness Calculation. We assign the performance per watt as fitness value. The performance increment of physical host Q_{xyz} is recorded as $\Delta F_{xyz} \times T$. ΔF_{xyz} is the CPU frequency increment of physical host Q_{xyz} and T is the VM work time. The energy consumption increment of physical host Q_{xyz} is recorded as ΔE_{xyz} . The VCPU frequencies of placement VMs are $(h_0, h_1, h_2, \dots, h_{n-1})$ Hz, so $\Delta F_{xy0} = h_0$, $\Delta F_{xy1} = h_1, \dots, \Delta F_{xy(n-1)} = h_{n-1}$. The relationship among

TABLE 1: An example of initial population in the first stage.

Physical host (x)	Initial solution vector (y)	Member of initial solution vector
0	0	$(P_0, P_{100}, P_{200}, P_{300}, P_{400}, P_{500}, P_{600}, P_{700}, P_{800}, P_{900})$
	1	$(P_{25}, P_{125}, P_{225}, P_{325}, P_{425}, P_{525}, P_{625}, P_{725}, P_{825}, P_{925})$
	2	$(P_{50}, P_{150}, P_{250}, P_{350}, P_{450}, P_{550}, P_{650}, P_{750}, P_{850}, P_{950})$
	3	$(P_{75}, P_{175}, P_{275}, P_{375}, P_{475}, P_{575}, P_{675}, P_{775}, P_{875}, P_{975})$
1	0	$(P_6, P_{106}, P_{206}, P_{306}, P_{406}, P_{506}, P_{606}, P_{706}, P_{806}, P_{906})$
	1	$(P_{31}, P_{131}, P_{231}, P_{331}, P_{431}, P_{531}, P_{631}, P_{731}, P_{831}, P_{931})$
	2	$(P_{56}, P_{156}, P_{256}, P_{356}, P_{456}, P_{556}, P_{656}, P_{756}, P_{856}, P_{956})$
	3	$(P_{81}, P_{181}, P_{281}, P_{381}, P_{481}, P_{581}, P_{681}, P_{781}, P_{881}, P_{981})$
2	0	$(P_{12}, P_{112}, P_{212}, P_{312}, P_{412}, P_{512}, P_{612}, P_{712}, P_{812}, P_{912})$
	1	$(P_{37}, P_{137}, P_{237}, P_{337}, P_{437}, P_{537}, P_{637}, P_{737}, P_{837}, P_{937})$
	2	$(P_{62}, P_{162}, P_{262}, P_{362}, P_{462}, P_{562}, P_{662}, P_{762}, P_{862}, P_{962})$
	3	$(P_{87}, P_{187}, P_{287}, P_{387}, P_{487}, P_{587}, P_{687}, P_{787}, P_{887}, P_{987})$
3	0	$(P_{18}, P_{118}, P_{218}, P_{318}, P_{418}, P_{518}, P_{618}, P_{718}, P_{818}, P_{918})$
	1	$(P_{43}, P_{143}, P_{243}, P_{343}, P_{443}, P_{543}, P_{643}, P_{743}, P_{843}, P_{943})$
	2	$(P_{68}, P_{168}, P_{268}, P_{368}, P_{468}, P_{568}, P_{668}, P_{768}, P_{868}, P_{968})$
	3	$(P_{93}, P_{193}, P_{293}, P_{393}, P_{493}, P_{593}, P_{693}, P_{793}, P_{893}, P_{993})$

energy, voltage, and frequency in CMOS circuits [27] is related by

$$E = C \times F \times V^2 \times T, \quad F = K \times \frac{(V - Vt)^2}{V}, \quad (2)$$

where E is energy consumption, C is CPU circuit switching capacity, F is CPU frequency, V is CPU voltage, K is a factor which depends on technology, and Vt is CPU threshold voltage. By formula (2), we can get the relationship between voltage and frequency as follows:

$$V = \sqrt{\frac{F \times Vt}{K} + \frac{F^2}{4 \times K^2}} + Vt + \frac{F}{2 \times K}. \quad (3)$$

We can also get the energy consumption increment of physical host Q_{xyz} as follows:

$$\begin{aligned} \Delta E_{xyz} &= C_{xyz} \times (F_{xyz} + h_z) \\ &\times \left(\sqrt{\frac{(F_{xyz} + h_z) \times Vt_{xyz}}{K_{xyz}} + \frac{(F_{xyz} + h_z)^2}{4 \times K_{xyz}^2}} \right. \\ &\quad \left. + Vt_{xyz} + \frac{F_{xyz} + h_z}{2 \times K_{xyz}} \right)^2 \times T \\ &- C_{xyz} \times F_{xyz} \times \left(\sqrt{\frac{F_{xyz} \times Vt_{xyz}}{K_{xyz}} + \frac{F_{xyz}^2}{4 \times K_{xyz}^2}} \right. \\ &\quad \left. + Vt_{xyz} + \frac{F_{xyz}}{2 \times K_{xyz}} \right)^2 \times T. \end{aligned} \quad (4)$$

It updates the $F_{xyz} = F_{xyz} + h_z$ dynamically and temporarily after ΔE_{xyz} calculation. The updated F_{xyz} only works in the process of the fitness value calculation for current solution vector. The fitness value of the algorithm is the ratio of the incremental performance and incremental power consumption after deploying the VMs according to the solution vector. Thus, the fitness value I_{xy} of the solution vector S_{xy} in the proposed VM placement strategy can be expressed as follows:

$$\begin{aligned} I_{xy} &= \frac{\sum_{z=0}^{n-1} \Delta F_{xyz} \times T}{\sum_{z=0}^{n-1} \Delta E_{xyz}} \\ &= \sum_{z=0}^{n-1} h_z \\ &\times \left(\sum_{z=0}^{n-1} C_{xyz} \times (F_{xyz} + h_z) \right. \\ &\quad \times \left(\sqrt{\frac{(F_{xyz} + h_z) \times Vt_{xyz}}{K_{xyz}} + \frac{(F_{xyz} + h_z)^2}{4 \times K_{xyz}^2}} \right. \\ &\quad \left. \left. + Vt_{xyz} + \frac{F_{xyz} + h_z}{2 \times K_{xyz}} \right)^2 \right. \\ &\quad \left. - C_{xyz} \times F_{xyz} \right. \\ &\quad \times \left(\sqrt{\frac{F_{xyz} \times Vt_{xyz}}{K_{xyz}} + \frac{F_{xyz}^2}{4 \times K_{xyz}^2}} \right. \\ &\quad \left. \left. + Vt_{xyz} + \frac{F_{xyz}}{2 \times K_{xyz}} \right)^2 \right)^{-1}. \end{aligned} \quad (5)$$

3.3. *Selection, Crossover, and Mutation in the First Stage.* Selecting operations choose the solution vectors according to the probability in the direction proportional to the fitness value. The selected solution vectors with higher fitness value will get more opportunities to be inherited by succeeding generation. The selection probability β_{xy} of solution vector S_{xy} is as follows:

$$\beta_{xy} = \frac{I_{xy}}{\sum_{a=0}^{g-1} I_{xa}}. \quad (6)$$

The selection probability area α_{xy} of solution vector S_{xy} between 0 and 1 is as follows:

$$\alpha_{xy} = \begin{cases} [0, \beta_{xy}), & y = 0, \\ \left[\sum_{a=0}^{y-1} \beta_{xa}, \sum_{a=0}^y \beta_{xa} \right], & 0 < y < g - 1, \\ \left(\sum_{a=0}^{y-1} \beta_{xa}, 1 \right], & y = g - 1. \end{cases} \quad (7)$$

Then selection process generates g random numbers between 0 and 1. It selects g solution vectors according to g random numbers which appear in probability area.

In crossover process, we use the multipoint crossover method with self-adaptive crossover rate [28]. We set the initial crossover rate with 1. Firstly, crossover process calculates the crossover rate for the solution vectors of current generation. We record ζ^{pre} as the crossover rate of previous generation solution vectors.

The average fitness value of current generation solution vectors is as follows:

$$I_{x \text{ average}} = \frac{\sum_{i=0}^{g-1} I_{xi}}{g}. \quad (8)$$

I_{xi} is the fitness value of the current generation solution vector. The average fitness value of previous generation solution vectors is as follows:

$$I_{x \text{ average}}^{\text{pre}} = \frac{\sum_{i=0}^{g-1} I_{xi}^{\text{pre}}}{g}. \quad (9)$$

I_{xi}^{pre} is the fitness value of the previous generation solution vector. The crossover rate ζ of the current generation solution vectors is as follows:

$$\zeta = \zeta^{\text{pre}} \times \left(1 - \frac{I_{x \text{ average}} - I_{x \text{ average}}^{\text{pre}}}{I_{x \text{ average}}^{\text{pre}}} \right). \quad (10)$$

We assume that $I_{x \text{ max}}$ is the largest fitness value of the solution vectors. Crossover process uses the random mating strategy for mating the population. For each pair of mating solution vectors, we assume that I_{xy} is the bigger fitness value of the two solution vectors. The crossover rate ζ_{xy} of this pair of mating solution vectors is as follows:

$$\zeta_{xy} = \zeta \times \frac{I_{x \text{ max}} - I_{xy}}{I_{x \text{ max}} - I_{x \text{ average}}}. \quad (11)$$

If this pair of mating solution vectors needs to be crossed according to crossover rate ζ_{xy} , crossover process generates n random numbers of 0 or 1. The position will be the crossover point if the random number is 1. The process crosses this pair of mating solution vectors according to the crossover points.

We use the multipoint mutation method with self-adaptive mutation rate [28] as in the crossover process. Firstly, the mutation process calculates the mutation rate for the solution vectors. We assume that $I_{x \text{ max}}$ is the largest fitness value of the solution vectors. I_{xy} is the fitness value of solution vector S_{xy} . The mutation rate δ_{xy} of S_{xy} is as follows:

$$\delta_{xy} = \frac{\zeta}{n} \times \frac{I_{x \text{ max}} - I_{xy}}{I_{x \text{ max}} - I_{x \text{ average}}}. \quad (12)$$

Then the mutation process sets the mutation points for S_{xy} according to mutation rate δ_{xy} . If the mutation process sets the point as a mutation point, it records the related number with 1; otherwise it records the related number with 0. The solution vector is an incorrect solution after crossover if the number θ of a physical host that appears in solution vector is bigger than the number φ of VMs that can be loaded in this physical host. In this case, we set $\theta - \varphi$ mutation points (set the related number with 1) randomly on the position of the physical host in solution vector. For example, there are two solution vectors ($P_5, P_{14}, P_{54}, P_{189}, P_{201}, P_{323}, P_{405}, P_{667}, P_{701}, P_{899}$) and ($P_{88}, P_{103}, P_{166}, P_{255}, P_{323}, P_{323}, P_{391}, P_{405}, P_{653}, P_{878}$), the crossover point is 8, and then the solution vectors after crossover are ($P_5, P_{14}, P_{54}, P_{189}, P_{201}, P_{323}, P_{405}, P_{405}, P_{653}, P_{878}$) and ($P_{88}, P_{103}, P_{166}, P_{255}, P_{323}, P_{323}, P_{391}, P_{667}, P_{701}, P_{899}$). The solution vector ($P_5, P_{14}, P_{54}, P_{189}, P_{201}, P_{323}, P_{405}, P_{405}, P_{653}, P_{878}$) is an incorrect solution if the remaining available resources of P_{405} only can load one VM. So we set P_{405} as mutation point.

After determining the mutation points, the mutation process continues to mutate the mutation points. In initial population generation process, we take the coverage of the solution space into consideration, so the mutation process mutates the mutation point with the scope of $w/n/g/g$. As for P_i , the mutation interval is as follows:

$$\begin{aligned} & [P_{i-w/n/g/g+w}, P_{w-1}] \cup [P_0, P_{i+w/n/g/g}] \\ & \quad i - w/n/g/g < 0, \\ & [P_{i-w/n/g/g}, P_{i+w/n/g/g}] \\ & \quad i - w/n/g/g \geq 0, \quad i + w/n/g/g \leq w - 1, \\ & [P_{i-w/n/g/g}, P_{w-1}] \cup [P_0, P_{i+w/n/g/g-(w-1)}] \\ & \quad i + w/n/g/g > w - 1. \end{aligned} \quad (13)$$

Because of the indeterminacy of the mutation points, mutation process mutates the mutation points according to the sequence of the mutation points. According to the nonmutation points (the relevant position of random number is 0), mutation process updates the information of members in mutation interval and deletes the members of mutation interval if the remaining resources of relevant physical

hosts cannot satisfy the requirement of users. The number of alternative mutation physical hosts after updating the mutation interval is l . The alternative mutation physical hosts are expressed as $(P'_0, P'_1, P'_2, \dots, P'_{l-1})$. If $l = 0$, the mutation process randomly selects a mutation physical host that can satisfy the requirements of users from solution space. If $l > 0$, the mutation process selects a physical host from mutation interval proportional to the benefit of the fitness value to mutate the mutation point.

If physical host P'_i loads the VM, the performance per watt I'_i is as follows:

$$I'_i = (h) \times \left(C_i \times (F_i + h) \times \left(\sqrt{\frac{(F_i + h) \times Vt_i}{K_i} + \frac{(F_i + h)^2}{4K_i^2}} + Vt_i + \frac{F_i + h}{2K_i} \right)^2 - C_i \times F_i \times \left(\sqrt{\frac{F_i \times Vt_i}{K_i} + \frac{F_i^2}{4K_i^2}} + Vt_i + \frac{F_i}{2K_i} \right)^2 \right)^{-1}. \quad (14)$$

The selection probability β'_i of alternative mutation physical host P'_i is as follows:

$$\beta'_i = \frac{I'_i}{\sum_{j=0}^{l-1} I'_j}. \quad (15)$$

The probability area α'_i of alternative mutation physical host P'_i between 0 and 1 is as follows:

$$\alpha'_i = \begin{cases} [0, \beta'_i], & i = 0, \\ \left[\sum_{a=0}^{i-1} \beta'_a, \sum_{a=0}^i \beta'_a \right], & 0 < i < l - 1, \\ \left[\sum_{a=0}^{i-1} \beta'_a, 1 \right], & i = l - 1. \end{cases} \quad (16)$$

Then mutation process generates a random number between 0 and 1. It selects an alternative physical host according to the probability area in which the random number appeared. After mutating the mutation point, mutation process sets the relevant position number of solution vector with 0.

3.4. Iteration and Termination in the First Stage. After the mutation process, the algorithm judges whether it reaches

the iterative termination conditions of the first stage of DPGA. If so, the algorithm stops iteration in the first stage; otherwise it continues the iteration. The solution vector with the maximum fitness value is the optimal solution vector in the first stage. The iterative termination conditions of the first stage are as follows.

- (1) Iterative times attain the preset maximum iterative times in the first stage. We set the maximum iterative times in the first stage with τ . The value of τ is related to w and n .
- (2) The difference between the largest fitness value and the average fitness value is less than a certain ratio of the average fitness value. We set the difference ratio of the second termination condition of the first stage with μ . We record the largest fitness value of the solution vectors as $I_{x \max}$. Thus, the first stage of the algorithm will terminate if it satisfies the following formula:

$$I_{x \max} \leq (1 + \mu) \times I_{x \text{ average}}. \quad (17)$$

- (3) The optimized proportion of the average fitness values between two adjacent generation populations is less than the preset ratio. We set the optimized proportion of the third termination condition of the first stage with σ . Thus, the first stage of the algorithm will terminate if it satisfies the following formula:

$$I_{x \text{ average}} \leq (1 + \sigma) \times I_{x \text{ average}}^{\text{pre}}. \quad (18)$$

3.5. Genetic Algorithm in the Second Stage. After completing the iteration in the first stage of DPGA, the algorithm collects the solution vectors obtained from the first stage as the initial population in the second stage. The selection process in the second stage chooses the solution vectors in the same way as in the first stage. We also use the multipoint crossover method with self-adaptive crossover rate as in the first stage.

The average fitness value of current generation solution vectors in the second stage is as follows:

$$I'_{\text{average}} = \frac{\sum_{i=0}^{g-1} I'_i}{g}. \quad (19)$$

I'_i is the fitness value of the current generation solution vector in the second stage. The average fitness value of previous generation solution vectors in the second stage is as follows:

$$I_{\text{average}}^{\text{pre}} = \frac{\sum_{i=0}^{g-1} I_i^{\text{pre}}}{g}. \quad (20)$$

I_i^{pre} is the fitness value of the previous generation solution vector in the second stage. The crossover rate ζ' of the current generation solution vectors in the second stage is as follows:

$$\zeta' = \zeta^{\text{pre}} \times \left(1 - \frac{I'_{\text{average}} - I_{\text{average}}^{\text{pre}}}{I_{\text{average}}^{\text{pre}}} \right). \quad (21)$$

$\zeta^{\text{pre}'}$ is the crossover rate of the previous generation solution vectors in the second stage. I'_i is the fitness value of the current generation solution vector in the second stage. $I_i^{\text{pre}'}$ is the fitness value of the previous generation solution vector in the second stage. The crossover rate ζ'_x of the pair of the mating solution vectors in the second stage is as follows:

$$\zeta'_x = \zeta' \times \frac{I'_{\max} - I'_x}{I'_{\max} - I'_{\text{average}}}. \quad (22)$$

I'_{\max} is the largest fitness value of the solution vectors in the second stage. I'_x is the bigger fitness value of the pair of the mating solution vectors in the second stage.

The mutation process uses the multipoint mutation method with self-adaptive mutation rate and confirms the mutation points as the way it used in the first stage. The mutation rate δ'_x of S_x in the second stage is as follows:

$$\delta'_x = \frac{\zeta'}{n} \times \frac{I'_{\max} - I'_x}{I'_{\max} - I'_{\text{average}}}. \quad (23)$$

I'_{\max} is the largest fitness value of the solution vectors in the second stage. I'_x is the fitness value of solution vector S'_x in the second stage. After confirming the mutation points in the second stage, being different from the process in the first stage, it mutates the mutation points with the scope of the whole solution space. Because of the indeterminacy of the mutation points, mutation process mutates the mutation points according to the sequence of the mutation points. Firstly, according to the nonmutation points, mutation process updates the information of all solution space members and deletes the members of mutation interval if the remaining resources of relevant physical hosts cannot satisfy the requirement of users. Then the mutation process mutates the mutation points in the same way as in the first stage.

After the mutation process in the second stage, the algorithm judges whether it reaches the iterative termination conditions of the second stage of DPGA. If so, the algorithm stops iteration in the second stage; otherwise it continues the iteration. The solution vector with the maximum fitness value is the optimal solution vector. The iterative termination conditions of the second stage are as follows.

- (1) Iterative times attain the preset maximum iterative times in the second stage. Because the solution vectors obtained from the first stage are relative optimal solutions, we decrease the maximum iterative times accordingly in the second stage. We set the maximum iterative times in the second stage with τ/g .
- (2) The difference between the largest fitness value and the average fitness value is less than a certain ratio of the average fitness value. As the result of the fact that the solution vectors obtained from the first stage are relative optimal solutions, we decrease the difference ratio accordingly in the second stage. We set the different ratio of the second termination condition of the second stage with μ/g . We record the largest fitness value of the solution vectors as I'_{\max} . Thus, the

second stage of the algorithm will terminate if it satisfies the following formula:

$$I'_{\max} \leq (1 + \mu/g) \times I'_{\text{average}}. \quad (24)$$

- (3) The optimized proportion of the average fitness values between two adjacent generation populations is less than the preset ratio. We decrease the optimized proportion accordingly in the second stage in consequence of the fact that the solution vectors obtained from the first stage are relative optimal solutions. We set the optimized proportion of the third termination condition of the second stage with σ/g . The second stage of the algorithm will terminate if it satisfies the following formula:

$$I'_{\text{average}} \leq (1 + \sigma/g) \times I_{\text{average}}^{\text{pre}'}. \quad (25)$$

4. Evaluation

In order to simulate a dynamic cloud platform, we utilize a cloud simulator named CloudSim toolkit [29], version 3.0.3. The CloudSim framework can create different kinds of entities and remove data center entities at run time. The CloudSim framework can also calculate the status information of entities such as resource utilization and power consumption during the simulation period. We choose 6 kinds of host models as shown in Table 2 for CloudSim platform in the experiments.

According to Table 3, we need to create power model classes for each kind of host models to calculate the power consumption of the hosts in CloudSim platform [30].

In the experiments, DPGA needs some parameters of the hosts. The CloudSim platform does not provide the parameters C , K , and Vt of the hosts which should have been obtained from the hardware providers. Therefore we need to calculate the approximate values of the parameters. Firstly, we pick up two groups of core voltage and core frequency for each kind of host model, and then we calculate their power consumption by the CloudSim platform. Finally, we utilize the matlab [31] to solve the multiple equations established by formula (2) according to the information of Table 4. The values of parameters are showed in Table 4.

The class PowerHost of the CloudSim platform does not contain the member variables of C , K , and Vt . We create a new class which extends the class PowerHost by adding the member variables of C , K , and Vt so that the entities in the experiments can record the information of parameters for DPGA. In the experiments, a data center consisting of w hosts is created. These hosts are averagely composed of the above 6 kinds of host models. Then the data center creates d VMs according to Table 5 averagely with the full utilization model as the original loads of the data center.

In the experiments, the data center creates n VMs according to Table 6 averagely as the requirements of users with full utilization model.

4.1. Performance per Watt with Different Original Loads. In this experiment, we set the hosts number of the data center

TABLE 2: Host models for CloudSim platform in the experiments.

Host	CPU	Memory (G)
IBM System X3650 M4	2 × [Intel Xeon E5-2660 2200 MHz, 10 cores]	64
IBM System X3300 M4	2 × [Intel Xeon E5-2470 2300 MHz, 8 cores]	24
Dell PowerEdge R710	2 × [Intel Xeon X5675 3066 MHz, 6 cores]	24
Dell PowerEdge R610	2 × [Intel Xeon X5670 2933 MHz, 6 cores]	12
Acer Altos AR580 F2	4 × [Intel Xeon X4607 2200 MHz, 6 cores]	64
Acer Altos R380 F2	2 × [Intel Xeon X2650 2000 MHz, 8 cores]	24

TABLE 3: Benchmark results summary of host models.

Host	Power consumption for the different target loads (W)										
	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
IBM System X3650 M4	52.7	80.5	90.3	100	110	120	131	143	161	178	203
IBM System X3300 M4	50.8	74.3	84.1	94.5	106	122	141	164	188	220	260
Dell PowerEdge R710	62.2	104	117	127	137	147	157	170	187	205	227
Dell PowerEdge R610	61.9	102	115	126	137	149	160	176	195	218	242
Acer Altos AR580 F2	109	155	170	184	197	211	226	252	280	324	368
Acer Altos R380 F2	52.9	77.1	85.4	94	102	110	124	141	162	186	215

$w = 1600$ and the VMs number $n = 10$ as the requirements of users. We adjust the VMs number d as the original loads from 0 to 5000 and allocate these VMs to the hosts randomly. It represents different load levels of the data center. All idle hosts are switched to Sleep state. The experiment is designed for verifying the efficiency of DPGA in performance per watt of a cloud center under different original loads. In this scenario, we compare performance per watt of DPGA with ST (static threshold) which sets the utilization threshold to 0.9, IQR (interquartile range) which sets the safety parameter to 1.5, LR (local regression) which sets the safety parameter to 1.2, LRR (local regression robust) which sets the safety parameter to 1.2, and MAD (median absolute deviation) which sets the safety parameter to 2.5 [30]. As illustrated in Figure 2, DPGA placement strategy for VMs deployment under different original loads gets higher performance per watt than other placement strategies. Further, when $d \leq 1000$, namely, the data center under an approximate idle state, performance per watt of DPGA placement strategy increases rapidly. When $1000 < d \leq 2000$, namely, the data center under a low loading state, performance per watt of DPGA placement strategy increases at a relatively flat rate. When $2000 < d \leq 4000$, namely, the data center under a moderate loading state, performance per watt of DPGA placement strategy is relatively stable. When $d > 4000$, namely, the data center under an overloading state, performance per watt of DPGA placement strategy begins to decline gradually. This is because the hosts under the state from idle to load or under the overload states consume more power than the hosts under the state of a certain load. In conclusion, DPGA has a better performance per watt and is relatively more stable because DPGA placement strategy is the heuristic approach. It takes the performance per watt as evaluation standard and tends towards stability by two step iterations.

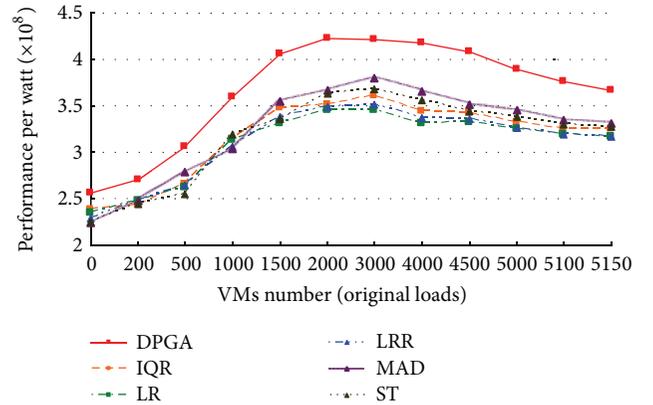


FIGURE 2: Comparison of performance per watt with different original loads.

4.2. Performance per Watt with Different User Requests. In this experiment, we set the hosts number of the data center $w = 1600$ and the VMs number $d = 3000$ as the original loads. Then we allocate these VMs to the hosts randomly. We adjust the VMs number n as the requirements of users from 10 to 50. All idle hosts are switched to Sleep state. The experiment is designed for verifying the efficiency of DPGA in performance per watt of a cloud center with different requirements of users. In this scenario, we compare performance per watt of DPGA with ST, IQR, LR, LRR, and MAD that take the same parameters as the experiment in Section 4.1. As illustrated in Figure 3, DPGA placement strategy for VMs deployment with different requirements of users gets higher performance per watt than other placement strategies. Further, with the increase of the requirements of users, DPGA placement strategy for VMs deployment

TABLE 4: Parameters summary of host models.

Host	Core voltage (V)	Frequency (Hz)	Power (W)	ΔE (W)	C (F)	K	Vt. (V)
IBM System X3650 M4	0.806	$800 * 10^6$	106.364	85.272	$0.501 * 10^{-12}$	$87.565 * 10^9$	0.422
	1.172	$2100 * 10^6$	191.636				
IBM System X3300 M4	0.986	$821.5 * 10^6$	101.075	130.386	$0.631 * 10^{-12}$	$57.787 * 10^9$	0.512
	1.433	$2135.9 * 10^6$	231.461				
Dell PowerEdge R710	0.857	$1066.4 * 10^6$	131.781	85.647	$0.526 * 10^{-12}$	$72.411 * 10^9$	0.468
	1.246	$2932.6 * 10^6$	217.428				
Dell PowerEdge R610	0.906	$980 * 10^6$	129.754	96.781	$0.566 * 10^{-12}$	$65.298 * 10^9$	0.502
	1.317	$2744 * 10^6$	226.535				
Acer Altos AR580 F2	0.994	$800 * 10^6$	192.273	191.727	$0.617 * 10^{-12}$	$85.299 * 10^9$	0.521
	1.445	$2100 * 10^6$	348				
Acer Altos R380 F2	0.953	$700 * 10^6$	98	102.5	$0.59 * 10^{-12}$	$55.441 * 10^9$	0.514
	1.386	$1900 * 10^6$	200.5				

TABLE 5: VM models for loads of data center.

Item	VM models							
	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8
VCPU (MHz)	$1000 * 1$	$1200 * 2$	$1300 * 2$	$1400 * 4$	$1500 * 4$	$1600 * 6$	$1800 * 6$	$2000 * 8$
RAM (M)	512	1024	1024	2048	2048	4096	4096	8192

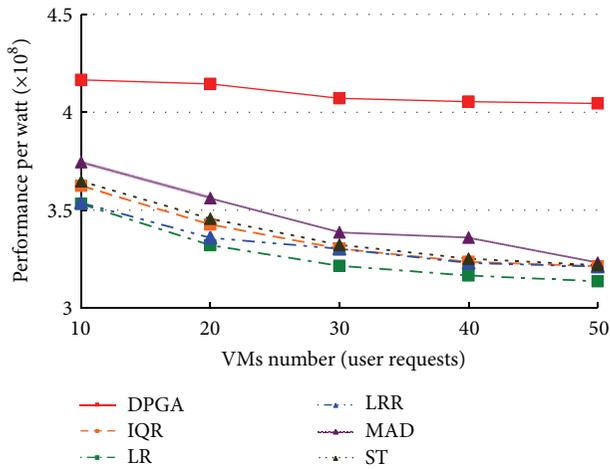


FIGURE 3: Comparison of performance per watt with different user requests.

gets more stable performance per watt than other placement strategies.

4.3. Performance per Watt with Different State of Idle Hosts.

In this experiment, we set the hosts number of the data center $w = 1600$ and the VMs number $n = 10$ as the requirements of users. We adjust the VMs number d as the original loads from 0 to 5000 and allocate these VMs to the hosts randomly. It represents different load levels of the data center. There are two policies to be formulated for idle hosts. The first policy is On/Off policy, wherein all idle hosts are switched off. The second policy is On/Sleep policy, wherein all idle hosts are switched to Sleep state. The experiment is designed for

verifying the efficiency of DPGA in performance per watt of a cloud center with different policies for idle hosts. In this scenario, we compare performance per watt of DPGA with On/Off policy for idle hosts and DPGA with On/Sleep policy for idle hosts. As illustrated in Figure 4, DPGA placement strategy for VMs deployment with On/Sleep policy gets higher performance per watt than DPGA placement strategy with On/Off policy when the data center is under an approximate idle state. DPGA placement strategy for VMs deployment with On/Sleep policy gets approximately the same performance per watt as DPGA placement strategy with On/Off policy when the data center is under a loading state. This is because the idle hosts at Sleep state consume certain power while the turned-off idle hosts do not consume any power. Therefore DPGA placement strategy for VMs deployment is more suitable for the cloud center under a loading state.

4.4. Actual and Theoretical Values of Performance per Watt.

In this experiment, we set the hosts number of the data center $w = 1600$ and the VMs number $n = 10$ as the requirements of users. We adjust the VMs number d as the original loads from 500 to 5000 and allocate these VMs to the hosts randomly. It represents different load levels of the data center. All idle hosts are switched to Sleep state. In this scenario, we compare actual performance per watt of DPGA with theoretical performance per watt of DPGA calculated by formula (5). As illustrated in Figure 5, theoretical performance per watt is higher than actual performance per watt when the data center is under a low loading state. Theoretical performance per watt is approximately the same as actual performance per watt when the data center is under a moderate loading state. Theoretical performance per watt is lower than actual performance per

TABLE 6: VM models for user requests.

Item	User request models									
	Model 1	Model 2	Model 3	Model 4	Model 5	Model 6	Model 7	Model 8	Model 9	Model 10
VCPU (MHz)	1000 * 1	1100 * 2	1300 * 2	1400 * 4	1500 * 4	1600 * 6	1700 * 6	1800 * 8	1900 * 8	2000 * 10
RAM (M)	256	512	512	1024	1024	2048	2048	4096	4096	8192

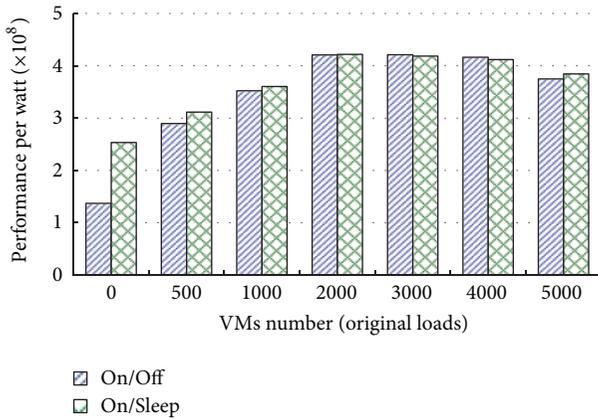


FIGURE 4: Comparison of performance per watt with different state of idle hosts.

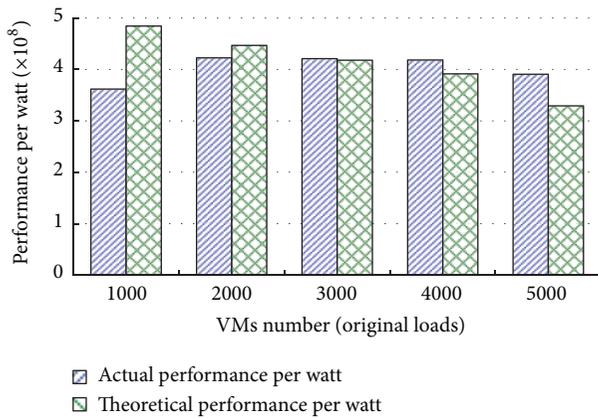


FIGURE 5: Comparison of actual performance per watt and theoretical performance per watt.

watt when the data center is under an overloading state. This is because the hosts under a moderate loading state can calculate a relatively more accurate value of power consumption by DVFS formula than the hosts under a low loading state or an overloading state. In conclusion, DPGA placement strategy for VMs deployment is more suitable for the cloud center under a moderate loading state.

5. Conclusion

In this paper, we present the design, implementation, and evaluation of a distributed parallel genetic algorithm of virtual machine placement strategy on cloud platform. The algorithm is divided into two stages to get a better and more

accurate solution. We assign the performance per watt as evaluation standard. We use the multipoint crossover method with self-adaptive crossover rate and the multipoint mutation method with self-adaptive mutation rate in the proposed approach. DPGA executes the first stage genetic algorithm with selected initial subpopulation and puts the solutions obtained into the second stage genetic algorithm as initial population. Then it finally gets a relatively optimal solution from the second stage. The experimental results show that our approach is an efficient, stable, and effective placement strategy for VM deployment.

To further improve the performance of placement strategy for VM deployment, there are also many problems that need to be solved in the future. The number of parallel executions in the first stage g should be related to the size of solution space w and the number of deployment VMs n . We plan to assign the value of g according to w and n . In population initialization process, we select initial subpopulation from solution space dispersedly and averagely. In crossover and mutation process, we use the multipoint crossover method with self-adaptive crossover rate and the multipoint mutation method with self-adaptive mutation rate. We plan to optimize the algorithm in detail. In the judgment of iterative termination conditions, the maximum iteration times should be related to the size of solution space w and the number of deployment VMs n . We plan to assign the maximum iteration times according to w and n . There are also two open questions on the termination of the two stages. One is to determine the difference ratio between the largest fitness value and average fitness value, and the other one is to determine the optimized proportion of the average fitness values between two adjacent generation populations. In this paper, to ensure that the algorithm is executed correctly, we assume that $w/n/g/g > 1$. In order to execute the algorithm efficiently in the case of $w/n/g/g \leq 1$, we plan to combine our approach with other methods. Our approach is appropriate for the case that all physical hosts of solution space are in a fast LAN and in the same network environment. We plan to extend our approach to WAN and different network environment. Our approach uses the parallel genetic algorithm. We plan to use other heuristic algorithms such as ant colony algorithm, bee colony algorithm, and particle swarm optimization to implement placement strategy of VMs deployment and compare their performance.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper. There is no direct financial relation that might lead to a conflict of interests for any of the authors.

References

- [1] L. Wang, J. Tao, M. Kunze, A. C. Castellanos, D. Kramer, and W. Karl, "Scientific cloud computing: early definition and experience," in *Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications (HPCC '08)*, pp. 825–830, Dalian, China, September 2008.
- [2] J. Nakajima, Q. Lin, S. Yang et al., "Optimizing virtual machines using hybrid virtualization," in *Proceedings of the 26th Annual ACM Symposium on Applied Computing (SAC '11)*, pp. 573–578, March 2011.
- [3] N. Regola and J. Ducom, "Recommendations for virtualization technologies in high performance computing," in *Proceedings of the 2nd IEEE International Conference on Cloud Computing Technology and Science*, pp. 409–416, December 2010.
- [4] P. Barham, B. Dragovic, K. Fraser et al., "Xen and the art of virtualization," in *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03)*, pp. 164–177, New York, NY, USA, October 2003.
- [5] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W. H. Freeman, San Francisco, Calif, USA, 1979.
- [6] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proceedings of the Spring Joint Computer Conference*, pp. 483–485, ACM, Atlantic City, NJ, USA, April 1967.
- [7] J. L. Gustafson, "Reevaluating Amdahl's law," *Communications of the ACM*, vol. 31, no. 5, pp. 532–533, 1988.
- [8] H. X. Sun and L. M. Ni, *Scalable Problems and Memory-Bounded Speedup*, Institute for Computer Applications in Science and Engineering, Hampton, Va, USA, 1992.
- [9] J. Tordsson, R. S. Montero, R. Moreno-Vozmediano, and I. M. Llorente, "Cloud brokering mechanisms for optimized placement of virtual machines across multiple providers," *Future Generation Computer Systems*, vol. 28, no. 2, pp. 358–367, 2012.
- [10] M. Steiner, B. G. Gaglianella, V. Gurbanli et al., "Network-aware service placement in a distributed cloud environment," *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 73–74, 2012.
- [11] W. Wang, H. Chen, and X. Chen, "An availability-aware virtual machine placement approach for dynamic scaling of cloud applications," in *Proceedings of the 9th International Conference on Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC '12)*, pp. 509–516, 2012.
- [12] Z. I. M. Yusoh and M. Tang, "A penalty-based genetic algorithm for the composite SaaS placement problem in the cloud," in *Proceedings of the 6th IEEE World Congress on Computational Intelligence (WCCI '10) and IEEE Congress on Evolutionary Computation (CEC '10)*, pp. 1–8, July 2010.
- [13] B. Li, J. Li, J. Huai, T. Wo, Q. Li, and L. Zhong, "EnaCloud: an energy-saving application live placement approach for cloud computing environments," in *Proceedings of the IEEE International Conference on Cloud Computing (CLOUD '09)*, pp. 17–24, Bangalore, India, September 2009.
- [14] Z. W. Ni, X. F. Pan, and Z. J. Wu, "An ant colony optimization for the composite SaaS placement problem in the cloud," *Applied Mechanics and Materials*, vol. 130–134, pp. 3062–3067, 2012.
- [15] Z. I. M. Yusoh and M. Tang, "A cooperative coevolutionary algorithm for the composite SaaS Placement Problem in the Cloud," in *Proceedings of the 17th International Conference on Neural Information Processing*, pp. 618–625, Springer, 2010.
- [16] H. Wang, W. Xu, F. Wang, and C. Jia, "A cloud-computing-based data placement strategy in high-speed railway," *Discrete Dynamics in Nature and Society*, vol. 2012, Article ID 396387, 15 pages, 2012.
- [17] D. Yuan, Y. Yang, X. Liu, and J. Chen, "A data placement strategy in scientific cloud workflows," *Future Generation Computer Systems*, vol. 26, no. 8, pp. 1200–1214, 2010.
- [18] L. Guo, Z. He, S. Zhao, N. Zhang, J. Wang, and C. Jiang, "Multi-objective optimization for data placement strategy in cloud computing," in *Information Computing and Applications*, vol. 308 of *Communications in Computer and Information Science*, Springer, Berlin, Germany, 2012.
- [19] J. Ding, H. Y. Han, and A. H. Zhou, "A data placement strategy for data-intensive cloud storage," *Advanced Materials Research*, vol. 354, pp. 896–900, 2012.
- [20] G. von Laszewski, L. Wang, A. J. Younge, and X. He, "Power-aware scheduling of virtual machines in DVFS-enabled clusters," in *Proceedings of the IEEE International Conference on Cluster Computing and Workshops (CLUSTER '09)*, pp. 1–10, September 2009.
- [21] W. Fang, X. Liang, S. Li, L. Chiaraviglio, and N. Xiong, "VMPlanner: optimizing virtual machine placement and traffic flow routing to reduce network power costs in cloud data centers," *Computer Networks*, vol. 57, no. 1, pp. 179–196, 2013.
- [22] R. Ge, X. Feng, and K. W. Cameron, "Performance-constrained distributed DVS scheduling for scientific applications on power-aware clusters," in *Proceedings of the ACM/IEEE Conference on Supercomputing (SC '05)*, November 2005.
- [23] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Liu, "A multi-objective ant colony system algorithm for virtual machine placement in cloud computing," *Journal of Computer and System Sciences*, vol. 79, no. 8, pp. 1230–1242, 2013.
- [24] B. Kantarci, L. Foschini, A. Corradi, and H. T. Mouftah, "Inter-and-intra data center VM-placement for energy-efficient large-scale cloud systems," in *Proceedings of the IEEE Globecom Workshops (GC Wkshps '12)*, pp. 708–713, Anaheim, Calif, USA, December 2012.
- [25] S. Chaisiri, B. Lee, and D. Niyato, "Optimal virtual machine placement across multiple cloud providers," in *Proceedings of the IEEE Asia-Pacific Services Computing Conference (APSCC '09)*, pp. 103–110, December 2009.
- [26] J. T. Piao and J. Yan, "A network-aware virtual machine placement and migration approach in cloud computing," in *Proceedings of the 9th International Conference on Grid and Cloud Computing (GCC '10)*, pp. 87–92, Nanjing, China, November 2010.
- [27] C. H. Hsu, U. Kremer, and M. Hsiao, "Compiler-directed dynamic frequency and voltage scheduling," in *Power-Aware Computer Systems*, pp. 65–81, Springer, Berlin, Germany, 2001.
- [28] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 24, no. 4, pp. 656–667, 1994.
- [29] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. de Rose, and R. Buyya, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience*, vol. 41, no. 1, pp. 23–50, 2011.

- [30] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers," *Concurrency Computation Practice and Experience*, vol. 24, no. 13, pp. 1397–1420, 2012.
- [31] *MathWorks T. Matlab*, The MathWorks, Natick, Mass, USA, 2004.

Research Article

An Artificial Bee Colony Algorithm for Uncertain Portfolio Selection

Wei Chen

School of Information, Capital University of Economics and Business, Beijing 100070, China

Correspondence should be addressed to Wei Chen; chenwei@cueb.edu.cn

Received 3 March 2014; Revised 10 June 2014; Accepted 10 June 2014; Published 26 June 2014

Academic Editor: T. O. Ting

Copyright © 2014 Wei Chen. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Portfolio selection is an important issue for researchers and practitioners. In this paper, under the assumption that security returns are given by experts' evaluations rather than historical data, we discuss the portfolio adjusting problem which takes transaction costs and diversification degree of portfolio into consideration. Uncertain variables are employed to describe the security returns. In the proposed mean-variance-entropy model, the uncertain mean value of the return is used to measure investment return, the uncertain variance of the return is used to measure investment risk, and the entropy is used to measure diversification degree of portfolio. In order to solve the proposed model, a modified artificial bee colony (ABC) algorithm is designed. Finally, a numerical example is given to illustrate the modelling idea and the effectiveness of the proposed algorithm.

1. Introduction

Portfolio selection deals with the problem of how to allocate investor's wealth among different assets such that the investment goal can be achieved. Markowitz [1] originally proposed the mean-variance (M-V) model for portfolio selection in 1952, which has played an important role in the development of modern portfolio selection theory. The key principle of the M-V model is to use the expected return of a portfolio as the investment return and to use the variance (or standard deviation) of the expected returns of the portfolio as the investment risk. Markowitz's M-V model for portfolio selection problems can be formulated mathematically in two ways: minimizing risk under prescribing a minimum acceptable expected return level and maximizing expected return under prescribing a maximum acceptable risk level. Since then, a number of scholars, including Sharp [2], Merton [3], Pang [4], Perold [5], Best and Grauer [6], Konno and Yamazaki [7], and Best and Hlouskova [8], proposed different mathematical methods for the development of portfolio models. All the above-mentioned researches assume that the security returns are random variables with probability distributions. In probability theory, probability density and probability distribution functions of a random variable are usually derived from historical data. However, in the real

securities market, market imperfections make the security returns present vagueness and ambiguity so that sometimes security returns cannot be well reflected by historical data. Therefore, the prediction of security returns is determined largely by experts' estimation and contains much subjective imprecision rather than randomness. With the extensive application of fuzzy set theory [9], many researchers, such as Tanaka and Guo [10], Carlsson et al. [11], Vercher et al. [12], Zhang et al. [13], Chen et al. [14], Tsaur [15], and Gupta et al. [16], have investigated the portfolio selection problem in fuzzy environment.

An important assumption for the above fuzzy portfolio problems is that security returns are fuzzy variables. However, as we research the problem deeper, we find that paradoxes will appear if fuzzy set is employed. For example, if a security return is regarded as a fuzzy variable, then we may assign it a membership function, suppose it is a triangular fuzzy variable $\xi = (-0.02, 0.03, 0.08)$. Based on the membership function, the possibility theory will immediately conclude the following three propositions: (a) the return is exactly 0.03 with possibility measure 1; (b) the return is not 0.03 with possibility measure 1; and (c) "return is exactly 0.03" and "return is not exactly 0.03" are equally likely. However, it is doubtless that the belief degree of "return is exactly 0.03" is almost zero. On the other hand, "return is exactly 0.03"

and “return is not exactly 0.03” have the same belief degree in possibility measure, which implies that the two events will happen equally likely. It seems that no human being can accept this conclusion. This paradox shows that those imprecise quantities like security returns cannot be quantified by possibility measure and then they are not fuzzy concepts. To avoid paradox, Liu [17] founded the uncertainty theory based on an axiomatic system of normality, self-duality, countable subadditivity, and product uncertain measure. Based on the uncertainty theory, much work has been done on the development of theoretical and practical work. Peng and Iwamura [18] proved a sufficient and necessary condition of uncertainty distribution. Chen and Liu [19] proved the existence and uniqueness theorem for uncertain differential equations. Gao [20] investigated the α -shortest path and the shortest path problem in uncertain networks. Ding and Gao [21] assumed that demand is uncertain variable and formulated an optimal (σ, S) policy for uncertain multiproduct newsboy problem. In the area of uncertain portfolio selection problems, Zhu [22] solved an uncertain optimal control problem and applied it to a portfolio selection model. Taking semiabsolute deviation as a risk measure, Liu and Qin [23] presented three mean semiabsolute deviation models under the assumption that security returns are uncertain variables. Huang [24] proposed two new mean-variance and mean-semivariance models in which security returns are given subject to experts’ estimations. Later, Huang [25] defined a new risk measurement, that is, a risk index, and further proposed a new mean-risk index selection method for portfolio selection based on the new risk measurement. Other portfolio selection methods based on uncertainty theory can be found in [26].

Artificial bee colony (ABC) algorithm is a fairly new metaheuristic proposed by Karaboga [27], which is based on simulating the foraging behavior of honey bee swarms. Based on some classic benchmark functions, the performance of the ABC algorithm was compared with that of some other population-based algorithms such as genetic algorithm (GA), differential evolution (DE), and particle swarm optimization (PSO) in [28–31]. Their research results demonstrated that the ABC algorithm is competitive to other population-based algorithms with an advantage of employing fewer control parameters. Recently, ABC algorithm has captured much attention and has been applied to many practical optimization problems including resource constrained project scheduling problem [32], reliability redundancy allocation problem [33], job shop scheduling problem [34], mechanical design problem [35], and traveling salesman problem [36]. A survey about applications of ABC algorithm is provided by Karaboga et al. in [37].

The purposes of this paper are to propose an uncertain portfolio adjusting model under the assumption that security returns are given mainly by experts’ estimations, in which four criteria, namely, return, risk, transaction cost, and diversification degree of portfolio, are considered, and to develop a modified ABC algorithm for solving the proposed portfolio problem. The rest of the paper is organized as follows. For better understanding of the paper, some necessary knowledge about uncertain variable will be introduced in

Section 2. In Section 3, we will propose an uncertain mean-variance-entropy model for portfolio selection. In Section 4, we develop a modified ABC algorithm to solve the proposed model. After that, an example is given to illustrate the proposed model and the effectiveness of the proposed algorithm in Section 5. Finally, some concluding remarks are given in Section 6.

2. Preliminary

Next, we will introduce some fundamental concepts and properties of uncertainty theory, which will be used throughout this paper.

Definition 1 (Liu [17]). Let Γ be a nonempty set and let \mathcal{L} be a σ -algebra over Γ . Each element $\Lambda \in \mathcal{L}$ is called an event. A set function $\mathcal{M}\{\Lambda\}$ is called an uncertain measure if it satisfies the following three axioms.

Axiom 1 (normality axiom): $\mathcal{M}\{\Gamma\} = 1$.

Axiom 2 (duality axiom): $\mathcal{M}\{\Lambda\} + \mathcal{M}\{\Lambda^c\} = 1$.

Axiom 3 (subadditivity axiom): for every countable sequence of events $\{\Lambda_i\}$, we have

$$\mathcal{M}\left\{\bigcup_{i=1}^{\infty}\Lambda_i\right\}\leq\sum_{i=1}^{\infty}\mathcal{M}\{\Lambda_i\}. \quad (1)$$

The triplet $(\Gamma, \mathcal{L}, \mathcal{M})$ is called an uncertainty space. In order to obtain an uncertain measure of compound event, a product uncertain measure was defined by Liu [38], thus producing the fourth axiom of uncertainty theory as follows.

Axiom 4 (product axiom): let $(\Gamma_k, \mathcal{L}_k, \mathcal{M}_k)$ be uncertainty spaces for $k = 1, 2, \dots$. Then the product uncertain measure \mathcal{M} is an uncertain measure satisfying

$$\mathcal{M}\left\{\prod_{k=1}^{\infty}\Lambda_k\right\}=\bigwedge_{k=1}^{\infty}\mathcal{M}_k\{\Lambda_k\}. \quad (2)$$

Definition 2 (Liu [17]). An uncertain variable is a measurable function ξ from an uncertainty space $(\Gamma, \mathcal{L}, \mathcal{M})$ to the set of real numbers; that is, for any Borel set B of real numbers, the set

$$\{\xi \in B\} = \{\gamma \in \Gamma \mid \xi(\gamma) \in B\} \quad (3)$$

is an event.

In order to describe an uncertain variable, a concept of uncertainty distribution is defined as follows.

Definition 3 (Liu [17]). The uncertainty distribution of an uncertain variable ξ is defined by

$$\Phi(x) = \mathcal{M}\{\xi \leq x\}, \quad (4)$$

for any real number x .

Let ξ be an uncertain variable with continuous uncertainty distribution Φ . Then the inverse function Φ^{-1} is called the inverse uncertainty distribution of ξ .

Theorem 4 (Liu [39]). *Let $\xi_1, \xi_2, \dots, \xi_n$ be independent uncertain variables with uncertainty distributions $\Phi_1, \Phi_2, \dots, \Phi_n$, respectively. If $f(x_1, x_2, \dots, x_n)$ is strictly increasing with respect to x_1, x_2, \dots, x_m and strictly decreasing with respect to $x_{m+1}, x_{m+2}, \dots, x_n$, then $\xi = f(\xi_1, \xi_2, \dots, \xi_n)$ is an uncertain variable with an inverse uncertainty distribution:*

$$\Psi^{-1}(\alpha) = f(\Phi_1^{-1}(\alpha), \dots, \Phi_m^{-1}(\alpha), \Phi_{m+1}^{-1}(1-\alpha), \dots, \Phi_n^{-1}(1-\alpha)). \tag{5}$$

Based on the uncertain measure, the definitions of the expected value and variance of an uncertain variable can be given as follows.

Definition 5 (Liu [17]). Let ξ be an uncertain variable. Then the expected value of ξ is defined by

$$E[\xi] = \int_0^{+\infty} \mathcal{M}\{\xi \geq x\} - \int_{-\infty}^0 \mathcal{M}\{\xi \leq x\} \tag{6}$$

provided that at least one of the two integrals is finite.

As a useful representation of expected value, it has been proved by Liu [17] that

$$E[\xi] = \int_0^1 \Psi^{-1}(\alpha) d\alpha, \tag{7}$$

where Ψ^{-1} is the inverse uncertainty distribution of uncertain variable ξ .

Definition 6 (Liu [17]). Let ξ be an uncertain variable with finite expected value e . Then the variance of ξ is defined by

$$V[\xi] = E[(\xi - e)^2]. \tag{8}$$

Let ξ be an uncertain variable with expected value e . If we only know its uncertainty distribution Ψ , then the variance is

$$\begin{aligned} V[\xi] &= \int_0^{+\infty} \mathcal{M}\{(\xi - e)^2 \geq x\} dx \\ &= \int_0^{+\infty} \mathcal{M}\{(\xi \geq e + \sqrt{x}) \cup (\xi \leq e - \sqrt{x})\} dx \\ &\leq \int_0^{+\infty} (\mathcal{M}\{\xi \geq e + \sqrt{x}\} + \mathcal{M}\{\xi \leq e - \sqrt{x}\}) dx \tag{9} \\ &= \int_0^{+\infty} (1 - \Psi(e + \sqrt{x}) + \Psi(e - \sqrt{x})) dx \\ &= 2 \int_0^{+\infty} x(1 - \Psi(e + x) + \Psi(e - x)) dx. \end{aligned}$$

In this case, it is always assumed that the variance is

$$V[\xi] = 2 \int_0^{+\infty} x(1 - \Psi(e + x) + \Psi(e - x)) dx. \tag{10}$$

The expected value and variance of an uncertain variable satisfy the following properties.

Theorem 7 (Liu [38]). *Let ξ and η be independent uncertain variables with finite expected values. Then for any real numbers a and b , one has*

$$\begin{aligned} E[a\xi + b\eta] &= aE[\xi] + bE[\eta], \\ V[a\xi + b] &= a^2V[\xi]. \end{aligned} \tag{11}$$

3. Uncertain Portfolio Selection Model

Suppose that an investor starts with an existing portfolio and considers to reallocate his/her wealth among n securities. In order to describe the problem conveniently, we use the following notations.

- x_j : the proportion invested in security j ,
- r_j : the return rate of security j ,
- l_j : the lower bound constraint on security j ,
- u_j : the upper bound constraint on security j ,
- k_j : the constant rate of transaction cost for the security j ,
- $j = 1, 2, \dots, n$.

Transaction cost is an important factor considered by investors in financial markets. Arnott and Wanger [40] suggested that ignoring transaction costs would lead to an inefficient portfolio, whereas Sadjadi et al. [41] showed that adding transaction costs would assist decision makers to better understand the behavior of an efficient frontier. Similar to the researches in [42] and others, we assume that the transaction cost is a V-shaped function of differences between a new portfolio $\mathbf{x} = (x_1, x_2, \dots, x_n)$ and the existing portfolio $\mathbf{x}^0 = (x_1^0, x_2^0, \dots, x_n^0)$. That is to say, the transaction cost of j th security can be expressed as

$$c_j = k_j |x_j - x_j^0|, \quad j = 1, 2, \dots, n. \tag{12}$$

Hence, the total transaction cost of the shift from \mathbf{x}^0 to \mathbf{x} is $C(\mathbf{x}; \mathbf{x}^0) = \sum_{j=1}^n k_j |x_j - x_j^0|$.

Furthermore, the net return on the portfolio after paying transaction costs is given by

$$\sum_{j=1}^n r_j x_j - \sum_{j=1}^n k_j |x_j - x_j^0|. \tag{13}$$

For a new investor, it can be taken that $x_j^0 = 0, j = 1, 2, \dots, n$.

Barak et al. [43] pointed out that very small weighting of an asset will have no distinct influence on the portfolio's return, but the administrative and monitoring costs will be increased. Similarly, very high weighting in any asset will cause investors to suffer from a larger risk. Thus, quantity constraints have to be included in the portfolio model. Specifically, a minimum l_j and a maximum u_j for each asset j are given, and we impose that either $x_j = 0$ or $l_j \leq x_j \leq u_j$.

In addition, one of the shortcomings of Markowitz’s model is that the portfolios are often extremely concentrated on a few assets, which is a contradiction to the notion of diversification [44]. As a result, some researchers, such as Jana et al. [44], Fang et al. [45], Kapur [46], and Zhang et al. [47], have applied the entropy measure proposed by Shannon [48] to infer how much the portfolio is diversified. In this paper, we also employ entropy to measure the diversification degree of portfolio. That is to say, we maximize the entropy function:

$$S(x) = -\sum_{j=1}^n x_j \ln x_j. \tag{14}$$

Assume that the objective of the investor wants to minimize the portfolio risk, and to maximize both the portfolio return after paying the transaction costs and the diversification degree of portfolio. Thus, the portfolio selection problem can be formulated as the following three-objective programming problem:

$$\begin{aligned} \min \quad & V \left[\sum_{j=1}^n r_j x_j - \sum_{j=1}^n k_j |x_j - x_j^0| \right] \\ \max \quad & E \left[\left(\sum_{j=1}^n r_j x_j \right) - \sum_{j=1}^n k_j |x_j - x_j^0| \right] \\ \max \quad & -\sum_{j=1}^n x_j \ln x_j \\ \text{s.t.} \quad & \sum_{j=1}^n x_j = 1, \\ & l_j \leq x_j \leq u_j, \quad j = 1, 2, \dots, n, \\ & x_j \geq 0, \quad j = 1, 2, \dots, n, \end{aligned} \tag{15}$$

where V denotes the variance operators and E denotes the expected value operator.

There are many methods to solve the above multiobjective optimization problem (15). One basic method is to transfer the multiobjective optimization problem into a single-objective optimization problem. We can divide these methods into two different types. The first alternative is to construct only one evaluation function for optimization by weighting the multiple objective functions. Initial work on the weighted-sum method can be found in [49] with many subsequent applications and citations. Rao and Roy proposed a method for determining weights based on fuzzy set theory [50]. Marler and Arora provided insight into how the weighted-sum method works and have explored the significance of the weights with respect to preferences, the Pareto optimal set, and the objective-function values [51]. The second alternative is to select one important objective function as the objective function to optimize, while the rest of objective functions are defined as constrained conditions. For example, Marglin developed the ϵ -constraint method, where

one individual objective function is minimized with an upper level constraint imposed on the other objective functions [52]. Ehrgott and Ruzika presented two modifications by first including slack variables in the formulation and second elasticizing the constraints and including surplus variables [53]. More researches for the multiobjective optimization methods can be found in [54–57]. In this paper, based on the second method, the problem (15) can be transformed into the following optimization problem:

$$\begin{aligned} \min \quad & V \left[\sum_{j=1}^n r_j x_j - \sum_{j=1}^n k_j |x_j - x_j^0| \right] \\ \text{s.t.} \quad & E \left[\left(\sum_{j=1}^n r_j x_j \right) - \sum_{j=1}^n k_j |x_j - x_j^0| \right] \geq \mu_0, \\ & -\sum_{j=1}^n x_j \ln x_j \geq h_0, \\ & \sum_{j=1}^n x_j = 1, \\ & l_j \leq x_j \leq u_j, \quad j = 1, 2, \dots, n, \\ & x_j \geq 0, \quad j = 1, 2, \dots, n, \end{aligned} \tag{16}$$

where μ_0 is a required rate of return of portfolio and h_0 is the minimum entropy level the investors can tolerate.

In order to apply model (16) in a practical investment problem, we need to estimate security returns. The favored method is to regard security returns as random variables and then according to the historical observation take the arithmetic mean as the expected returns. However, since the security returns, especially short term security returns, are sensitive to various economic and noneconomic factors, it is found in reality that sometimes the historical data can hardly reflect the future security returns. In this situation, security returns have to be given mainly by experts’ judgements and estimations rather than historical data. As is mentioned before, uncertainty theory provides a new tool to deal with subjective or empirical data. In this paper, we employ uncertain variables to describe the security returns. For the sake of simplicity, throughout this paper, the uncertain return r_j is a zigzag uncertain variable $r_j \sim \mathcal{Z}(a_j, b_j, c_j)$, $j = 1, 2, \dots, n$. r_j can be described with the following uncertainty distribution:

$$\Phi_j(r) = \begin{cases} 0, & \text{if } r \leq a_j, \\ \frac{(r - a)}{2(b - a)}, & \text{if } a_j \leq r \leq b_j, \\ \frac{(r + c - 2b)}{2(c - b)}, & \text{if } b_j \leq r \leq c_j, \\ 1, & \text{if } r \geq c_j, \end{cases} \tag{17}$$

where $a_j < b_j < c_j$.

According to Definitions 5 and 6, the expected value and the variance of the zigzag uncertain variable $\xi \sim \mathcal{Z}(a, b, c)$ are given as

$$E[\xi] = \frac{a + 2b + c}{4}, \tag{18}$$

$$V[\xi] = \frac{33\alpha^3 + 21\alpha^2\beta + 11\alpha\beta^2 - \beta^3}{384\alpha},$$

where $\alpha = \max\{b - a, c - b\}$ and $\beta = \min\{b - a, c - b\}$.

Theorem 8 (Liu [38]). Assume that ξ_1 and ξ_2 are independent zigzag uncertain variables $\mathcal{Z}(a_1, b_1, c_1)$ and $\mathcal{Z}(a_2, b_2, c_2)$, respectively. Then the sum $\xi_1 + \xi_2$ is also a zigzag uncertain variable $\mathcal{Z}(a_1 + a_2, b_1 + b_2, c_1 + c_2)$; that is,

$$\mathcal{Z}(a_1, b_1, c_1) + \mathcal{Z}(a_2, b_2, c_2) = \mathcal{Z}(a_1 + a_2, b_1 + b_2, c_1 + c_2). \tag{19}$$

The product of a zigzag uncertain variable $\mathcal{Z}(a, b, c)$ and a scalar number $k > 0$ is also a zigzag uncertain variable $\mathcal{Z}(ka, kb, kc)$; that is,

$$k \cdot \mathcal{Z}(a, b, c) = \mathcal{Z}(ka, kb, kc). \tag{20}$$

Therefore, for any real numbers $x_j \geq 0, j = 1, 2, \dots, n$, the total uncertain return is still a zigzag uncertain variable in the form of

$$r_p = \sum_{j=1}^n r_j x_j \tag{21}$$

$$\sim \mathcal{Z}\left(\sum_{j=1}^n a_j x_j, \sum_{j=1}^n b_j x_j, \sum_{j=1}^n c_j x_j\right).$$

Furthermore, according to Theorem 7, the net uncertain expected return of the portfolio after paying transaction costs is given as

$$E[r_p - C(x)] = E\left[\sum_{j=1}^n r_j x_j - \sum_{j=1}^n k_j |x_j - x_j^0|\right]$$

$$= E\left[\sum_{j=1}^n r_j x_j\right] - \sum_{j=1}^n k_j |x_j - x_j^0|$$

$$= \frac{1}{4} \sum_{j=1}^n (a_j + 2b_j + c_j) x_j - \sum_{j=1}^n k_j |x_j - x_j^0|, \tag{22}$$

and the uncertain variance of the portfolio after paying transaction costs is

$$V[r_p - C(x)] = V\left[\sum_{j=1}^n r_j x_j - \sum_{j=1}^n k_j |x_j - x_j^0|\right]$$

$$= V\left[\sum_{j=1}^n r_j x_j\right] \tag{23}$$

$$= \frac{33M^3 + 21M^2m + 11Mm^2 - m^3}{384M}$$

$$= \frac{11M^2}{128} + \frac{7Mm}{128} + \frac{11m^2}{384} - \frac{m^3}{384M},$$

where

$$M = \max\left\{\sum_{j=1}^n (b_j - a_j) x_j, \sum_{j=1}^n (c_j - b_j) x_j\right\}, \tag{24}$$

$$m = \min\left\{\sum_{j=1}^n (b_j - a_j) x_j, \sum_{j=1}^n (c_j - b_j) x_j\right\}.$$

To simplify the portfolio selection model, it seems reasonable to choose the securities in such a way that $b_j - a_j \leq c_j - b_j, j = 1, 2, \dots, n$. Therefore, model (16) can be converted into the following crisp form:

$$\min \frac{11}{128} \left(\sum_{j=1}^n (c_j - b_j) x_j\right)^2 + \frac{7}{128} \left(\sum_{j=1}^n (c_j - b_j) x_j\right)$$

$$\times \left(\sum_{j=1}^n (b_j - a_j) x_j\right) + \frac{11}{384} \left(\sum_{j=1}^n (b_j - a_j) x_j\right)^2$$

$$- \frac{(\sum_{j=1}^n (b_j - a_j) x_j)^3}{384 (\sum_{j=1}^n (c_j - b_j) x_j)}$$

$$\text{s.t. } \frac{1}{4} \sum_{j=1}^n (a_j + 2b_j + c_j) x_j - \sum_{j=1}^n k_j |x_j - x_j^0| \geq \mu_0,$$

$$- \sum_{j=1}^n x_j \ln x_j \geq h_0,$$

$$\sum_{j=1}^n x_j = 1,$$

$$l_j \leq x_j \leq u_j, \quad j = 1, 2, \dots, n,$$

$$x_j \geq 0, \quad j = 1, 2, \dots, n. \tag{25}$$

It should be noted that the proposed uncertain portfolio model (25) is different from the Markowitz M-V model because it is based on the different theories. That is to say, if the security returns are characterized as random variables with probability distributions, we can study portfolio selection problems by probability theory, while if security returns

are regarded as uncertain variables with uncertainty distributions, we can apply uncertainty theory to do researches on the relevant problems.

4. Modified ABC Algorithm

4.1. Standard ABC Algorithm. Artificial bee colony (ABC) algorithm is a new swarm intelligence method which simulates intelligent foraging behavior of honey bees and it is initially proposed by Karaboga [27]. In the ABC algorithm, there are three types of bees: the employed bee, the onlooker bee, and the scout bee. Each of them plays different roles in the process: the employed bees stay on a food source and provide the neighborhood of the source in its memory; the onlooker bee gets the information of food sources from the employed bees in the hive and selects one of the food sources to gather the nectar; and the scout bee is responsible for finding a new food source.

In each successful algorithm, a robust search process, including exploitation and exploration process, must be implemented effectively. In the ABC algorithm, the employed bees and onlookers execute the exploitation process in the search space, while the scout bees execute the exploration process. There is only one employed bee around each food source. In other words, the number of employed bees is equal to the number of food sources around the hive. The positions of food sources represent possible solutions to the optimization problem and the nectar amount of a food source corresponds to the quality or fitness of the associated solution. The number of the employed bees or the onlooker bees is equal to the number of solutions in the population.

At the first step, the ABC generates a randomly distributed initial population P of SN solutions (food source positions), where SN denotes the size of population. Each solution X_i ($i = 1, 2, \dots, SN$) is a D -dimensional vector. Here, D is the number of optimization parameters. After initialization, the population is subjected to repeated cycles of four major steps: updating feasible solutions by employed bees, selection of feasible solutions by onlooker bees, updating feasible solutions by onlooker bees, and avoidance of suboptimal solutions by scout bees.

Updating Feasible Solutions by Employed Bees. In order to produce a new feasible food source from the old one in memory, the ABC uses the following expression:

$$v_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}), \quad (26)$$

where $k \in 1, 2, \dots, SN$ and $j \in 1, 2, \dots, D$ are randomly chosen indexes. Although k is determined randomly, it has to be different from i . ϕ_{ij} is a random number in $[-1, 1]$.

If a new food source V_i is better than an old food source X_i , the old food source is replaced by the new food source. Otherwise, the old one is retained.

Selection of Feasible Solutions by Onlooker Bees. After all employed bees complete the search process, they share the information about their food sources with onlooker bees. An onlooker bee evaluates the nectar information obtained from

- (1) Initialize the population
- (2) set cycle = 1
- (3) **while** cycle \leq MCN **do**
- (4) Update feasible solutions by employed bees
- (5) Select feasible solutions by onlooker bees
- (6) Update feasible solutions by onlooker bees
- (7) Avoid suboptimal solutions by scout bees
- (8) set cycle = cycle + 1
- (9) **end while**

ALGORITHM 1: The framework of the standard ABC algorithm.

all employed bees and chooses a food source depending on the probability value p_i associated with that food source:

$$p_i = \frac{\text{fit}_i}{\sum_{n=1}^{SN} \text{fit}_n}, \quad (27)$$

where fit_i is the fitness value of the solution i which is proportional to nectar amount of the food source in the position i .

Updating Feasible Solutions by Onlooker Bees. Based on the information obtained from the employed bees, the onlooker bees select their feasible food sources. The selected food sources are then updated using (26); that is, an old food source is replaced by a new food source if the new food source is of a better quality.

Avoidance of Suboptimal Solutions by Scout Bees. If a position cannot be improved further through a predetermined number of cycles called limit, then employed bees will abandon the food source positions and become scout bees. This helps avoid suboptimal solutions. Assume that the abandoned source is X_i and $j \in 1, 2, \dots, D$; then the scout discovers a new food source by using the equation below:

$$x_{i,j} = x_{\min}^j + \text{rand}(0, 1)(x_{\max}^j - x_{\min}^j), \quad (28)$$

where j is determined randomly; it should be noticed that it has to be different from i .

The main steps of the standard ABC algorithm are given in Algorithm 1.

4.2. Modified Artificial Bee Colony Algorithm

4.2.1. Chaotic Initialization. Generally speaking, diversity in initial population helps escape from local optima and good-quality initial solutions accelerate convergence speed in an evolutionary algorithm. If no information about the solution is available, then random initialization is the most commonly used method to initialize the population. However, this method may affect the algorithm performance on the convergence speed and the quality of the final solution. Recently, chaotic maps with ergodicity, irregularity, and the stochastic property have been adopted to initialize the population, and some good results have been shown in many applications

```

(1) set population size SN and dimension of solution D
    // Randomly generate the initial value of chaotic variable  $cx_{i,j}$ 
(2) for  $j = 1$  to  $D$  do
(3)    $cx_{1,j} = \text{rand}(0, 1)$ 
(4)   while  $cx_{1,j} \in \{0, 0.25, 0.5, 0.75, 1\}$  do
(5)      $cx_{1,j} = \text{rand}(0, 1)$ 
(6)   end while
(7) end for
    // The iterative process of chaotic initialization
(8) for  $i = 1$  to SN do
(9)   for  $j = 1$  to  $D$  do
(10)     $x_{i,j} = x_{\min}^j + cx_{i,j} (x_{\max}^j - x_{\min}^j)$ 
(11)     $cx_{i+1,j} = 4cx_{i,j}(1 - cx_{i,j})$ 
(12)   end for
(13) end for
    
```

ALGORITHM 2: Chaotic initialization.

[58, 59]. In this paper, chaos-based initialization method is employed to increase the population diversity and improve the global convergence.

A simplest chaotic map is logistic map, whose equation is the following:

$$x_{n+1} = \mu x_n (1 - x_n), \tag{29}$$

where μ is a control parameter and x_n is the n th chaotic number, where n denotes the iteration number. Obviously, $x_n \in (0, 1)$ under the conditions that the initial $x_0 \in (0, 1)$. In particular, x_n behaves as chaotic dynamics when $\mu = 4$ and $x_0 \notin \{0, 0.25, 0.5, 0.75, 1\}$.

The pseudocode of the proposed chaotic initialization is given in Algorithm 2.

4.2.2. *New Search Mechanism.* The employed bee and onlooker bee produce a new feasible food source according to (26). However, the convergence performance of the algorithm is not good in some cases. Therefore, we proposed a modified search method to produce new solutions, in which forgetting factor and neighborhood factor are considered. This operation can be defined as follows:

$$v_{i,j} = \varphi x_{i,j} + \psi \phi_{ij} (x_{i,j} - x_{k,j}), \tag{30}$$

where φ is the forgetting factor, which expresses the memory strength for current food source, and it is decreased gradually. In addition, the quality of the neighborhood food source X_k also affects the convergence speed and quality of the final solution. Thus, the neighborhood factor ψ is introduced to accelerate the convergence speed by adjusting the radius of the search for new candidates. The φ and ψ are defined as follows:

$$\begin{aligned} \varphi &= \omega_{\max} - \frac{\text{iteration}}{\text{MCN}} (\omega_{\max} - \omega_{\min}), \\ \psi &= \omega_{\min} + \frac{\text{iteration}}{\text{MCN}} (\omega_{\max} - \omega_{\min}), \end{aligned} \tag{31}$$

where the values of ω_{\max} and ω_{\min} represent the maximum and minimum percentage of the position adjustment for the

TABLE 1: The zigzag uncertain returns of 8 securities.

Security	1	2	3	4	5	6	7	8
a_j	0.006	-0.017	0.085	0.050	0.163	-0.035	0.095	0.116
b_j	0.011	-0.035	0.103	0.062	0.172	-0.082	0.161	0.235
c_j	0.028	0.013	0.128	0.095	0.193	0.022	0.232	0.527
x_j^0	0.05	0.07	0.15	0.06	0.15	0.22	0.13	0.17
k_j	0.002	0.005	0.003	0.001	0.004	0.002	0.005	0.003
l_j	0.1	0.05	0.09	0.06	0.07	0.05	0.06	0.06
u_j	0.35	0.35	0.35	0.35	0.35	0.35	0.35	0.35

employed bees or onlooker bees. With these selected values, the value of φ will linearly decrease and the value of ψ will linearly increase.

Moreover, in the scout bee phase, if the food source is abandoned, then the scouts produce a new food source by (28). However, to some extent, it has the defects of prematurity and stagnation. Therefore, the chaotic search technique is used to get out of the local optima and is specifically illustrated in Algorithm 3.

4.2.3. *The Proposed Method.* Based on the above discussions, the modified ABC algorithm can be well balanced between the exploration and exploitation. The pseudocode of the proposed algorithm is given in Algorithm 4.

5. Numerical Example

In this section, we give a numerical example to illustrate the application of the uncertain mean-variance-entropy model with transaction costs and the effectiveness of the proposed algorithm. In the example, security returns cannot be well reflected by the historical data and are given by experts' evaluations. The investor wants to select portfolios from 8 individual securities. Thus, we assume that the return of security j is a zigzag uncertain variable $r_j \sim \mathcal{Z}(a_j, b_j, c_j)$, $j = 1, 2, \dots, 8$. The uncertainty distributions of security returns are given in Table 1. And other related portfolio parameters are also listed in Table 1.

```

(1) Set the maximum number of chaotic iteration  $K = 100$  and iterative variable  $k = 1$ 
    // The variable  $cx_{k,j}$  denotes the  $j$ th chaotic variable in the  $k$ th iteration
(2) for  $j = 1$  to  $D$  do
(3)    $cx_{k,j} = (x_{i,j} - x_{\min}^j) / (x_{\max}^j - x_{\min}^j)$ 
(4) end for
(5) while  $k < K$  do
(6)   for  $j = 1$  to  $D$  do
(7)      $cx_{k+1,j} = 4cx_{k,j}(1 - cx_{k,j})$ 
        //  $V = [v_1, \dots, v_D]$  represents a candidate solution
(8)      $v_j = x_{\min}^j + cx_{k+1,j}(x_{\max}^j - x_{\min}^j)$ 
(9)   end for
        // Greedy selection is applied between  $\{V, X_i\}$ 
(10)  if  $f(V) > f(x_i)$ 
(11)    Replace solution  $X_i$  with candidate solution  $V$ 
(12)    Set  $trial = 0$ 
(13)    Break
(14)  else
(15)    Set  $trial(i) = trial(i) + 1$ 
(16)  end if
(17)  Set  $k = k + 1$ 
(18) end while

```

ALGORITHM 3: Chaotic search for scout bees.

```

(1) Set population size SN, the number of maximum cycles MCN and the control parameter limit
(2) Perform Algorithm 2 to fulfill the chaotic initialization
(3) while iteration  $\leq$  MCN do
    // The employed bees phase
(4) for  $i = 1$  to SN do
(5)   Produce a new candidate food source  $V_i$  corresponding to food source  $X_i$  using (29)
(6)   if  $(f(V_i) > f(X_i))$  then  $trial(i) = 0$ 
(7)   else  $trial(i) = trial(i) + 1$ 
(8)   end if
(9) end for
(10) Calculate the fitness values of all food source and the probability values  $p_i$  by using (27)
    // The onlooker bees phase
(11) Set  $t = 0, i = 1$ 
(12) while  $t < SN$  do
(13)   if  $random() < p_i$  then
(14)     Set  $t = t + 1$ 
(15)     Produce a new candidate food source  $V_i$  for the onlooker bee
(16)     if  $f(V_i) > f(X_i)$  then  $trial(i) = 0$ 
(17)     else  $trial(i) = trial(i) + 1$ 
(18)     end if
(19)     Set  $i = i + 1$ 
(20)   if  $i > SN$  then Set  $i = 1$ 
(21)   end if
(22) end while
    // The scout bees phase
(23) for  $i = 1$  to SN do
(24)   if  $trial(i) \geq SN$  then
(25)     Perform Algorithm 3 to implement chaotic search
(26)   end if
(27) end for
(28) Set  $iteration = iteration + 1$ 
(29) end while

```

ALGORITHM 4: Modified ABC algorithm.

TABLE 2: The efficient portfolios with $h_0 = 1.8$.

Security	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	Risk
$\mu_0 = 0.07$	0.349	0.050	0.137	0.072	0.222	0.050	0.060	0.060	0.0002236
$\mu_0 = 0.09$	0.309	0.050	0.090	0.098	0.283	0.050	0.060	0.060	0.0002255
$\mu_0 = 0.11$	0.136	0.063	0.212	0.068	0.350	0.050	0.060	0.060	0.0002406

TABLE 3: The efficient portfolios under different transaction costs ($\mu_0 = 0.08, h_0 = 1.7$).

Security	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	TC
$k_5 = 0$	0.341	0.050	0.090	0.060	0.289	0.050	0.060	0.060	0.00188
$k_5 = 0.004$	0.350	0.050	0.102	0.060	0.268	0.050	0.060	0.060	0.00234
$k_5 = 0.008$	0.332	0.050	0.096	0.060	0.292	0.050	0.060	0.060	0.00298
$k_5 = 0.015$	0.322	0.050	0.105	0.060	0.293	0.050	0.060	0.060	0.00394

TABLE 4: Robustness analysis for the modified ABC algorithm ($u_0 = 0.08, h_0 = 1.8$).

SN	MCN	Limit	Objective value	Relative error (%)
10	1500	50	$2.2474E - 4$	0
20	2000	100	$2.2445E - 4$	0.13
20	1500	150	$2.2424E - 4$	0.22
30	2500	150	$2.2469E - 4$	0.02
30	2000	100	$2.2369E - 4$	0.47
40	3000	200	$2.2384E - 4$	0.4
40	1500	100	$2.2406E - 4$	0.3
50	1500	100	$2.2372E - 4$	0.45
60	2000	150	$2.2371E - 4$	0.46
70	2000	50	$2.2358E - 4$	0.52

TABLE 5: The comparison of the modified ABC, ABC, and GA ($u_0 = 0.09, h_0 = 1.75$).

	Modified ABC	ABC	GA
x_1	0.327	0.319	0.311
x_2	0.050	0.050	0.061
x_3	0.090	0.132	0.158
x_4	0.060	0.060	0.060
x_5	0.303	0.269	0.240
x_6	0.050	0.050	0.050
x_7	0.060	0.060	0.060
x_8	0.060	0.060	0.060
Risk	$2.203E - 4$	$2.234E - 4$	$2.270E - 4$

To solve the proposed model (25) by the modified ABC algorithm, we let the number of food sources $SN = 20$, the dimension of the problem $D = 8$, maximum cycle number $MCN = 1000$, and the control parameter limit = 100. Moreover, in this section, a total of 10 runs for each experimental setting are conducted and the average results are given. The values of ω_{max} and ω_{min} are fixed to 1 and 0.2, respectively.

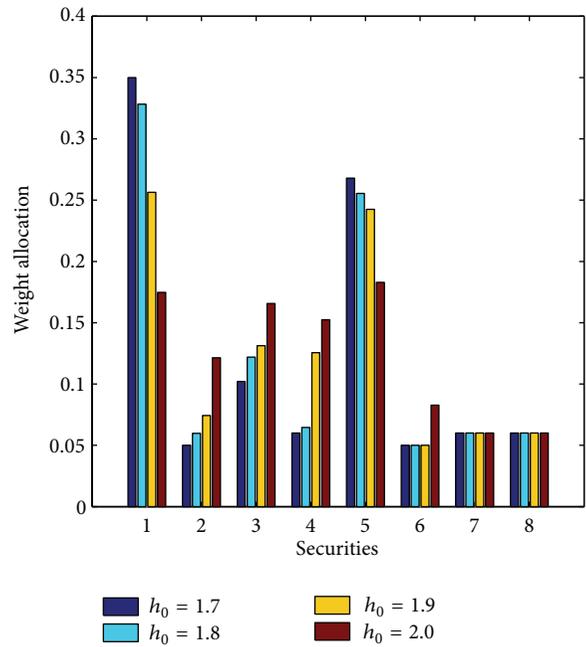


FIGURE 1: Comparison results under different diversification degrees.

Suppose that the minimum entropy level $h_0 = 1.8$ and the efficient portfolios under different expected returns are shown in Table 2. If the investor is not satisfied with any of the portfolios obtained, more portfolios can be obtained by varying the value of μ . From Table 2, we can see that the larger the return of portfolio is, the larger the risk of portfolio is. There is a tradeoff between the return and the risk. Moreover, for the different expected return μ , the optimal portfolios are also different.

Next, to demonstrate that the transaction costs have effect on the optimal portfolio selection, given $h_0 = 1.7$ and $\mu = 0.08$, the efficient portfolios under different transaction costs are given in Table 3. It should be noted that, for simplicity, we only vary the transaction cost of security 5. We can see

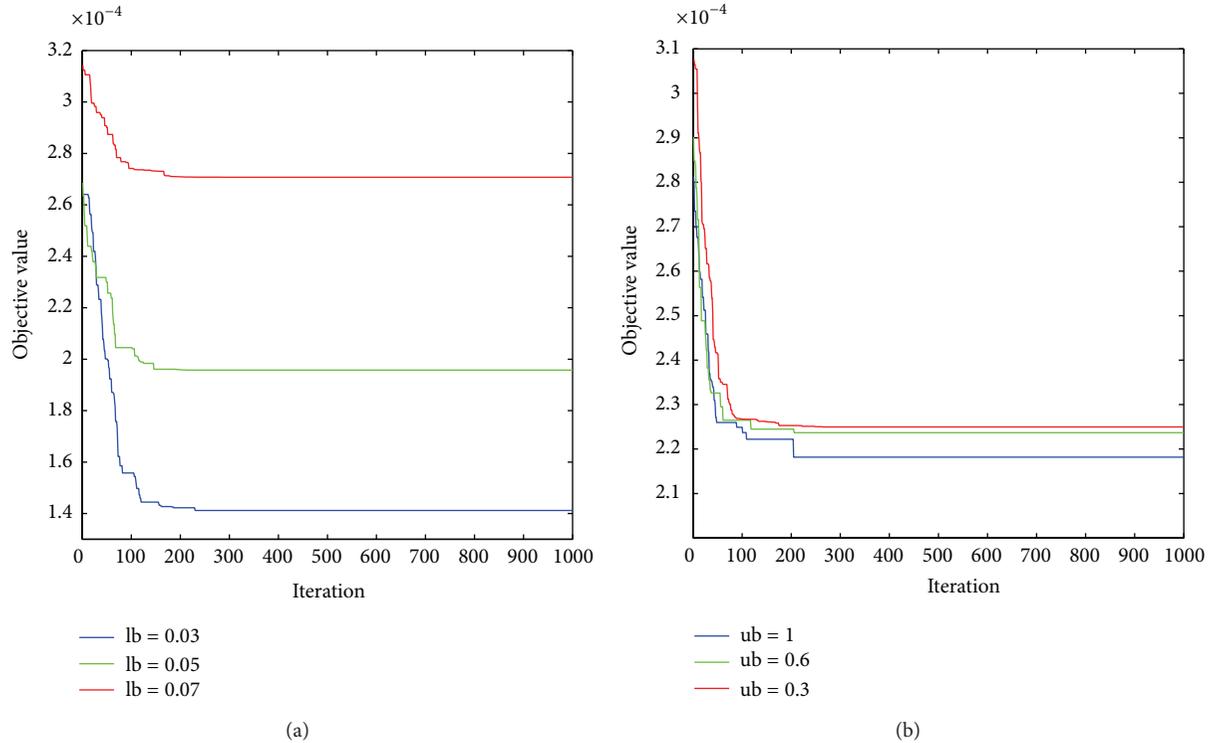


FIGURE 2: The performance of the ABC algorithm under different bound constraints.

that although the proportions of securities 2, 4, and 6–8 are the same, the holdings of other securities are different. Simultaneously, we also see that the total transaction cost also increases.

Figure 1 gives comparison results of the optimal portfolio strategies under different diversification degrees; that is, $h_0 = 1.7$, $h_0 = 1.8$, $h_0 = 1.9$, and $h_0 = 2.0$. The result indicates that, as the diversification degree of the portfolio changes, the investors adopt different investment strategies.

Figure 2 presents the performance of the ABC algorithm with respect to the choice of the lower and upper bounds. Moreover, to show the robustness of the proposed algorithm, we use relative error as the index; that is, $(\text{maximum} - \text{actual value}) / \text{maximum} \times 100\%$, where the maximum is the maximal value of all the objective values calculated. The detailed results are shown in Table 4. Obviously, the relative errors do not exceed 1%. That is, the proposed algorithm is robust to set parameters.

Table 5 presents a comparison of the modified ABC, ABC, and GA. From Table 5, it can be observed that, for the $u_0 = 0.09$ and $h_0 = 1.75$, the investment strategy obtained by modified ABC algorithm outperforms those produced by standard ABC algorithm and GA, because the global optimal value obtained by modified ABC algorithm is minimal. In addition, Figure 3 shows the convergence characteristic of modified ABC, ABC, and GA. Generally speaking, the modified ABC algorithm converges within 200 iterations, while ABC and GA algorithms converge within 400 and 430 iterations, respectively. All these results show

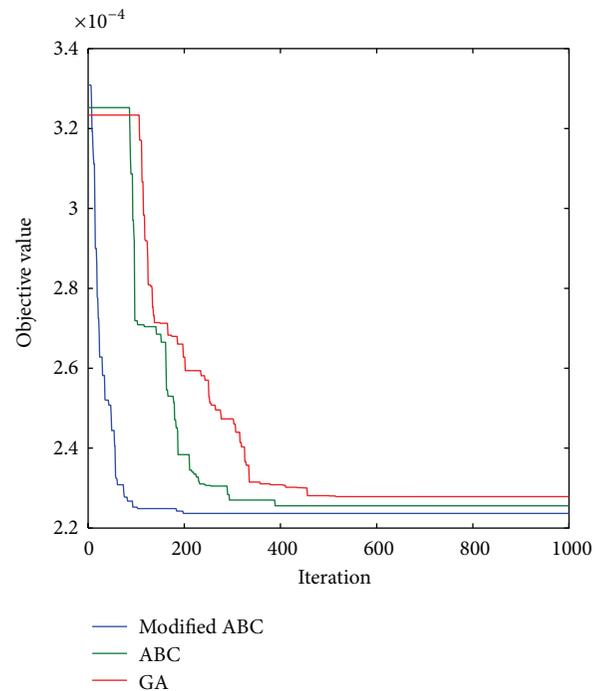


FIGURE 3: Convergence characteristics of the modified ABC, ABC, and GA ($u_0 = 0.08$, $h_0 = 1.8$).

that the modified ABC algorithm has a better performance and is efficient in finding optimal portfolios.

6. Conclusion

Since the security market is so complex, sometimes security returns have to be predicted mainly by experts' evaluations rather than historical data. This paper discusses a portfolio selection problem with returns given by experts' evaluations. By taking the security returns as uncertain variables, this paper makes use of the uncertain expected value and uncertain variance to measure the return and risk of securities. Furthermore, we propose an uncertain mean-variance-entropy model for portfolio selection with transaction costs. After that, a modified ABC algorithm is developed to solve the proposed problem. Finally, a numerical example is presented to illustrate this modeling concept and to demonstrate the effectiveness of the proposed algorithm. The results show that the proposed method is applicable and effective for solving the portfolio selection problem.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The author thanks Professor Zhijun Wu for his valuable comments and suggestions. This research was supported by the Humanity and Social Science Youth Foundation of Ministry of Education of China (no. 13YJC630012), the National Natural Science Foundation of China (nos. 71240002 and 71102150), the Beijing Natural Science Foundation of China (no. 9142003), and the Beijing Philosophical Social Science Project of China (no. 13SHB015).

References

- [1] H. Markowitz, "Portfolio selection," *The Journal of Finance*, vol. 7, no. 1, pp. 77–91, 1952.
- [2] W. Sharp, "A simplified model for portfolio analysis," *Management Science*, vol. 9, pp. 277–293, 1963.
- [3] R. C. Merton, "An analytic derivation of the efficient frontier," *Journal of Finance and Quantitative Analysis*, vol. 10, pp. 1851–1872, 1972.
- [4] J. S. Pang, "A new and efficient algorithm for a class of portfolio selection problems," *Operations Research*, vol. 28, no. 3, part 2, pp. 754–767, 1980.
- [5] A. F. Perold, "Large-scale portfolio optimization," *Management Science*, vol. 30, no. 10, pp. 1143–1160, 1984.
- [6] M. J. Best and R. R. Grauer, "The efficient set mathematics when mean-variance problems are subject to general linear constraints," *Journal of Economics and Business*, vol. 42, no. 2, pp. 105–120, 1990.
- [7] H. Konno and H. Yamazaki, "Mean absolute deviation portfolio optimization model and its application to Tokyo stock exchange," *Management Science*, vol. 37, pp. 519–531, 1991.
- [8] M. J. Best and J. Hlouskova, "The efficient frontier for bounded assets," *Mathematical Methods of Operations Research*, vol. 52, no. 2, pp. 195–212, 2000.
- [9] L. A. Zadeh, "Fuzzy sets," *Information and Computation*, vol. 8, pp. 338–353, 1965.
- [10] H. Tanaka and P. Guo, "Portfolio selection based on upper and lower exponential possibility distributions," *European Journal of Operational Research*, vol. 114, no. 1, pp. 115–126, 1999.
- [11] C. Carlsson, R. Fullér, and P. Majlender, "A possibilistic approach to selecting portfolios with highest utility score," *Fuzzy Sets and Systems*, vol. 131, no. 1, pp. 13–21, 2002.
- [12] E. Vercher, J. D. Bermúdez, and J. V. Segura, "Fuzzy portfolio optimization under downside risk measures," *Fuzzy Sets and Systems*, vol. 1, pp. 361–377, 2007.
- [13] W. G. Zhang, Y. L. Wang, Z. P. Chen, and Z. K. Nie, "Possibilistic mean-variance models and efficient frontiers for portfolio selection problem," *Information Sciences*, vol. 177, no. 13, pp. 2787–2801, 2007.
- [14] W. Chen, Y. Yang, and H. Ma, "Fuzzy portfolio selection problem with different borrowing and lending rates," *Mathematical Problems in Engineering*, vol. 2011, Article ID 263240, 15 pages, 2011.
- [15] R. Tsaur, "Fuzzy portfolio model with different investor risk attitudes," *European Journal of Operational Research*, vol. 227, no. 2, pp. 385–390, 2013.
- [16] P. Gupta, G. Mittal, and M. K. Mehlaawat, "Multiobjective expected value model for portfolio selection in fuzzy environment," *Optimization Letters*, vol. 7, no. 8, pp. 1765–1791, 2013.
- [17] B. Liu, *Uncertainty Theory*, vol. 154 of *Studies in Fuzziness and Soft Computing*, Springer, Berlin, Germany, 2nd edition, 2007.
- [18] Z. X. Peng and K. Iwamura, "A sufficient and necessary condition of uncertainty distribution," *Journal of Interdisciplinary Mathematics*, vol. 13, no. 3, pp. 277–285, 2010.
- [19] X. W. Chen and B. Liu, "Existence and uniqueness theorem for uncertain differential equations," *Fuzzy Optimization and Decision Making*, vol. 9, no. 1, pp. 69–81, 2010.
- [20] Y. Gao, "Shortest path problem with uncertain arc lengths," *Computers & Mathematics with Applications*, vol. 62, no. 6, pp. 2591–2600, 2011.
- [21] S. Ding and Y. Gao, "The (σ, S) policy for uncertain multi-product newsboy problem," *Expert Systems with Applications*, vol. 41, pp. 3769–3776, 2014.
- [22] Y. Zhu, "Uncertain optimal control with application to a portfolio selection model," *Cybernetics and Systems*, vol. 41, no. 7, pp. 535–547, 2010.
- [23] Y. Liu and Z. Qin, "Mean semi-absolute deviation model for uncertain portfolio optimization problem," *Journal of Uncertain Systems*, vol. 6, no. 4, pp. 299–307, 2012.
- [24] X. Huang, "Mean-variance models for portfolio selection subject to experts' estimations," *Expert Systems with Applications*, vol. 39, no. 5, pp. 5887–5893, 2012.
- [25] X. Huang, "A risk index model for portfolio selection with returns subject to experts' estimations," *Fuzzy Optimization and Decision Making*, vol. 11, no. 4, pp. 451–463, 2012.
- [26] X. Huang, *Portfolio Analysis: From Probabilistic to Credibilistic and Uncertain Approaches*, Springer, Berlin, Germany, 2010.
- [27] D. Karaboga, "An idea based on honeybee swarm for numerical optimization," Tech. Rep. TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.
- [28] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.
- [29] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 687–697, 2008.

- [30] D. Karaboga and B. Akay, "A comparative study of artificial Bee colony algorithm," *Applied Mathematics and Computation*, vol. 214, no. 1, pp. 108–132, 2009.
- [31] D. Karaboga and B. Akay, "A modified Artificial Bee Colony (ABC) algorithm for constrained optimization problems," *Applied Soft Computing Journal*, vol. 11, no. 3, pp. 3021–3031, 2011.
- [32] K. Ziarati, R. Akbari, and V. Zeighami, "On the performance of bee algorithms for resource-constrained project scheduling problem," *Applied Soft Computing Journal*, vol. 11, no. 4, pp. 3720–3733, 2011.
- [33] W. Yeh and T. Hsieh, "Solving reliability redundancy allocation problems using an artificial bee colony algorithm," *Computers & Operations Research*, vol. 38, no. 11, pp. 1465–1473, 2011.
- [34] A. Banharnsakun, B. Sirinaovakul, and T. Achalakul, "Job shop scheduling with the Best-so-far ABC," *Engineering Applications of Artificial Intelligence*, vol. 25, no. 3, pp. 583–593, 2012.
- [35] B. Akay and D. Karaboga, "Artificial bee colony algorithm for large-scale problems and engineering design optimization," *Journal of Intelligent Manufacturing*, vol. 23, no. 4, pp. 1001–1014, 2012.
- [36] D. Karaboga and B. Gorkemli, "A combinatorial Artificial Bee Colony algorithm for traveling salesman problem," in *Proceedings of the International Symposium on Innovations in Intelligent Systems and Applications (INISTA '11)*, pp. 50–53, Istanbul, Turkey, June 2011.
- [37] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga, "A comprehensive survey: artificial bee colony (ABC) algorithm and applications," *Artificial Intelligence Review*, vol. 42, pp. 21–57, 2014.
- [38] B. Liu, "Why is there a need for uncertainty theory?" *Journal of Uncertain Systems*, vol. 6, no. 1, pp. 3–10, 2012.
- [39] B. Liu, *Uncertainty Theory: A Branch of Mathematics for Modeling Human Uncertainty*, Springer, Berlin, Germany, 2010.
- [40] R. D. Arnott and W. H. Wanger, "The measurement and control of trading costs," *Financial Analysts Journal*, vol. 46, no. 6, pp. 73–80, 1990.
- [41] S. J. Sadjadi, M. B. Aryanezhad, and B. F. Moghaddam, "A dynamic programming approach to solve efficient frontier," *Mathematical Methods of Operations Research*, vol. 60, no. 2, pp. 203–214, 2004.
- [42] W. Chen and W. G. Zhang, "The admissible portfolio selection problem with transaction costs and an improved PSO algorithm," *Physica A: Statistical Mechanics and its Applications*, vol. 389, no. 10, pp. 2070–2076, 2010.
- [43] S. Barak, M. Abessi, and M. Modarres, "Fuzzy turnover rate chance constraints portfolio model," *European Journal of Operational Research*, vol. 228, no. 1, pp. 141–147, 2013.
- [44] P. Jana, T. K. Roy, and S. K. Mazumder, "Multi-objective possibilistic model for portfolio selection with transaction cost," *Journal of Computational and Applied Mathematics*, vol. 228, no. 1, pp. 188–196, 2009.
- [45] S. Fang, J. R. Rajasekera, and H. J. Tsao, *Entropy Optimization and Mathematical Programming*, Kluwer Academic, 1997.
- [46] J. N. Kapur, *Maximum Entropy Models in Science and Engineering*, Wiley Eastern Limited, New Delhi, India, 1990.
- [47] W. Zhang, Y. Liu, and W. Xu, "A possibilistic mean-semi-variance-entropy model for multi-period portfolio selection with transaction costs," *European Journal of Operational Research*, vol. 222, no. 2, pp. 341–349, 2012.
- [48] C. E. Shannon, *The Mathematical Theory of Communication*, The University of Illinois Press, Urbana, Ill, USA, 1949.
- [49] L. A. Zadeh, "Optimality and non-scalar-valued performance criteria," *IEEE Transaction on Automation Control*, vol. 8, no. 1, pp. 59–60, 1963.
- [50] J. R. Rao and N. Roy, "Fuzzy set-theoretic approach of assigning weights to objectives in multicriteria decision making," *International Journal of Systems Science. Principles and Applications of Systems and Integration*, vol. 20, no. 8, pp. 1381–1386, 1989.
- [51] R. T. Marler and J. S. Arora, "The weighted sum method for multi-objective optimization: new insights," *Structural and Multidisciplinary Optimization*, vol. 41, no. 6, pp. 853–862, 2010.
- [52] S. Marglin, *Public Investment Criteria*, MIT Press, Cambridge, Mass, USA, 1967.
- [53] M. Ehrgott and S. Ruzika, "Improved ϵ -constraint method for multiobjective programming," *Journal of Optimization Theory and Applications*, vol. 138, no. 3, pp. 375–396, 2008.
- [54] J. L. Cohon, *Multiobjective Programming and Planning*, vol. 140, Academic Press, New York, NY, USA, 1978.
- [55] V. Chankong and Y. Y. Haimes, *Multiobjective Decision Making Theory and Methodology*, Elsevier, New York, NY, USA, 1983.
- [56] H. K. Singh, T. Ray, and W. Smith, "C-PSA: constrained Pareto simulated annealing for constrained multi-objective optimization," *Information Sciences*, vol. 180, no. 13, pp. 2499–2513, 2010.
- [57] M. K. Luhandjula and M. J. Rangoaga, "An approach to solving a fuzzy multiobjective programming problem," *European Journal of Operational Research*, vol. 232, no. 2, pp. 249–255, 2014.
- [58] B. Alatas, "Chaotic bee colony algorithms for global numerical optimization," *Expert Systems with Applications*, vol. 37, no. 8, pp. 5682–5687, 2010.
- [59] C. H. Yang, S. W. Tsai, and L. Y. Chuang, "An improved particle swarm optimization with double-bottom chaotic maps for numerical optimization," *Applied Mathematics and Computation*, vol. 219, no. 1, pp. 260–279, 2012.

Research Article

Fault Detection of Aircraft System with Random Forest Algorithm and Similarity Measure

Sanghyuk Lee,¹ Wookje Park,² and Sikhang Jung³

¹ Department of Electrical and Electronic Engineering, Xi'an Jiaotong-Liverpool University, Suzhou 215123, China

² Aviation Technical Division, Aviation Safety Technology Center, 557 Yongyu-ro, Jung-gu, Incheon 400-420, Republic of Korea

³ Department of Aerospace, Automobile & Mechanical Engineering, Chung Cheong University, 38 Wolgok-Gil, Gangnae-Myeon, Cheongwon-Gun, Chungcheongbuk-Do 363-792, Republic of Korea

Correspondence should be addressed to Wookje Park; parkwookje@gmail.com

Received 4 April 2014; Accepted 10 June 2014; Published 26 June 2014

Academic Editor: T. O. Ting

Copyright © 2014 Sanghyuk Lee et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Research on fault detection algorithm was developed with the similarity measure and random forest algorithm. The organized algorithm was applied to unmanned aircraft vehicle (UAV) that was readied by us. Similarity measure was designed by the help of distance information, and its usefulness was also verified by proof. Fault decision was carried out by calculation of weighted similarity measure. Twelve available coefficients among healthy and faulty status data group were used to determine the decision. Similarity measure weighting was done and obtained through random forest algorithm (RFA); RF provides data priority. In order to get a fast response of decision, a limited number of coefficients was also considered. Relation of detection rate and amount of feature data were analyzed and illustrated. By repeated trial of similarity calculation, useful data amount was obtained.

1. Introduction

Fault-tolerance and adaptation of aircraft system with actual faults/healthy data have been studied. In order to process the adaptation of the pilot or the flight control system under abnormal condition, critical mission or return to a safe region should be followed with clear and right decision [1]. First of all, data for fault decision should be available; data acquisition system is required to take data accurately and quickly for the guarantee of precise fault decision. A lightweight fault detection model of DURUMI-II, which is an unmanned aerial vehicle (UAV), was considered and fault decision process was carried out with random forests algorithm (RFA) [2]. RFA is a state-of-the-art classification algorithm and has shown high classification accuracy. Additionally, RFA generates priority for each feature. The proposed approach enables one to figure out stable, important features with high detection rates. As a result, parameter optimization and feature selection were performed to make guaranteed high detection rates.

In order to get fault detection result, discrimination measure has to be considered. Similarity measure [3–6] represents the degree of similarity between comparable data; it has also been done by numerous works [1, 2, 7–9]. Similarity measures between two vague datasets mean that it roughly depends on inverse value of distance. Therefore, similarity can be considered as common information between two data distributions; hence, the obtained similarity measure was based upon the distance measure. Hence, the computation of similarity between two fuzzy sets could be followed with the obtained similarity measure.

12 cases operating data are used to design discriminating measure for healthy/faults condition. Hence, it is represented as multivariate dataset; it is also understood as multidimensional data. Each component has different importance to determine either faulty or healthy. To get more accurate decision, it is needed to consider weighting factor to each 12 coefficients. Depending on expert's opinion could be a strong candidate to solve the problem. It seems, like the heuristic approach, results depend on individual. It means

that results cannot guarantee the consistency. RFA provides important values with respect to each feature. It means that more rationale can be obtained through RFA.

In the previous research [1, 10, 11], fault detection and isolation (FDI) operation were accompanied by the fault-tolerant control system to control process failure. With the assumption of the controllability and the trimmability of the UAV at postfailure conditions, it means that the aircraft can keep on flying with the help of 6 flight control computer even the control surface stuck happen. It makes restructure and reconfigure controllers according to the grade of system failure. To get a decision or analysis for faults, statistical information of elevator deflection, aileron deflection, and others has been analyzed.

By applying RFA to multivariate similarity measure, different weighting factors were considered. Detection results showed good performance with actual data. Specially, when data was placed within overlapping region, RFA showed satisfactory performance compared to unitary weighting case [7, 8]. In the next chapter, RFA was introduced. Normal operation and longitudinal faulty operation were performed, and operation data were obtained. Considered airplane model and state equation were also illustrated in Chapter 3. RFA procedure was introduced in Chapter 4, and the importance value was obtained through RFA. Computation result was compared with similarity measure calculation. Similarity was also weighted with importance variable. Finally, conclusions are derived.

2. Random Forests Algorithm

A lot of interest in ensemble learning algorithm generated many results about classifier and regression by way of boosting and bagging [2, 7, 8, 12–14]. Random forests algorithm was proposed by Breiman [2], the method is categorized under “ensemble learning” method, and it adds an additional layer of randomness to bagging. Applying ensembles of trees can achieve important gains in classification and regression accuracy and each tree in the ensemble is developed according to the random parameter. Applying an injection of randomness, each of these trees is generated. Dietterich proposed random split selection approach, where the split was selected randomly from the K best splits at each node [8].

The common element in all RFA procedures is multiple trees. Random vector Θ_k is generated from past random vectors $\Theta_1, \dots, \Theta_{k-1}$, each has the same distribution independently. By the derivation of Breiman, random vector Θ is generated as the counts in N boxes. It is resulting from N darts thrown at random at the boxes, where N is the number of examples in the training set. Then, Θ consists of a number of independent random integers between 1 and K . A tree is also grown with the training set and Θ_k and resulting in a classifier $h(\mathbf{x}, \Theta_k)$ where \mathbf{x} is an input vector [2]. With the definition of Breiman, RFA is defined in Definition 1.

Definition 1 (see [2]). A random forest is a classifier consisting of a collection of tree-structured classifiers $\{h(\mathbf{x}, \Theta_k), k = 1, \dots\}$ where the $\{\Theta_k\}$ are independent identically distributed

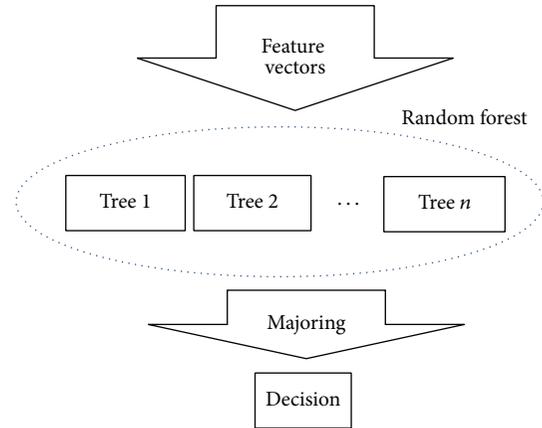


FIGURE 1: Structure of RFA.

random vectors and each tree casts a unit vote for the most popular class at input \mathbf{x} .

With the property of Definition 1, two missions are needed to design RFA, one is a tree structure and the other is the number of trees. Tree structure has no rule for the design; it depends on computational requirement and designer himself/herself. The number of trees should be decided through heuristic consideration such as by using trial and error. Well-known tree algorithm is considered in reference [12].

Total structure of RFA is shown in Figure 1. Trees are repeated n times. As in reference [12], RFA has several advantages and disadvantages. It is one of the effective learning algorithms, because it is convenient to process large database, and possible to handle many input variables. However, it often shows overfit for some datasets, and not easy to interpret, and others. Even some uncomfortable RFA results provide useful information such as variable importance, because decision result could be obtained through probability.

Majoring could be obtained by averaging each class result. By overlapping the results of each tree decision, it also provides ensemble mean of each decision result. Effect of deleting uncertainty was done by averaging node outputs. Furthermore, if feature vectors have wide characteristics, then it is obviously guarantee ergodicity.

3. Model for Similarity Measure

3.1. Airplane Model. Consider the aircraft system of combining with longitudinal mode and lateral-directional mode. Then, the state space equation is as follows [1, 10, 11, 15–17]:

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \\ \mathbf{y}(t) &= \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t), \end{aligned} \quad (1)$$

TABLE 1: Elevator trim value in longitudinal mode (normal mode elevator trim value: 5.752).

		Elevator stuck angle				
		-10°	-5°	0°	+5°	+10°
Rudder stuck angle	-10°	—	—	-5.098	-10.008	—
	-5°	—	11.218	1.006	-9.150	—
	0°	—	10.173	5.796	-1.796	—
	5°	—	—	8.880	3.076	—
	10°	—	—	14.719	—	—
Elevator trim value (elevator stuck only)		12.640	10.173	5.796	-1.796	-9.898



FIGURE 2: The UAV configuration.

where \mathbf{x} and \mathbf{u} denote state vector and elevator control input variable, respectively. Output vector \mathbf{y} is identified by \mathbf{x} itself [11, 15–17]. Consider

$$\begin{aligned} \mathbf{x} &= [\alpha \ u \ q \ \theta \ \beta \ p \ \gamma \ \phi]^T, \\ \mathbf{u} &= \delta_e, \end{aligned} \quad (2)$$

where $\mathbf{u} = \delta_e$ is elevator control input. In the state vector, p , q , and γ are the angular velocities and α and β are the angle of attack and the sideslip angle. Finally, ϕ and θ represent the roll and pitch angle, respectively.

In Figure 2, an experimental model consisted of one elevator, one rudder, and two ailerons. In order to get two types of data, normal and fault, two times of data acquisitions were carried out. It is notified that one or two of elevator, aileron, and rudder were broken intentionally. It was made after taking normal operating data. In order to simulate our fault detection procedure, the experimental model also has been equipped with one-piece elevator. It was separated into two—one is normal and the other is faulty. Hence, it was difficult to know whether it was faulty or not. Therefore, control surface has been added at the other vertical stabilizer. Considered UAV is illustrated in Figure 2; fault was applied artificially. Two different flight tests were carried out for normal and faulty operations separately [1, 10, 11].

In order to carry out the experiment, intentional damages were applied to the right elevator, the left rudder, and the left aileron stuck and the combination of this surface stuck was considered. Without the uncontrollability and the untrimmability of the aircraft at postfailure conditions, the flight test was scheduled. For left rudder only and right elevator with the left rudder stuck cases were considered. Stuck angles of the first case (left rudder only) were considered from -10° , -5° , 0° ,

and $+5^\circ$, to $+10^\circ$. Same stuck angles of the second case (right elevator with the left rudder stuck) were also considered.

In the first flight test, the control input for the real-time parameter estimation was applied with the knob and switching method, and the flight data was obtained from the exciting dynamics of UAV operation with the mentioned method [1, 10, 14]. However, as it was pointed out in the result, result showed that the applied time interval was slightly inaccurate. So, in the second flight test, for the purpose of constant control realization and the time interval, the control input device was developed to use an RF modem and a R/C transmitter [1, 10, 11].

3.2. Parameters of Longitudinal Mode. In order to control UAV during the occurrence of surface stuck and combination stuck, the aircraft should possess the controllability and the trimmability under the postfailure conditions. Because flight test procedure contains the ability to recover to the normal state from the fault state [1, 10, 11], Table 1 shows that the experimental results of UAV (DURUMI-II) show the trim value of available primary control surface at postfailure conditions. Blank was considered as the uncontrollability and the untrimmability cases.

Research on longitudinal mode fault detection was carried out in the previous research [10]. In order to obtain the failure status of the elevator, analyses of $C_{m_{\delta_e}}$, $C_{m_{\alpha}}$, and $C_{L_{\alpha}}$ were essential to obtain information on the aircraft performance characteristics. In the research, instead of statistical information such as mean and variance similarity measure was proposed to overcome the ambiguity of big standard deviation. Because it invokes scattering results, the analysis results of the stability and controllability derivatives are not clear.

4. Numerical Results and Their Analysis

Fault detection algorithm was proposed in this chapter. Total of 89 data were considered, 38 normal and 51 fault. Twelve features were also included within dataset.

4.1. Random Forest with Numerical Interpolation. Considering RFA, the number of variables in the random subset at each node (m_{try}) and the number of trees in the forest (n_{tree}) are needed. In order to get the best classification rate (correction decision rate), determination of optimal number of two parameters is required.

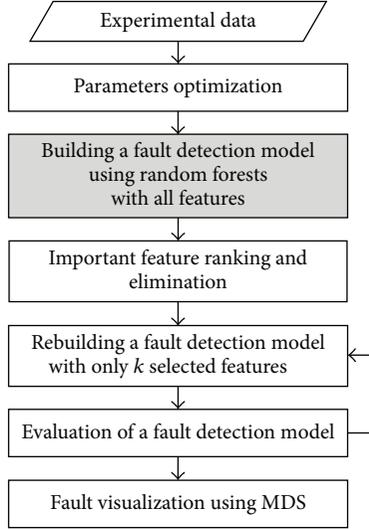


FIGURE 3: Overall diagram of proposed sequences.

Requirements to decide operation condition are considered as follows.

Requirement 1. Parameters optimization is conducted to guarantee high detection rates.

Requirement 2. Fault detection model is built using RFA with all features. The RFA generates variable importance values in numerical form.

Requirement 3. Rank whole features and eliminate the irrelevant features.

Requirement 4. Rebuild a fault detection model with only k selected features.

Requirement 5. Evaluate the rebuilt detection model. If the detection rates and error rates satisfy requirements, terminate the computation. Otherwise, iterate with less number of features.

To evaluate the feasibility of our approach, longitudinal experiments on the flight test data of Table 3 are considered [18].

In Figure 3 sequence, the highest detection rate was satisfied when m_{try} was used only with 2 features. For n_{tree} , there is no specific function that figures out the optimal value as m_{try} . Thus, the optimal value of n_{tree} was considered by choosing the n_{tree} value as high and stable detection rates.

As results of experiments, two optimized parameter values were considered when $m_{\text{try}} = 2$ and $n_{\text{tree}} = 260$. With these two parameters, feature selection of the flight test data has been carried out by employing the feature selection algorithm supported by RFA.

RFA provides the variable importance of each feature; its results are illustrated in Table 2. The proposed approach shows reasonable context information by their important feature. Here, the meaning of $C_{m_{\delta e}}$, $C_{m_{\alpha}}$, and $C_{L_{\alpha}}$ and other parameters are expressed in reference [10, 11]. By the results

TABLE 2: Variable importance with respect to feature.

Features	Variable importance
$C_{m_{\delta e}}$	2.389784762
$C_{m_{\alpha}}$	1.601497696
$C_{L_{\alpha}}$	1.461898295
$C_{D_{\delta e}}$	1.399035508
C_{D_0}	0.939209972
$C_{L_{\delta e}}$	0.734541155
$C_{m_{\alpha}}$	0.711052447
C_{m_q}	0.693304391
C_{L_0}	0.408650397
u_0	0.24089344
C_{D_0}	0.196847678
$C_{l_{\delta e}}$	0.121865038

TABLE 3: Polynomial coefficients and standard error.

	Value	Standard error
a	85.44917	6.34784
b_1	20.15267	10.6106
b_2	-11.5519	6.09265
b_3	2.9505	1.61962
b_4	-0.3802	0.21744
b_5	0.0241	0.0143
b_6	$-5.97E - 04$	$3.66E - 04$

of Table 1, the pitching moment coefficient with changes in the elevator deflection $C_{m_{\delta e}}$ shows bigger difference than the other parameters. However, to get more reliable data, detection ratio versus number of parameters was also carried out.

This approach shows that the feature selection should be important to decide decision performance because the number of features determines detection rate. With combinations of the highest importance variable, decision rate was obtained. Results were illustrated in Figures 4 and 5. By numerical conclusion, the highest decision rate, 97.75%, was obtained when the highest two features were used.

Interpolating the data with sixth order polynomial was obtained as

$$y = a + b_1x + b_2x^2 + \dots + b_6x^6. \quad (3)$$

Intersecting value and coefficients b_1 to b_6 are illustrated in Table 3.

By differentiating (3), three maxima satisfy 1.67, 6.00, and 11.22.

These values mean number of features. Hence, it is definite that two features selection guarantees highest detection rate as in Figure 4. Similarly, 9th order polynomial interpolation was obtained as follows:

$$y = a + b_1x + b_2x^2 + \dots + b_9x^9. \quad (4)$$

Next, three maxima are also obtained as 2.11, 6.11, and 10.11 (Table 4).

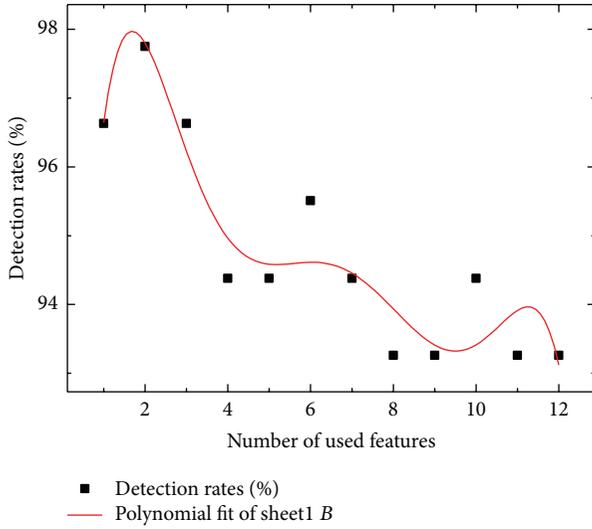


FIGURE 4: Detection rate with respect to number of features.

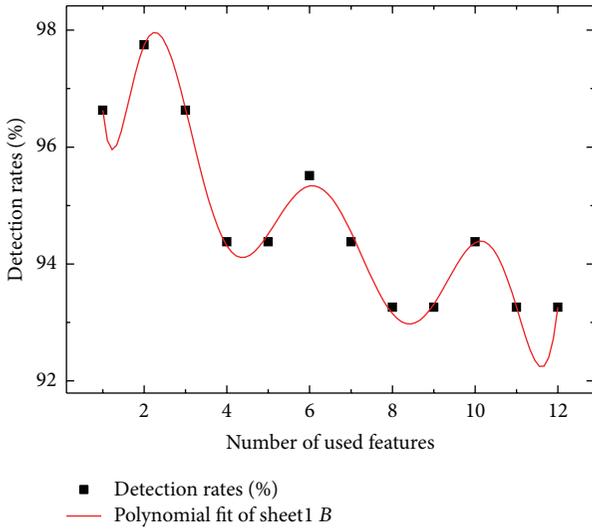


FIGURE 5: Detection rate with respect to number of features.

TABLE 4: Polynomial coefficients and standard error.

	Value	Standard error
a	151.055	21.69855
b_1	-142.582	54.63766
b_2	143.4329	53.61102
b_3	-73.217	27.62498
b_4	21.30309	8.39364
b_5	-3.73895	1.58132
b_6	$4.02E - 01$	$1.87E - 01$
b_7	$-2.59E - 02$	$1.34E - 02$
b_8	$9.11E - 04$	$5.38E - 04$
b_9	$-1.35E - 05$	$9.19E - 06$

It shows the same result with 6th order polynomial interpolation.

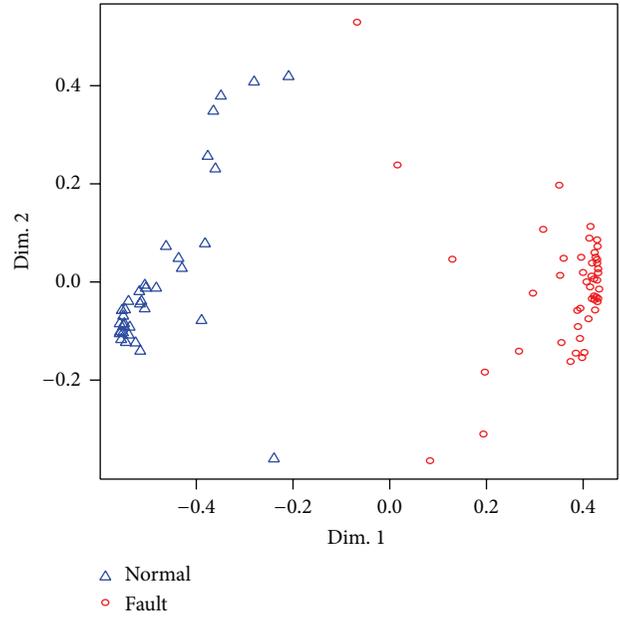


FIGURE 6: MDS plots of normal and fault patterns.

By considering multidimensional scaling (MDS), it provides a method for discovering “hidden” structures in multidimensional data. MDS is designed by considering similarity measure and mapped on a lower dimensional spatial representation [12, 13]. With coefficients $C_{m_{\delta e}}$, $C_{m_{\alpha}}$, and $C_{L_{\alpha}}$, normal/fault patterns are implemented by multidimensional scaling (MDS) methodology in Figure 6 [7]. It is obtained with open source R-project [12].

Above results provide two parameters, $C_{m_{\delta e}}$ and $C_{m_{\alpha}}$, which are enough and most efficient to decide whether it is faulty or not. Now, normal and fault patterns are illustrated via MDS. Normal and fault patterns are shown via blue triangles and red circles in Figure 6. This indicates that the fault monitoring and flight control system organization can be feasible by visualization, without expert’s knowledge.

4.2. Comparison with Similarity Measure Results. Similarity measure is designed through using the definition of Liu [19]. Following similarity measure will be used as the calculation of the degree of similarity between normal and fault operating conditions. Proposed similarity measure has strong point by the point of computation in comparison to the result of the literatures [3–6, 20–22]. They required at least $2n$ comparisons and $2n$ additions by the formations of

$$s(A, B) = 1 - d(A, A \cap B) - d(B, A \cap B),$$

$$s(A, B) = 2 - d((A \cap B), [1]_X) - d((A \cup B), [0]_X). \tag{5}$$

Theorem 2. For any sets $A, B \in F(X)$,

$$s(A, B) = 1 - d(\mu_A(x), \mu_B(x)) \tag{6}$$

is the similarity measure between set A and set B , where d satisfies Hamming distance measure.

TABLE 5: Computation of similarity measure.

Similarity measure	A (normal)	B (normal)	C (fault)	D (fault)
$s_{m_{\delta e}}(F_N, p)$	1.00	0.05	0.05	0.05
$s_{m_{\delta e}}(p, F_F)$	0.00	0.00	1.00	1.00
$s_{m_{\alpha}}(F_N, p)$	0.33	0.08	1.00	0.92
$s_{m_{\alpha}}(p, F_F)$	1.00	0.00	0.16	0.42
Results only (6)				
$s(F_N, p)$	1.33	0.13	1.05	0.97
$s(p, F_F)$	0.78	0.00	1.16	1.42
Results with Table 2				
$s(F_N, p)$	2.92	0.25	1.72	1.59
$s(p, F_F)$	1.60	0.00	2.64	3.06

Proof. Commutative property of (S1) is easy to prove; it is clear from (6) itself. To show (S2);

$$\begin{aligned} s(D, D^C) &= 1 - d(\mu_D(x), \mu_{D^C}(x)) \\ &= 1 - 1 = 0 \end{aligned} \quad (7)$$

is obtained. Because $\mu_D(x)$ and $\mu_{D^C}(x)$ are complements, difference $d(\mu_D(x), \mu_{D^C}(x))$ always satisfies one. (S3) is rather easy to prove:

$$s(C, C) = 1 - d(\mu_C(x), \mu_C(x)) = 1. \quad (8)$$

□

From above statement, it is rational that $s(C, C)$ satisfies maximal value. Finally, triangular equality is obvious by the definition of Liu [19]; hence (S4), is also satisfied.

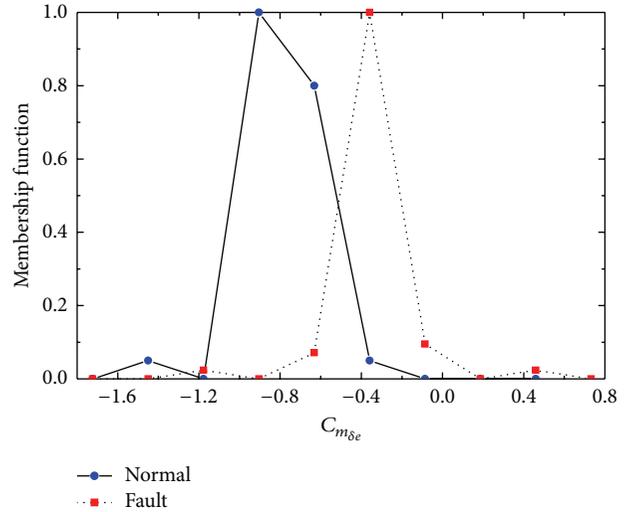
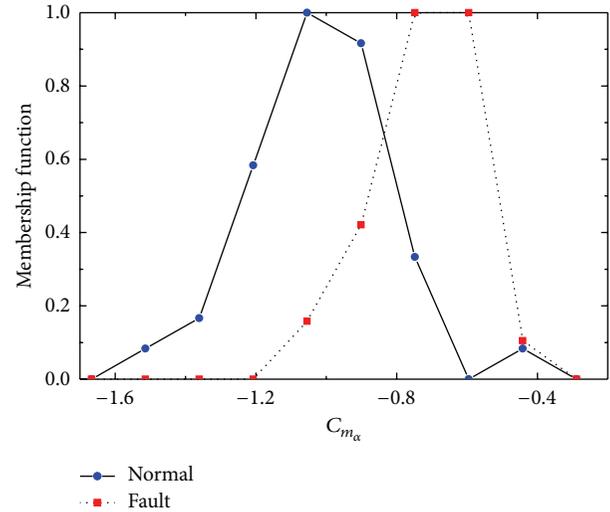
By applying this similarity measure, calculation reduced n comparisons and $n + 1$ additions. Two parameter $C_{m_{\delta e}}$ and $C_{m_{\alpha}}$ membership functions are illustrated in Figures 7 and 8. Normal and fault distributions are also shown. With similarity measure (6), decision results are clearly discriminative. In Table 5, calculation results of (6) are emphasized by considering variable importance of Table 2 as weighting factor.

5. Conclusions

Fault detection problem of aircraft system was carried out with RFA; it was applied to build a fault detection methodology for unmanned aircraft system, named URUMI-II. With obtained performance of RFA, results provide importance of each parameter or feature. The feasibility of fault detection algorithm with RFA was validated.

With experimental data on the flight test of DURUMI-II, fault decision algorithm showed the approach is able to detect faults with high detection rates (Figure 5). Additionally, the visualization of normal and fault patterns using MDS was able to easily figure it out with context information. Similarity measure weighting calculation with importance variable was applied for detection problem. Decision results were emphasized more than with only similarity measure.

By the calculation of RFA, meaningful result was provided that detection algorithm was effective even with

FIGURE 7: Normal/fault data distribution of $C_{m_{\delta e}}$.FIGURE 8: Normal/fault data distribution of $C_{m_{\alpha}}$.

a limited amount of operation data. Consequently, the flight supporting control system with fault detection algorithm is reconfigured. Then, the reliability increases without additional sensors such as a potentiometer on the control surface.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research was supported by Grant no. (13-ASTRP-A02) from Aviation Safety Technology Research Program funded by Ministry of Land, Infrastructure and Transport of Korean government.

References

- [1] W. J. Park, S. H. Lee, and J. I. Song, "Fault detection and isolation of DURUMI-II using similarity measure," *Journal of Mechanical Science and Technology*, vol. 23, no. 2, pp. 302–310, 2009.
- [2] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
- [3] S. Lee and T. O. Ting, "The evaluation of data uncertainty and entropy analysis for multiple events," in *Advances in Swarm Intelligence*, vol. 7332 of *Lecture Notes in Computer Science*, pp. 175–182, Springer, Berlin, Germany, 2012.
- [4] S. Park, S. Lee, S. Lee, and T. O. Ting, "Design similarity measure and application to fault detection of lateral directional mode flight system," in *Advances in Swarm Intelligence*, vol. 7332 of *Lecture Notes in Computer Science*, pp. 183–191, Springer, New York, NY, USA, 2012.
- [5] S. Lee and T. O. Ting, "Uncertainty evaluation via fuzzy entropy for multiple facts," *International Journal of Electronic Commerce*, vol. 4, no. 2, pp. 345–354, 2013.
- [6] S. Lee, W. He, and T. O. Ting, "Study on similarity measure for overlapped and non-overlapped data," in *Proceedings of the International Conference on Information Science and Technology (ICIST '13)*, pp. 48–53, March 2013.
- [7] R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee, "Boosting the margin: a new explanation for the effectiveness of voting methods," *The Annals of Statistics*, vol. 26, no. 5, pp. 1651–1686, 1998.
- [8] T. G. Dietterich, "Experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization," *Machine Learning*, vol. 40, no. 2, pp. 139–157, 2000.
- [9] F. W. Young and R. M. Hamer, *Theory and Applications of Multidimensional Scaling*, Erlbaum Associates, Hillsdale, NJ, USA, 1994.
- [10] W. J. Park, E. T. Kim, K. J. Seong, and Y. C. Kim, "A study on the parameter estimation of DURUMI-II for the fixed right elevator using flight test data," *Journal of Mechanical Science and Technology*, vol. 20, no. 8, pp. 1224–1231, 2006.
- [11] W. Park, E. Kim, Y. Song, and B. Ko, "A study on the real-time parameter estimation of DURUMI-II for control surface fault using flight test data," *International Journal of Control, Automation and Systems*, vol. 5, no. 4, pp. 410–418, 2007.
- [12] Random forest, http://en.wikipedia.org/wiki/Random_forest.
- [13] The R Project for Statistical Computing, <http://www.r-project.org/>.
- [14] G. Biau, "Analysis of a random forests model," *Journal of Machine Learning Research*, vol. 13, pp. 1063–1095, 2012.
- [15] Y. D. Kim, "A Study on Fault Detection and Redundancy Management System," SUDP-P1-G4, 2005.
- [16] M. R. Napolitano, Y. Song, and B. Seanor, "On-line parameter estimation for restructurable flight control systems," *Aircraft Design*, vol. 4, no. 1, pp. 19–50, 2001.
- [17] R. C. Nelson, *Flight Stability and Automatic Control*, McGraw-Hill, New York, NY, USA, 1998.
- [18] W. J. Park, S. M. Lee, S. K. Lee, and J. S. Park, "Lightweight fault detection of DURUMI-II using random forests," in *Proceedings of the International Conference on Materials and Reliability*, Busan, Republic of Korea, 2011.
- [19] X. C. Liu, "Entropy, distance measure and similarity measure of fuzzy sets and their relations," *Fuzzy Sets and Systems*, vol. 52, no. 3, pp. 305–318, 1992.
- [20] S. H. Lee, W. J. Park, and D. Y. Jung, "Similarity measure design and similarity computation for discrete fuzzy data," *Journal of Central South University of Technology (English Edition)*, vol. 18, no. 5, pp. 1602–1608, 2011.
- [21] J. Fan and W. Xie, "Distance measure and induced fuzzy entropy," *Fuzzy Sets and Systems*, vol. 104, no. 2, pp. 305–314, 1999.
- [22] S. Lee, Y. Kim, S. Cheon, and S. Kim, "Reliable data selection with fuzzy entropy," in *Proceedings of the 2nd International Conference on Fuzzy Systems and Knowledge Discovery (FSKD '05)*, vol. 3613 of *Lecture Notes in Computer Science*, pp. 203–212, August 2005.

Research Article

Adaptive MANET Multipath Routing Algorithm Based on the Simulated Annealing Approach

Sungwook Kim

Department of Computer Science, Sogang University, Sinsu dong 1, Mapo-ku, Seoul 121-742, Republic of Korea

Correspondence should be addressed to Sungwook Kim; swkim01@sogang.ac.kr

Received 11 April 2014; Revised 17 May 2014; Accepted 24 May 2014; Published 16 June 2014

Academic Editor: T. O. Ting

Copyright © 2014 Sungwook Kim. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Mobile ad hoc network represents a system of wireless mobile nodes that can freely and dynamically self-organize network topologies without any preexisting communication infrastructure. Due to characteristics like temporary topology and absence of centralized authority, routing is one of the major issues in ad hoc networks. In this paper, a new multipath routing scheme is proposed by employing simulated annealing approach. The proposed metaheuristic approach can achieve greater and reciprocal advantages in a hostile dynamic real world network situation. Therefore, the proposed routing scheme is a powerful method for finding an effective solution into the conflict mobile ad hoc network routing problem. Simulation results indicate that the proposed paradigm adapts best to the variation of dynamic network situations. The average remaining energy, network throughput, packet loss probability, and traffic load distribution are improved by about 10%, 10%, 5%, and 10%, respectively, more than the existing schemes.

1. Introduction

Due to the explosive growth of wireless communication technology, mobile ad hoc networks (MANETs) have been used in many practical applications in the commercial, military, and private sectors. MANETs are self-creating, self-organizing, and autonomous systems of mobile hosts connected by wireless links with no static infrastructure such as base station. When making such networks operational, a key question is how to effectively decide routing paths, given the dynamic nature of the system and the limited knowledge of the network topology. In recent times, a lot of attention has been attracted to designing efficient routing protocols for efficient MANET operations [1–4].

During the operation of MANETs, unexpected growth of traffic may develop in a specific routing path; it may create local traffic congestion. In order to alleviate this kind of traffic overload condition, load balancing strategy should be employed. In MANETs, the meaning of load balancing is to ease out the heavy traffic load in a specific path, which can ensure the balanced network resource assumption. To ensure the load balancing, multipath routing algorithms have been developed. Multipath routing algorithm establishes multiple paths between a source and a destination node and spreads

the traffic load along multiple routes. It can alleviate traffic congestion in a specific path. Therefore, multipath routing algorithms can provide the route resilience while ensuring the reliability of data transmission [5, 6].

Nowadays, metaheuristic approach is widely recognized as a practical perspective to be implemented for real world network operations [7–9]. Traditionally, metaheuristic algorithms try to improve a candidate solution iteratively with regard to a given measure of quality. Even though this approach does not guarantee an optimal solution, it can be widely applied to various network control problems. Simulated annealing is a well-known probabilistic metaheuristic algorithm for finding an effective solution [10–12]. To adaptively make a routing decision, the basic concept of simulated annealing approach can be adopted.

Motivated by the facts presented in the above discussion, a new multipath routing scheme is proposed based on the simulated annealing approach. In this work, we do not focus on trying to get an optimal solution itself, but, instead, an adaptive online feedback model is adopted. Therefore, the proposed scheme repeatedly estimates the current network situations and dynamically makes a control decision. This approach can significantly reduce the computational

complexity and overheads. Due to this reason, wireless nodes are assumed to be self-interested agents and make local decisions in a distributed manner. Therefore, routing packets are adaptively distributed through multiple paths in pursuit of the main goals such as load balancing and network reliability. Under diverse network environment changes, the proposed scheme tries to approximate an optimal network performance. The important features of our proposed scheme are (i) interactive process to get an efficient network performance, (ii) distributed approach for large-scale network operations, (iii) dynamic adaptability to current network, and (iv) feasibility for practical implementation.

1.1. Related Work. Recently, several routing schemes for ad hoc networks have been presented in research literature. Leu et al. developed the multicast power greedy clustering (MPGC) scheme [13], which is an adaptive power-aware and on-demand multicasting algorithm. The MPGC scheme uses greedy heuristic clustering, power-aware multicasting, and clustering maintenance techniques that maximize energy efficiency and prolong network lifetime.

To improve the network reliability and reduce the network traffic, Kim et al. propose the double-layered effective routing (DLER) scheme for peer-to-peer network systems [14]. This scheme first chooses the shortest routing paths among possible routing paths and selects the path associated with the relay peer who has lower mobility to improve the reliability of the system. Therefore, in the DLER scheme, the lower mobility of relay peers contributes to both the stability of clusters and the robustness of the system.

Hieu and Hong proposed the entropy-based multirate routing (EMRR) scheme [15]. This scheme introduces a new approach to modeling relative distance among nodes under a variety of communication rates, due to node's mobility in MANETs. When mobile nodes move to another location, the relative distance between communicating nodes will directly affect the data rate of transmission. Therefore, the stability of a route is related to connection entropy. Taking into account these issues, the link weight and route stability based on connection entropy are considered as a new routing metric. In the EMRR scheme, the problem of determining the best route is formulated as the minimization of an object function formed as a linear combination of the link weight and the connection uncertainty of that link.

The ant-colony based routing algorithm (ARA) scheme was proposed [16]; in this scheme, swarm intelligence and ant-colony metaheuristic techniques are used. This scheme consists of three phases: route discovery, route maintenance, and route failure handling. In the route discovery phase, new routes between nodes are discovered using forward and backward ants. Routes are maintained by subsequent data packets; that is, as the data traverse the network, node pheromone values are modified to reinforce the routes.

Wang et al. developed the logical hypercube-based virtual dynamic backbone (HVDB) scheme for an n -dimensional hypercube in a large-scale MANET [17]. The HVDB scheme is a proactive, QoS-aware, and hybrid multicast routing protocol. Owing to the regularity, symmetry properties,

and small diameter of the hypercube, every node plays almost the same role. In addition, no single node is more loaded than any other node, and bottlenecks do not exist, unlike the case in tree-based architectures. In particular, the HVDB scheme can satisfy the new QoS requirements—high availability and good load balancing—by using location information.

Barolli et al. proposed the genetic algorithm based routing (GAR) scheme for mobile ad hoc networks [18]. In the GAR scheme, only a small number of nodes are involved in route computation because a small population size is used. As a result, routing information is transmitted only for the nodes present in that population. Different routes are ranked by sorting; the first route is the best one, and the remaining routes are used as backup routes. Because a tree-based genetic algorithm method is used in the GAR scheme, the delay and transmission success rate are considered as QoS parameters in this scheme.

The incentive-based repeated routing (IRR) scheme in [19] is an incentive-based routing model that captures the notion of repetition. To provide a desirable solution, the IRR scheme examines certain fundamental properties to govern the behavior of autonomous agents. The distributed routing mechanism (DRM) scheme in [20] is an adaptive and scalable routing scheme for wireless ad hoc networks. This scheme provides a cost-efficient routing mechanism for strategic agents. In addition, the DRM scheme is designed to maximize the benefit of each agent.

The proactive congestion reduction (PCR) scheme in [5] focuses on adaptive routing strategies to help congestion reduction. Based on a nonlinear optimization method for multipath routings, the PCR scheme calculates a traffic splitting vector that determines a near-optimal traffic distribution over routing paths. The shortest multipath source (SMS) scheme [6] is one of the most generally accepted on-demand dynamic routing schemes that build multiple shortest partial disjoint paths. The SMS scheme uses node-disjoint secondary paths to exploit fault tolerance, load balancing, and bandwidth aggregation. All the earlier work has attracted a lot of attention and introduced unique challenges. However, these existing schemes have several shortcomings as described in Section 3. Compared to the PCR scheme and the SMS scheme [5, 6], the proposed scheme attains better performance for wireless network managements.

This paper is organized as follows. Section 2 describes the proposed algorithms in detail. In Section 3, performance evaluation results are presented along with comparisons with the schemes proposed in [5, 6]. Finally, in Section 4, concluding remarks are given and some directions are identified for future work.

2. Proposed MANET Routing Algorithms

Multipath routing algorithms are designed to split and transmit the traffic load through two or more different paths to a destination simultaneously. In this paper, we propose a new multipath routing scheme to balance the network load while ensuring efficient network performance.

2.1. Path Setup Algorithm. Usually, wireless link capacity continually varies because of the impacts from transmission power, interference, and so forth. Therefore, it is important to estimate the current link status by considering several control factors. To configure the adaptive multihop routing path, the proposed algorithm defines a link cost ($L.P$) for each link to estimate the degree of communication adaptability [21, 22]. In order to relatively handle dynamic network conditions, the $L.P$ value from the node i to the node j is obtained as

$$L.P_{ij} = \left[(1 - \alpha) \times \mathcal{E}_{ij} + \alpha \times (1 - \Theta_j(t)) \right] + \left[\omega \times (1 - \Psi_{ij}(t)) \right], \tag{1}$$

$$\text{s.t., } \alpha = \frac{E_i}{E_M}, \quad \mathcal{E}_{ij} = \frac{d_{ij}}{D_M}, \quad \Psi_{ij}(t) = \frac{\kappa_t^{ij}}{(\kappa_t^{ij} + \vartheta_t^{ij})},$$

where d_{ij} is the distance from the node i to the node j and E_i is the remaining energy of the node i . E_M and D_M are the initial energy and the maximum coverage range of each node. Therefore, the d_{ij} and E_i are normalized by the D_M and E_M ; the range is varied from 0 to 1. $\Theta_j(t)$ is the entropy for the node j at the time (t). Usually, entropy is the uncertainty and a measure of the disorder in a system. It represents the topological change, which is a natural quantification of the effect of node mobility on MANET's connectivity service [23]. In this work, the basic concept of entropy is adopted for supporting and evaluating stable routing routes. For the mobile node j , the entropy $\Theta_j(t)$ is calculated as follows [23]:

$$\Theta_j(t) = \frac{-\sum_{k \in F_j} P_k(t, \Delta_t) \times \log P_k(t, \Delta_t)}{\log C(F_j)}, \tag{2}$$

$$\text{s.t., } P_k(t, \Delta_t) = \frac{a_{j,k}}{\sum_{i \in F_j} a_{j,i}},$$

where Δ_t is a time interval. F_j denotes the set of the neighboring nodes of node j , and $C(F_j)$ is the cardinality (degree) of set F_j . To estimate the stability of a part of a specific route, $a_{j,i}$ represents a measure of the relative mobility among two nodes j and i as

$$a_{j,i} = \frac{1}{I.T} \times \sum_{l=1}^{I.T} |v(j, i, t_l)|, \tag{3}$$

$$\text{s.t., } v(j, i, t) = v(j, t) - v(i, t),$$

where $v(j, t)$ and $v(i, t)$ are the velocity vectors of node j and node i at time t , respectively. $I.T$ is the number of discrete times t_l that mobility information can be calculated and disseminated to other neighboring nodes within time interval Δ_t . $v(j, i, t)$ is the relative velocity between nodes j and i at time t . Any change can be described as a change of variable values $a_{j,i}$ in the course of time t such as $a_{j,i}(t) \rightarrow a_{j,i}(t + \Delta_t)$. The entropy $\Theta_j(t)$ is normalized as $0 \leq \Theta_j(t) \leq 1$. If $\Theta_j(t)$ value is close to 1, the part of the route that represents the links of the path associated with an intermediate node j is stable. If $\Theta_j(t)$ value is close to 0, the local route is unstable [23]. In (1), $\Psi_{ij}(t)$ is the link ij 's trust value at the time t . After the t th

iteration, $\Psi_{ij}(t)$ is using the number of packets successfully serviced in the link ij (κ_t^{ij}) divided by the total number of packets that have been sent from the node i to the relay node j ($\kappa_t^{ij} + \vartheta_t^{ij}$).

To relatively estimate the current link situation by using (1), the control parameters α and ω should be adjusted dynamically. The \mathcal{E}_{ij} reflects the cost of the wireless communication; the closer a next node, the more attractive for routing due to the less communication cost. The E_i is the current residual energy of node i , which reflects the remaining lifetime of a wireless node. Due to the characteristics of wireless propagation, the energy consumption rate for wireless communications is strongly related to the internode distance. The parameter α controls the relative weights given to distance and entropy of corresponding relay node. Under diverse network environments, a fixed value of α cannot effectively adapt to the changing conditions [21, 22]. In this paper, we treat it as an online decision problem and adaptively modify α value. When the remaining energy of the node i is high, we can put more emphasis on the stability status of next node j , that is, on $(1 - \Theta_j(t))$. In this case, a higher value of α is more suitable. When the remaining energy of the node i is not enough due to traffic overhead, the path selection should strongly depend on the energy dissipation for data transmission. In this case, a lower value of α is more suitable for the energy consumption rate, that is, on \mathcal{E}_{ij} , since the distance between two neighbor nodes directly affects the energy consumption rate. In the proposed algorithm, the value of α of the node i is dynamically adjusted based on the current rate of its remaining energy per initially assigned energy (E_i/E_M). Therefore, the system can be more responsive to current network conditions by the real-time network monitoring. The parameter ω is an impact factor to evaluate the trust level of the link. In this paper, to avoid the detrimental packet loss effect, each link's trust level is fully considered to estimate $L.P$ value; the ω value is fixed as 1.

The $L.P$ value can represent the normalized communication cost of each link. With the $L.P$ value, we define the path cost parameter (PC) to calculate total routing path cost; PC is computed as the sum of all link costs from the source node to the current node. Based on the PC value, the proposed routing algorithm constructs adaptive multihop routing paths to reach the destination node. At the initial time for routing operations, the source node broadcasts its initial PC value (i.e., PC = 0). Within the power coverage area, message receiving relay nodes individually estimate the link cost according to (1) and estimate its PC value as PC + $L.P$. Some nodes can receive multiple PC values from reachable different neighbor nodes. For self-organizing and independent-effective controlling, each node keeps this information. For example, the node i can have received multiple PC values, that is, PC_1, PC_k , and PC_{N_i} , where PC_k is the receiving PC value of the message-sending neighbor node k ($1 \leq k \leq N_i$) and N_i is the number of total reachable neighbor nodes. In this case, the node i calculates its own PC_i value as follows:

$$PC_i = \arg \min_{k \in N_i} (PC_k + L.P_{ik}). \tag{4}$$

According to (4), the node i adaptively selects one neighbor node as a relay node while minimizing PC_i value, which potentially incorporates more global network information. The estimated PC value is recursively forwarded to establish the routing path. This route formation process is repeated until all available multipaths from the source to the destination node are configured.

2.2. Simulated Annealing Routing Algorithm. Generally, multipath routing algorithms face an essential challenge—how to distribute the volume of traffic to a specific path. In order to produce good solutions within a reasonable amount of computer time, the proposed scheme does not seek the optimal allocation. Based on feedbacks of the real-time traffic measurements, it is designed in a simple but efficient metaheuristic algorithm.

Simulated annealing (SA) is a well-known metaheuristic method that has been applied successfully to combinatorial optimization problems [8]. The term simulated annealing derives from the roughly analogous natural phenomena of annealing of solids, which is accomplished by heating up a solid and allowing it to cool down slowly so that thermal equilibrium is maintained. Each step of the SA process replaces the current solution by a random “nearby” solution, chosen with a probability that depends on the difference between the corresponding function values and on a global parameter T (called the temperature). The T is gradually decreased during the process to reach steady state or thermal equilibrium [8, 10, 12].

In the proposed algorithm, the SA approach is used to solve the multipath routing problem. The basic concept of the proposed algorithm is to proportionally load traffic on each route according to its adaptability. To transmit packets, each node selects a next relay node based on the PC information. From the point of view of the node i , selection probability of the neighbor node k (SP_k) is defined as follows:

$$SP_k = \frac{TC_k}{\sum_{j=1}^n TC_j}, \quad \text{where } TC_k = 1 - \frac{(PC_j + L.P_{ij})}{\sum_{j=1}^n (PC_j + L.P_{ij})},$$

$$k \in N_i, \quad (5)$$

where n is the total number of neighbor nodes. Based on the *roulette-wheel* function [24] of SP values, a next relay node is temporarily selected. For example, the probability of node k 's selection is SP_k . Therefore, we can make the more adaptable nodes more likely to be selected than the less adaptable nodes. In addition, to avoid a local optimal solution, the Boltzmann probability (BP) is adopted. The BP is defined as follows [8]:

$$BP = \exp\left(-\frac{N_{pc} - C_{pc}}{T(t)}\right), \quad (6)$$

where N_{pc} is the SP value of new selected node and C_{pc} is the SP value of previously connected node. In (6), the difference between N_{pc} and C_{pc} (i.e., $N_{pc} - C_{pc}$) means the path adaptability alteration. $T(t)$ is a parameter to control the BP value. Metaphorically, it is the time t 's *temperature* of

the system. As an annealing process, the $T(t)$ is decreased according to a cooling schedule. At the beginning of the annealing algorithm run, the initialization temperature is high enough so that possibility of accepting any decision changes whether it improves the solution or not. While time is ticking away, the $T(t)$ value decreases until the stopping condition is met. In this paper, $T(t)$ value is set to the current ratio of the remaining packet amount to the total routing packet amount.

At the routing decision time, there are two cases.

- (i) If the N_{pc} value is higher than the C_{pc} (i.e., $N_{pc} - C_{pc} \geq 0$), the new selected neighbor node replaces the current relay node.
- (ii) If the N_{pc} value is less than the C_{pc} value (i.e., $N_{pc} - C_{pc} < 0$), the new selected neighbor node is not eligible to replace the current relay node. However, this node might still be accepted as a new relay node to potentially avoid local optima. It is analogous to the uphill move acceptance to reach an optimal point. In this case, a random number X is generated, where X is in the range of $\{0 \cdots 1\}$.
 - (a) If the X is less than BP (i.e., $X < BP$), the new selected neighbor node replaces the current relay node.
 - (b) Otherwise, the current routing route is not changed.

Based on the SA approach, individual nodes in our proposed scheme locally make routing decisions to select next relay nodes. In an entirely distributed fashion, this hop-by-hop path selection procedure is recursively repeated until the packet reaches the destination node. Therefore, our proposed routing algorithm can have the self-adaptability for network dynamics.

2.3. The Main Steps of MANET Routing Algorithm. In this paper, we propose a new multipath routing algorithm for wireless mobile ad hoc networks. In the proposed scheme, routing is guided by employing a simulated annealing process. Therefore, self-interested ad hoc nodes make routing decisions according to private preferences while adapting the current network situations. To solve the dynamic and distributed routing problem, the main steps of the proposed multipath routing algorithm are given next.

Step 1. Each node dynamically estimates the d , \mathcal{C} , E , $\Theta(\cdot)$, $\Psi(\cdot)$, and α values based on the real-time measurement.

Step 2. The $L.P$ value is locally calculated according to (1).

Step 3. At the initial time for routing operations, the source node broadcasts the initial PC value to neighbor nodes. Each node calculates its PC value by using (4) and recursively forwards this information.

Step 4. Based on the PC value, route configuration process continues repeatedly until all available multipaths from the source to the destination node are configured.

TABLE 1: Type of traffic and system parameters used in the simulation experiments.

(a)		
Traffic type	Bandwidth requirement	Connection duration (ave./sec)
I	128 Kbps	60 sec (1 min)
II	256 Kbps	120 sec (2 min)
III	512 Kbps	180 sec (3 min)

(b)		
Parameter	Value	Description
<i>unit_time</i>	1 second	Equal interval of time axis
e_{dis}	1 pJ/bit/m ²	Energy dissipation coefficient for the packet transmission
E_{co}	10 nJ/bit	System parameter for the electronic digital coding energy dissipation
D_M	10 m	Maximum wireless coverage range of each node
E_M	10 joules	Initial assigned energy amount of each node
ω	1	The weighted factor for the trust level
$I.T$	10 seconds	The number of discrete times to estimate entropy
X	0~1	Generated random number

(c)			
Parameter	Initial	Description	Values
α	1	The ratio of remaining to initial energy of node	0~1 (E_i/E_M)
$T(t)$	1	The ratio of remaining to initial packet amount at time t	0~1

Step 5. To transmit packets, each relay node temporarily selects a next relay node with the selection probability, which is estimated according to (5).

Step 6. If the N_{pc} value is higher than the C_{pc} (i.e., $N_{pc} - C_{pc} > 0$), the new selected neighbor node replaces the current relay node; proceed to Step 8. Otherwise, go to Step 7.

Step 7. When the N_{pc} value is less than the C_{pc} value (i.e., $N_{pc} - C_{pc} < 0$), a random number X is generated. If a generated X is less than the BP (i.e., $X < BP$), the new selected neighbor node replaces the current relay node. Otherwise, the established routing route is not changed.

Step 8. In an entirely distributed fashion, this hop-by-hop path selection procedure is recursively repeated until the packet reaches the destination node.

3. Performance Evaluation

In this section, the effectiveness of the proposed algorithms is validated through simulation; we propose a simulation model for the performance evaluation. With a simulation study, the performance superiority of the proposed multipath routing scheme can be confirmed. The assumptions implemented in our simulation model were as follows.

- (i) 100 nodes are distributed randomly over an area of 500×500 meter square.
- (ii) Each data message is considered CBR traffic with the fixed packet size.

- (iii) Network performance measures obtained on the basis of 50 simulation runs are plotted as functions of the packet generation per second (packets/s).

- (iv) Data packets are generated at the source according to the rate λ (packets/s), and the range of offered load was varied from 0 to 3.0.

- (v) The bandwidth of the wireless link was set to 5 Mb/s and the *unit_time* is one second.

- (vi) The source and destination nodes are randomly selected.

- (vii) For simplicity, we assume the absence of noise or physical obstacles in our experiments.

- (viii) The mobility of each mobile node is randomly selected from the range of 0–10 m/s, and mobility model is random way point model.

- (ix) At the beginning of simulation, all nodes started with an initial energy of 10 joules.

- (x) Three different traffic types were assumed; they were generated with equal probability.

Table 1 shows the traffic types and system parameters used in the simulation. Each type of traffic has its own requirements in terms of bandwidth and service time. In order to emulate a real wireless network and for a fair comparison, we used the system parameters for a realistic simulation model [21, 22].

Recently, the PCR scheme [5] and the SMS scheme [6] have been published and introduced unique challenges for the issue of multipath routing in MANETs. Even though these existing schemes have presented novel multipath routing algorithms, there are several disadvantages. First, these

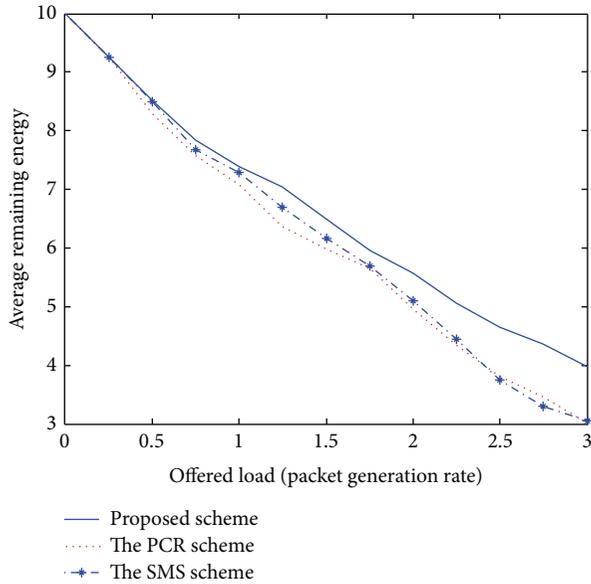


FIGURE 1: Average remaining energy.

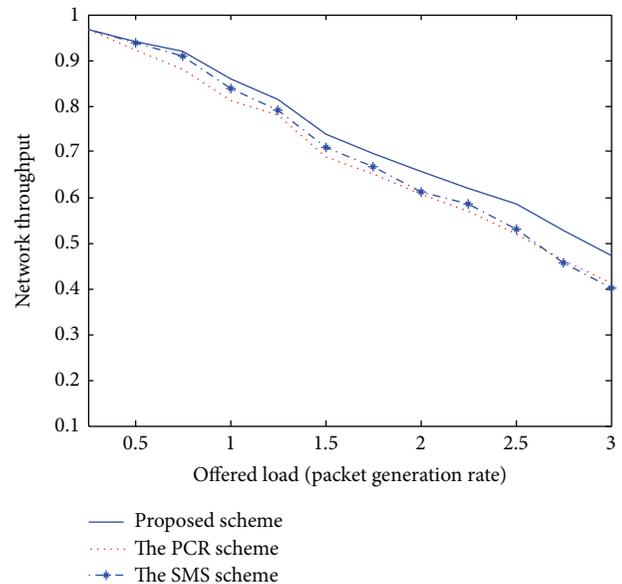


FIGURE 2: Network throughput.

schemes cannot adaptively estimate the current network conditions. Therefore, each node is unaware of effective routing paths to reach a destination. Second, some nodes carry a disproportionately large amount of the entire traffic, drastically decreasing the throughput of the flows they forward. Third, the PCR and SMS schemes are based on a centralized approach. The ideas for practical implementations are left for future study. As mentioned earlier, we compare the performance of the proposed scheme with these existing schemes to confirm the superiority of the proposed approach. In our simulation analysis of Figures 1–5, the x -axis (a horizontal line) marks the traffic intensities, which is varied from 0 to 3.0. The y -axis (a vertical line) represents the normalized value for each performance criterion.

Figure 1 compares the performance of each scheme in terms of the average remaining energy of wireless nodes. To maximize a network lifetime, the remaining energy is an important performance metric. All the schemes have similar trends. However, based on (1), the proposed scheme effectively selects the next routing link by considering the remaining energy information. Therefore, we attain much remaining energy under heavy traffic load intensities; it guarantees a longer node lifetime.

Figure 2 shows the performance comparison of network throughput. Usually, network throughput is the rate of successful message delivery over a communication channel. The throughput is usually measured in bits per second (bit/s or bps) and sometimes in data packets per second or data packets per time slot. In this work, network throughput is defined as the ratio of data amount received at the destination nodes to the total generated data amount. For a fair comparison, it is the best realistic way. Due to the inclusion of the adaptive online approach, the proposed scheme can have the best throughput gain.

In Figure 3, the packet loss probabilities are presented; packet loss means the failure of one or more transmitted

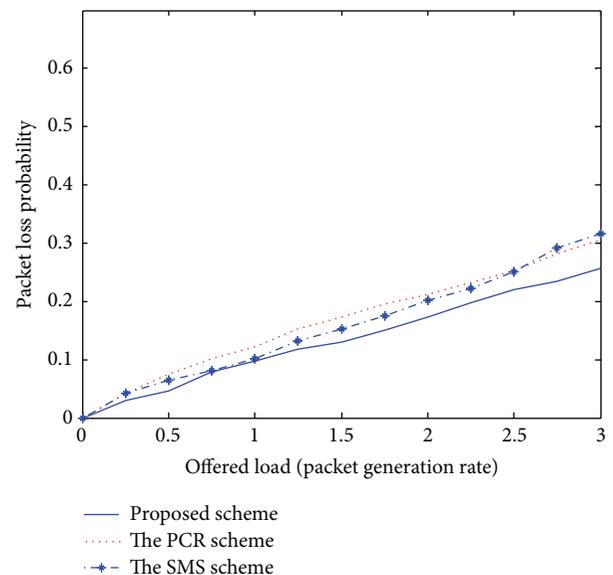


FIGURE 3: Packet loss probability.

packets to arrive at their destinations. As the offered traffic load increases, wireless nodes will run out of the energy or capacity for data transmissions and data packets are likely to be dropped. Therefore, the packet loss probability increases linearly with the traffic load. Based on the real-time online manner, our dynamic SA approach can improve the system reliability, so we achieve a lower packet loss rate than other schemes under various traffic loads.

The curves in Figures 4 and 5 indicate the average energy-exhaustion ratio and normalized traffic load distribution. In this paper, traffic load distribution means the average rate of traffic dispersion among wireless nodes. In an entirely

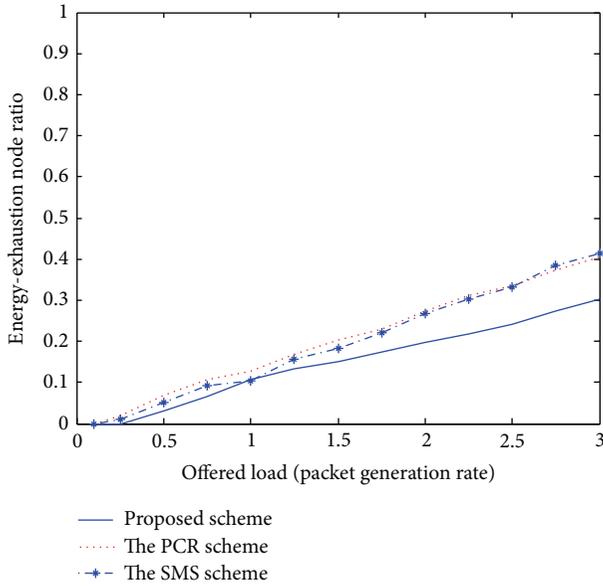


FIGURE 4: Energy-exhaustion ratio.

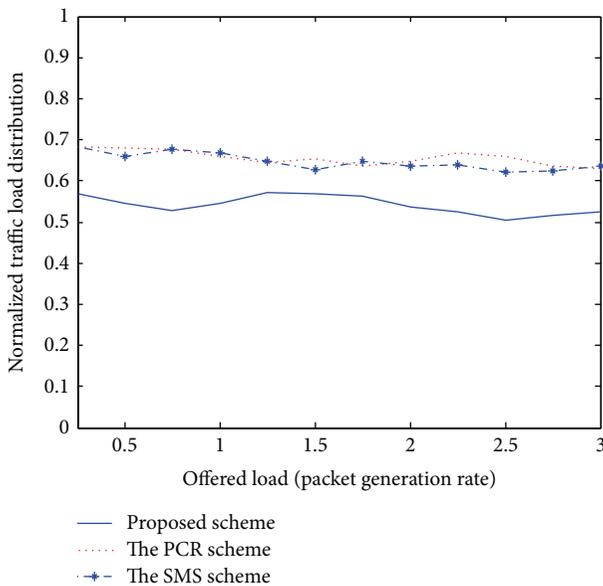


FIGURE 5: Normalized traffic load distribution.

distributed fashion, individual node in our scheme monitors the current network situation and updates all control parameters periodically for the adaptive routing. Therefore, under various system constraints, the proposed scheme is able to decrease the number of energy expiration nodes and adaptively distribute routing packets to avoid traffic congestions, which is highly desirable property for the MANET management.

The simulation results shown in Figures 1–5 demonstrate that the proposed multipath routing scheme generally exhibits better performance compared with the other existing schemes [5, 6]. Based on the adaptive simulated annealing

approach, the proposed scheme constantly monitors the current traffic conditions and gets an efficient solution. Through the simulation experiments, it could be seen that the proposed strategy is proved to be an effective paradigm to solve complex routing problems in a dynamic network environment.

4. Summary and Conclusions

Recent advances in wireless technology and availability of mobile computing devices have generated a lot of interest in mobile ad hoc networks. For these networks, the biggest challenge is to find routing paths to satisfy varying requirements. In this paper, new multipath routing algorithms are developed based on the effective simulated annealing approach. For real network implementation, the proposed scheme is designed in self-organizing, dynamic online, and interactive process. Therefore, each individual node has an ability to provide more adaptive control mechanism and makes a local routing decision to find an efficient path. Under dynamic network environments, this approach can dynamically reconfigure the established path to adapt to network changes. From simulation results, the proposed scheme outperforms existing schemes in terms of network reliability, energy efficiency, and so forth.

In the future, we expect our methodology to be useful in developing new adaptive ad hoc routing algorithms. In particular, the metaheuristic approach can be extended to support delay sensitive data services. In addition, the basic concept of adaptive online algorithms has become an interesting research topic in highly mobile ad hoc networks.

Conflict of Interests

The author, Sungwook Kim, declares that there is no conflict of interests regarding the publication of this paper.

References

- [1] P. Deepalakshmi and S. Radhakrishnan, "QoS routing algorithm for mobile ad hoc networks using ACO," in *Proceedings of the International Conference on Control Automation, Communication and Energy Conservation (INCACEC '09)*, pp. 1–6, June 2009.
- [2] J. C.-P. Wang, M. Abolhasan, D. R. Franklin, and F. Safaei, "End-to-end path stability of reactive routing protocols in IEEE 802.11 ad hoc networks," in *Proceedings of the IEEE 34th Conference on Local Computer Networks (LCN '09)*, pp. 20–23, October 2009.
- [3] F. Qin and Y. Liu, "Multipath based QoS routing in MANET," *Journal of Networks*, vol. 4, no. 8, pp. 771–778, 2009.
- [4] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile ad hoc networks," *Journal of Ad Hoc Networks*, vol. 2, no. 1, pp. 1–22, 2004.
- [5] M. Klinkowski, D. Careglio, and J. Solé-Pareta, "Reactive and proactive routing in labelled optical burst switching networks," *IET Communications*, vol. 3, no. 3, pp. 454–464, 2009.
- [6] H. Zafar, D. Harle, I. Andonovic, and Y. Khawaja, "Performance evaluation of shortest multipath source routing scheme," *IET Communications*, vol. 3, no. 5, pp. 700–713, 2009.

- [7] J. Leino, *Applications of game theory in Ad Hoc networks [M.S. thesis]*, Helsinki University of Technology, 2003.
- [8] M. Randall, G. McMahon, and S. Sugden, "A simulated annealing approach to communication network design," *Journal of Combinatorial Optimization*, vol. 6, no. 1, pp. 55–65, 2002.
- [9] M. Dirani and T. Chahed, "Framework for resource allocation in heterogeneous wireless networks using game theory," in *Proceedings of the 3rd International Workshop of the EURO-NGI Network of Excellence*, pp. 144–154, 2006.
- [10] T. K. Varadharajan and C. Rajendran, "A multi-objective simulated-annealing algorithm for scheduling in flowshops to minimize the makespan and total flowtime of jobs," *European Journal of Operational Research*, vol. 167, no. 3, pp. 772–795, 2005.
- [11] T. Hussain and S. J. Habib, "Optimization of network clustering and hierarchy through simulated annealing," in *Proceedings of the 7th IEEE/ACS International Conference on Computer Systems and Applications (AICCSA '09)*, pp. 712–716, May 2009.
- [12] K. Bouleimen and H. Lecocq, "A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode version," *European Journal of Operational Research*, vol. 149, no. 2, pp. 268–281, 2003.
- [13] J. J. Y. Leu, M. H. Tsai, C. Tzu-Chiang et al., "Adaptive power aware clustering and multicasting protocol for mobile ad-hoc networks," in *Ubiquitous Intelligence and Computing*, pp. 331–340, 2006.
- [14] J. Kim, K. Lee, T. Kim, and S. Yang, "Effective routing schemes for double-layered peer-to-peer systems in MANET," *Journal of Computing Science and Engineering*, vol. 5, no. 1, pp. 19–31, 2011.
- [15] C. T. Hieu and C. Hong, "A connection entropy-based multi-rate routing protocol for mobile Ad Hoc networks," *Journal of Computing Science and Engineering*, vol. 4, no. 3, pp. 225–239, 2010.
- [16] M. Gunes, U. Sorges, and I. Bouazizi, "ARA—the ant-colony based routing algorithm for MANETs," in *Proceedings of the International Conference on Parallel Processing Workshops*, pp. 79–85, August 2002.
- [17] G. Wang, J. Cao, L. Zhang, K. C. C. Chan, and J. Wu, "A novel QoS multicast model in mobile ad hoc networks," in *Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS '05)*, pp. 206–211, April 2005.
- [18] L. Barolli, A. Koyama, T. Sukanuma, and N. Shiratori, "GAMAN: a GA based QoS routing method for mobile ad hoc networks," *Journal of Interconnection Networks*, vol. 4, no. 3, pp. 251–270, 2003.
- [19] M. Afergan, "Using repeated games to design incentive-based routing systems," in *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM '06)*, pp. 1–13, April 2006.
- [20] M.-Y. Wu and W. Shu, "RPF: a distributed routing mechanism for strategic wireless ad hoc networks," in *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM '04)*, pp. 2885–2889, December 2004.
- [21] S. Kim, "Cooperative game theoretic online routing scheme for wireless network managements," *IET Communications*, vol. 4, no. 17, pp. 2074–2083, 2010.
- [22] S. Kim, "Game theoretic multi-objective routing scheme for wireless sensor networks," *Ad-Hoc & Sensor Wireless Networks*, vol. 10, no. 4, pp. 343–359, 2010.
- [23] H. Shen, B. Shi, L. Zou, and H. Gong, "A Distributed entropy-based long-life QoS routing algorithm in Ad Hoc network," in *Proceedings of the Canadian Conference on Electrical and Computer Engineering: Toward a Caring and Humane Technology (CCECE '03)*, pp. 1535–1538, May 2003.
- [24] Y. Zou, Z. Mi, and M. Xu, "Dynamic load balancing based on roulette wheel selection," in *Proceedings of the International Conference on Communications, Circuits and Systems (ICCCAS '06)*, pp. 1732–1734, June 2006.

Research Article

A Solution Quality Assessment Method for Swarm Intelligence Optimization Algorithms

Zhaojun Zhang,¹ Gai-Ge Wang,² Kuansheng Zou,¹ and Jianhua Zhang¹

¹ School of Electrical Engineering and Automation, Jiangsu Normal University, Xuzhou, Jiangsu 221116, China

² School of Computer Science and Technology, Jiangsu Normal University, Xuzhou, Jiangsu 221116, China

Correspondence should be addressed to Zhaojun Zhang; zj921@gmail.com

Received 21 April 2014; Accepted 21 May 2014; Published 11 June 2014

Academic Editor: Xin-She Yang

Copyright © 2014 Zhaojun Zhang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Nowadays, swarm intelligence optimization has become an important optimization tool and widely used in many fields of application. In contrast to many successful applications, the theoretical foundation is rather weak. Therefore, there are still many problems to be solved. One problem is how to quantify the performance of algorithm in finite time, that is, how to evaluate the solution quality got by algorithm for practical problems. It greatly limits the application in practical problems. A solution quality assessment method for intelligent optimization is proposed in this paper. It is an experimental analysis method based on the analysis of search space and characteristic of algorithm itself. Instead of “value performance,” the “ordinal performance” is used as evaluation criteria in this method. The feasible solutions were clustered according to distance to divide solution samples into several parts. Then, solution space and “good enough” set can be decomposed based on the clustering results. Last, using relative knowledge of statistics, the evaluation result can be got. To validate the proposed method, some intelligent algorithms such as ant colony optimization (ACO), particle swarm optimization (PSO), and artificial fish swarm algorithm (AFS) were taken to solve traveling salesman problem. Computational results indicate the feasibility of proposed method.

1. Introduction

Swarm intelligence (SI) optimization [1] is a class of certain population-based metaheuristics which are inspired by the behavior of swarm of agents (i.e., living beings) interacting locally with each other and with their environment. SI is relatively new subfield of artificial intelligence. The behavior of every agent in SI is simple and does not have intelligence. But a number of simple agents through local rules are able to have the emergence of collective intelligence and come to intelligent solutions for complex problems. In recent years, SI has received widespread attention in research. Typical SI schemes include ant colony optimization (ACO) [2], particle swarm optimization (PSO) [3], artificial bee colony (ABC) [4], and artificial fish swarm algorithm (AFS) [5].

ACO is a class of optimization algorithms modeled on the foraging behavior of an ant colony. In ACO, a colony of artificial ants with the artificial pheromone trails and heuristic information are stochastic constructive heuristics

that build better and better solutions by using and updating pheromone trail. New solutions are generated using a parameterized probabilistic model, the parameters of which are updated using previously generated solutions so as to direct the search towards promising areas of the solution space. The first ACO algorithm is ant system (AS) [6]. In the next years, many kinds of ACO algorithms have been developed to improve the performance of AS, such as ant colony system (ACS) [7], max-min ant system (MMAS) [8], and two-stage updating pheromone for invariant ant colony optimization algorithm (TSIACO) [9]. PSO is a metaheuristic search method that simulates the movements of a flock of birds which aim to find food. PSO optimizes a problem by having a population of candidate solutions, called particles, and moving these particles around in the search space according to simple mathematical formulae over the particle's position and velocity. Each particle's movement is influenced by its local best known position and also guided toward the best known positions in the search space, which are updated as

better positions founded by other particles. The first PSO algorithm was introduced by Kennedy and Eberhart. ABC is an optimization algorithm based on the intelligent foraging behavior of honey bee swarm, proposed by Karaboga in 2005. In the ABC model, the colony consists of three groups of bees: employed bees, onlookers, and scouts. It is assumed that there is only one artificial employed bee for each food source. Employed bees go to their food source and come back to hive and dance on this area. The employed bee whose food source has been abandoned becomes a scout and starts to search for finding a new food source. Onlookers watch the dances of employed bees and choose food sources depending on dances. The scout bee moves in the solution space to discover new food sources. SI has been applied to many applications problems, such as knapsack problems, scheduling problems, assignment problems, multiobjective optimization problem, and cluster analysis.

Although great progress has been achieved in application, there is also a basic question, which is how to quantify the goodness of the solution obtained in finite time, needed to be answered. We call it solution quality evaluation problem. At present, the existing researches focus on the solution “value performance,” namely, the difference between the solution obtained by algorithm and the optimal solution of the problem. The general use of the method is ratio analysis, namely, ratio between solution obtained by algorithm and optimal solution. If the ratio is closer to 1, it means that higher quality is obtained by algorithm and the algorithm is more effective. Competitive analysis [10] of online algorithm also can be employed. The drawback of two methods is that they need optimal solution of problem. There are some approximation methods used to estimate optimal for example, extreme value theory [11] and Lagrange’s relaxation method [12], to get the solution value or bound to replace the optimal solution in practical problems. This analysis method generally requires strong theoretical basis of mathematic and strong math skills, and even it is difficult or impossible to give this kind of boundary for most of the problems. In addition to bias in the theoretical study of evaluation methods, some scholars pay more attention to the experimental analysis method. Hoos and Stützle [13] proposed to analyze the performance and behavior of stochastic local search algorithm by experimental analysis method. The performance of several existing particle swarm optimization algorithms was compared by using this method, and an improved particle swarm optimization algorithm was introduced according to the law in [14].

With development of the ordinal optimization (OO) theory [15], the research changes the angle to solution “ordinal performance” to evaluate solution quality of optimization method. Here the solution “ordinal performance” refers to the judgment about whether the solution is belonging to the good enough solution set. Shen et al. [16] used solution comparison between heuristic methods and uniform sampling to evaluate the solution. The evaluation criterion is alignment probability used in OO. As the extension of this work, author used the knowledge of hypothesis testing to develop it into a theory in [17]. In this paper, we proposed an experimental analysis method based on the analysis of search space and

characteristic of algorithm itself to evaluate the solution quality for SI.

The rest of this paper is organized as follows: Section 2 reviews the basic idea of OO and indicates the difficulty of quantifying solution quality by analyzing the existing method. Section 3 describes our experimental analysis method detailed. Some simulation results are presented in Section 4 to show the feasibility of proposed method. Finally, Section 5 concludes the paper.

2. Basics of Ordinal Performance

The ordinal performance is concerned with whether the solution belongs to the good enough set. The evaluation criterion is alignment probability. The definition of good enough set and alignment probability is introduced in OO. So, in this section, we briefly overview OO.

2.1. Brief Overview of OO. OO was first introduced by Ho et al. in 1992 [15], which has become an important tool for optimizing discrete event dynamic system (DEDS). There are two basic ideas in OO. The first idea is ordinal comparison; that is, “order” is easier to ascertain than “value.” The second idea is goal softening. Instead of only caring about optimal solution, OO is willing to settle for the “good enough” solution.

In OO, Θ is the search space and satisfies $|\Theta| = N$. The “good enough” set G is defined as the top- g of the search space Θ or top $p\%$ of the search space Θ . It satisfies $|G| = g$. Selected set S is selected by rule and satisfies $|S| = s$. OO can guarantee that S contains top- g solutions of the search space with a high probability. It is called alignment probability in OO and denoted by P_{AP} .

2.2. Ordinal Performance. The research of solution quality evaluation method transfers from the value performance to the ordinal performance, after the definition of the good enough set, selected set, and alignment probability introduced. Based on this knowledge, Shen et al. [17] proposed evaluation method, called ordinal optimization ruler (OO ruler), using the related knowledge of hypothesis testing. So we can use OO ruler to qualify the ordinal performance of solution. One of the intuitive understandings of OO ruler is that uniform samples are taken out from the whole search space and evaluated with a crude but computationally easy model when applying OO. After ordering via the crude performance estimates, the lined-up uniform samples can be seen as an approximate ruler. By comparing the heuristic design with such a ruler, we can quantify the heuristic design, just as we measure the length of an object with a ruler. If the OO ruler gets from all the solutions, it is an accurate ruler. But this is obviously an ideal situation for practical problems. It is proved that approximate OO ruler is also effective.

Theorem 1 (see [17]). *If the k solution obtained by optimization algorithm is better than t solution of selected set obtained by uniform sampling, we can judge that the k solution belongs to the top $p\%$ of the search space Θ at least. And the type II*

error probability is not larger than β_0 . The relation between s , β_0 , t , and $p\%$ is determined by

$$\sum_{j=0}^{t-1} \binom{s}{j} (p\%)^j (1-p\%)^{s-j} \leq \beta_0, \quad (1)$$

where $\binom{s}{j}$ represents the number of different choices of s designed out of j distinguished ones.

In the case of given parameters of s and β_0 , we can get relation between t and $p\%$ through the list method.

For an arbitrary solution obtained by heuristic algorithm, we only need to compare it whether satisfies the conditions of Theorem 1, then we can make the corresponding judgment, so as to realize the evaluation ordinal performance of solution. But OO ruler has a premise. To get OO ruler, uniform sampling for search space is needed. It is also prerequisite for OO. The so-called uniform sampling refers to the same probability of getting arbitrary solution. It is also the reason why the uniform sampling can provide quantitative reference. But, for some problems, it is difficult to achieve uniform sampling, and thus it will not be able to get OO ruler. In addition, the price of getting OO ruler for huge solution space is very high. These two problems limit the application of OO ruler in solution evaluation. However, the introduction of ordinal performance has great inspiration for the research of solution quality evaluation for SI.

3. The Framework of Assessment Method

In this section, we take traveling salesman problem (TSP) as an example to describe experimental analysis method of solution quality evaluation.

3.1. Sample Characteristics of SI. For SI, the feature of the algorithm itself determines that the sampling method in the search space is not uniform. Especially by the partial reinforcement effect, it makes the algorithm more and more concentrated in certain regions. So it is not suitable for evaluating method directly using OO ruler. In addition, the algorithm produces a large number of feasible solutions. The feasible solution contains the search characteristics of some algorithms and the distribution of the solution space. To obtain the hidden information and its rational utilization through some analysis methods, we need to do some research. It plays an important role in the research of quality evaluation and improving the algorithm performance.

3.2. The Framework of Assessment Method. Based on the above analysis, this paper presents a general framework of the quality evaluation method for SI. The framework contains three procedures. First, to get some internal approximate uniform subclass, using cluster method, the solution samples (corresponding to selected subset of OO) were homogeneous processing. Second, discrete probability distribution solution samples of each subclass and the scale relationship of the subclass are estimated in the fitness space. Based on the characteristics of the subclass, the presupposition ratio of

the good enough set is distributed to each subclass. Last, alignment probability is calculated according to the model of solution quality evaluation, so as to complete the evaluation of the solution quality.

3.2.1. Uniform Clustering for Nonuniform Samples. According to the characteristics of discrete space, uniform clustering of samples is that obtaining probability of solution is approximating same. Compared with the continuous space, clustering is very different from discrete space. General discrete spatial distance features are defined with the question, and not as the continuous space as a distance to define general way. This makes clustering method based on grid no longer applicable, which is used in continuous space such as density clustering and clustering method based on grid. And the huge solution sample set also limits the use of some special clustering method. Therefore, we need to design a suitable and efficient clustering algorithm based on demand.

Approximate sampling probability is the purpose of clustering. The approximate sampling probability here refers to the neighbor characteristics (including the distance and number of nearest neighbors) consistent approximation. A feasible method for TSP is to calculate the distance between all solution samples. Then clustering is done according to the nearest neighbor statistical feature of each sample distance. But it is only applicable to the small size of the solution sample. Another possible method is that the clustering centers are selected from the best solutions. The distance is calculated between each feasible solution and the cluster center. Then the solution samples are clustered according to the distance. The calculation complexity of this algorithm is low. It is more suitable for clustering large scale solution samples. In the next section, we use this clustering method.

3.2.2. The "Good Enough" Set Decomposition. The solution alignment probability is calculated using a priori ratio of the good enough set (the ration between the good enough set and search space) in OO. The ratio of each kind of the good enough sets is needed to know after clustering. The prior ratio requires decomposing prior ratio of each class. This decomposition has a certain relationship with each class distribution of samples and the class size. Therefore, the distribution characteristics of solution in the fitness value, as well as proportional relation of class size, are needed to estimate.

Estimation of distribution of solution in the fitness value is problem of one-dimensional distribution sequence estimation. The purpose of distribution estimation is to obtain the good enough set distribution. If the fitness value is arranged according to the order from small to large, ordered performance curve (OPC) can be obtained. For the minimization problem, the good enough set is in the first half of the OPC. To obtain a true estimation of the good enough set, you need to consider the types of OPC.

3.2.3. Ordinal Performance Estimation. The original search space after clustering is divided into l approximate uniform partition. Search space Θ_k , the good enough set G_i , and

Step 1. Delete the same solution of S according to distance, and denote the new set by S1;
 Step 2. Find the best solution from the set S1, and denote s^0 ;
 Step 3. Calculate the distance between the solution of S1 and s^0 , divide the set S1 into some subset S_i according to distance, and let $|S_i| = s_i$;
 Step 4. Distribute the good enough solution set G;
 Step 4.1. Get M solutions from S1 according to the order of fitness;
 Step 4.2. Delete the infeasible and same solutions, and get M1 solutions;
 Step 4.3. Assemble M1 solutions and S1 solutions, and get g solutions by ordering fitness;
 Step 4.4. Calculate the distance between g solutions and s^0 , and count the number of g_i according to the distance;
 Step 5. Calculate the alignment probability P_{AP} by (4)

ALGORITHM 1: The main steps of assessment method.

selected set S_i of each partition and search space Θ , good enough set G, and selected set S of the original search space have the following correspondence in the collection and base:

$$\begin{aligned} \Theta_1 \cup \Theta_2 \cup \dots \cup \Theta_l &= \Theta, \\ G_1 \cup G_2 \cup \dots \cup G_l &= G, \\ S_1 \cup S_2 \cup \dots \cup S_l &= S, \\ N_1 + N_2 + \dots + N_l &= N, \\ N \equiv |\Theta|, \quad N_i \equiv |\Theta_i|, \quad \forall i = 1, \dots, l, \\ g_1 + g_2 + \dots + g_l &= g, \\ g \equiv |G|, \quad g_i \equiv |G_i|, \quad \forall i = 1, \dots, l, \\ s_1 + s_2 + \dots + s_l &= s, \\ s \equiv |S|, \quad s_i \equiv |S_i|, \quad \forall i = 1, \dots, l, \end{aligned} \tag{2}$$

where $|\cdot|$ is the base of set \cdot .

Since the probability of any feasible solution pumped into each subclass is the same, for a sampling result θ_i has

$$P(\theta_s = \theta_i) = \frac{1}{N_i}, \quad \forall \theta_i \in \Theta_i. \tag{3}$$

In this paper, we only concern the selected set whether has at least one solution in good enough set. So we can draw the following conclusions:

$$P_{AP} \{ |S \cap G| \geq 1 \} = 1 - \prod_{i=1}^l (1 - p_i\%)^{s_i}. \tag{4}$$

3.2.4. *Procedures of Assessment Method.* The main steps to get the evaluation method by the above analysis are described in Algorithm 1.

4. Experimental Results and Analysis

In this section, we take the Hopfield 10-city problem, which is also used in [17], as the example to demonstrate our experimental analysis method. The coordinates of the 10 cities are

$\{(0.4000, 0.4439); (0.2439, 0.1463); (0.1707, 0.2293); (0.2293, 0.7610); (0.5171, 0.9414); (0.8732, 0.6536); (0.6878, 0.5219); (0.8488, 0.3609); (0.6683, 0.2536); (0.6195, 0.2634)\}$. There is $(10-1)! = 362880$ solutions in the search space. The best path is $[1, 4, 5, 6, 7, 8, 9, 10, 2, 3]$ or $[1, 3, 2, 10, 9, 8, 7, 6, 5, 4]$. Here we define $|G| = 0.005N$. We use two groups of experimental simulation to demonstrate effectiveness of proposed method, where P_{AP} is alignment probability. Statistics value represents the alignment probability by our methods. Computational value is the alignment probability, and the error represents the difference of two alignment probabilities.

4.1. *Evaluation Index.* Alignment probability is a measure of whether optimal solution belongs to the good enough set. It is a probability value. Therefore, studying this value has little significance in one experiment. It is needed to do many experiments to study the statistical laws. So, each kind of experiment independently does K times. If the optimal of i time belongs to the good enough set, let $s_i = 1$; otherwise $s_i = 0$. Let P_g be statistical frequency. Then, for K times experiment, we have

$$P_g = \frac{\sum_{i=1}^r s_i}{r}, \quad r = 1, 2, \dots, K. \tag{5}$$

From (5), the following can be seen, when K tends to infinity:

$$\lim_{K \rightarrow +\infty} P_g = P'_g, \tag{6}$$

where P'_g is the alignment probability value, but it is generally difficult to obtain. In general, we only need to compute the P_g value which may be tested experimentally.

Let $\bar{P}_A(r)$ be the alignment probability in an experiment by the evaluation method; P_A is average value of $\bar{P}_A(r)$. Consider

$$P_A = \frac{\sum_{i=1}^r \bar{P}_A(r)}{r}, \quad r = 1, 2, \dots, K. \tag{7}$$

Let e_r be the absolute value of error of P_g and P_A ; that is,

$$e_r = |P_A - P_g|. \tag{8}$$

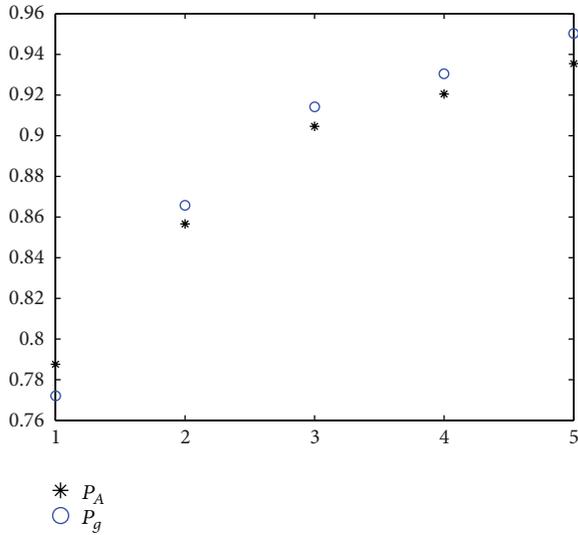


FIGURE 1: Comparison of two probabilities with ratio ≥ 1 .

In the following experiments, we are using e_r as the standard evaluation index.

4.2. Ordinal Performance Evaluation of Nonuniform Sampling. The solution space is sorted according to the fitness values and gets the whole solution space of the sample set, denoted by Ω . We deliberately partition the search space into the same two parts Ω_1 and Ω_2 . Then we sample, respectively, in parts Ω_1 and Ω_2 , respectively. Times are denoted by n_1 and n_2 . Then the total number of samplings is n . Then

$$\frac{n_1}{n_2} = \text{ratio}, \tag{9}$$

$$n_1 + n_2 = n.$$

Let $K = 5000$ and $n = 3000$. Because the value of ration can be divided into two cases. One is no less than 1, and the other is less than 1. So, the following points are discussed.

4.2.1. Ratio ≥ 1 . This case illustrates the sampling times in area Ω_1 more than in area Ω_2 , and the good enough set is in area Ω_1 . The experiment results can be seen in Figures 1 and 2. The abscissa is value of ratio. The values from left to right, respectively, are 1, 2, 5, 10, and 100. In Figure 1, we can see that, with the increasing value of ratio, the sampling point in area Ω_1 is increasing. The probability of obtaining the good enough solution increases as the good enough set is in area Ω_1 . In addition, except for the case of ratio = 1, P_A is slightly higher than P_g . The rest of P_A are lower than P_g . The error of two probabilities seen from Figure 2 is lower and no more than 2% generally.

4.2.2. Ratio < 1 . This case illustrates the sampling times in area Ω_2 more than in area Ω_1 . The experiment results can be seen in Figures 3 and 4. The abscissa is value of ratio. The values from left to right, respectively, are 0.01, 0.1, 0.2, 0.5, and

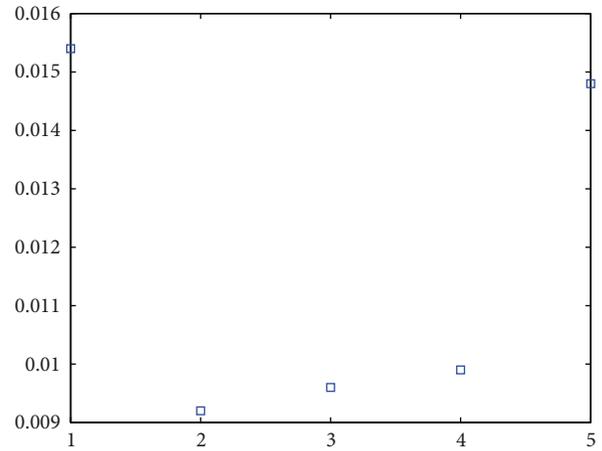


FIGURE 2: Error of two probabilities with ratio ≥ 1 .

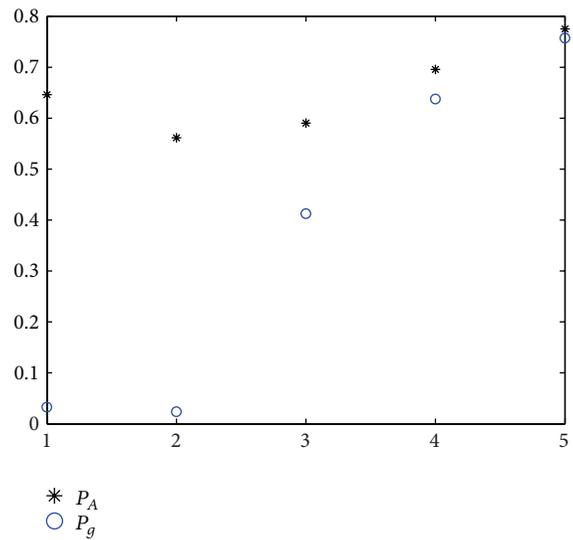


FIGURE 3: Comparison of two probabilities with ratio < 1 .

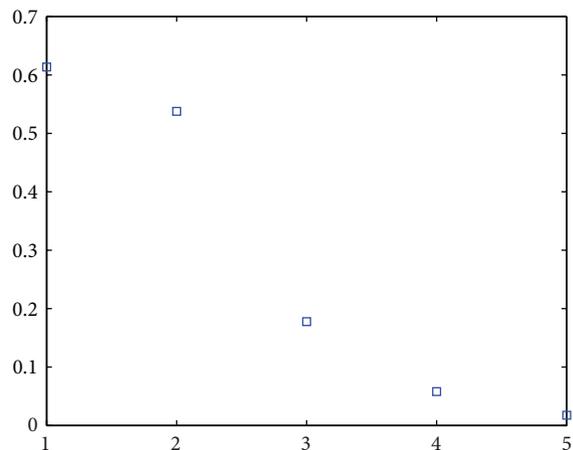


FIGURE 4: Error of two probabilities with ratio < 1 .

TABLE 1: Experimental comparison for ant number m .

m	Best	Worst	Average	STD	P_A	P_g	e_r
2	2.6907	3.2025	2.7671	0.0840	0.8496	0.9985	0.1489
4	2.6907	2.9689	2.7200	0.0521	0.9626	1.0000	0.0374
5	2.6907	2.9689	2.7111	0.0410	0.9792	1.0000	0.0208
8	2.6907	2.8982	2.6966	0.0222	0.9972	1.0000	0.0028
10	2.6907	2.8982	2.6937	0.0163	0.9992	1.0000	0.0008

The best solution, the worst solution, the average solution quality, and the standard deviation in K times running are given.

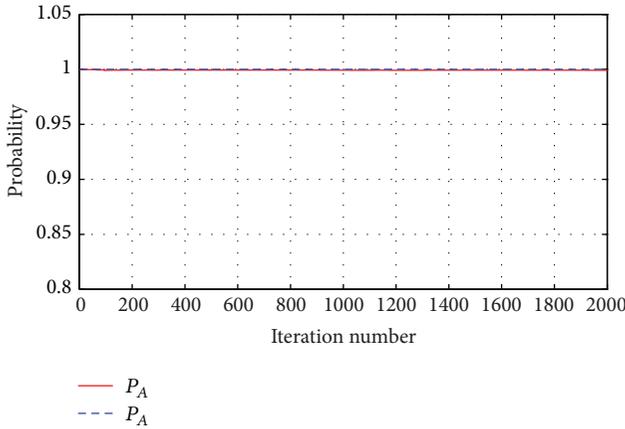


FIGURE 5: Comparison of two probabilities with ACO.

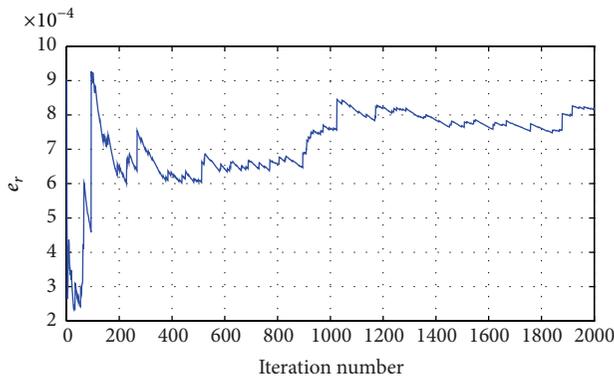


FIGURE 6: Error of two probabilities with ACO.

0.9. In Algorithm 1 and Figure 1, we can see that the error of two probabilities is high. But the error decreases with the ratio increasing.

4.3. *Ordinal Performance Evaluation of ACO.* Let $K = 2000$; the computational results can be seen from Figures 5 and 6. From Figure 5 we can see that the alignment probability of P_A and P_g is close to 1 and the difference is low. The P_A is slightly lower than P_g . It is showed that the evaluation method is conservative. The error range is less than 0.1%. This shows that the calculation result is credible.

In order to further study the relation between the parameters of ant colony algorithm and evaluation results, we focus on the relationship between the maximum number of iterations changes and ant number changes and evaluation of results. The results can be seen from Tables 1 and 2.

First, we study the ant number. The ant number m belongs to the set $\{2, 4, 5, 8, 10\}$. From Table 1 we can see that the value of P_A is increasing with the m increasing. The error of probability is reducing with the m increasing. This shows that the size of solution has some influence on the evaluation method. Second, we study the iteration number N_{\max} which is selected from the set $\{10, 20, 30, 50, 100, 200\}$. From Table 2 we can see that P_A is much less than P_g when N_{\max} is 10. But, with N_{\max} increasing, the error is reducing. The reason is that the information of space is accumulated with N_{\max} increasing. It is showed that the more the utilization of information of the solution space, the more accurate the result.

4.4. *Ordinal Performance Evaluation of PSO and AFS.* We also do the same comparison for PSO and AFS. The results can be seen from Tables 3 and 4. m is particle number in Table 3 and m is fish number in Table 4. From Tables 3 and 4 we can see that the value of P_A is increasing with the m increasing and the maximum number of iterations N_{\max} . The average solution is also improved. It is showed that the solution quality is effect on P_A .

5. Conclusion

A solution assessment method of SI is presented in this paper. Based on the analysis of the existing knowledge foundation, combined with the ordinal optimization theory, the ordinal performance is research target to evaluate solution. Then based on the analysis of characteristics of SI algorithms, the framework of evaluation method is given. The detailed steps of the method are presented. Finally, taking the Hopfield 10-city problem as an example, some simulation experiments are done. The experimental results show that the proposed method is feasible.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

TABLE 2: Experimental comparison for maximum iteration number.

N_{\max}	Best	Worst	Average	STD	P_A	P_g	e_r
10	2.6907	3.1879	2.7695	0.0791	0.7750	0.9985	0.2235
20	2.6907	2.9844	2.7138	0.0435	0.9586	1.0000	0.0414
30	2.6907	3.0504	2.7118	0.0440	0.9748	1.0000	0.0252
50	2.6907	2.9390	2.7094	0.0393	0.9803	1.0000	0.0197
80	2.6907	2.9669	2.7073	0.0384	0.9835	1.0000	0.0165
100	2.6907	2.9669	2.7061	0.0358	0.9853	1.0000	0.0147
200	2.6907	2.8982	2.7022	0.0327	0.9909	1.0000	0.0091

TABLE 3: Experimental comparison for PSO.

m	N_{\max}	Best	Worst	Average	STD	P_A	P_g	e_r
10	30	2.6907	3.5976	3.0174	0.1543	0.6720	0.7425	0.0713
10	50	2.6907	3.3618	2.9483	0.1310	0.7861	0.8890	0.1029
10	60	2.6907	3.3582	2.9227	0.1222	0.8257	0.9340	0.1083
10	80	2.6907	3.3038	2.8923	0.1104	0.8894	0.9745	0.0851
10	100	2.6907	3.2328	2.8685	0.1029	0.9262	0.9840	0.0578
20	60	2.6907	3.2275	2.8516	0.0947	0.9480	0.9970	0.0490
20	100	2.6907	3.0861	2.8068	0.0736	0.9898	1.0000	0.0102

TABLE 4: Experimental comparison for AFS.

m	N_{\max}	Best	Worst	Average	STD	P_A	P_g	e_r
5	50	2.6907	3.1556	2.7439	0.1017	0.7036	0.9985	0.2949
8	50	2.6907	3.0909	2.7108	0.0569	0.8215	1.0000	0.1785
10	50	2.6907	3.0783	2.7034	0.0440	0.8639	1.0000	0.1361
15	50	2.6907	3.0302	2.6929	0.0153	0.9397	1.0000	0.0603
10	80	2.6907	3.0344	2.6970	0.0293	0.8855	1.0000	0.1145
10	100	2.6907	3.0830	2.6956	0.0280	0.8898	1.0000	0.1102
20	80	2.6907	2.7782	2.6909	0.0039	0.9743	1.0000	0.0257
20	100	2.6907	2.7782	2.6909	0.0048	0.9750	1.0000	0.0250

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant nos. 61305149, 61104222, and 61304175) and the Science Fundamental Research Project of Jiangsu Normal University (Grant no. 12XL063). The authors are also thankful to the anonymous referees.

References

- [1] J. Kennedy, R. C. Eberhart, and Y. H. Shi, *Swarm Intelligence*, Morgan Kaufmann, San Mateo, Calif, USA, 2001.
- [2] M. Dorigo and T. Stützle, *Ant Colony Optimization*, MIT Press, Cambridge, Mass, USA, 2004.
- [3] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.
- [4] D. Karaboga and B. Akay, "A survey: algorithms simulating bee swarm intelligence," *Artificial Intelligence Review*, vol. 31, no. 1–4, pp. 61–85, 2009.
- [5] X.-L. Li, Z.-J. Shao, and J.-X. Qian, "An optimizing method based on autonomous animals: fish-swarm algorithm," *System Engineering Theory and Practice*, vol. 22, no. 11, pp. 32–38, 2002.
- [6] M. Dorigo, V. Maniezzo, and A. Coloni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, B: Cybernetics*, vol. 26, no. 1, pp. 29–41, 1996.
- [7] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.
- [8] T. Stützle and H. H. Hoos, "MAX-MIN ant system," *Future Generation Computer Systems*, vol. 16, no. 8, pp. 889–914, 2000.
- [9] Z. J. Zhang and Z. R. Feng, "Two-stage updating pheromone for invariant ant colony optimization algorithm," *Expert Systems with Applications*, vol. 39, no. 1, pp. 706–712, 2012.
- [10] J. Aspnes and O. Waarts, "Compositional competitiveness for distributed algorithms," *Journal of Algorithms*, vol. 54, no. 2, pp. 127–151, 2005.
- [11] J. Hsler, P. Cruz, A. Hall, and C. M. Fonseca, "On optimization and extreme value theory," *Methodology and Computing in Applied Probability*, vol. 5, no. 2, pp. 183–195, 2003.
- [12] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific Press, Belmont, Mass, USA, 1999.

- [13] H. H. Hoos and T. Stützle, *Stochastic Local Search: Foundations and Applications*, Morgan Kaufmann Press, San Francisco, Calif, USA, 2005.
- [14] M. A. Montes de Oca, T. Stützle, M. Birattari, and M. Dorigo, "Frankenstein's PSO: a composite particle swarm optimization algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 1120–1132, 2009.
- [15] Y. C. Ho, R. S. Sreenivas, and P. Vakili, "Ordinal optimization of DEDS," *Discrete Event Dynamic Systems*, vol. 2, no. 1, pp. 61–88, 1992.
- [16] Z. Shen, Y.-C. Ho, and Q.-C. Zhao, "Ordinal optimization and quantification of heuristic designs," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 19, no. 3, pp. 317–345, 2009.
- [17] Z. Shen, Q.-C. Zhao, and Q.-S. Jia, "Quantifying heuristics in the ordinal optimization framework," *Discrete Event Dynamic Systems: Theory and Applications*, vol. 20, no. 4, pp. 441–471, 2010.

Review Article

Application of Reinforcement Learning in Cognitive Radio Networks: Models and Algorithms

Kok-Lim Alvin Yau,¹ Geong-Sen Poh,² Su Fong Chien,³ and Hasan A. A. Al-Rawi¹

¹ Faculty of Science and Technology, Sunway University, No. 5 Jalan Universiti, Bandar Sunway, 46150 Petaling Jaya, Selangor, Malaysia

² University Malaysia of Computer Science & Engineering, Jalan Alamanda 2, Presint 16, 62150 Putrajaya, Wilayah Persekutuan Putrajaya, Malaysia

³ Department of Mathematical Modeling Laboratory, Mimos Berhad, Technology Park Malaysia, 57000 Kuala Lumpur, Malaysia

Correspondence should be addressed to Kok-Lim Alvin Yau; koklimy@sunway.edu.my

Received 11 April 2014; Accepted 25 April 2014; Published 5 June 2014

Academic Editor: T. O. Ting

Copyright © 2014 Kok-Lim Alvin Yau et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Cognitive radio (CR) enables unlicensed users to exploit the underutilized spectrum in licensed spectrum whilst minimizing interference to licensed users. Reinforcement learning (RL), which is an artificial intelligence approach, has been applied to enable each unlicensed user to observe and carry out optimal actions for performance enhancement in a wide range of schemes in CR, such as dynamic channel selection and channel sensing. This paper presents new discussions of RL in the context of CR networks. It provides an extensive review on how most schemes have been approached using the traditional and enhanced RL algorithms through state, action, and reward representations. Examples of the enhancements on RL, which do not appear in the traditional RL approach, are rules and cooperative learning. This paper also reviews performance enhancements brought about by the RL algorithms and open issues. This paper aims to establish a foundation in order to spark new research interests in this area. Our discussion has been presented in a tutorial manner so that it is comprehensive to readers outside the specialty of RL and CR.

1. Introduction

Cognitive radio (CR) [1] is the next generation wireless communication system that enables unlicensed or Secondary Users (SUs) to explore and use underutilized licensed spectrum (or white spaces) owned by the licensed or Primary Users (PUs) in order to improve the overall spectrum utilization. The CR technology improves the availability of bandwidth at each SU, and so it enhances the SU network performance. Reinforcement learning (RL) has been applied in CR so that the SUs can observe, learn, and take optimal actions on their respective local operating environment. For example, a SU observes its spectrum to identify white spaces, learns the best possible channels for data transmissions, and takes actions such as to transmit data in the best possible channel. Examples of schemes in which RL has been applied are dynamic channel selection [2], channel sensing [3], and routing [4]. To the best of our knowledge, the discussion

on the application of RL in CR networks is new albeit the importance of RL in achieving the fundamental concept of CR, namely, cognition cycle (see Section 2.2.1). This paper provides an extensive review on various aspects of the application of RL in CR networks, particularly, the components, features, and enhancements of RL. Most importantly, we present how the traditional and enhanced RL algorithms have been applied to approach most schemes in CR networks. Specifically, for each new RL model and algorithm which is our focus, we present the purpose(s) of a CR scheme, followed by in-depth discussion on its associated RL model (i.e., state, action, and reward representations) which characterizes the purposes, and finally the RL algorithm which aims to achieve the purpose. Hence, this paper serves as a solid foundation for further research in this area, particularly, for the enhancement of RL in various schemes in the context of CR, which can be achieved using new extensions in existing schemes, and for the application of RL in new schemes.

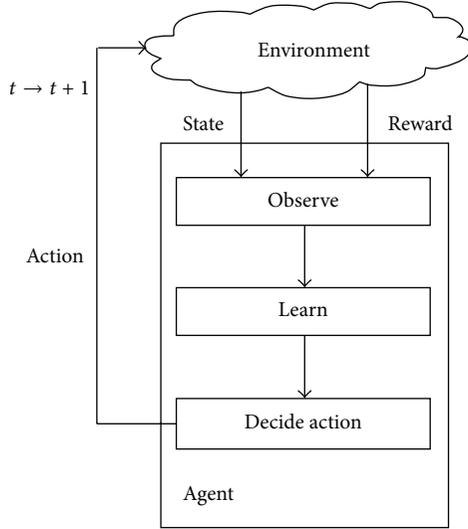


FIGURE 1: A simplified RL model.

The rest of this paper is organized as follows. Section 2 presents RL and CR networks. Section 3 presents various components, features, and enhancements of RL in the context of CR networks. Section 4 presents various RL algorithms in the context of CR networks. Section 5 presents performance enhancements brought about by the RL algorithms in various schemes in CR networks. Section 6 presents open issues. Section 7 presents conclusions.

2. Reinforcement Learning and Cognitive Radio Networks

This section presents an overview of RL and CR networks.

2.1. Reinforcement Learning. Reinforcement learning is an unsupervised and online artificial intelligence technique that improves system performance using simple modeling [5]. Through unsupervised learning, there is no external teacher or critic to oversee the learning process, and so, an agent learns knowledge about the operating environment by itself. Through online learning, an agent learns knowledge on the fly while carrying out its normal operation, rather than using empirical data or experimental results from the laboratory.

Figure 1 shows a simplified version of a RL model. At a particular time instant, a learning agent or a decision maker observes state and reward from its operating environment, learns, decides, and carries out its action. The important representations in the RL model for an agent are as follows.

- (i) *State* represents the decision-making factors, which affect the reward (or network performance), observed by an agent from the operating environment. Examples of states are the channel utilization level by PUs and channel quality.
- (ii) *Action* represents an agent's action, which may change or affect the state (or operating environment) and reward (or network performance), and so

the agent learns to take optimal actions at most of the times.

- (iii) *Reward* represents the positive or negative effects of an agent's action on its operating environment in the previous time instant. In other words, it is the consequence of the previous action on the operating environment in the form of network performance (e.g., throughput).

At any time instant, an agent observes its state and carries out a proper action so that the state and reward, which are the consequences of the action, improve in the next time instant. Generally speaking, RL estimates the reward of each state-action pair, and this constitutes knowledge. The most important component in Figure 1 is the learning engine that provides knowledge to the agent. We briefly describe how an agent learns. At any time instant, an agent's action may affect the state and reward for better or for worse or maintain the status quo; and this in turn affects the agent's next choice of action. As time progresses, the agent learns to carry out a proper action given a particular state. As an example of the application of the RL model in CR networks, the learning mechanism is used to learn channel conditions in a dynamic channel selection scheme. The state represents the channel utilization level by PUs and channel quality. The action represents a channel selection. Based on an application, the reward represents distinctive performance metrics such as throughput and successful data packet transmission rate. Lower channel utilization level by PUs and higher channel quality indicate better communication link, and hence the agent may achieve better throughput performance (reward). Therefore, maximizing reward provides network performance enhancement.

Q-learning [5] is a popular technique in RL, and it has been applied in CR networks. Denote decision epochs by $t \in T = \{1, 2, \dots\}$; the knowledge possessed by agent i for a particular state-action pair at time t is represented by Q-function as follows:

$$Q_{t+1}^i(s_t^i, a_t^i) \leftarrow (1 - \alpha) Q_t^i(s_t^i, a_t^i) + \alpha \left[r_{t+1}^i(s_{t+1}^i) + \gamma \max_{a \in A} Q_t^i(s_{t+1}^i, a) \right], \quad (1)$$

where

- (i) $s_t^i \in S$ represents state,
- (ii) $a_t^i \in A$ represents action,
- (iii) $r_{t+1}^i(s_{t+1}^i) \in R$ represents delayed rewards, which is received at time $t + 1$ for an action taken at time t ,
- (iv) $0 \leq \gamma \leq 1$ represents discount factor. The higher the value of γ , the greater the agent relies on the discounted future reward $\gamma \max_{a \in A} Q_t^i(s_{t+1}^i, a)$ compared to the delayed reward $r_{t+1}^i(s_{t+1}^i)$,
- (v) $0 \leq \alpha \leq 1$ represents learning rate. The higher the value of α , the greater the agent relies on the delayed reward $r_{t+1}^i(s_{t+1}^i)$ and the discounted future reward $\gamma \max_{a \in A} Q_t^i(s_{t+1}^i, a)$, compared to the Q-value $Q_t^i(s_t^i, a_t^i)$ at time t .

At decision epoch t , agent i observes its operating environment to determine its current state s_t^i . Based on the s_t^i , the agent chooses an action a_t^i . Next, at decision epoch $t + 1$, the state s_t^i changes to s_{t+1}^i as a consequence of the action a_t^i , and the agent receives delayed reward $r_{t+1}^i(s_{t+1}^i)$. Subsequently, the Q-value $Q_{t+1}^i(s_t^i, a_t^i)$ is updated using (1). Note that, in the remaining decision epochs at time $t, t + 1, \dots$, the agent is expected to take optimal actions with regard to the states; hence, Q-value is updated using a maximized discounted future reward $\gamma \max_{a \in A} Q_t^i(s_{t+1}^i, a)$. As this procedure evolves through time, agent i receives a sequence of rewards and the Q-value converges. Q-learning searches for an optimal policy at all time instants through maximizing value function $V^\pi(s_t^i)$ as shown below:

$$V^\pi(s_t^i) = \max_{a \in A} (Q_t^i(s_t^i, a)). \quad (2)$$

Hence, the policy (or action selection) for agent i is as follows:

$$\pi_i(s_t^i) = \arg \max_{a \in A} (Q_t^i(s_t^i, a)). \quad (3)$$

The update of the Q-value in (1) does not cater for the actions that are never chosen. Exploitation chooses the best-known action, or the greedy action, at all time instants for performance enhancement. Exploration chooses the other nonoptimal actions once in a while to improve the estimates of all Q-value in order to discover better actions. While Figure 1 shows a single agent, the presence of multiple agents is feasible. In the context of CR networks, a rigorous proof of the convergence of Q-value in the presence of multiple SUs has been shown in [6].

The advantages of RL are as follows:

- (i) instead of tackling every single factor that affects the system performance, RL models the system performance (e.g., throughput) that covers a wide range of factors affecting the throughput performance including the channel utilization level by PUs and channel quality and, hence, its simple modeling approach;
- (ii) prior knowledge of the operating environment is not necessary; and so a SU can learn the operating environment (e.g., channel quality) as time goes by.

2.2. Cognitive Radio Networks. Traditionally, spectrum allocation policy has been partitioning radio spectrum into smaller ranges of licensed and unlicensed frequency bands (also called channels). The licensed channels provide exclusive channel access to licensed users or PUs. Unlicensed users or SUs, such as the popular wireless communication systems IEEE 802.11, access unlicensed channels without incurring any monetary cost, and they are forbidden to access any of the licensed channels. Examples of unlicensed channels are Industrial, Scientific, and Medical (ISM) and Unlicensed National Information Infrastructure (UNII) bands. While the licensed channels have been underutilized, the opposite phenomenon has been observed among the unlicensed channels.

Cognitive radio enables SUs to explore radio spectrum and use white spaces whilst minimizing interference to PUs.

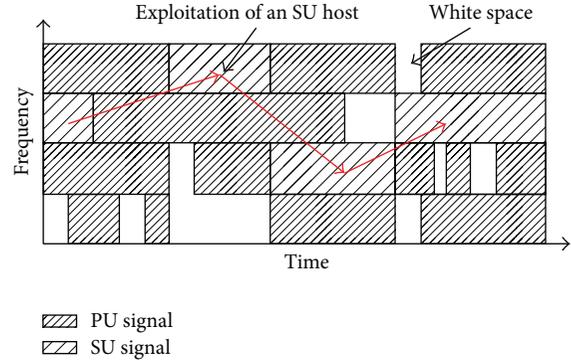


FIGURE 2: A SU exploits white spaces across various channels.

The purpose is to improve the availability of bandwidth at each SU, hence improving the overall utilization of radio spectrum. CR helps the SUs to establish a “friendly” environment, in which the PUs and SUs coexist without causing interference with each other as shown in Figure 2. In Figure 2, a SU switches its operating channel across various channels from time to time in order to utilize white spaces in the licensed channels. Note that each SU may observe different white spaces, which are location dependent. The SUs must sense the channels and detect the PUs’ activities whenever they reappear in white spaces. Subsequently, the SUs must vacate and switch their respective operating channel immediately in order to minimize interference to PUs. For a successful communication, a particular white space must be available at both SUs in a communication node pair.

The rest of this subsection is organized as follows. Section 2.2.1 presents cognition cycle, which is an essential component in CR. Section 2.2.2 represents various application schemes in which RL has been applied to provide performance enhancement.

2.2.1. Cognition Cycle. Cognition cycle [7], which is a well-known concept in CR, is embedded in each SU to achieve context awareness and intelligence in CR networks. Context awareness enables a SU to sense and be aware of its operating environment; while intelligence enables the SU to observe, learn, and use the white spaces opportunistically so that a static predefined policy is not required while providing network performance enhancement.

The cognition cycle can be represented by a RL model as shown in Figure 1. The RL model can be tailored to fit well with a wide range of applications in CR networks. A SU can be modeled as a learning agent. At a particular time instant, the SU agent observes state and reward from its operating environment, learns, decides, and carries out action on the operating environment in order to maximize network performance. Further description on RL-based cognition cycle is presented in Section 2.1.

2.2.2. Application Schemes. Reinforcement learning has been applied in a wide range of schemes in CR networks for SU performance enhancements, whilst minimizing interference

to PUs. The schemes are listed as follows, and the nomenclatures (e.g., (A1) and (A2)) are used to represent the respective application schemes throughout the paper.

- (A1) *Dynamic Channel Selection (DCS)*. The DCS scheme selects operating channel(s) with white spaces for data transmission whilst minimizing interference to PUs. Yau et al. [8, 9] propose a DCS scheme that enables SUs to learn and select channels with low packet error rate and low level of channel utilization by PUs in order to enhance QoS, particularly throughput and delay performances.
- (A2) *Channel Sensing*. Channel sensing senses for white spaces and detects the presence of PU activities. In [10], the SU reduces the number of sensing channels and may even turn off channel sensing function if its operating channel has achieved the required successful transmission rate in order to enhance throughput performance. In [11], the SU determines the durations of channel sensing, time of channel switching, and data transmission, respectively, in order to enhance QoS, particularly throughput, delay, and packet delivery rate performances. Both [10, 11] incorporate DCS (A1) into channel sensing in order to select operating channels. Due to the environmental factors that can deteriorate transmissions (e.g., multipath fading and shadowing), Lo and Akyildiz [3] propose a cooperative channel sensing scheme, which combines sensing outcomes from cooperating one-hop SUs, to improve the accuracy of PU detection.
- (A3) *Security Enhancement*. Security enhancement scheme [12] aims to ameliorate the effects of attacks from malicious SUs. Vucevic et al. [13] propose a security enhancement scheme to minimize the inaccurate sensing outcomes received from neighboring SUs in channel sensing (A2). A SU becomes malicious whenever it sends inaccurate sensing outcomes, intentionally (e.g., Byzantine attacks) or unintentionally (e.g., unreliable devices). Wang et al. [14] propose an antijamming scheme to minimize the effects of jamming attacks from malicious SUs, which constantly transmit packets to keep the channels busy at all times so that SUs are deprived of any opportunities to transmit.
- (A4) *Energy Efficiency Enhancement*. Energy efficiency enhancement scheme aims to minimize energy consumption. Zheng and Li [15] propose an energy-efficient channel sensing scheme to minimize energy consumption in channel sensing. Energy consumption varies with activities, and it increases from sleep, idle, to channel sensing. The scheme takes into account the PU and SU traffic patterns and determines whether a SU should enter sleep, idle, or channel sensing modes. Switching between modes should be minimized because each transition between modes incurs time delays.
- (A5) *Channel Auction*. Channel auction provides a bidding platform for SUs to compete for white spaces.

Chen and Qiu [16] propose a channel auction scheme that enables the SUs to learn the policy (or action selection) of their respective SU competitors and place bids for white spaces. This helps to allocate white spaces among the SUs efficiently and fairly.

- (A6) *Medium Access Control (MAC)*. MAC protocol aims to minimize packet collision and maximize channel utilization in CR networks. Li et al. [17] propose a collision reduction scheme that reduces the probability of packet collision among PUs and SUs, and it has been shown to increase throughput and to decrease packet loss rate among the SUs. Li et al. [18] propose a retransmission policy that enables a SU to determine how long it should wait before transmission in order to minimize channel contention.
- (A7) *Routing*. Routing enables each SU source or intermediate node to select its next hop for transmission in order to search for the best route(s), which normally incurs the least cost or provides the highest amount of rewards, to the SU destination node. Each link within a route has different types and levels of costs, such as queuing delay, available bandwidth or congestion level, packet loss rate, energy consumption level, and link reliability, as well as changes in network topology as a result of irregular node's movement speed and direction.
- (A8) *Power Control*. Yao and Feng [19] propose a power selection scheme that selects an available channel and a power level for data transmission. The purpose is to improve its Signal-to-Noise Ratio (SNR) in order to improve packet delivery rate.

3. Reinforcement Learning in the Context of Cognitive Radio Networks: Components, Features, and Enhancements

This section presents the components of RL, namely, state, action, reward, discounted reward, and Q-function; as well as the features of RL, namely, exploration and exploitation, updates of learning rate, rules and cooperative learning. The components and features of RL (see Section 2.1) are presented in the context of CR. For each component and feature, we show the traditional approach and subsequently the alternative or enhanced approaches with regard to modeling, representing, and applying them in CR networks. This section serves as a foundation for further research in this area, particularly, the application of existing features and enhancements in current schemes in RL models for either existing or new schemes.

Note that, for improved readability, the notations (e.g., s_t^i and a_t^i) used in this paper represent the same meaning throughout the entire paper, although different references in the literature may use different notations for the same purpose.

3.1. State. Traditionally, each state is comprised of a single type of information. For instance, in [11], each state

$s_t^i \in S = \{1, 2, \dots, K\}$ represents a single channel out of K channels available for data transmission. The state may be omitted in some cases. For instance, in [10], the state and action representations are similar, so the state is not represented. The traditional state representation can be enhanced in the context of CR as described next.

Each state can be comprised of several types of information. For instance, Yao and Feng [19] propose a joint DCS (A1) and power allocation (A8) scheme in which each state is comprised of three-tuple information; specifically, $\mathbf{s}_t^i = (s_{1,t}^i, s_{2,t}^i, s_{3,t}^i) \in S_1 \times S_2 \times S_3$. The substate $s_{1,t}^i \in S_1 = \{1, 2, \dots, N_{\text{SU}}\}$ represents the number of SU agents, $s_{2,t}^i \in S_2 = \{1, 2, \dots, N_{\text{SU-SU}}\}$ represents the number of communicating SU agents, and $s_{3,t}^i \in S_3 = \{p_1, p_2, \dots, p_{N_p}\}$ represents the received power on each channel.

The value of a state may deteriorate as time goes by. For instance, Lundén et al. [20] propose a channel sensing (A2) scheme in which each state $s_{k,t}^i \in \{0 \leq p_{\text{idle},k}^i \leq 1\}$ represents SU agent i 's belief (or probability) that channel k is idle (or the absence of PU activity). Note that the belief value of channel k deteriorates whenever the channel is not sensed recently, and this indicates the diminishing confidence in the belief that channel k remains idle. Denote a small step size by δ (i.e., $\delta = 0.01$); the state value of channel k deteriorates if it is not updated at each time instant; specifically, $s_{k,t+1}^i = s_{k,t}^i - \delta$.

3.2. Action. Traditionally, each action represents a single action a_t^i out of a set of possible actions A . For instance, in [10], each action $a_t^i \in A = \{1, 2, \dots, K\}$ represents a single channel out of the K channels available for data transmission. The traditional action representation can be enhanced in the context of CR as described next.

Each action $a_t^i \in A$ can be further divided into various levels. As an example, Yao and Feng [19] propose a joint DCS (A1) and power allocation (A8) scheme in which each action $a_t^i \in A = \{p_1, p_2, \dots, p_K\}$ represents a channel selection, and each $p_k \in P_{\text{PA}} = \{p_1, p_2, \dots, p_{N_{\text{PA}}}\}$ represents a power level allocation with N_{PA} being the number of power levels. As another example, Zheng and Li [15] propose an energy efficiency enhancement (A4) scheme in which there are four kinds of actions, namely, transmit, idle, sleep, and sense channel. The sleepaction $a_{\text{sp},t}^i \in A = \{a_{\text{sp}1}, a_{\text{sp}2}, \dots, a_{N_{\text{sp}}}\}$ represents a sleep level with N_{sp} being the number of sleep levels. Note that different sleep level incurs different amount of energy consumption.

3.3. Delayed Reward. Traditionally, each delayed reward represents the amount of performance enhancement achieved by a state-action pair. A single reward computation approach is applicable to all state-action pairs. As an example, in [2], $r_{t+1}^i(a_{t+1}^i) \in R = \{1, -1\}$ represents the reward and cost values of 1 and -1 for each successful and unsuccessful transmission, respectively. As another example, in [8], $r_{t+1}^i(a_{t+1}^i)$ represents the amount of throughput achieved within a time window. The traditional reward representation can be enhanced in the context of CR as described next.

The delayed reward can be computed differently for distinctive actions. As an example, in a joint DCS (A1) and channel sensing (A2) scheme, Felice et al. [21] compute the delayed rewards in two different ways based on the types of actions: channel sensing a_{se} and data transmission a_{tx} . Firstly, a SU agent calculates delayed reward $r_{t+1}^i(s_t^i, a_{\text{se},t}^i)$ at time instant $t+1$. The $r_{t+1}^i(s_t^i, a_{\text{se},t}^i)$ indicates the likelihood of the existence of PU activities in channel s_t^i whenever action $a_{\text{se},t}^i$ is taken. Specifically, $r_{t+1}^i(s_t^i, a_{\text{se},t}^i) = \sum_{j=0}^{N_{\text{nbr},i}} d_{i,j} / N_{\text{nbr},i}$ where $N_{\text{nbr},i}$ indicates the number of neighboring SU agents, while $d_{i,j}$, which is a binary value, indicates the existence of PU activities as reported by SU neighbor agent $j \in N_{\text{nbr},i}$. Secondly, a SU agent calculates delayed reward $r_{t+1}^i(s_t^i, a_{\text{tx},t}^i)$ at time instant $t+1$. The $r_{t+1}^i(s_t^i, a_{\text{tx},t}^i)$ indicates the successful transmission rate, which takes into account the aggregated effect of interference from PU activities whenever action $a_{\text{tx},t}^i$ is taken. Specifically, $r_{t+1}^i(s_t^i, a_{\text{tx},t}^i) = \sum_{j=0}^{N_{\text{DATA},i}} \text{ACK}_{i,j} / \sum_{j=0}^{N_{\text{DATA},i}} \text{DATA}_{i,j}$ where $N_{\text{DATA},i}$ indicates the number of data packets sent by SU agent i , $\text{ACK}_{i,j}$ indicates the number of acknowledgment packets received by SU agent i , and $\text{DATA}_{i,j}$ indicates the number of data packets being transmitted by SU agent i .

Jouini et al. [22] apply an Upper Confidence Bound (UCB) algorithm to compute delayed rewards in a dynamic and uncertain operating environment (e.g., operating environment with inaccurate sensing outcomes), and it has been shown to improve throughput performance in DCS (A1). The main objective of this algorithm is to determine the upper confidence bounds for all rewards and subsequently use them to make decisions on action selection. The rewards are uncertain, and the uncertainty is caused by the dynamicity and uncertainty of the operating environment. Let $N_{a^i}(t)$ represent the number of times an action $a^i \in A$ has been taken on the operating environment up to time t ; an agent i calculates the upper confidence bounds of all delayed rewards as follows:

$$B_t^i(a^i, N_{a^i}(t)) = \bar{r}_t^i(a^i, N_{a^i}(t)) + U_t^i(a^i, N_{a^i}(t)), \quad (4)$$

where $\bar{r}_t^i(a^i, N_{a^i}(t)) = \sum_{j=0}^{t-1} r_j^i(a_j^i) / N_{a^i}(t)$ is the mean reward, and $U_t^i(a^i, N_{a^i}(t))$ is the upper confidence bias being added to the mean. Note that $r_j^i(a_j^i) = 0$ if a_j^i is not chosen at time instant j . The $U_t^i(a^i, N_{a^i}(t))$ is calculated as follows:

$$U_t^i(a^i, N_{a^i}(t)) = \sqrt{\frac{\beta \cdot \ln(t)}{N_{a^i}(t)}}, \quad (5)$$

where exploration coefficient $\beta > 1$ is a constant empirical factor. For instance, $\beta = 1.2$ in [22, 23].

The UCB algorithm selects actions with the highest upper confidence bounds, and so (3) is rewritten as follows:

$$\pi_i(a_t^i) = \arg \max_{a \in A} B_t^i(a, N_{a^i}(t)). \quad (6)$$

3.4. Discounted Reward. Traditionally, the discounted reward has been applied to indicate the dependency of Q -value

on future rewards. Based on an application, the discounted reward may be omitted with $\gamma = 0$ to show the lack of dependency on future rewards, and this approach is generally called the myopic approach. As an example, Li [6] and Chen et al. [24] apply Q-learning in DCS (A1), and the Q-function in (1) is rewritten as follows:

$$Q_{t+1}^i(a_t^i) \leftarrow (1 - \alpha)Q_t^i(a_t^i) + \alpha \cdot r_{t+1}^i(a_t^i). \quad (7)$$

3.5. Q-Function. The traditional Q-function (see (1)) has been widely applied to update Q-value in CR networks. The traditional Q-function can be enhanced in the context of CR as described next.

Lundén et al. [20] apply a linear function approximation-based approach to reduce the dimensionality of the large state-action spaces (or reduce the number of state-action pairs) in a collaborative channel sensing (A2) scheme. A linear function $f(s_t^i, a_t^i)$ provides a matching value $\theta_t(s_t^i, a_t^i)$ for a state-action pair. The matching value $\theta_t(s_t^i, a_t^i)$, which shows the appropriateness of a state-action pair, is subsequently applied in Q-value computation. The linear function $f(s_t^i, a_t^i)$ is normally fixed (or hard-coded), and various kinds of linear functions are possible to indicate the appropriateness of a state-action pair based on prior knowledge. For instance, $f(s_t^i, a_t^i)$ yields a value that represents the level of desirability of a certain number of SU agents sensing a particular channel [20]. Higher $f(s_t^i, a_t^i)$ value indicates that the number of SU agents sensing a particular channel is closer to a desirable number. Using a fixed linear function $f(s_t^i, a_t^i)$, the learning problem is transformed into learning the matching value $\theta_t(s_t^i, a_t^i)$ as follows:

$$Q_t^i(s_t^i, a_t^i) = \theta_t(s_t^i, a_t^i) \cdot f(s_t^i, a_t^i). \quad (8)$$

The parameter $\theta_t(s_t^i, a_t^i)$ is updated as follows:

$$\begin{aligned} \theta_{t+1}(s_t^i, a_t^i) = & \theta_t(s_t^i, a_t^i) + \alpha \left[r_{t+1}^i(s_t^i) + \gamma \cdot Q_t^i(s_{t+1}^i, a_{t+1}^i) \right. \\ & \left. - Q_t^i(s_t^i, a_t^i) \right] \cdot f(s_t^i, a_t^i). \end{aligned} \quad (9)$$

3.6. Exploration and Exploitation. Traditionally, there are two popular approaches to achieve a balanced trade-off between exploration and exploitation, namely, softmax and ϵ -greedy [5]. For instance, Yau et al. [8] use the ϵ -greedy approach in which an agent explores with a small probability ϵ (i.e., $\epsilon = 0.1$) and exploits with probability $1 - \epsilon$. Essentially, these approaches aim to control the frequency of exploration so that the best-known action is taken at most of the times. The traditional exploration and exploitation approach can be enhanced in the context of CR as described next.

In [3, 25], using the softmax approach, an agent selects actions based on a Boltzman distribution; specifically, the probability of selecting an action a_t in state s_t is as follows:

$$P(s_t^i, a_t^i) = \frac{e^{Q_t^i(s_t^i, a_t^i)/\tau_t}}{\sum_{j=1}^K e^{Q_t^i(s_t^i, a_j^i)/\tau_t}}, \quad (10)$$

where τ_t is a time-varying parameter called temperature. Higher temperature value indicates more exploration, while smaller temperature value indicates more exploitation. Denote the time duration during which exploration actions are being chosen by T_e ; the temperature τ_t is decreased as time goes by so that the agent performs more exploitation as follows:

$$\tau_t = -\frac{(\tau_0 - \tau_e) \cdot t}{T_e} + \tau_0, \quad (11)$$

where τ_0 and τ_e are initial and final values of temperature, respectively. Note that, due to the dynamicity of the operating environment, exploration is necessary at all times, and so $\tau_t \geq \tau_0$.

In [21], using the ϵ -greedy approach, an agent uses a simple approach to decrease exploration probability as time goes by as follows:

$$\epsilon_{t+1} = \max\{\delta \cdot \epsilon_t, \epsilon_{\min}\}, \quad (12)$$

where $0 \leq \delta \leq 1$ is a discount factor and ϵ_{\min} is the minimum exploration probability.

3.7. Other Features and Enhancements. This section presents other features and enhancements on the traditional RL approach found in various schemes for CR networks, including updates of learning rate, rules, and cooperative learning.

3.7.1. Updates of Learning Rate. Traditionally, the learning rate α is a constant value [16]. The learning rate α may be adjusted as time goes by because higher value of α may compromise the RL algorithm's accuracy to converge to a correct action in a finite number of steps [26]. In [27], the learning rate reduces as time goes by using $\alpha(t) = \alpha(t - 1) - \Delta$, where Δ is a small value to provide smooth transition between steps. In [14], the learning rate is updated using $\alpha(t) = \Delta \cdot \alpha(t - 1)$.

3.7.2. Rules. Rules determine a feasible set of actions for each state. The traditional RL algorithm does not apply rules although it is an important component in CR networks. For instance, in order to minimize interference with PUs, the SUs must comply with the timing requirements set by the PUs, such as the time interval that a SU must vacate its operating channel after any detection of PU activities.

As an example, Zheng and Li [15] propose an energy efficiency enhancement scheme in which there are four kinds of actions, namely, transmit, idle, sleep, and sense channel. Rules are applied so that the feasible set of actions is comprised of idle and sleep whenever the state indicates that there is no packet in the buffer. As another example, Peng et al. [4] propose a routing scheme, specifically, a next hop selection scheme in which the action represents the selection of a next hop out of a set of SU next hops. Rules are applied so that the feasible set of actions is limited to SU next hops with a certain level of SNR, as well as with shorter distance between next hop and the hop after next. The purposes of the rules are to reduce transmission delays and to ensure

high-quality reception. Further description about [4, 15] is found in Table 1.

3.7.3. Cooperative Learning. Cooperative learning enables neighbor agents to share information among themselves in order to expedite the learning process. The exchanged information can be applied in the computation of Q-function. The traditional RL algorithm does not apply cooperative learning, although it has been investigated in multiagent reinforcement learning (MARL) [28].

Felice et al. [11] propose a cooperative learning approach to reduce exploration. The Q-value is exchanged among the SU agents, and it is used in the Q-function computation to update Q-value. Each SU agent i keeps track of its own Q-value $Q_t^i(s_t^i)$, and it is updated using the similar way to [6] (see Section 3.4). At any time instant, each agent i receives Q-value from its neighbor agent $j \in J = \{1, 2, \dots, N_{\text{nbr},i}\}$. The agent keeps a vector of Q-value $\mathbf{Q}_t^i(s_t^i)$ with $s_t^i \in S$. For the case $s_t^j = s_t^i$, the Q-value $Q_t^i(s_t^i)$ is updated as follows:

$$Q_t^i(s_t^i) = Q_t^i(s_t^i) + w(s_t^i, j) \cdot (Q_t^j(s_t^i) - Q_t^i(s_t^i)), \quad (13)$$

where $w(s_t^i, j)$ defines the weight assigned to cooperation with neighbor agent j . Similar approach has been applied in [25], and the Q-value $Q_t^i(s_t^i)$ is updated based on the weight $w(s_t^i, j)$ as follows:

$$Q_t^i(s_t^i) = (1 - w(s_t^i, j)) \cdot Q_t^i(s_t^i) + w(s_t^i, j) \cdot Q_t^j(s_t^i). \quad (14)$$

In [11], the weight $w(s_t^i, j)$ depends on how much a neighbor agent j can contribute to the accurate estimation of value function $V_t^i(s_t^i)$, such as the physical distance between agent i and j . In [25], the weight $w(s_t^i, j)$ depends on the accuracy of the exchanged Q-value $Q_t^i(s_t^i)$ (or expert value $E_t^i(s_t^i)$ as described next) and the physical distance between agent i and j .

In [25], an agent exchanges its Q-value with its neighboring agents only if the expert value $E_t^i(s_t^i)$ for Q-value $Q_t^i(s_t^i)$ is greater than a particular threshold. The expert value $E_t^i(s_t^i)$ indicates the accuracy of the Q-value $Q_t^i(s_t^i)$. For instance, in [25], the Q-value $Q_t^i(s_t^i)$ indicates the availability of white spaces in channel s_t^i , and so greater deviation in the signal strengths reduces the expert value $E_t^i(s_t^i)$. By reducing the exchanges of Q-value with low accuracy, this approach reduces control overhead, and hence it reduces interference to PUs.

Application of cooperative learning in the CR context has been very limited. More description on cooperative learning is found in Section 4.8. Further research could be pursued to investigate how to improve network performance using this approach in existing and new schemes.

4. Reinforcement Learning in the Context of Cognitive Radio Networks: Models and Algorithms

Direct application of the traditional RL approach (see Section 2.1) has been shown to provide performance enhancement in CR networks. Reddy [29] presents a preliminary investigation in the application of RL to detect PU signals in channel sensing (A2). Table 1 presents a summary of the schemes that apply the traditional RL approach. For each scheme, we present the purpose(s) of the CR scheme, followed by its associated RL model.

Most importantly, this section presents a number of new additions to the RL algorithms, which have been applied to various schemes in CR networks. A summary of the new algorithms, their purposes, and references, is shown in Table 2. Each new algorithm has been designed to suit and to achieve the objectives of the respective schemes. For instance, the collaborative model (see Table 2) aims to achieve an optimal global reward in the presence of multiple agents, while the traditional RL approach achieves an optimal local reward in the presence of a single agent only. The following subsections (i.e., Sections 4.1–4.9) provide further details to each new algorithm, including the purpose(s) of the CR scheme(s), followed by its associated RL model (i.e., state, action, and reward representations) which characterize the purposes, and finally the enhanced algorithm which aims to achieve the purpose. Hence, these subsections serve as a foundation for further research in this area, particularly, the application of existing RL models and algorithms found in current schemes to either apply them in new schemes or extend the RL models in existing schemes to further enhance network performance.

4.1. Model 1: Model with $\gamma = 0$ in Q-Function. This is a myopic RL-based approach (see Section 3.4) that uses $\gamma = 0$ so that there is lack of dependency on future rewards, and it has been applied in [10, 17, 18]. Li et al. [10] propose a joint DCS (A1) and channel sensing (A2) scheme, and it has been shown to increase throughput, as well as to decrease the number of sensing channels (see performance metric (P4) in Section 5) and packet retransmission rate. The purposes of this scheme are to select operating channels with successful transmission rate greater than a certain threshold into a sensing channel set and subsequently to select a single operating channel for data transmission.

Table 3 shows the RL model for the scheme. The action $a_t^i \in A_p$ is to select whether to remain at the current operating channel or to switch to another operating channel with higher successful transmission rate. A preferred channel set A_p is composed of actions a_t^i with Q-value $Q_t^i(a_t^i)$ greater than a fixed threshold Q_{th} (e.g., $Q_{\text{th}} = 5$ in [10]). Since the state and action are similar in this model, the state representation is not shown in Table 3, and we represent $r_{t+1}^i(a_t^i) = r_{t+1}^i(s_{t+1}^i)$. Note that $a_{t+1}^i = a_t^i$ if there is no channel switch. The reward $r_{t+1}^i(a_t^i)$ represents different kinds of events, specifically, $r_{t+1}^i(a_t^i) = 1$ in case of successful transmission, and $r_{t+1}^i(a_t^i) = -1$ in case

TABLE 1: RL models with direct application of the traditional RL approach for various schemes in CR networks.

References	Purpose	State	Action	Reward/cost
(A1) Dynamic channel selection (DCS)				
Tang et al. [2]	Each SU (agent) selects the operating channel with the least channel utilization level by PUs in order to improve throughput and to reduce end-to-end delay and the number of channel switches	—	Selecting an available channel for data transmission	Fixed positive/negative values to be rewarded/punished for successful/unsuccessful transmission
Li [6]	Each SU (agent) selects different operating channel with other SUs in order to reduce channel contention	—	Selecting an available channel for data transmission	Amount of successful data packet transmission
Yao and Feng [19]	SU base station (agent) selects an available channel and a power level for data transmission in order to improve its SNR. This scheme aims to increase packet delivery rate	Three-tuple information: (i) SU hosts of the SU base station, (ii) transmitting SU hosts, (iii) received power on each channel	Selecting a set of actions (see Section 3.2): (i) available channel for data transmission, (ii) transmission power level	SNR level
Li et al. [18]	Each SU link (agent) aims to maximize its individual SNR level. Note that the agent is a SU link, instead of the SU itself as seen in the other schemes	The availability of a channel for data transmission. States $s_k = 0$ and $s_k = 1$ indicate that channel k is idle and busy, respectively	Selecting an available channel for data transmission	SNR level, which takes into account the interference from neighboring SUs
(A2) Channel sensing				
Lo and Akyildiz [3]	Each SU (agent) (i) finds a set of neighboring SUs for cooperative channel sensing, (ii) minimizes cooperative channel sensing delay. This scheme aims to increase the probability of PU detection	A set of SU neighbor nodes that may cooperate with the SU agent to perform cooperative channel sensing	Selecting SU neighbor nodes that may cooperate with the SU agent. The SU neighbor nodes cooperate through sending their respective local sensing outcome to the SU agent	The reward (or cost) is dependent on the reporting delay, which is the time between a SU agent requesting for cooperation from a SU neighbor node and the arrival of its sensing outcome
(A4) Energy efficiency enhancement				
Zheng and Li [15]	Each SU (agent) selects a suitable action (transmit, idle, sleep, or sense channel) whenever it does not have any packets to send in order to reduce energy consumption	Four-tuple information: (i) operation mode: transmit, idle, and sleep, (ii) number of packets in the buffer, (iii) availability of PU activities, (iv) countdown timer for periodic channel sensing	Selecting an action: transmit, idle, sleep, or sense channel	Amount of energy consumption for each operation mode throughout the duration of the operation mode
(A7) Routing				
Peng et al. [4]	Each SU (agent) selects a SU neighbor node (or next hop) for data transmission to SU destination node in order to reduce end-to-end delay and energy consumption	A set of SU next hops	Selecting a SU next hop	Ratio of the residual energy of the SU next hop to energy consumption incurred by sending, receiving, encoding, and decoding data while transmitting data to the SU next hop

TABLE 2: Summary of RL models and algorithms for various schemes in CR networks.

Model	Purpose	References
Model with $\gamma = 0$ in Q-function	This model uses $\gamma = 0$ so that there is lack of dependency on future rewards	Li et al. [10, 17, 18]
Model with a set of Q-functions	This model uses a set of distinctive Q-functions to keep track of the Q-values of different actions	Di Felice et al. [11, 21]
Dual Q-function Model	This model updates two Q-functions for the next and previous states, respectively, simultaneously in order to expedite the learning process	Xia et al. [33]
Partial observable model	This model computes belief state, which is the probability of the environment operating in a particular state, in a dynamic and uncertain operating environment	Bkassiny et al. [34]
Actor-critic model	This model adjusts the delayed reward value using reward corrections in order to expedite the learning process	Vucevic et al. [13]
Auction model	This model allows agents to place bids during auctions conducted by a centralized entity so that the winning agents receive rewards	Chen and Qiu [16], Jayaweera et al. [36], Fu and van der Schaar [37], and Xiao et al. [38]
Internal self-learning model	This model enables an agent to exchange its virtual actions continuously with rewards generated by a simulated internal environment within the agent itself in order to expedite the learning process	Bernardo et al. [27]
Collaborative model	This model enables an agent to collaborate with its neighbor agents and subsequently make local decisions independently in distributed networks. A local decision is part of an optimal joint action, which is comprised of the actions taken by all the agents in a network	Lundén et al. [20] Liu et al. [39]
Competitive model	This model enables an agent to compete with its neighbor agents and subsequently make local decisions independently in worst-case scenarios in the presence of competitor agents, which attempt to minimize the accumulated rewards of the agent	Wang et al. [14]

TABLE 3: RL model for joint dynamic channel selection and channel sensing [10].

Action	$a_t^i \in A_p = \{a_t^i \in A \mid Q_t^i(a_t^i) > Q_{th}\}$; each action represents a single channel available for data transmission
Reward	$r_{t+1}^i(a_t^i) = \begin{cases} 1, & \text{if successful transmission} \\ -1, & \text{if unsuccessful transmission} \end{cases}$

of unsuccessful transmission or channel a_{t+1}^i is sensed busy. The RL model is embedded in a centralized entity such as a base station.

Algorithm 1 presents the RL algorithm for the scheme. The action $a_t^i \in A_p$ is chosen from a preferred channel set. The update of the Q-value $Q_{t+1}^i(a_t^i)$ is self-explanatory. Similar approach has been applied in DCS (A1) [30, 31].

Li et al. [18] propose a MAC protocol, which includes both DCS (A1) and a retransmission policy (A6), to minimize channel contention. The DCS scheme enables the SU agents to minimize their possibilities of operating in the same channel. This scheme uses the RL algorithm in Algorithm 1, and the reward representation is extended to more than a single performance enhancement. Specifically, the reward $r_{t+1}^i(a_t^i)$ represents the successful transmission rate and transmission delay. Higher reward indicates higher successful transmission rate and lower transmission delay,

and vice versa. To accommodate both transmission rate and transmission delay in Q-function, the reward representation becomes $r_{t+1}^i(a_t^i) = r_{t+1}^{i'}(a_t^i) + r_{t+1}^{i''}(a_t^i)$, and so the Q-function becomes $Q_{t+1}^i(a_t^i) = Q_t^i(a_t^i) + r_{t+1}^{i'}(a_t^i) + r_{t+1}^{i''}(a_t^i)$. The retransmission policy determines the probability a SU agent transmits at time t , and so $Q_{t+1}^i(a_t^i)$ indicates the probability a SU agent transmits at time t . The reward $r_{t+1}^{i'}(a_t^i) = 1, 0$, and -1 if the transmission delay at time t is smaller than, equal to, and greater than the average transmission delay, respectively. The reward $r_{t+1}^{i''}(a_t^i)$ represents different kinds of events; specifically, $r_{t+1}^{i''}(a_t^i) = 2, 0$, and -2 in case of successful transmission, idle transmission, and unsuccessful transmission, respectively; note that idle indicates that channel a_t^i is sensed busy, and so there is no transmission.

Li et al. [17] propose a MAC protocol (A6) to reduce the probability of packet collision among PUs and SUs, and it has been shown to increase throughput and to decrease packet loss rate. Since both successful transmission rate and the presence of idle channels are important factors, it keeps track of the Q-functions for channel sensing $Q_t^i(a_{se}^i)$ and transmission $Q_t^i(a_{tx}^i)$ using RL algorithm in Algorithm 1, respectively. Hence, similar to Algorithm 2 in Section 4.2, there is a set of two Q-functions. The action a_t^i is to select whether to remain at the current operating channel or to switch to another operating channel. The sensing reward $r_{t+1}^i(a_{se}^i) = 1$ and -1 if the channel is sensed idle and busy,

Repeat

(a) Choose action $a_t^i \in A_p$

(b) Update Q-value:

$$Q_{t+1}^i(a_t^i) = Q_t^i(a_t^i) + r_{t+1}^i(a_t^i)$$

(c) Update preferred channel set $A_p \in A$

$$a_t^i \in A_p = \{a_t^i \in A \mid Q_{t+1}^i(a_t^i) > Q_{th}\}$$

ALGORITHM 1: RL algorithm for joint DCS and channel sensing [10].

Repeat

(a) Choose action a_t^i

(b) Update Q-value $Q_t^i(s_t^i, a_t^i) \in \{Q_t^i(s_t^i, a_{se}^i), Q_t^i(s_t^i, a_{tx}^i), Q_t^i(s_t^i, a_{sw}^i)\}$ as follows:

$$Q_{t+1}^i(s_t^i, a_t^i) = \begin{cases} Q_{t+1}^i(s_t^i, a_{se}^i) = Q_t^i(s_t^i, a_{se}^i) + \alpha \cdot e^i(s_t^i) \cdot (T_{se}^i - Q_t^i(s_t^i, a_{se}^i)) & \text{if } a_t^i = a_{se}^i \\ Q_{t+1}^i(s_t^i, a_{tx}^i) = Q_t^i(s_t^i, a_{tx}^i) + \alpha \cdot (1 - e^i(s_t^i)) \cdot (T_{tx}^i - Q_t^i(s_t^i, a_{tx}^i)) & \text{if } a_t^i = a_{tx}^i \\ \text{Switch channel (Change channel from } s_t^i \text{ to } s_{t+1}^i) & \text{if } a_t^i = a_{sw}^i \end{cases}$$

(c) Update Q-value:

$$Q_t^i(s_t^i, a_{sw}^i) = \max_{1 \leq s \leq K} (V_t^i(s) - V_t^i(s_t^i) - \theta)$$

(d) Update policy:

$$\pi^i(s_{t+1}^i, a_{t+1}^i) = \frac{e^{Q_t^i(s_{t+1}^i, a_{t+1}^i)}}{Q_t^i(s_{t+1}^i, a_{se}^i) + Q_t^i(s_{t+1}^i, a_{tx}^i) + Q_t^i(s_{t+1}^i, a_{sw}^i)}$$

ALGORITHM 2: RL algorithm for joint DCS and channel sensing [11].

respectively. The transmission reward $r_{t+1}^i(a_{tx}^i) = 1$ and -1 if the transmission is successful and unsuccessful, respectively. Action selection is based on the maximum average Q-value; specifically, $Q_t^i(a_t^i) = [Q_t^i(a_{se}^i) + Q_t^i(a_{tx}^i)]/2$.

4.2. Model 2: Model with a Set of Q-Functions. A set of distinctive Q-functions can be applied to keep track of the Q-value of different actions, and it has been applied in [11, 21]. Di Felice et al. [11] propose a joint DCS (A1) and channel sensing (A2) scheme, and it has been shown to increase goodput and packet delivery rate, as well as to decrease end-to-end delay and interference level to PUs. The purposes of this scheme are threefold:

- (i) firstly, it selects an operating channel that has the lowest channel utilization level by PUs;
- (ii) secondly, it achieves a balanced trade-off between the time durations for data transmission and channel sensing;
- (iii) thirdly, it reduces the exploration probability using a knowledge sharing mechanism.

Table 4 shows the RL model for the scheme. The state $s_t^i \in S$ represents a channel for data transmission. The actions $a_t^i \in A$ are to sense channel, to transmit data, or to switch its operating channel. The reward $r_{t+1}^i(s_{t+1}^i)$ represents the difference between two types of delays, namely, the maximum allowable single-hop transmission delay and a successful single-hop transmission delay. A single-hop transmission delay covers four kinds of delays including backoff, packet

transmission, packet retransmission, and propagation delays. Higher reward level indicates shorter delay incurred by a successful single-hop transmission. The RL model is embedded in a centralized entity such as a base station.

Algorithm 2 presents the RL algorithm for the scheme. Denote learning rate by $0 \leq \alpha \leq 1$, eligible trace by $e^i(s_t^i)$, and the amount of time during which the SU agent is involved in successful transmissions or was idle (i.e., no packets to transmit) by T_{tx}^i , as well as the temporal differences by $(T_{se}^i - Q_t^i(s_t^i, a_{se}^i))$ and $(T_{tx}^i - Q_t^i(s_t^i, a_{tx}^i))$. A single type of Q-function is chosen to update the Q-value $Q_t^i(s_t^i, a_t^i)$ based on the current action $a_t^i \in A = \{a_{se}^i, a_{tx}^i, a_{sw}^i\}$ being taken. The temporal difference indicates the difference between the actual outcome and the estimated Q-value.

In step (b), the eligible trace $e^i(s_t^i)$ represents the temporal validity of state s_t^i . Specifically, in [11], eligible trace $e^i(s_t^i)$ represents the existence of PU activities in channel s_t^i , and so it is only updated when channel sensing operation a_{se}^i is taken. Higher eligible trace $e^i(s_t^i)$ indicates greater presence of PU activities, and vice versa. Hence, the term $e^i(s_t^i)$ is in the update of Q-value $Q_{t+1}^i(s_t^i, a_{se}^i)$, and $(1 - e^i(s_t^i))$ is in the update of Q-value $Q_{t+1}^i(s_t^i, a_{tx}^i)$ in Algorithm 2. Therefore, higher eligible trace $e^i(s_t^i)$ results in higher value of $Q_{t+1}^i(s_t^i, a_{se}^i)$ and lower value of $Q_{t+1}^i(s_t^i, a_{tx}^i)$, and this indicates more channel sensing tasks and lesser data transmission in channels with greater presence of PU activities. The action a_{sw}^i switches channel from state s_t^i to state s_{t+1}^i . The ϵ -greedy approach is applied to choose the next channel s_{t+1}^i . In [21], eligible trace

TABLE 4: RL model for joint dynamic channel selection and channel sensing [11].

State	$s_t^i \in S = \{1, 2, \dots, K\}$; each state represents an available channel
Action	$a_t^i \in A = \{a_{se}, a_{tx}, a_{sw}\}$, where action a_{se} senses a channel for the duration of T_{se}^i , a_{tx} transmits a data packet, and a_{sw} switches the current operating channel to another one which has the lowest best-known average transmission delay for a single-hop
Reward	$r_{t+1}^i(s_{t+1}^i)$ represents the difference between a successful single-hop transmission delay and the maximum allowable single-hop transmission delay

TABLE 5: RL model for the routing scheme [33].

State	$s_t^i \in S = \{1, 2, \dots, N-1\}$; each state represents a SU destination node n . N represents the number of SUs in the entire network
Action	$a_t^i \in A = \{1, 2, \dots, J\}$; each action represents the selection of a next-hop SU node j . J represents the number of SU node i 's neighbor SUs
Reward	$r_{t+1}^i(s_{t+1}^i, a_{t+1}^i)$ represents the number of available common channels among nodes i and j

$e^i(s_t^i)$, which represents the temporal validity or freshness of the sensing outcome, is only updated when the channel sensing operation a_{se}^i is taken as shown in Algorithm 2. The eligible trace $e^i(s_t^i)$ is discounted whenever a_{se}^i is not chosen as follows:

$$e^i(s_{t+1}^i) = \begin{cases} 1, & \text{if } a_t^i = a_{se}^i \\ \delta \cdot e^i(s_t^i), & \text{otherwise,} \end{cases} \quad (15)$$

where $0 \leq \delta \leq 1$ is a discount factor for the eligible trace. Equation (15) shows that the eligible trace of each state s_t^i is set to the maximum value of 1 whenever action a_{se}^i is taken; otherwise, it is decreased with a factor of δ .

In step (c), the $Q_t^i(s_t^i, a_{sw}^i)$ value keeps track of the channel that provides the best-known lowest estimated average transmission delay. In other words, the channel must provide the maximum amount of reward that can be achieved considering the cost of a channel switch θ . Hence, $Q_t^i(s_t^i, a_{sw}^i)$ can keep track of a channel s_{t+1}^i that provides the best-known state value $V_t^i(s_{t+1}^i)$ the SU agent receives compared to the average state value $V_t^i(s)$ by switching its current operating channel s_t^i to the operating channel s_{t+1}^i . Note that the state value $V_t^i(s_t^i)$ is exchanged among the SU agents to reduce exploration through cooperative learning (see Section 3.7.3).

In step (d), the policy $\pi^i(s_{t+1}^i, a_{t+1}^i)$ is applied at the next time instant. The policy provides probability distributions over the three possible types of actions $A = \{a_{se}^i, a_{tx}^i, a_{sw}^i\}$ using a modified Boltzmann distribution (see Section 3.6). Next, the policy is applied to select the next action a_{t+1}^i in step (a).

4.3. Model 3: Dual Q-Function Model. The dual Q-function model has been applied to expedite the learning process [32]. The traditional Q-function (see (1)) updates a single Q-value at a time, whereas the dual Q-function updates two Q-values simultaneously. For instance, in [33], the traditional Q-function updates the Q-value for the next state only (e.g., SU destination node), whereas the dual Q-function updates the Q-value for the next and previous states (e.g., SU source and destination nodes, respectively). The dual Q-function model updates a SU agent's Q-value in both directions (i.e., towards the source and destination nodes) and speeds up the learning

process in order to make more accurate decisions on action selection; however, at the expense of higher network overhead incurred by more Q-value exchanges among the SU neighbor nodes.

Xia et al. [33] propose a routing (A7) scheme, and it has been shown to reduce SU end-to-end delay. Generally speaking, the availability of channels in CR networks is dynamic, and it is dependent on the channel utilization level by PUs. The purpose of this scheme is to enable a SU node to select a next-hop SU node with higher number of available channels. The higher number of available channels reduces the time incurred in seeking for an available common channel for data transmission among a SU node pair, and hence it reduces the MAC layer delay.

Table 5 shows the RL model for the scheme. The state $s_t^i \in S$ represents a SU destination node n . The action $a_t^i \in A$ represents the selection of a next-hop SU neighbor node j . The reward $r_{t+1}^i(s_{t+1}^i, a_{t+1}^i)$ represents the number of available common channels among nodes i and $a_t^i = j$. The RL model is embedded in each SU agent.

This scheme applies the traditional Q-function (see (1)) with $\gamma = 1$. Hence, the Q-function is rewritten as follows:

$$Q_{t+1}^i(s_t^i, j) \leftarrow (1 - \alpha) Q_t^i(s_t^i, j) + \alpha \left[r_{t+1}^i(s_{t+1}^i, j) + \max_{k \in a_t^i} Q_t^j(s_{t+1}^i, k) \right], \quad (16)$$

where $k \in a_t^j$ is an upstream node of SU neighbor node j , so node j must estimate and send information on $\max_{k \in a_t^j} Q_t^j(s_{t+1}^i, k)$ to SU node i .

The dual Q-function model in this scheme is applied to update the Q-value for the SU source and destination nodes. While the traditional Q-function enables the SU intermediate node to update the Q-value for the SU destination node only (or next state), which is called forward exploration, the dual Q-function model enables an intermediate SU node to achieve backward exploration as well by updating the Q-value for the SU source node (or previous state). Forward exploration is achieved by updating the Q-value at SU node i for the SU destination node whenever it receives an estimate $\max_{k \in a_t^j} Q_t^j(s_{t+1}^i, k)$ from SU node j , while backward

TABLE 6: RL model for joint DCS and channel sensing [34].

State	$\mathbf{s}_t^i = (s_{1,t}^i, s_{2,t}^i, \dots, s_{K,t}^i) \in S_1 \times S_2 \times \dots \times S_K$; each substate $s_{k,t}^i \in S_k = \{0, 1\}$ indicates an idle or busy channel; specifically, $s_{k,t}^i = 0$ if PU activity does not exist in channel k , and $s_{k,t}^i = 1$ if PU activity exists in channel k
Action	$a_t^i \in A = \{1, 2, \dots, K\}$; each action represents a single channel available for data transmission
Reward	$r_{t+1}^i(\mathbf{s}_{t+1}^i) = \begin{cases} 1, & \text{if successful transmission} \\ 0, & \text{if unsuccessful transmission because the sensed channel is busy} \\ -0.5, & \text{if unsuccessful transmission and backoff because there is collision with other SUs} \end{cases}$

exploration is achieved by updating the Q-value at SU node j for the SU source node whenever it receives a data packet from node i . Note that, in the backward exploration case, node i 's packets are piggybacked with its Q-value so that node j is able to update Q-value for the respective SU source node. Although the dual Q-function approach increases the network overhead, it expedites the learning process since SU nodes along a route update Q-value of the route in both directions.

4.4. Model 4: Partial Observable Model. The partial observable model has been applied in a dynamic and uncertain operating environment. The uniqueness of the partial observable model is that the SU agents are uncertain about their respective states, and so each of them computes belief state $b(s_t^i)$, which is the probability that the environment is operating in state s_t^i .

Bkassiny et al. [34] propose a joint DCS (A1) and channel sensing (A2) scheme, and it has been shown to improve the overall spectrum utilization. The purpose of this scheme is to enable the SU agents to select their respective operating channels for sensing and data transmission in which the collisions among the SUs and PUs must be minimized.

Table 6 shows the RL model for the scheme. The state $\mathbf{s}_t^i \in S_1 \times S_2 \times \dots \times S_K$ represents the availability of a set of channels for data transmission. The action $a_t^i \in A$ represents a single channel out of K channels available for data transmission. The reward represents fixed positive (negative) values to be rewarded (punished) for successful (unsuccessful) transmissions. The RL model is embedded in each SU agent so that it can make decision in a distributed manner.

Algorithm 3 presents the RL algorithm for the scheme. The action $a_t^i \in A$ is chosen from a preferred channel set. The chosen action has the maximum belief-state Q-value, which is calculated using belief vector $\mathbf{b}(\mathbf{s}_t^i) = (b(s_{1,t}^i), b(s_{2,t}^i), \dots, b(s_{K,t}^i))$ as weighting factor. The belief vector $\mathbf{b}(\mathbf{s}_t^i)$ is the probability of a possible set of state $\mathbf{s}_t^i = (s_{1,t}^i, s_{2,t}^i, \dots, s_{K,t}^i)$ being idle at time $t + 1$. Upon receiving reward $r_{t+1}^i(\mathbf{s}_{t+1}^i, a_t^i)$, the SU agent updates the entire set of belief vectors $\mathbf{b}(\mathbf{s}_t^i)$ using Bayes' formula [34]. Next, the SU agent updates the Q-value $Q_{t+1}^i(\mathbf{s}_t^i, a_t^i)$. Note that $\max_{a \in A} Q_{b,t+1}^i(\mathbf{s}_{t+1}^i, a) = \max_{a \in A} \sum_{s \in \mathbf{s}_t^i} b(s) Q_t^i(s, a)$.

It shall be noted that Bkassiny et al. [34] apply the belief vector $\mathbf{b}(\mathbf{s}_t^i)$ as a weighting vector in its computation of Q-value $Q_{t+1}^i(\mathbf{s}_t^i, a_t^i)$, while most of the other approaches, such

TABLE 7: RL model for security enhancement [13].

Action	$a_t^i \in A = \{1, 2, \dots, N_{nbr,i}\}$; each action represents a neighboring SU chosen for channel sensing purpose, where $N_{nbr,i}$ indicates the number of SU node i 's neighbor SUs
Reward	$r_{t+1}^i(a_t^i) = \begin{cases} R, & \text{if correct sensing outcome} \\ -R, & \text{if incorrect sensing outcome} \end{cases}$

as [20], use belief vector $\mathbf{b}(\mathbf{s}_t^i)$ as the actual state, specifically, $Q_{t+1}^i(\mathbf{b}(\mathbf{s}_t^i), a_t^i)$. This approach has been shown to achieve a near-optimal solution with a very low complexity in [35].

4.5. Model 5: Actor-Critic Model. Traditionally, the delayed reward has been applied directly to update the Q-value. The actor-critic model adjusts the delayed reward value using reward corrections, and this approach has been shown to expedite the learning process. In this model, an actor selects actions using suitability value, while a critic keeps track of temporal difference, which takes into account reward corrections in delayed rewards.

Vucevic et al. [13] propose a collaborative channel sensing (A2) scheme, and it has been shown to minimize error detection probability in the presence of inaccurate sensing outcomes. The purpose of this scheme is that it selects neighboring SU agents that provide accurate channel sensing outcomes for security enhancement purpose (A3). Table 7 shows the RL model for the scheme. The state is not represented. An action $a_t^i \in A$ represents a neighboring SU chosen by SU agent i for channel sensing purpose. The reward $r_{t+1}^i(a_t^i)$ represents fixed positive (negative) values to be rewarded (punished) for correct (incorrect) sensing outcomes compared to the final decision, which is the fusion of the sensing outcomes. The RL model is embedded in each SU agent.

The critic keeps track of $c_{t+1}^i(a_t^i) = c_t^i(a_t^i) + \beta \cdot \Delta c_t^i(a_t^i)$, where $\Delta c_t^i(a_t^i)$ is the temporal difference and β is a constant (e.g., $\beta = 0.01$). In [13], $\Delta c_t^i(a_t^i)$ depends on the difference between the delayed reward $r_{t+1}^i(a_t^i)$ and the long-term delayed reward $\bar{r}_{t+1}^i(a_t^i) = \alpha \cdot r_{t+1}^i(a_t^i) + (1-\alpha) \cdot \bar{r}_t^i(a_t^i)$, the number of incorrect sensing outcomes, and the suitability value $\pi_t^i(a_t^i)$. Next, the actor selects actions using $c_{t+1}^i(a_t^i)$ given by the critic. The probability of selecting action a_t^i is based on the suitability value of action i ; $\pi_t^i(a_t^i) = e^{c_{t+1}^i(a_t^i)} / \sum_{a \in N_{nbr,i}} e^{c_{t+1}^i(a)}$.

<p>Repeat</p> <p>(a) Choose action $a_t^i \in A$</p> $a_t^i = \operatorname{argmax}_a Q_{b,t}^i(\mathbf{s}_t, a) = \operatorname{argmax}_a \sum_{\mathbf{s} \in \mathcal{S}} \mathbf{b}(\mathbf{s}) Q_t^i(\mathbf{s}, a)$ <p>(b) Receive delayed reward $r_{t+1}^i(\mathbf{s}_{t+1}^i, a_t^i)$</p> <p>(c) Update belief $\mathbf{b}(\mathbf{s}_t^i)$</p> <p>(d) Update Q-value:</p> $Q_{t+1}^i(\mathbf{s}_t^i, a_t^i) = Q_t^i(\mathbf{s}_t^i, a_t^i) + \alpha \cdot \mathbf{b}(\mathbf{s}_t^i) \cdot \left[r_{t+1}^i(\mathbf{s}_{t+1}^i, a_t^i) + \gamma \max_{a \in A} Q_{b,t+1}^i(\mathbf{s}_{t+1}^i, a) - Q_t^i(\mathbf{s}_t^i, a_t^i) \right]$
--

ALGORITHM 3: RL algorithm for joint dynamic channel selection and channel sensing [34].

TABLE 8: RL model for the channel auction scheme [16].

State	$s_t^i \in S = \{0, 1, \dots, L_b \cdot L_c\}$, each state represents a two-tuple information composed of buffer fullness index L_b and credit ratio index L_c
Action	$a_t^i \in A = \{1, 2, \dots, L_a\}$; each action represents the amount of a bid for white spaces
Reward	$r_{t+1}^i(s_{t+1}^i) = \begin{cases} \text{Positive value of the amount of data sent,} & \text{if successful bid} \\ \text{Negative value of the amount of data that could have sent,} & \text{if unsuccessful bid} \end{cases}$

TABLE 9: RL model for the channel auction scheme [36].

Reward	$r_{t+1}^i(a_{k,t}^i) = \begin{cases} R, & \text{if successful bid} \\ -R, & \text{if unsuccessful bid} \end{cases}$
--------	--

4.6. *Model 6: Auction Model.* The auction model has been applied in centralized CR networks. In the auction model, a centralized entity, such as a base station, conducts auctions and allows SU hosts to place bids so that the winning SU hosts receive rewards. The centralized entity may perform simple tasks, such as allocating white spaces to SU hosts with winning bids [16], or it may learn using RL to maximize its utility [36]. The RL model may be embedded in each SU host in a centralized network [16, 36–38], or in the centralized entity only [36].

Chen and Qiu [16] propose a channel auction scheme (A5), and it has been shown to allocate white spaces among SU hosts (or agents) efficiently and fairly. The purpose of this scheme is to enable the SU agents to select the amount of bids during an auction, which is conducted by centralized entity, for white spaces. The SU agents place the right amount of bids in order to secure white spaces for data transmission, while saving their credits, respectively. The RL model is embedded in each SU host.

Table 8 shows the RL model for the scheme. The state $s_t^i \in S$ indicates a SU agent’s information, specifically, the amount of data for transmission in its buffer and the amount of credits (or “wealth”) it owns. The action $a_t^i \in A$ is the amount of a bid for white spaces. The reward $r_{t+1}^i(s_{t+1}^i)$ indicates the amount of data sent. This scheme applies the traditional Q-learning approach (see (1)), to update Q-values.

Jayaweera et al. [36] propose another channel auction scheme (A5) that allocates white spaces among SUs, and it has been shown to increase transmission rates of the SUs and to reduce energy consumption of the PUs. In [36], the PUs

adjust the amount of white spaces and allocate them to the SUs with winning bids. The winning SUs transmit their packets, as well as relaying PUs’ packets using the white spaces so that the PUs can reduce its energy consumption. In other words, the SUs use their power as currency to buy the bandwidth. Two different kinds of RL models are embedded in PUs and SUs, respectively, so that the PUs can learn to adjust the amount of white spaces to be allocated to the SUs, and the SUs can learn to select the amount of bids during an auction for white spaces.

The state is not represented, and we show the action and reward representations of the scheme. Table 9 shows the reward representation of the RL model. The reward $r_{t+1}^i(a_{k,t}^i)$ indicates a constant positive reward in case of successful bid and a constant negative reward in case of unsuccessful bid. The reward representation is embedded in both PUs and SUs. The actions for both PUs and SUs are different. Each SU i selects the amount of bid $a_{k,t}^i \in A$ during an auction for white spaces in channel k , while each PU adjusts the amount of white spaces $a_{k,t}^i \in A$ to be offered for auction in its own channel k . Higher amount of white spaces encourages the SUs to participate in auctions.

This scheme applies Q-function $Q_{k,t+1}^i(a_{k,t}^i) = Q_{k,t}^i(a_{k,t}^i) + r_{t+1}^i(a_{k,t}^i)$ with $\gamma = 0$ (see Section 4.1) at both PUs and SUs. The SUs’ Q-function indicates the appropriate amount of bids for white spaces, while the PUs’ Q-function indicates the appropriate amount of white spaces to be offered for auction.

Fu and Van der Schaar [37] propose a channel auction scheme (A5) that improves the bidding policy of SUs, and it has been shown to reduce SUs’ packet loss rate. The purpose of this scheme is to enable SU agents to learn and adapt the amount of bids during an auction for time-varying white spaces in dynamic wireless networks with environmental disturbance and SU-SU disturbance. Examples of *environmental disturbance* are dynamic level of channel utilization by PUs, channel condition (i.e., SNR), and SU traffic rate, while

TABLE 10: RL model for the channel auction scheme [37].

State	$s_t^i = (b_t^i, \mathbf{p}_t^i) \in S$; each state represents a two-tuple information composed of the fullness of the buffer state b_t^i and channel states $\mathbf{p}_t^i = (p_{t,1}^i, p_{t,2}^i, \dots, p_{t,k}^i)$, where $p_{t,k}^i$ represents the state of channel k in terms of SNR
Action	$a_t^i \in A = \{a_{t,1}^i, a_{t,2}^i, \dots, a_{t,k}^i\}$; each action represents the amount of a bid for white spaces in channel k . K represents the number of available channels
Reward	$r_{t+1}^i(\mathbf{s}_{t+1}, \mathbf{w}_{t+1}, \mathbf{a}_{t+1}) = g_{t+1}^i + c_{t+1}^i$ represents the sum of the number of lost packets g_{t+1}^i and the channel cost c_{t+1}^i that SU i must pay for using the channel. Note that the packet loss g_{t+1}^i and channel cost c_{t+1}^i depend on the global state \mathbf{s}_{t+1} , available channels \mathbf{w}_{t+1} , and bidding actions \mathbf{a}_{t+1} of all competing SUs

TABLE 11: RL model for a power control scheme [38].

Action	$a_t^i \in A = \{a_{sh}^i, a_{mh}^i\}$, with a_{sh}^i and a_{mh}^i being transmitting SU i 's packets to the SU destination node using single-hop transmission and multihop relaying, respectively
Reward	$r_{t+1}^i(a_{t+1}^i)$ represents the revenue obtained from the other SUs for relaying their packets. Higher rewards indicate higher transmission rate and transmission power of SU node i

an example of SU-SU disturbance is the effect from other competing SUs, who are noncollaborative and autonomous in nature. Compared to traditional centralized auction schemes, SUs compute their bids based on their knowledge and observation of the operating environment with limited information received from other SUs and the centralized base station. Note that the joint bidding actions of SUs affect the allocation of white spaces and bidding policies of the other SUs, and so the proposed learning algorithm improves the bidding policy of SUs based on the observed white space allocations and rewards.

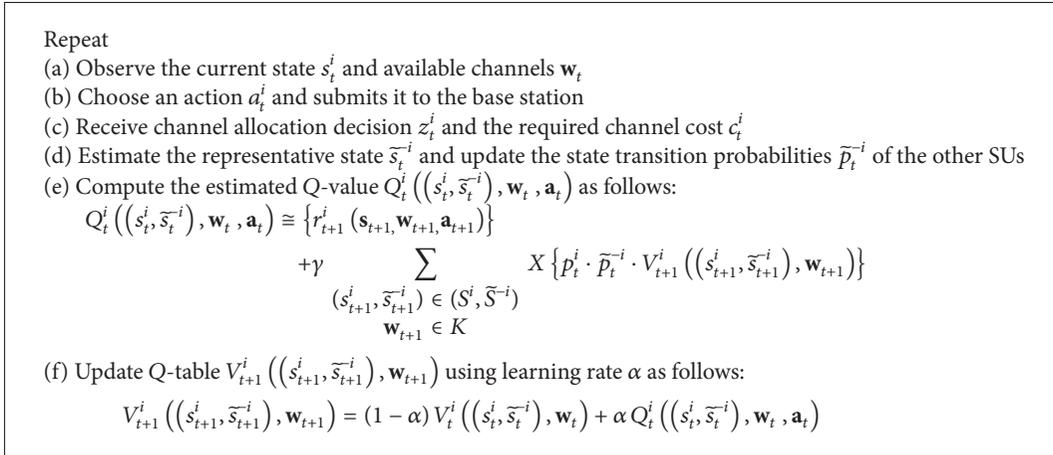
Table 10 shows the RL model for the scheme. The state $s_t^i \in S$ indicates SU agent's information, specifically, its buffer state, as well as the states of the available channels in terms of SNR. The action $a_t^i \in A$ is the amount of bids for white spaces. The reward $r_{t+1}^i(\mathbf{s}_{t+1}, \mathbf{w}_{t+1}, \mathbf{a}_{t+1})$ represents the sum of the number of lost packets g_{t+1}^i and the channel cost c_{t+1}^i that SU i must pay for using the channel. Note that the channel cost c_{t+1}^i represents network congestion, and hence higher cost c_{t+1}^i indicates higher congestion level. The RL model is embedded in each SU host.

Algorithm 4 presents the RL algorithm for the scheme. In step (a), SU agent i observes its current state s_t^i and available channels (or white spaces) \mathbf{w}_t advertised by the centralized base station. In step (b), it decides and submits its bids to the base station, and the bids are estimated based on SU i 's state s_t^i and other SUs' representative (or estimated) state \tilde{s}_t^{-i} . Note that, since SU i needs to know all the states and transition probabilities of other SUs, which may not be feasible, it estimates the representative state \tilde{s}_t^{-i} based on its previous knowledge of channel allocation z_t^j and channel cost c_{t+1}^j (or network congestion). In step (c), SU i receives its channel allocation decision z_t^i and the required channel cost c_t^i from the base station. In step (d), the representative state \tilde{s}_t^{-i} and transition probabilities \tilde{p}_t^{-i} of the other SUs are updated based on the newly received channel allocation decision z_t^i and the required channel cost c_t^i information. In step (e), SU i computes its estimated Q-value, which is inspired by the traditional Q-function approach, and this approach explicitly

takes into account the effects of the bidding actions of the other SUs based on their estimated representative state \tilde{s}_t^{-i} and transition probabilities \tilde{p}_t^{-i} . Note that \mathbf{a}_t also denotes Markov-based policy profile that represents the bidding policies of all the other SUs. In step (f), the Q-table is updated if there are changes in the SU states and channel availability.

Xiao et al. [38] propose a power control scheme (A8), and it has been shown to increase the transmission rates and payoffs of SUs. There are two main differences compared to the traditional auction schemes, which have been applied to centralized networks. Firstly, the interactions among all nodes, including PUs and SUs, are coordinated in a distributed manner. A SU source node transmits its packets to the SU destination node using either single-hop transmission or multihop relaying. In multihop relaying, a SU source node must pay the upstream node, which helps to relay the packets. Secondly, the PUs treat each SU equally, and so there is lack of competitiveness in auctions. Each SU may accumulate credits through relaying. Game theory is applied to model the network in which SUs pay credits to PUs for using licensed channels and to other SUs for relaying their packets. The purpose of this scheme is to enable a SU node to choose efficient actions in order to improve its payoff, as well as to collect credits through relaying, and to minimize the credits paid to PUs and other SU relays. A RL model is embedded in each SU.

The state is not represented, and we show the action and reward representations of the scheme. Table 11 shows the RL model for the scheme. The action $a_t^i \in A$ represents transmission of SU i 's packets by either using single-hop transmission or multihop relaying. The reward $r_{t+1}^i(a_t^i)$ indicates the revenue (or profit) received by SU node i for providing relaying services to other SUs, and so higher reward indicates higher transmission rate and increased transmission power of SU node i . Denote the payoff of SU i by \mathbf{p}_t^i , as shown in (17). The payoff indicates the difference between SU i 's revenue and costs. There are two types of costs represented by $c_t^{i,j}$ and $c_t^{i,PU}$. The $c_t^{i,j}$ represents the cost charged by the upstream SU node j for relaying SU node i 's packets, and the $c_t^{i,PU}$ represents the cost charged by all



ALGORITHM 4: RL algorithm for the channel auction scheme [37].

PU's for using the white spaces in licensed channels. The $c_t^{i,PU}$ increases with the SU i 's interference power in the respective channel. Consider

$$\mathbf{p}_t^i = \sum_{j=1}^N (r_t^{i,j} + c_t^{i,j} + c_t^{i,PU}). \quad (17)$$

This scheme applies Q-function $Q_{t+1}^i(a_t^i) = Q_t^i(a_t^i) + \delta(\mathbf{p}_t^i \cdot P_t^i(a_t^i))$, which indicates the average payoff, where δ is a constant step size and $P_t^i(a_t^i)$ is the probability of SU i choosing action a_t^i , which is computed according to Boltzmann distribution (see Section 3.6).

4.7. Model 7: Internal Self-Learning Model. The internal self-learning model has been applied to expedite the learning process. The uniqueness of the internal self-learning model lies in the learning approach in which the learning mechanism continuously interacts with a simulated internal environment within the SU agent itself. The learning mechanism continuously exchanges its actions with rewards generated by the simulated internal environment so that the SU agent learns the optimal actions for various settings of the operating environment, and this helps Q-value and the optimal action to converge.

Bernardo et al. [27] propose a DCS (A1) scheme, and it has been shown to improve the overall spectrum utilization and throughput performances. Note that, unlike the previous schemes in which the RL models are embedded in the SU agents, the RL model is embedded in each PU base station (or agent) in this scheme, and it is applied to make medium-term decisions (i.e., from tens of seconds to tens of minutes). The purpose of this scheme is to enable a PU agent to select its operating channels for transmission in its own cell. In order to improve the overall spectrum utilization, the PU agent preserves its own QoS while generating white spaces and sells them off to SU agents.

Table 12 shows the RL model for the scheme. The action $\mathbf{a}_t^i \in A_1 \times A_2 \times \dots \times A_K$ is a set of chosen available channels for the entire cell. The reward $r_{t+1}^i(\mathbf{a}_t^i)$ has a zero value if

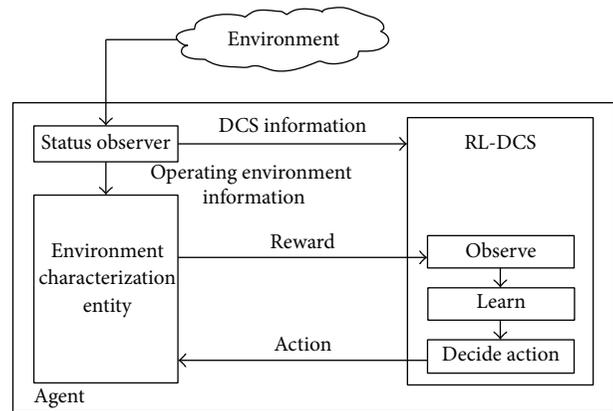


FIGURE 3: Internal self-learning model.

the estimated throughput of an action selection \mathbf{a}_t^i is less than a throughput threshold T_{th} ; otherwise, the reward is based on the spectrum efficiency $\hat{\eta}(\mathbf{a}_t^i)$ and the amount of white spaces $\bar{W}(\mathbf{a}_t^i)$, which may be sold off to SU agents. Both λ and μ are constant weight factors.

Figure 3 shows the internal self-learning model. The learning mechanism, namely, RL-DCS, continuously interacts with a simulated internal environment, namely, Environment Characterization Entity (ECE). Based on the information observed from the real operating environment (i.e., the number of PU hosts and the average throughput per PU host), which is provided by status observer, the ECE implements a model of the real operating environment (i.e., spectrum efficiency $\hat{\eta}(\mathbf{a}_t^i)$ and the amount of white spaces $\bar{W}(\mathbf{a}_t^i)$) and computes reward $r_{t+1}^i(\mathbf{a}_t^i)$. Hence, the ECE evaluates the suitability of action \mathbf{a}_t^i in its simulated internal model of the operating environment. By exchanging action \mathbf{a}_t^i and reward $r_{t+1}^i(\mathbf{a}_t^i)$ between RL-DCS and ECE, the RL-DCS learns an optimal action \mathbf{a}_t^i at a faster rate compared to the conventional learning approach, and this process stops when the optimal action \mathbf{a}_t^i converges.

Repeat

(a) Choose action $\mathbf{a}_t^i = (a_{1,t}^i, a_{2,t}^i, \dots, a_{K,t}^i) \in A_1 \times A_2 \times \dots \times A_K$

(b) Receive delayed reward $r_{t+1}^i(\mathbf{a}_t^i)$ from ECE

(c) Update Q-value:
For each $a_{k,t}^i \in \mathbf{a}_t^i$

$$Q_{t+1}^i(a_{k,t}^i) = Q_t^i(a_{k,t}^i) + \alpha \cdot [r_{t+1}^i(a_{k,t}^i) - \bar{r}_t^i(a_{k,t}^i)] \cdot [a_{k,t}^i - P_t^i(a_{k,t}^i)] \cdot x$$

(d) Update probability $P_t^i(a_{k,t}^i)$, which is the probability of taking action $a_{k,t}^i$:
For each $a_{k,t}^i \in \mathbf{a}_t^i$

$$P_t^i(a_{k,t}^i) = \max \left[\min \left\{ \frac{1}{1 + e^{-Q_{t+1}^i(a_{k,t}^i) \cdot x}}, 1 - \varepsilon \right\}, \varepsilon \right]$$

ALGORITHM 5: RL algorithm for RL-DCS [27].

Repeat

(a) Take action $a_t^i \in A$

(b) Exchange collaboration message $D_{t,1}^i$ with SU neighbor agents // First round of collaboration

(c) Determine delayed reward $r_{t+1}^i(\mathbf{s}_{t+1}^i, a_t^i)$

(d) Exchange collaboration message $D_{t,2}^i$ with SU neighbor agents // Second round of collaboration

(e) Choose action $a_{t+1}^i \in A$

(f) Update θ_{t+1} (see (9))

(g) Update Q-value, $Q_{t+1}^i(\mathbf{s}_t^i, a_t^i)$

ALGORITHM 6: RL algorithm for the channel sensing scheme [20].

Algorithm 5 presents the RL algorithm for the scheme. The action $\mathbf{a}_t^i = (a_{1,t}^i, a_{2,t}^i, \dots, a_{K,t}^i) \in A_1 \times A_2 \times \dots \times A_K$ is chosen using a Bernoulli random variable [27]. The PU agent receives reward $r_{t+1}^i(\mathbf{a}_t^i)$ computed by ECE and computes the average reward $\bar{r}_t^i(a_{k,t}^i)$ for each subaction $a_{k,t}^i$ at time t using the exponential moving average [27]. Denote the probability of taking action $a_{k,t}^i$ by $P_t^i(a_{k,t}^i)$ and the current overall unused spectrum, which is the ratio of the unused bandwidth to the total bandwidth of a cell, by x . Upon receiving reward $r_{t+1}^i(\mathbf{a}_t^i)$, the PU agent updates the Q-value $Q_{t+1}^i(a_{k,t}^i)$ for each action $a_{k,t}^i \in \mathbf{a}_t^i$. Finally, the probability of taking action $a_{k,t}^i$, specifically, $P_t^i(a_{k,t}^i)$, is updated. Note that the exploration probability is ε .

4.8. Model 8: Collaborative Model. Collaborative model enables a SU agent to collaborate with its SU neighbor agents and subsequently make local decisions independently in distributed CR networks. It enables the agents to learn and achieve an optimal joint action. A joint action is defined as the actions taken by all the agents throughout the entire network. An optimal joint action is the actions taken by all the agents throughout the entire network that provides an ideal and optimal network-wide performance. Hence, the collaborative model reduces the selfishness of each agent through taking other agents' actions or strategies into account. The collaboration may take the form of exchanging local information, including knowledge (Q-value), observations, and decisions, among the SU agents.

Lundén et al. [20] propose a collaborative channel sensing (A2) scheme, and it has been shown to maximize the amount of white spaces found. The purposes of this scheme are twofold:

- (i) firstly, it selects channels with more white spaces for channel sensing purpose;
- (ii) secondly, it selects channels so that the SU agents diversify their sensing channels. In other words, the SU agents perform channel sensing in various channels.

Table 13 shows the RL model for the scheme. The state $\mathbf{s}_t^i \in S_1 \times S_2 \times \dots \times S_K$ represents the belief on the availability of a set of channels for data transmission. An action $a_t^i \in A$, which is part of the joint action \mathbf{a}_t representing all the actions taken by SU agent i and its SU neighbor agents, represents a single channel chosen by SU agent i for channel sensing purpose. The reward $r_{t+1}^i(\mathbf{s}_{t+1}^i, a_t^i)$ represents the number of channels identified as being idle (or free) at time $t + 1$ by SU agent i . The RL model is embedded in each SU agent.

Algorithm 6 presents the RL algorithm for the scheme, and it is comprised of two rounds of collaboration message exchanges. After taking action $a_t^i \in A$, the SU agent i exchange collaboration messages $D_{t,1}^i = (a_t^i, \beta_t^i)$ with its SU neighbor agents. The $D_{t,1}^i$ is comprised of two-tuple information, namely, SU agent i 's action a_t^i and SU agent i 's sensing outcomes β_t^i . SU agent i determines the delayed reward based on $D_{t,1}^i$. Next, the SU agent i exchanges collaboration messages

TABLE 12: RL model for the DCS scheme [27].

Action	$\mathbf{a}_t^i = (a_{1,t}^i, a_{2,t}^i, \dots, a_{K,t}^i) \in A_1 \times A_2 \times \dots \times A_K$; each subaction $a_{k,t}^i \in A_k = \{0, 1\}$ represents the presence of PU activities. Specifically, $a_{k,t}^i = 0$ if a PU agent cannot transmit in channel k and so it becomes white space, and $a_{k,t}^i = 1$ if the PU agent can transmit in channel k .
Reward	$r_{t+1}^i(\mathbf{a}_t^i) = \begin{cases} 0, & \text{if } \widehat{TH}(\mathbf{a}_t^i) < T_{th} \\ \lambda \cdot \widehat{\eta}(\mathbf{a}_t^i) + \mu \cdot \overline{W}(\mathbf{a}_t^i), & \text{otherwise.} \end{cases}$

TABLE 13: RL model for the channel sensing scheme [20].

State	$\mathbf{s}_t^i = (b(s_{1,t}^i), b(s_{2,t}^i), \dots, b(s_{K,t}^i)) \in S_1 \times S_2 \times \dots \times S_K$; each substate $b(s_{k,t}^i) \in S_k = \{0, 1\}$ indicates SU i 's belief about channel k , and it has a value of 0 (busy) or 1 (idle)
Action	$a_t^i \in A = \{1, 2, \dots, K\}$; each action represents a single channel chosen for channel sensing purpose
Reward	$r_{t+1}^i(\mathbf{s}_{t+1}^i, a_t^i)$ represents the number of channels identified as being idle by SU node i

$D_{t,2}^i = a_{t+1}^i$ with its SU neighbor agents. During the second round of collaboration message exchange, a SU agent i chooses its action a_{t+1}^i for the next time instance upon receiving $D_{t,2}^j$ from SU neighbor agent j . Note that the SU agent transmission order affects the action selection. This is because a SU agent may receive and use information obtained from its preceding agents, and so it can make decisions using more updated information in the second round. Since one of the main purposes is to enable the SU agents to diversify their sensing channels, the SU agents choose action a_{t+1}^i from a preferred channel set. The preferred channel set is comprised of sensing channels which are yet to be chosen by the preceding SU agents. The SU agent chooses channels with the maximum Q-value from the preferred channel set. Finally, the SU agent updates Q-value $Q_{t+1}^i(s_t^i, a_t^i)$ and $\theta_{t+1}(s_t^i, a_t^i)$ (see Section 3.5).

Liu et al. [39] propose a collaborative DCS (A1) scheme that applies a collaborative model, and it has been shown to achieve a near-optimal throughput performance. The purpose of this scheme is to enable each SU link to maximize its individual delayed rewards, specifically, the SNR level. Note that this collaboration approach assumes that an agent has full observation of the actions and policies adopted by all the other SU links at any time instance. Hence, (1) is rewritten as follows:

$$Q_{t+1}^i(s_t^i, a_t^i, \mathbf{a}_t^{-i}) \leftarrow (1 - \alpha) Q_t^i(s_t^i, a_t^i, \mathbf{a}_t^{-i}) + \alpha \left[r_{t+1}^i(s_{t+1}^i) + \gamma \max_{a_i, \mathbf{a}_{-i} \in A} Q_t^i(s_{t+1}^i, a_i, \mathbf{a}_{-i}^{-i}) \right], \quad (18)$$

where a_t^i represents the action taken by agent i and \mathbf{a}_t^{-i} represents the joint action taken by all the SU agents throughout the entire CR network except agent i . Note that $a_t^i \cap \mathbf{a}_t^{-i} \in A$, where A represents joint actions by all the SU agents throughout the entire CR network. Therefore, (19) is similar to the traditional RL approach except when an action a_t^i becomes a joint action $a_t^i \cap \mathbf{a}_t^{-i}$ (or set of actions). To take into account actions taken by the other agents \mathbf{a}_t^{-i} , agent i updates an average Q-value $\overline{Q}_t^i(s_t^i, a_t^i)$, which is the average Q-value of

agent i in state s_t^i if it takes action a_t^i , while the other agents take action \mathbf{a}_t^{-i} . The $\overline{Q}_t^i(s_t^i, a_t^i)$ is updated as follows:

$$\overline{Q}_t^i(s_t^i, a_t^i) = \sum_{\mathbf{a}_{-i}} Q_t^i(s_t^i, a_t^i, \mathbf{a}_{-i}^{-i}) \prod_{j=1, j \neq i}^N \pi_j(s_t^j, a_t^j, \mathbf{a}_t^{-i}), \quad (19)$$

where N is the number of agents.

Next, $\overline{Q}_t^i(s_t^i, a_t^i)$ is applied in action selection using the Boltzmann equation (see Section 3.6). Further research can be pursued to reduce communication overheads and to enable indirect coordination among the agents.

4.9. Model 9: Competitive Model. Competitive model enables a SU agent to compete with its SU neighbor agents and subsequently make local decisions independently in CR networks. The competitive model enables an agent to make optimal actions in worst-case scenarios in the presence of competitor agents, which attempt to minimize the accumulated rewards of the agent. Note that the competitor agent may also possess the capability to observe, learn, and carry out the optimal actions in order to deteriorate the agents' accumulated rewards.

Wang et al. [14] propose an antijamming approach (A3) scheme called channel hopping, and it applies minimax-Q learning to implement the competitive model. This approach has been shown to maximize the accumulated rewards (e.g., throughput) in the presence of jamming attacks. Equipped with a limited number of transceivers, the malicious SUs aim to minimize the accumulated rewards of SU agents through constant packet transmission in a number of channels in order to prevent spectrum utilization by SU agents. The purposes of the channel hopping scheme are twofold:

- (i) firstly, it introduces randomness in channel selection so that the malicious SUs do not jam its selected channels for data transmission;
- (ii) secondly, it selects a proper number of control and data channels in a single frequency band for control and data packet transmissions. Note that each frequency band consists of a number of channels. Due to the criticality of control channel, duplicate control

TABLE 14: RL model for the channel hopping scheme [14].

State	$\mathbf{s}_{k,t}^i = (P_{k,t}^i, g_{k,t}^i, N_{C,k,t}^i, N_{D,k,t}^i) \in S_1 \times S_2 \times S_3 \times S_4$; substate $P_{k,t}^i \in S_1 = \{0, 1\}$ indicates an idle or busy channel; specifically, $P_{k,t}^i = 0$ if PU activity does not exist, and $P_{k,t}^i = 1$ if PU activity exists; substate $g_{k,t}^i \in S_2 = \{q_1, q_2, \dots, q_{N_g}\}$ represents gain, while $N_{C,k,t}^i \in S_3$ and $N_{D,k,t}^i \in S_4$ represent the numbers of control and data channels that get jammed, respectively
Action	$\mathbf{a}_t^i = \{a_{1,t}^i, a_{2,t}^i, \dots, a_{K,t}^i\} \in A$; subaction $a_{k,t}^i = (a_{C_1,k,t}^i, a_{D_1,k,t}^i, a_{C_2,k,t}^i, a_{D_2,k,t}^i)$, where action $a_{C_1,k,t}^i$ (or $a_{D_1,k,t}^i$) indicates that the agent will transmit control (or data) packets in $a_{C_1,k,t}^i$ (or $a_{D_1,k,t}^i$) channels uniformly selected from the previously unjammed channels, while action $a_{C_2,k,t}^i$ (or $a_{D_2,k,t}^i$) indicates that the agent will transmit control (or data) packets in $a_{C_2,k,t}^i$ (or $a_{D_2,k,t}^i$) channels uniformly selected from the previously jammed channels
Reward	$r_{t+1}^i(\mathbf{s}_{k,t+1}^i, \mathbf{a}_t^i, \mathbf{a}_t^m)$ represents the channel gain

<p>Repeat</p> <p>(a) Choose action \mathbf{a}_t^i</p> <p>(b) Update Q-value $Q_t^i(\mathbf{s}_{k,t}^i, \mathbf{a}_t^i, \mathbf{a}_t^m)$ as follows:</p> $Q_{t+1}^i(\mathbf{s}_{k,t}^i, \mathbf{a}_t^i, \mathbf{a}_t^m) = (1 - \alpha) Q_t^i(\mathbf{s}_{k,t}^i, \mathbf{a}_t^i, \mathbf{a}_t^m) + \alpha [r_{t+1}^i(\mathbf{s}_{k,t+1}^i, \mathbf{a}_t^i, \mathbf{a}_t^m) + \gamma V(\mathbf{s}_{k,t+1}^i)]$ <p>(c) Update optimal strategy $\pi^{i,*}(\mathbf{s}_{k,t}^i)$ as follows:</p> $\pi^{i,*}(\mathbf{s}_{k,t}^i) = \arg \max_{\pi^{i,*}(\mathbf{s}_{k,t}^i)} \min_{\pi^m(\mathbf{s}_{k,t}^i)} \sum_{\mathbf{a}} \pi^i(\mathbf{s}_{k,t}^i, \mathbf{a}) Q_t^i(\mathbf{s}_{k,t}^i, \mathbf{a}, \mathbf{a}_t^m)$ <p>(d) Update value function $V(\mathbf{s}_{k,t}^i)$ as follows:</p> $V(\mathbf{s}_{k,t}^i) = \min_{\pi^m(\mathbf{s}_{k,t}^i)} \sum_{\mathbf{a}} \pi^{i,*}(\mathbf{s}_{k,t}^i, \mathbf{a}) Q_t^i(\mathbf{s}_{k,t}^i, \mathbf{a}, \mathbf{a}_t^m)$
--

ALGORITHM 7: RL algorithm for the channel hopping scheme [14].

packets may be transmitted in multiple channels to minimize the effects of jamming, and so a proper number of control channels are necessary.

Note that, as competitors, the malicious SUs aim to minimize the accumulated rewards of SU agents. Table 14 shows the RL model for the scheme. Each state is comprised of four-tuple information; specifically, $\mathbf{s}_{k,t}^i = (P_{k,t}^i, g_{k,t}^i, N_{C,k,t}^i, N_{D,k,t}^i) \in S_1 \times S_2 \times S_3 \times S_4$. With respect to frequency band k , the substate $P_{k,t}^i \in S_1 = \{0, 1\}$ represents the presence of PU activities and $g_{k,t}^i \in S_2 = \{q_1, q_2, \dots, q_{N_g}\}$ represents gain, while $N_{C,k,t}^i \in S_3$ and $N_{D,k,t}^i \in S_4$ represent the numbers of control and data channels that get jammed, respectively. An action $\mathbf{a}_t^i \in A$ represents channel selections within a single frequency band for control and data packet transmissions purpose, and the channels may be jammed or not jammed in the previous time slot. The reward $r_{t+1}^i(\mathbf{s}_{k,t+1}^i, \mathbf{a}_t^i, \mathbf{a}_t^m)$ represents the gain (e.g., throughput) of using channels that are not jammed. Note that the reward $r_{t+1}^i(\mathbf{s}_{k,t+1}^i, \mathbf{a}_t^i, \mathbf{a}_t^m)$ is dependent on the malicious SU's (or competitor's) action \mathbf{a}_t^m . The RL model is embedded in each SU agent.

Algorithm 7 presents the RL algorithm for the scheme. In step (b), the Q-function is dependent on the competitor's action \mathbf{a}_t^m , which is the channels chosen by the malicious SUs for jamming purpose. In step (c), the agent determines its optimal policy $\pi^{i,*}(\mathbf{s}_{k,t}^i)$, in which the competitor is assumed to take its optimal action that minimizes the Q-value, and hence the term $\min_{\pi^m(\mathbf{s}_{k,t}^i)}$. Nevertheless, in this worst-case scenario, the agent chooses an optimal action and hence

the term $\arg \max_{\pi^{i,*}(\mathbf{s}_{k,t}^i)}$. In step (d), the agent updates its value function $V(\mathbf{s}_{k,t}^i)$, which is applied to update the Q-value in step (b) in the next time instant. Using the optimal policy $\pi^{i,*}(\mathbf{s}_{k,t}^i)$ obtained in step (c), the agent calculates its value function $V(\mathbf{s}_{k,t}^i)$, which is an approximate of the discounted future reward. Again, the competitor is assumed to take its optimal action that minimizes the agent's Q-value and hence the term $\min_{\pi^m(\mathbf{s}_{k,t}^i)}$.

5. Performance Enhancements

Table 15 shows the performance enhancements brought about by the application of the traditional and enhanced RL algorithms in various schemes in CR networks. The RL approach has been shown to achieve the following performance enhancement.

- (P1) *Higher Throughput/Goodput*. Higher throughput (or goodput) indicates higher packet delivery rate, higher successful packet transmission rate, and lower packet loss rate.
- (P2) *Lower End-to-End Delay/Link Delay*. Lower end-to-end delay, which is the summation of link delays along a route, indicates shorter time duration for packets to traverse from a source node to its destination node.
- (P3) *Lower Level of Interference to PUs*. Lower level of interference to PUs indicates lower number of collisions with PU activities.

- (P4) *Lower Number of Sensing Channels.* The lower number of sensing channels indicates lower sensing overheads (i.e., delays and energy consumption).
- (P5) *Higher Overall Spectrum Utilization.* In order to increase the overall spectrum utilization, Chen et al. [24] increase channel access time, while Jiang et al. [30, 31] reduce blocking and dropping probabilities of calls, respectively.
- (P6) *Lower Number of Channel Switches.* Chen et al. [24] reduce number of channel switches in order to reduce channel switching time.
- (P7) *Lower Energy Consumption.* Lower energy consumption indicates longer network lifetime and number of survival nodes.
- (P8) *Lower Probability of False Alarm.* Lo and Akyildiz [3] reduce false alarm, which occurs when a PU is mistakenly considered present in an available channel, in channel sensing (A2).
- (P9) *Higher Probability of PU Detection.* Lo and Akyildiz [3] increase the probability of PU detection in order to reduce miss detection in channel sensing (A2). Miss detection occurs whenever a PU is mistakenly considered absent in a channel with PU activities.
- (P10) *Higher Number of Channels Being Sensed Idle.* Lundén et al. [20] increase the number of channels being sensed idle, which contains more white spaces.
- (P11) *Higher Accumulated Rewards.* Wang et al. [14] increase the accumulated rewards, which represent gains, such as throughput performance. Xiao et al. [38] improve SU's total payoff, which is the difference between gained rewards (or revenue) and total cost incurred.

6. Open Issues

This section discusses open issues that can be pursued in this research area.

6.1. Enhanced Exploration Approaches. While larger value of exploration probability may be necessary if the dynamicity of the operating environment is high, the opposite holds whenever the operating environment is rather stable. Generally speaking, exploration helps to increase the convergence rate of a RL scheme. Nevertheless, higher exploration rate may cause fluctuation in performance (e.g., end-to-end delay and packet loss) due to the selection of nonoptimal actions. For instance, in a dynamic channel selection scheme (A1), the performance may fluctuate due to the frequent exploration of nonoptimal channels. Similarly, in a routing scheme (A7), the performance may fluctuate due to the frequent exploration of nonoptimal routes. Further research could be pursued to investigate the possibility of achieving exploration without compromising the application performance. Additionally, further research could be pursued to investigate how to achieve an optimal trade-off between exploration and exploitation in a diverse range of operating environments. For

instance, through simulation, Li [6] found that, with higher learning rate α and lower temperature τ_t , the convergence rate of the Q-value is faster.

6.2. Fully Decentralized Channel Auction Models. To the best of our knowledge, most of the existing RL-based channel auction models (see Section 4.6) have been applied in centralized CR networks, in which a centralized entity (e.g., base station) allocates white spaces to SU hosts with winning bids. The centralized entity may perform simple tasks, such as allocating white spaces to SU hosts with winning bids [16], or it may learn using RL to maximize its utility [36]. The main advantage of the centralized entity is that it simplifies the management of the auction process and the interaction among nodes. Nevertheless, it introduces challenges to implementation due to additional cost and feasibility of having a centralized entity in all scenarios. While there have been increasing efforts to enhance the performance of the RL-based auction models, further research is necessary to investigate fully decentralized RL-based auction models, which do not rely on a centralized entity, along with their requirements and challenges. For instance, by incorporating the cooperative learning feature (see Section 3.7.3) into the RL auction model, SUs can exchange auction information with PUs and other SUs in a decentralized manner, which may enable them to perform bidding decisions without the need of a centralized entity. However, this may introduce other concerns such as security and nodes' selfishness, which can be interesting directions for further research.

6.3. Enhancement on the Efficiency of RL Algorithm. The application of RL in various application schemes in CR networks may introduce complexities, and so the efficiency of the RL algorithm should be further improved. As an example, the collaborative model (see Section 4.8) requires explicit coordination in which the neighboring agents exchange information among themselves in order to expedite convergence to optimal joint action. This enhances the network performance at the expense of higher amount of control overhead. Hence, further research is necessary to investigate the possibility of indirect coordination. Moreover, the network performance may further improve with reduced overhead incurred by RL. As another example, while RL has been applied to address security issues in CR networks (see application (A3)), the introduction of RL into CR schemes may introduce more vulnerabilities into the system. This is because the malicious SUs or attackers may affect the operating environment or manipulate the information so that the honest SUs' knowledge is adversely affected.

6.4. Application of RL in New Application Schemes. The wide range of enhanced RL algorithms, including the dual Q-function, partial observable, actor-critic, auction, internal self-learning, collaborative, and competitive models (see Sections 4.3–4.9), can be extended to other applications in CR networks, including emerging networks such as cognitive maritime wireless ad hoc networks and cognitive radio sensor networks [40], in order to achieve context awareness and

intelligence, which are the important characteristics of cognition cycle (see Section 2.2.1). For instance, the collaborative model (see Section 4.8) enables an agent to collaborate with its neighbor agents in order to make decisions on action selection, which is part of an optimal joint action. This model is suitable to be applied in most application schemes that require collaborative efforts, such as trust and reputation system [41] and cooperative communications, although the application of RL in those schemes is yet to be explored. In trust and reputation management, SUs make collaborative effort to detect malicious SUs, such that malicious SUs are assigned low trust and reputation values. Additionally, Section 3 presents new features of each component of RL, which can be applied to enhance the performance of existing RL-based applications schemes in CR networks. Further research could also be pursued to

- (i) apply new RL approaches, such as two-layered multi-agent RL model [42], to CR network applications,
- (ii) investigate RL models and algorithms applied to other kinds of networks such as cellular radio access networks [43] and sensor networks [44], which may be leveraged to provide performance enhancement in CR networks,
- (iii) apply or integrate the RL features and enhancements (e.g., state, action, and reward representations) to other learning-based approaches, such as the neural network-based approach [45].

6.5. Lack of Real Implementation of RL in CR Testbed. Most of the existing RL-based schemes have been evaluated using simulations, which have been shown to achieve performance enhancements. Nevertheless, to the best of our knowledge, there is lack of implementation of RL-based schemes in CR platform. Real implementation of the RL algorithms is important to validate their correctness and performance in real CR environment, which may also allow further refinements on these algorithms. To this end, further research is necessary to investigate the implementation and challenges of the RL-based scheme on CR platform.

7. Conclusions

Reinforcement learning (RL) has been applied in cognitive radio (CR) networks to achieve context awareness and intelligence. Examples of schemes are dynamic channel selection, channel sensing, security enhancement mechanism, energy efficiency enhancement mechanism, channel auction mechanism, medium access control, routing, and power control mechanism. To apply the RL approach, several representations may be necessary including state and action, as well as delayed and discounted rewards. Based on the CR context, this paper presents an extensive review on the enhancements of these representations, as well as other features including Q-function, trade-off between exploration and exploitation, updates of learning rate, rules, and cooperative learning. Most importantly, this paper presents an extensive review on a wide range of enhanced RL algorithms in CR context. Examples of

the enhanced RL models are dual Q-function, partial observable, actor-critic, auction, internal self-learning, and collaborative and competitive models. The enhanced algorithms provide insights on how various schemes in CR networks can be approached using RL. Performance enhancements achieved by the traditional and enhanced RL algorithms in CR networks are presented. Certainly, there is a great deal of future works in the use of RL, and we have raised open issues in this paper.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported by the Malaysian Ministry of Science, Technology and Innovation (MOSTI) under Science Fund 01-02-16-SF0027.

References

- [1] I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, "NeXt generation/dynamic spectrum access/cognitive radio wireless networks: a survey," *Computer Networks*, vol. 50, no. 13, pp. 2127–2159, 2006.
- [2] Y. Tang, D. Grace, T. Clarke, and J. Wei, "Multichannel non-persistent CSMA MAC schemes with reinforcement learning for cognitive radio networks," in *Proceedings of the 11th International Symposium on Communications and Information Technologies (ISCIT '11)*, pp. 502–506, October 2011.
- [3] B. F. Lo and I. F. Akyildiz, "Reinforcement learning-based cooperative sensing in cognitive radio ad hoc networks," in *Proceedings of the IEEE 21st International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC '10)*, pp. 2244–2249, September 2010.
- [4] J. Peng, J. Li, S. Li, and J. Li, "Multi-relay cooperative mechanism with Q-learning in cognitive radio multimedia sensor networks," in *Proceedings of the 10th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom '11)*, pp. 1624–1629, 2011.
- [5] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, MIT Press, Cambridge, Mass, USA, 1998.
- [6] H. Li, "Multi-agent Q-learning of channel selection in multi-user cognitive radio systems: a two by two case," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC '09)*, pp. 1893–1898, October 2009.
- [7] J. Mitola III and G. Q. Maguire Jr., "Cognitive radio: making software radios more personal," *IEEE Personal Communications*, vol. 6, no. 4, pp. 13–18, 1999.
- [8] K.-L. A. Yau, P. Komisarczuk, and P. D. Teal, "Enhancing network performance in Distributed Cognitive Radio Networks using single-agent and multi-agent Reinforcement Learning," in *Proceedings of the 35th Annual IEEE Conference on Local Computer Networks (LCN '10)*, pp. 152–159, October 2010.
- [9] K.-L. A. Yau, P. Komisarczuk, and P. D. Teal, "Achieving context awareness and intelligence in distributed cognitive radio networks: a payoff propagation approach," in *Proceedings of the 25th IEEE International Conference on Advanced Information*

- Networking and Applications Workshops (WAINA '11)*, pp. 210–215, March 2011.
- [10] H. Li, D. Grace, and P. D. Mitchell, "Cognitive radio multiple access control for unlicensed and open spectrum with reduced spectrum sensing requirements," in *Proceedings of the 7th International Symposium on Wireless Communication Systems (ISWCS '10)*, pp. 1046–1050, September 2010.
 - [11] M. Di Felice, K. R. Chowdhury, W. Meleis, and L. Bononi, "To sense or to transmit: a learning-based spectrum management scheme for cognitive radio mesh networks," in *Proceedings of the 5th Annual IEEE Workshop on Wireless Mesh Networks (WiMesh '10)*, pp. 19–24, June 2010.
 - [12] S. Parvin, F. K. Hussain, O. K. Hussain, S. Han, B. Tian, and E. Chang, "Cognitive radio network security," *Journal of Network and Computer Applications*, vol. 35, no. 6, pp. 1691–1708, 2012.
 - [13] N. Vucevic, I. F. Akyildiz, and J. Pérez-Romero, "Cooperation reliability based on reinforcement learning for cognitive radio networks," in *Proceedings of the 5th IEEE Workshop on Networking Technologies for Software-Defined Radio (SDR '10)*, pp. 19–24, June 2010.
 - [14] B. Wang, Y. Wu, K. J. R. Liu, and T. C. Clancy, "An anti-jamming stochastic game for cognitive radio networks," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 4, pp. 877–889, 2011.
 - [15] K. Zheng and H. Li, "Achieving energy efficiency via drowsy transmission in cognitive radio," in *Proceedings of the 53rd IEEE Global Communications Conference (GLOBECOM '10)*, pp. 1–6, December 2010.
 - [16] Z. Chen and R. C. Qiu, "Q-learning based bidding algorithm for spectrum auction in cognitive radio," in *Proceedings of the IEEE SoutheastCon 2011 Building Global Engineers*, pp. 409–412, March 2011.
 - [17] H. Li, D. Grace, and P. D. Mitchell, "Collision reduction in cognitive radio using multichannel 1-persistent CSMA combined with reinforcement learning," in *Proceedings of the 5th International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCom '10)*, pp. 1–5, June 2010.
 - [18] H. Li, D. Grace, and P. D. Mitchell, "Multiple access with multi-dimensional learning for cognitive radio in open spectrum," in *Proceedings of the 11th International Symposium on Communications and Information Technologies (ISCIT '11)*, pp. 298–302, October 2011.
 - [19] Y. Yao and Z. Feng, "Centralized channel and power allocation for cognitive radio networks: a Q-learning solution," in *Proceedings of the Future Network and Mobile Summit (FutureNetworkSummit '10)*, pp. 1–8, June 2010.
 - [20] J. Lundén, V. Koivunen, S. R. Kulkarni, and H. V. Poor, "Reinforcement learning based distributed multiagent sensing policy for cognitive radio networks," in *Proceedings of the IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN '11)*, pp. 642–646, May 2011.
 - [21] M. Di Felice, K. R. Chowdhury, A. Kassler, and L. Bononi, "Adaptive sensing scheduling and spectrum selection in cognitive wireless mesh networks," in *Proceedings of the 20th International Conference on Computer Communications and Networks (ICCCN '11)*, pp. 1–6, August 2011.
 - [22] W. Jouini, D. Ernst, C. Moy, and J. Palicot, "Upper confidence bound based decision making strategies and dynamic spectrum access," in *Proceedings of the IEEE International Conference on Communications (ICC '10)*, pp. 1–5, May 2010.
 - [23] W. Jouini, C. Moy, and J. Palicot, "Upper confidence bound algorithm for opportunistic spectrum access with sensing errors," in *Proceedings of the 5th International Conference on Cognitive Radio Oriented Wireless Networks & Communications (CROWNCOM '10)*, 2010.
 - [24] S. Chen, R. Vuyyuru, O. Altintas, and A. M. Wyglinski, "On optimizing vehicular dynamic spectrum access networks: automation and learning in mobile wireless environments," in *Proceedings of the IEEE Vehicular Networking Conference (VNC '11)*, pp. 39–46, 2011.
 - [25] K. Chowdhury, R. Doost-Mohammady, W. Meleis, M. Di Felice, and L. Bononi, "Cooperation and communication in cognitive radio networks based on TV spectrum experiments," in *Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM '11)*, pp. 1–9, June 2011.
 - [26] M. A. L. Thathachar and P. S. Sastry, "Varieties of learning automata: an overview," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 32, no. 6, pp. 711–722, 2002.
 - [27] F. Bernardo, R. Agustí, J. Pérez-Romero, and O. Sallent, "Distributed spectrum management based on reinforcement learning," in *Proceedings of the 4th International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM '09)*, pp. 1–6, June 2009.
 - [28] J. R. Kok and N. Vlassis, "Collaborative multiagent reinforcement learning by payoff propagation," *Journal of Machine Learning Research*, vol. 7, pp. 1789–1828, 2006.
 - [29] Y. B. Reddy, "Detecting primary signals for efficient utilization of spectrum using Q-learning," in *Proceedings of the International Conference on Information Technology: New Generations (ITNG '08)*, pp. 360–365, April 2008.
 - [30] T. Jiang, D. Grace, and Y. Liu, "Two-stage reinforcement-learning-based cognitive radio with exploration control," *IET Communications*, vol. 5, no. 5, pp. 644–651, 2011.
 - [31] T. Jiang, D. Grace, and P. D. Mitchell, "Efficient exploration in reinforcement learning-based cognitive radio spectrum sharing," *IET Communications*, vol. 5, no. 10, pp. 1309–1317, 2011.
 - [32] S. Kumar and R. Miikkulainen, "Dual reinforcement Q-routing: an on-line adaptive routing algorithm," in *Proceedings of the Artificial Neural Networks in Engineering Conference (ANNIE '97)*, pp. 231–238, 1997.
 - [33] B. Xia, M. H. Wahab, Y. Yang, Z. Fan, and M. Sooriyabandara, "Reinforcement learning based spectrum-aware routing in multi-hop cognitive radio networks," in *Proceedings of the 4th International Conference on Cognitive Radio Oriented Wireless Networks and Communications (CROWNCOM '09)*, pp. 1–6, June 2009.
 - [34] M. Bkassiny, S. K. Jayaweera, and K. A. Avery, "Distributed Reinforcement Learning based MAC protocols for autonomous cognitive secondary users," in *Proceedings of the 20th Annual Wireless and Optical Communications Conference (WOCC '11)*, pp. 1–6, April 2011.
 - [35] M. L. Littman, A. R. Cassandra, and L. P. Kaelbling, *Learning Policies For Partially Observable Environments: Scaling Up*. Readings in Agents, Morgan Kaufmann, San Francisco, Calif, USA, 1998.
 - [36] S. K. Jayaweera, M. Bkassiny, and K. A. Avery, "Asymmetric cooperative communications based spectrum leasing via auctions in cognitive radio networks," *IEEE Transactions on Wireless Communications*, vol. 10, no. 8, pp. 2716–2724, 2011.

- [37] F. Fu and M. van der Schaar, "Learning to compete for resources in wireless stochastic games," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 4, pp. 1904–1919, 2009.
- [38] Y. Xiao, G. Bi, and D. Niyato, "Game theoretic analysis for spectrum sharing with multi-Hop relaying," *IEEE Transactions on Wireless Communications*, vol. 10, no. 5, pp. 1527–1537, 2011.
- [39] X. Liu, J. Wang, Q. Wu, and Y. Yang, "Frequency allocation in dynamic environment of cognitive radio networks based on stochastic game," in *Proceedings of the 3rd International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC '11)*, pp. 497–502, October 2011.
- [40] F. Zeng, Y. Tang, and J. Pu, "Multichannel broadcast based on home channel for cognitive radio sensor networks," *The Scientific World Journal*, vol. 2014, Article ID 725210, 6 pages, 2014.
- [41] M. H. Ling, K. L. A. Yau, and G. S. Poh, "Trust and reputation management in cognitive radio networks: a survey," *Security and Communication Networks*, 2014.
- [42] B.-N. Wang, Y. Gao, Z.-Q. Chen, J.-Y. Xie, and S.-F. Chen, "A two-layered multi-agent reinforcement learning model and algorithm," *Journal of Network and Computer Applications*, vol. 30, no. 4, pp. 1366–1376, 2007.
- [43] R. Li, Z. Zhao, X. Chen, J. Palicot, and H. Zhang, "TACT: a transfer actor-critic learning framework for energy saving in cellular radio access networks," *IEEE Transactions on Wireless Communications*, vol. 13, no. 4, pp. 2000–2011, 2014.
- [44] M. Maalej, S. Cherif, and H. Besbes, "QoS and Energy aware cooperative routing protocol for wildlife monitoring wireless sensor networks," *The Scientific World Journal*, vol. 2013, Article ID 437926, 11 pages, 2013.
- [45] W. Zame, J. Xu, and M. van der Schaar, "Cooperative multi-agent learning and coordination for cognitive radio networks," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 3, pp. 464–477, 2014.

Research Article

Firefly Algorithm for Cardinality Constrained Mean-Variance Portfolio Optimization Problem with Entropy Diversity Constraint

Nebojsa Bacanin and Milan Tuba

Faculty of Computer Science, Megatrend University Belgrade, 11070 Belgrade, Serbia

Correspondence should be addressed to Milan Tuba; tuba@iee.org

Received 9 April 2014; Accepted 5 May 2014; Published 29 May 2014

Academic Editor: Xin-She Yang

Copyright © 2014 N. Bacanin and M. Tuba. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Portfolio optimization (selection) problem is an important and hard optimization problem that, with the addition of necessary realistic constraints, becomes computationally intractable. Nature-inspired metaheuristics are appropriate for solving such problems; however, literature review shows that there are very few applications of nature-inspired metaheuristics to portfolio optimization problem. This is especially true for swarm intelligence algorithms which represent the newer branch of nature-inspired algorithms. No application of any swarm intelligence metaheuristics to cardinality constrained mean-variance (CCMV) portfolio problem with entropy constraint was found in the literature. This paper introduces modified firefly algorithm (FA) for the CCMV portfolio model with entropy constraint. Firefly algorithm is one of the latest, very successful swarm intelligence algorithm; however, it exhibits some deficiencies when applied to constrained problems. To overcome lack of exploration power during early iterations, we modified the algorithm and tested it on standard portfolio benchmark data sets used in the literature. Our proposed modified firefly algorithm proved to be better than other state-of-the-art algorithms, while introduction of entropy diversity constraint further improved results.

1. Introduction

Since most real-life problems can be modeled as optimization tasks, many methods and techniques that could tackle such problems were developed. Thus, the optimization became one of the most applicable fields in mathematics and computer science.

The difficulty of an optimization problem depends on the mathematical relationships between the objective function, potential constraints, and decision variables. Hard optimization problems can be combinatorial (discrete) or continuous (global optimization), while continuous problems can be further be classified as constrained or unconstrained (bound constrained).

The optimization problem becomes even harder when some variables are real-valued, while others can take only integer values. Such mixed continuous/discrete problems usually require problem-specific search techniques in order to generate optimal, or near optimal solution.

One representative example of such hard optimization problems is portfolio optimization, a well-known issue in economics and finance. Many methods and heuristics were developed to optimize various models and formulations of the portfolio problem [1]. Various portfolio optimization problem models may or may not include different constraints in their formulations. Also, to enhance the diversity of portfolio, some approaches include entropy in its formulations [2].

Unconstrained (bound constrained) optimization is formulated as D -dimensional minimization or maximization problem:

$$\min \text{ (or max) } f(x), \quad x = (x_1, x_2, x_3, \dots, x_D) \in S, \quad (1)$$

where x represents a real vector with $D \geq 1$ components and $S \in R^D$ is hyper-rectangular search space with D dimensions constrained by lower and upper bounds:

$$lb_i \leq x_i \leq ub_i, \quad i \in [1, D]. \quad (2)$$

In (2), lb_i and ub_i are lower and upper bounds for the i th problem component, respectively.

The nonlinear constrained optimization problem in the continuous space can be formulated in the same way as in (1), but in this case $x \in F \subseteq S$, where S is D -dimensional hyper-rectangular space as defined in (2) while $F \subseteq S$ represents the feasible region defined by the set of m linear or nonlinear constraints:

$$\begin{aligned} g_j(x) &\leq 0, \quad \text{for } j \in [1, q], \\ h_j(x) &= 0, \quad \text{for } j \in [q + 1, m], \end{aligned} \quad (3)$$

where q is the number of inequality constraints, and $m - q$ is the number of equality constraints.

Fundamental versions of algorithms and metaheuristics for constrained numerical optimization problems do not include methods for dealing with constraints. Therefore, constraint handling techniques are usually added to these algorithms to improve and redirect the search process towards the feasible region of the search space. Equality constraints make optimization even harder by shrinking the feasible search space which becomes very small compared to the entire search space. To tackle such problem, equality constraints are replaced with the inequality constraints [3]:

$$|h(x)| - v \leq 0, \quad (4)$$

where $v > 0$ is some small violation tolerance, usually dynamically adjusted.

Since hard optimization problems are unsolvable in a reasonable amount of time, the exact methods which trace optimal solution cannot be applied. For such problems, it is more appropriate to employ nondeterministic metaheuristics.

Metaheuristics are iterative, population based, and stochastic approaches that do not guarantee finding the optimal solution, but they can obtain satisfying solution within acceptable time [4]. In metaheuristics implementations, the processes of exploitation and exploration conduct the search process. Exploitation directs search around the current best solutions, while the exploration randomly discovers new regions of a search domain.

During the last few decades, we witnessed development of nature-inspired sophisticated intelligent systems that can be used as optimization tools for many complex and hard problems. Metaheuristics that incorporate and simulate natural principles and rules are called nature-inspired metaheuristics.

Nature-inspired metaheuristics [5] can roughly be divided into two categories: evolutionary algorithms (EA) and swarm intelligence. The most prominent representative of EA is genetic algorithms (GA). GA can obtain good results for many kinds of optimization problems [6].

Social behavior of swarms of ants, bees, worms, birds, and fish was an inspiring source for the emerge of swarm intelligence [7]. Although swarm system consists of relatively unsophisticated individuals, they exhibit coordinated behavior that directs swarm towards the desired goal with no central component that manages the system as a whole.

Ant colony optimization (ACO) was founded on ant's ability to deploy a substance called pheromone for marking

the discovered path between the food source and ant's nests. It is one of the oldest members of swarm intelligence family [8] but it is constantly being improved and applied to different problems [9–11].

Artificial bee colony (ABC) algorithm mimics the behavior of honey bee swarm. In this paradigm, three types of artificial bees perform search. Each type of bees has its particular role in the search process. ABC was originally proposed by Karaboga for problems of continuous optimization [12]. This algorithm proves to be robust and capable of solving high dimensionality problems [13–15].

Cuckoo search (CS) is an iterative approach that models search process by employing Levy flights (series of straight flight paths with sudden 90 degrees turn). It was first proposed by Yang and Deb [16] and proved to be a robust optimization technique [17], obtaining satisfying results in real-life optimizations like image thresholding [18].

Also, swarm intelligence metaheuristics which mimic the human search process were developed. Seeker optimization algorithm (SOA) is established on human memory, reasoning, past experience, and social interactions. It has been proven that the SOA is a robust technique for solving global numerical and real-life optimization problems [19] and is continuously being improved [20].

As a result of the literature survey, it can be concluded that for portfolio optimization problem there are some genetic algorithm (GA) implementations. However, there are only few swarm intelligence algorithms adapted for this problem. There are papers which refer to solving portfolio problem with nondominating sorting genetic algorithm (NSGA) which was first proposed by Srinivas and Deb [21]. Newer version NSGA-II improves the convergence and the spread of solutions in the population [22]. Lin et al. presented NSGA-II based algorithm with integer encoding for solving MV portfolio model with minimum transaction lots (MTL), fixed transaction costs (TC), and linear constraints on capital investments. The optimization of MV portfolio problem with cardinality and holding weights constraints by GA is shown in [23]. Soleimani et al. [24] presented GA with RAR crossover operator for solving MV portfolio problem where cardinality constraints, MTL, and constraints on sector capitalization are taken into account.

As mentioned, only few swarm intelligence metaheuristics exist for portfolio optimization. Deng and Lin presented ant colony optimization (ACO) for solving the cardinality constraints Markowitz MV portfolio model [25]. Haqiqi and Kazemi [26] employed the same algorithm to MV portfolio model. We emphasize on one ACO implementation based on the average entropy for real estate portfolio optimization [27]. This is one of the rare papers that incorporates entropy in portfolio model. Cura investigated PSO approach to cardinality constrained MV portfolio optimization [1]. The test data set contains weekly prices from March 1992 to September 1997 from the following five indexes: Hang Seng in Hong Kong, DAX 100 in Germany, FTSE 100 in UK, S&P 100 in USA, and Nikkei in Japan. The results of this study are compared with those from genetic algorithm, simulated annealing, and tabu search approaches and showed that PSO has potential in portfolio optimization. Zhu et al. [28]

presented PSO algorithm for nonlinear constrained portfolio optimization with multiobjective functions. The model is tested on various restricted and unrestricted risky investment portfolios and a comparative study with GA is showed. PSO demonstrated high computational efficiency in constructing optimal risky portfolios and can be compared with other state-of-the-art algorithms.

ABC algorithm for mixed quadratic and integer programming problem of cardinality constrained MV portfolio model was presented by Wang et al. [29]. Some modifications of classical ABC algorithm for constrained optimization problems were adopted. The approach was tested on a standard benchmark data set and proved to be a robust portfolio optimizer.

One of the first implementations for portfolio optimization problem by the firefly algorithms was developed by Tuba et al. [30, 31]. Framework for solving this problem was devised. Metaheuristic was tested on standard five-asset data set. FA proved to be robust and effective technique for portfolio problem. Among other metaheuristics for portfolio problem, one approach based on neural networks (NN) should be distinguished [32].

In this paper, we propose a modified firefly algorithm (FA) for cardinality constrained mean-variance (CCMV) portfolio optimization with entropy constraint. FA was first introduced by Yang for unconstrained optimization [33]. Later, it was adapted [34] for solving various numerical and practical optimization problems [35–37].

We modified pure FA algorithm to adjust it for constrained problems and to improve its performance. We intensified exploration during early phase and eliminated it during late iterations when the proper part of the search space has been reached. Details will be given in Section 4.

The implementation of metaheuristics for the CCMV portfolio model with entropy constraint was not found in the literature survey. Thus, we conducted three experiments.

- (i) First, we compared original FA with our modified mFA applied to portfolio optimization problem. We wanted to see what is the real improvement of our approach.
- (ii) Then, we compared results of our algorithm for the CCMV portfolio model with and without entropy constraint. In this test, we analyzed the influence of entropy constraint to portfolio diversification.
- (iii) Finally, in the third test, we made comparative analysis between our modified mFA and other state-of-the-art metaheuristics. We compared our proposed algorithm to Cura’s PSO [1] and also to GA, TS, and SA, indirectly from [23].

The rest of the paper is organized as follows. Section 2 presents mathematical formulations of variety portfolio optimization models. The presentation of the original FA is given in Section 3. Our proposed modified FA approach for the CCMV portfolio problem with entropy constraint is discussed in Section 4. Section 5 first shows algorithm parameter settings that are used in experiments. Then, we present three experiments which we conducted along with

the comparative analyses with other metaheuristics. Finally, Section 6 gives conclusions and recommendations for future research.

2. Portfolio Optimization Problem Definitions

Portfolio optimization, as one of the most important issues in modern financial management, tackles the problem of distribution of financial resources across a number of assets to maximize return and control the risk.

When making financial decisions, investors follow the principle of diversification by investing their capital into different types of assets. By investment in portfolios, rather than in single assets, the risk is mitigated by diversification of the investments, without negative impact on expected returns.

The essential form of portfolio optimization is formulated as bicriterion optimization problem where the reward, which is measured by the mean of return, should be maximized, while the risk, measured by the variance of return, should be minimized [38]. This problem deals with the selection of the portfolio of securities that minimizes the risk subject to the constraints, while guaranteeing a given level of returns [39].

By literature research many methods for solving portfolio problem can be found. Markowitz’s standard mean-variance (MV) model chooses one important objective function that is subject to optimization, while the remaining objective functions are being threatened as constraints [40]. The key point in the MV formulation is to employ the expected returns of a portfolio as the investment return and the variance of returns of the portfolio as the investment risk [41]. Its basic assumptions are that the investors are rational with either multivariate normally distributed asset returns or in the case of arbitrary returns, a quadratic utility function [42]. This approach is widely adapted and plays an important role in the modern portfolio theory.

Markowitz’s MV model considers the selection of risky portfolio as objective function, and the mean return of an asset as one of the constraints [43]. This model can mathematically be defined as

$$\min \sigma_{R_p}^2 = \sigma_p^2 = \sum_{i=1}^N \sum_{j=1}^N \omega_i \omega_j \text{Cov}(\bar{R}_i, \bar{R}_j) \quad (5)$$

subject to

$$\bar{R}_p = E(R_p) = \sum_{i=1}^N \omega_i \bar{R}_i \geq R, \quad (6)$$

$$\sum_{i=1}^N \omega_i = 1, \quad \omega_i \geq 0, \quad \forall i \in \{1, 2, \dots, N\}, \quad (7)$$

where N is the total number of available assets, \bar{R}_i is the mean return on asset i , and $\text{Cov}(\bar{R}_i, \bar{R}_j)$ is covariance of returns of assets i and j , respectively. Constraint defined in (7) guarantees that the whole disposable capital is invested. Weight variable ω_i has a role of control parameter that

determines the proportion of the capital that is placed in asset i . Weight variable has a real value in the range $[0, 1]$.

In the presented MV formulation, the objective is to minimize the portfolio risk σ_p^2 , for a given value of portfolio expected return \bar{R}_p .

Efficient frontier model, which is often called single-objective function model, constructs only one evaluation function that models portfolio optimization problem. In this model, the desired mean return R is varying for the purpose of finding different objective function values. Risk aversion indicator $\lambda \in [0, 1]$ controls this process [28].

Efficient frontier definition is

$$\min \lambda \left[\sum_{i=1}^N \sum_{j=1}^N \omega_i \omega_j \text{Cov}(\bar{R}_i, \bar{R}_j) \right] - (1 - \lambda) \left[\sum_{i=1}^N \omega_i \bar{R}_i \right] \quad (8)$$

subject to

$$\sum_{i=1}^N \omega_i = 1 \quad (9)$$

$$\omega_i \geq 0, \quad \forall i \in \{1, 2, \dots, N\}.$$

In the presented formulation λ is critical parameter. It controls the relative importance of the mean return to the risk for the investor. When the value of λ is set to 0, mean return of portfolio is being maximized without considering the risk. Alternatively, when λ has a value of 1, risk of the portfolio is minimized regardless of the mean return. Thus, when the value of λ increases, the relative importance of the risk to the mean return for the investor rises, and vice versa.

Each λ value generates different objective function value which is composed of the mean return and the variance (risk). By tracing the mean return and variance intersections for different λ , a continuous curve can be drawn which is called an efficient frontier in the Markowitz's modern portfolio theory.

Another model worth mentioning is Sharpe ratio (SR) which uses the information from mean and variance of an asset [44]. In this model, the measure of mean return is adjusted with the risk and can be described by

$$SR = \frac{R_p - R_f}{\text{StdDev}(p)}, \quad (10)$$

where p denotes portfolio, R_p is the mean return of the portfolio p , and R_f is a test available rate of return on a risk-free asset. $\text{StdDev}(p)$ is a measure of the risk in portfolio (standard deviation of R_p). By adjusting the portfolio weights w_i , portfolio's Sharpe ratio can be maximized.

However, all three models: the MV, efficient frontier, and Sharpe ratio were constructed under strict and simplified assumptions that do not consider real-world parameters and limitations and as such are not always suitable for real applications. For these reason extended MV model is devised.

Extended MV formulation takes into account additional constraints such as budget, cardinality, transaction lots, and sector capitalization. These constraints model real-world

legal and economic environment where the financial investments are being done [45]. Budget constraint controls the minimum and maximum total budget proportion that can be invested in particular asset. Cardinality constraint controls whether a particular asset will be included in the portfolio. The minimum transaction lots constraint ensures that each security can only be purchased in a certain number of units. Sector capitalization constraint directs the investments towards the assets that belong to the sectors where higher value of market capitalization can be accomplished. The review of this constraint is given in [24].

When all the above-mentioned constraints are being applied to the basic portfolio problem formulation, the problem becomes a combinatorial optimization problem whose feasible region is not continuous. Thus, the extended MV model belongs to the group of quadratic mixed-integer programming models. Its formulation is

$$\min \sigma_{R_p}^2 = \sigma_p^2 = \sum_{i=1}^N \sum_{j=1}^N \omega_i \omega_j \text{Cov}(\bar{R}_i, \bar{R}_j), \quad (11)$$

where

$$\omega_i = \frac{x_i c_i z_i}{\sum_{j=1}^N x_j c_j z_j}, \quad i = 1, \dots, N, \quad (12)$$

$$\sum_{i=1}^N z_i = M \leq N, \quad M, N \in \mathcal{N}, \quad (13)$$

$$z_i \in \{0, 1\}, \quad i = 1, \dots, N$$

subject to

$$\sum_{i=1}^N x_i c_i z_i \bar{R}_i \geq BR \quad (14)$$

$$\sum_{i=1}^N x_i c_i z_i \leq B \quad (15)$$

$$0 \leq B_{\text{low}_i} \leq x_i c_i \leq B_{\text{up}_i} \leq B, \quad i = 1, \dots, N \quad (16)$$

$$\sum_{i_s} W_{i_s} \geq \sum_{i_{s'}} W_{i_{s'}} \quad (17)$$

$$\forall y_s y_{s'} \neq 0, \quad i_s, i_{s'} \in \{1, 2, \dots, N\}$$

$$s, s' \in \{1, 2, \dots, S\}, \quad s < s'$$

where

$$y_s = \begin{cases} 1 & \text{if } \sum_{i_s} z_i > 0 \\ 0 & \text{if } \sum_{i_s} z_i = 0, \end{cases} \quad (18)$$

In (11)–(18) M represents the number of selected assets among possible N assets. B is the total disposable budget, and B_{low_i} and B_{up_i} are lower and upper limits of the budget that can be invested in asset i , respectively. S represents the number of sectors in the market where the investment is being placed,

c_i is the size of transaction lot for asset i , and x_i denotes the number of such lots (of asset i) that is purchased.

Decision variable z_i is used to apply cardinality constraint: z_i is equal to 1 if an asset i is present in the portfolio, otherwise its value is 0. Equation (17) models sector capitalization constraint. Despite the fact that a certain sector has high capitalization, security from this sector that has low return and/or high risk must be excluded from the final portfolio's structure. To make such exclusion, variable y_s is defined and it has a value of 1 if the corresponding sector has at least one selected asset, and 0 otherwise. In (17) i_s is a set of assets which can be found in sector s . Sectors are sorted in descending order by their capitalization value.

Entropy was introduced by Jaynes [46] for wide application in optimization, crystallography in the beginning, networks [47], and so forth, but it also becomes an important tool in portfolio optimization and asset pricing. Entropy is widely recognized measure of portfolio diversification [2]. In multiobjective portfolio optimization models, the entropy can be used as an objective function. Here, we will address entropy as diversity constraint in portfolio models, because we employed it in portfolio model that is used for testing of our modified FA approach.

The entropy constraint defines lower bound L_E of entropy $E(P)$ of portfolio P according to the following equation [48]:

$$E(P) = -\sum_{i=1}^N x_i \ln x_i \geq L_E, \quad (19)$$

where N is the number of assets in portfolio P and x_i is weight variable of the security i .

In one extreme, when the weigh variable of only one asset in portfolio P is 1, and for the rest of the assets is 0, $E(P)$ reaches its minimum at $-1 * \ln 1 = 0$ [49]. This is the least diverse scenario. Contrarily, in the most diverse condition that, for $\forall i, x_i = 1/N, E(P)$ obtains its maximum in $-N(1/N \ln 1/N) = \ln N$. According to this, L_E is in the range $[0, \ln N]$. Greater value of entropy denotes better portfolio's diversity, and L_E is used to make sure that the diversity of P is not too low.

Entropy constraint equation (19) can be transformed into the upper-bound constraint [49]:

$$F(P) = e^{-E(P)} = e^{\sum_{i=1}^N x_i \ln x_i} \leq U_E. \quad (20)$$

As shown previously, $0 \leq E(P) \leq \ln N$, which implicates that $0 \geq -E(P) \geq -\ln N$. Then, the condition $e^0 = 1 \geq e^{-E(P)} = F(P) \geq e^{-\ln N} = 1/N$ holds. Thus, the range of upper-bound constraint U_E is $[1/N, 1]$.

In this paper, for testing purposes, we used model which employs some of the constraints that can be found in the extended MV formulation. In the experimental study, we implemented modified FA for optimizing cardinality constrained mean-variance model (CCMV) which is derived from the standard Markowitz's and the efficiency frontier models.

We were inspired by Usta's and Kantar's multiobjective approach based on a mean-variance-skewness-entropy portfolio selection model (MVSEM) that employs Shannon's

entropy measure to the mean-variance-skewness portfolio model (MVSM) to generate a well-diversified portfolio [50, 51]. Thus, we added entropy measure to the CCMV portfolio formulation to generate well-diversified portfolio.

Formulation of the CCMV model with entropy constraint is

$$\min \lambda \left[\sum_{i=1}^N \sum_{j=1}^N x_i x_j \sigma_{i,j} \right] - (1 - \lambda) \left[\sum_{i=1}^N x_i \mu_i \right] \quad (21)$$

subject to

$$\sum_{i=1}^N x_i = 1, \quad (22)$$

$$\sum_{i=1}^N z_i = K, \quad (23)$$

$$\epsilon_i z_i \leq x_i \leq \delta_i z_i, \quad z \in \{0, 1\}, \quad i = 1, 2, 3, \dots, N, \quad (24)$$

$$-\sum_{i=1}^N z_i x_i \ln x_i \geq L_E. \quad (25)$$

As already mentioned in this section, N is the number of potential securities that will be included in portfolio, λ is risk aversion parameter, x_i and x_j are weight variables of assets i and j , respectively, $\delta_{i,j}$ is their covariance, and μ_i is i th asset's return. K is the desired number of assets that will be included in the portfolio. Decision variable z_i controls whether the asset i will be included in portfolio. If its value is 1, asset i is included, and if the value is 0, asset i is excluded from the portfolio. ϵ and δ are lower and upper bounds of the asset that is included in portfolio and they make sure that the asset's proportion in the portfolio is within the predefined range.

We applied entropy constraint equation (25) with lower bounds, as in (19), to ensure that the diversity of portfolio is not too low. L_E is lower bound of the entropy in the range $[0, \ln K]$. In (25), z_i ensures that only assets that are included in portfolio are taken into account.

From the CCMV formulation with entropy constraint it can be seen that this problem belongs to the group of mixed quadratic and integer programming problems. It employs both real and integer variables with equity and inequity constraints.

3. Presentation of the Original FA

Firefly algorithm (FA) was originally proposed by Yang in 2008 [33], with later improvements [52]. It was applied to continuous [53], discrete [54], and mixed [55] optimization problems. The emergence of this metaheuristic was inspired by the social and flashing behavior of fireflies.

Fireflies inhabit moderate and tropical climate environments all around the world. Their flashing behavior has many different roles. Synchronized flashing by males is unique in the animal world and involves a capacity for visually coordinated, rhythmically coincident, and interindividual behavior. Also, flashing is used to alleviate communication for mating

and to frighten the predators. These flashing properties can be incorporated into swarm intelligence metaheuristic in such a way that they are associated with the objective function which is subject to optimization.

With respect to the facts that the real firefly system is sophisticated and that the metaheuristics are approximations of real systems, three idealized rules are applied with the goal to enable algorithm's implementation [33]: (1) all fireflies are unisex, so the attractions between fireflies do not depend on their sex; (2) attractiveness of a firefly is directly proportional to their brightness, and the less brighter firefly will move towards the brighter one. Brightness increases as the distance between fireflies decreases; (3) the brightness of a firefly is determined by the value of objective function. For minimization problems, brightness increases as the objective function value decreases. There are also other forms of brightness which can be defined similar to fitness function in genetic algorithms (GA) [6].

In the implementation of FA, one of the most important issues that should be considered is the formulation of attractiveness. For the sake of simplicity, a good approximation is that the attractiveness of a firefly is determined by its brightness which depends on the encoded objective function.

In the case of maximization problems, the brightness of a firefly at a particular location x can be chosen as $I(x) \sim f(x)$, where $I(x)$ is the attractiveness and $f(x)$ is the value of objective function at this location. Otherwise, if the goal is to minimize function, the following expression can be used:

$$I(x) = \begin{cases} \frac{1}{f(x)}, & \text{if } f(x) > 0 \\ 1 + |f(x)|, & \text{otherwise.} \end{cases} \quad (26)$$

The variations of light intensity and attractiveness are monotonically decreasing functions because as the light intensity and the attractiveness decrease, the distance from the source increases, and vice versa. This can be formulated as [56]

$$I(r) = \frac{I_0}{1 + \gamma r^2}, \quad (27)$$

where $I(r)$ is the light intensity, r is distance, and I_0 is the light intensity at the source. Besides that, the air also absorbs part of the light, and the light becomes weaker. Air absorption is modeled by the light absorption coefficient γ .

In most FA implementations that can be found in the literature survey, the combined effect of both the inverse square law and absorption can be approximated using the following Gaussian form:

$$I(r) = I_0 e^{-\gamma r^2}. \quad (28)$$

Attractiveness β of a firefly is relative because it depends on the distance between the firefly and the beholder. Thus, it varies with the distance $r_{i,j}$ between fireflies i and j . The attractiveness is direct proportional to fireflies light intensity (brightness), as shown in the following:

$$\beta(r) = \beta_0 e^{-\gamma r^2}, \quad (29)$$

where β_0 is the attractiveness at $r = 0$. Equation (29) determines a characteristic distance $\Gamma = 1/\sqrt{\gamma}$ over which the attractiveness changes significantly from β_0 to $\beta_0 e^{-1}$.

But, in practical applications, the above expression is usually replaced with

$$\beta(r) = \frac{\beta_0}{1 + \gamma r^2}. \quad (30)$$

Main reason for this replacement is that the calculation of exponential function in (29) demands much more computational power than simple division in (30).

The movement of a firefly i (its new position in iteration $t + 1$) towards the brighter, and thus more attractive firefly j is calculated using

$$x_i(t+1) = x_i(t) + \beta_0 r^{-\gamma r_{ij}^2} (x_j - x_i) + \alpha(\kappa - 0.5), \quad (31)$$

where β_0 is attractiveness at $r = 0$, α is randomization parameter, κ is random number drawn from uniform or Gaussian distribution, and $r_{i,j}$ is distance between fireflies i and j . The positions of fireflies are updated sequentially by comparing and updating each pair of them at every iteration.

The distance between fireflies i and j is calculated using Cartesian distance form [56]:

$$r_{i,j} = \|x_i - x_j\| = \sqrt{\sum_{k=1}^D (x_{i,k} - x_{j,k})^2}, \quad (32)$$

where D is the number of problem parameters. For most problems, $\beta_0 = 0$ and $\alpha \in [0, 1]$ are adequate settings.

The parameter γ has crucial impact on the convergence speed of the algorithm. This parameter shows the variation of attractiveness and in theory it has a value of $[0, +\infty)$, but in practice it is determined by the characteristic distance Γ of the system that is being optimized. In most implementations γ parameter varies between 0.01 and 100.

There are two special cases of the FA, and they are both associated with the value of γ as follows [33]:

- (i) if $\gamma = 0$, then $\beta = \beta_0$. That means that the air around firefly is completely clear. In this case, β is always the largest it could possibly be, and fireflies advance towards other fireflies with the largest possible steps. The exploration-exploitation is out of balance because the exploitation is maximal, while the exploration is minimal;
- (ii) if $\gamma = \infty$, then $\beta = 0$. In this case, there is a thick fog around fireflies and they could not see each other. The movement is performed in a random steps, and exploration is more intensive with practically no exploitation at all.

The pseudocode for the original FA is given as Algorithm 1.

In the presented pseudocode, SN is total number of fireflies in the population, IN is total number of algorithm's iterations, and t is the current iteration.

```

Generate initial population of fireflies  $x_i$ , ( $i = 1, 2, 3, SN$ )
Light intensity  $I_i$  at point  $x_i$  is defined by  $f(x)$ 
Define light absorption coefficient  $\gamma$ 
Define number of iterations  $IN$ 
while  $t < IN$  do
  for  $i = 1$  to  $SN$  do
    for  $j = 1$  to  $i$  do
      if  $I_j < I_i$  then
        Move firefly  $j$  towards firefly  $i$  in  $d$  dimension
        Attractiveness varies with distance  $r$  via  $\exp[-\gamma r]$ 
        Evaluate new solution, replace the worst with
        better solution and update light intensity
      end if
    end for
  end for
  Rank all fireflies and find the current best
end while
    
```

ALGORITHM 1: Original firefly algorithm.

4. Proposed mFA for the CCMV Portfolio Problem with Entropy Constraint

As mentioned in Section 1, we propose a modified firefly algorithm for cardinality constrained mean-variance portfolio optimization with entropy constraint.

By analyzing FA we noticed that, as most other swarm intelligence algorithms, the pure version of the algorithm, developed for unconstrained problems, exhibits some deficiencies when applied to constrained problems. In the early cycles of algorithm's execution established balance between exploitation and exploration is not completely appropriate for this class of problems. During early phase exploration is not intensive enough. However, during late cycles when FA was able to discover the right part of the search space, the exploration is no longer needed. To control whether the exploration will be triggered or not, we introduced exploration breakpoint *EBP* control parameter.

In this section, we show implementation details of our modified FA algorithm which we named mFA.

4.1. Initialization Phase and Fitness Calculation. At the initialization step, FA generates random population of SN fireflies (artificial agents) using

$$x_{i,j} = lb_j + \text{rand}(0, 1) * (ub_j - lb_j), \tag{33}$$

where $x_{i,j}$ is the weight of the j th portfolio's asset of the i th agent, $\text{rand}(0, 1)$ is random number uniformly distributed between 0 and 1, and ub_j and lb_j are upper and lower weight bounds of the j th asset, respectively.

If the initially generated value for the j th parameter of the i th firefly does not fit in the scope $[lb_j, ub_j]$, it is being modified using the following expression:

$$\begin{aligned} \text{if } (x_{i,j}) > ub_j, \quad \text{then } x_{i,j} &= ub_j \\ \text{if } (x_{i,j}) < lb_j, \quad \text{then } x_{i,j} &= lb_j. \end{aligned} \tag{34}$$

Moreover, in the initialization phase, decision variables $z_{i,j}$ ($i = 1, \dots, SN, j = 1, \dots, N$) are also initialized for each firefly agent i . N is the number of potential assets in portfolio. According to this, each firefly is modeled using $2 * N$ dimensions. z_i is a binary vector, with values 1, when an asset is included in portfolio, and 0, when it is excluded from it.

Decision variables are generated randomly by applying

$$z_{i,j} = \begin{cases} 1, & \text{if } \phi < 0.5 \\ 0 & \text{if } \phi \geq 0.5, \end{cases} \tag{35}$$

where ϕ is random real number between 0 and 1.

To guarantee the feasibility of solutions, we used similar arrangement algorithm as proposed in [1]. The arrangement algorithm is applied first time in the initialization phase.

In the arrangement algorithm, i is the current solution that consists of Q the distinct set of K_i^* assets in the i th solution, $z_{i,j}$ is the decision variable of asset j , and $x_{i,j}$ is the weight proportion for asset j . Arrangement algorithm pseudocode is shown as Algorithm 2.

For the constraint $\sum_{i=1}^N x_i = 1$ we set $\psi = \sum_{j \in Q} x_{i,j}$ and put $x_{i,j} = x_{i,j} / \psi$ for all assets that satisfy $j \in Q$. The same approach for satisfying this constraint was used in [1]. To make sure that each asset's proportion is within predefined lower and upper bounds, ϵ and δ , respectively, we used *if* $x_{i,j} > \delta_{i,j}$ *then* $x_{i,j} = \delta_{i,j}$ *and if* $x_{i,j} < \epsilon_{i,j}$ *then* $x_{i,j} = \epsilon_{i,j}$.

We did not apply c -value based approach for adding and removing assets from the portfolio as in [1]. According to our experiments, using c -value does not improve FA performance. It only increases computational complexity.

In modified FA, the fitness is employed to model the attractiveness of the fireflies. Attractiveness is directly proportional to the fitness.

After generating SN number of agents, fitness value is calculated for each firefly in the population. Fitness (brightness) is calculated as in the original FA implementation (26).

```

while  $K_i^* < K$  do
  select random asset  $j$  such that  $j \notin Q$ 
   $z_{i,j} = 1$ ,  $Q = Q \cup [j]$ ,  $K_i^* = K_i^* + 1$ 
end while
while  $K_i^* > K$  do
  select random asset  $j$  such that  $j \in Q$ 
   $z_{i,j} = 1$ ,  $Q = Q - [j]$ ,  $K_i^* = K_i^* - 1$ 
end while
while true do
   $\theta = \sum_{j \in Q} x_{i,j}$ ,  $x_{i,j} = \frac{x_{i,j}}{\psi}$ ,  $\eta = \sum_{j \in Q} \max(0, x_{i,j} - \delta_i)$ ,  $\phi = \sum_{j \in Q} \max(0, \eta_j - x_{i,j})$ 
  if  $\eta = 0$  and  $\phi = 0$  then
    exit algorithm
  end if
  for  $j = 1$  to  $N$  do
    if  $z_{i,j} = 1$  then
      if  $x_{i,j} > \delta_j$  then
         $x_{i,j} = \delta_j$ 
      end if
      if  $x_{i,j} < \varepsilon_j$  then
         $x_{i,j} = \varepsilon_j$ 
      end if
    end if
  end for
end while

```

ALGORITHM 2: Arrangement algorithm.

In the initialization phase, for each firefly in the population, constraint violation CV is being calculated. CV is a measure of how much the agents violate constraints in the problem definition:

$$CV_i = \sum_{g_j(x_i) > 0} g_j(x_i) + \sum_{j=q+1}^m h_j(x_i). \quad (36)$$

CV calculation is necessary, because it is later used for performing selection based on Deb's method [57, 58].

4.2. Firefly Movement. The movement of a firefly i towards the firefly that has a higher fitness j is calculated as in the original FA implementation [56]:

$$x_i(t+1) = x_i(t) + \beta_0 r^{-\gamma r_{i,j}^2} (x_j - x_i) + \alpha(\kappa - 0.5), \quad (37)$$

where $x_i(t+1)$ is new solution generated in iteration $(t+1)$, β_0 is attractiveness at $r = 0$, α is randomization parameter, κ is random number drawn from uniform or Gaussian distribution, and $r_{i,j}$ is distance between fireflies i and j .

Also, when moving a firefly, new decision variables are calculated:

$$z_{i,k}^{t+1} = \text{round} \left(\frac{1}{1 + e^{-z_{i,k}^t + \phi_{i,j}(z_{i,k}^t - z_{j,k}^t)}} - 0.06 \right), \quad (38)$$

where $z_{i,k}^{t+1}$ is decision variable for the k th asset of the new solution, $z_{i,k}$ is a decision variable of the k th parameter of the

old solution, and $z_{j,k}$ is decision variable of k th parameter of the brighter firefly j .

It should be noticed that the decision variables in the employed bee phase are generated differently than in the initialization phase equation (35).

After the new i th solution is generated in exploitation process using (37) and (38) the winner between new $x_i(t+1)$ and old $x_i(t)$ solution is retained using the selection process based on Deb's rules.

4.3. Exploration. As mentioned before, we noticed insufficient exploration power in the original FA implementation, particularly in early iterations of algorithm's execution. In this phase of algorithm's execution, exploitation-exploration balance is not well established for this type of problems. This balance was also discussed in [14]. Thus, we adopted mechanism similar to scout bee with *limit* parameter from the ABC metaheuristic.

We introduced parameter abandonment threshold (AT) that represents the allowed predetermined number of unsuccessful tries to improve particular solution. When a potential solution (firefly) is stagnating (not being improved) for AT iterations, it is replaced by a new, random one using (33), (34), and (35). Hence, fireflies that exploited exhausted solutions are transformed into scouts that perform the exploration process. The value of AT is empirically determined and will be shown in the experimental section.

Also, during late iterations, with the assumption that the right part of the search space has been found, the

```

Generate initial population of fireflies  $x_i$  and  $z_i$  ( $i = 1, 2, 3, \dots, SN$ ) by using (33) and (35)
Apply arrangement algorithm
Light intensity  $I_i$  at point  $x_i$  is defined by  $f(x)$ 
Define light absorption coefficient  $\gamma$ 
Define number of iterations  $IN$ 
Calculate fitness and CV for all fireflies using (26) and (36)
Set initial values for  $\alpha$ 
Set  $t$  value to 0
while  $t < IN$  do
    for  $i = 1$  to  $SN$  do
        for  $j = 1$  to  $i$  do
            if  $I_i < I_j$  then
                Move firefly  $i$  towards firefly  $j$  in  $d$  dimension using (37) and (38)
                Attractiveness varies with distance  $r$  via  $\exp[-\gamma r]$ 
                Evaluate new solution, replace worse with better solution
                using Deb's method and update light intensity
                if solution  $i$  is not improved and  $t_i < EBP$  then
                     $UIC_i$  increment by 1
                else
                     $UIC_i$  set to 0
                end if
            end if
        end for
    end for
    if  $t < EBP$  then
        replace all agents whose  $UIC > AT$  with random agents using (33)
    end if
    Apply arrangement algorithm
    Rank all fireflies and find the current best
    Recalculate values for  $\alpha$  using (39)
end while
    
```

ALGORITHM 3: Modified firefly algorithm.

intensive exploration is not needed any more. In that case, the exploration is not being triggered. For this purpose, we introduce new control parameter, exploration breakpoint (*EBP*) which controls whether the exploration will be triggered. The discussion of this parameter is also given in experimental section.

Also, we should note that the parameter α for FA search process is being gradually decreased from its initial value according to

$$\alpha(t) = \left(1 - \left(1 - \left(\left(\frac{10^{-4}}{9} \right)^{1/IN} \right) \right) \right) * \alpha(t - 1), \quad (39)$$

where t is the current iteration and IN is the maximum number of iterations.

Pseudocode of mFA is given as Algorithm 3. Some implementation's details are omitted for the sake of simplicity.

In the pseudocode, SN is total number of fireflies in the population, IN is total number of algorithm's iterations, and t is the current iteration. As explained, AT is the maximum number of unsuccessful attempts to improve particular solution after which it will be considered exhausted and replaced by a new, random solution.

5. Algorithm Settings and Experimental Results

In this section, we first present parameter settings which were adjusted for testing purposes of our proposed mFA. Then, we show experimental results, discussion, and comparative analysis with other state-of-the-art algorithms.

5.1. *Parameter Settings.* To test the performance and robustness of our modified FA, we set algorithm parameters similar to [1]. Number of firefly agents in the population SN is calculated by employing the following expression:

$$SN = 20\sqrt{N}, \quad (40)$$

where N is the number of potential assets in portfolio.

The value of maximum number of iterations IN in one algorithm's run is set according to

$$IN = \frac{1000 * N}{SN}. \quad (41)$$

As mentioned in Section 4, to improve the exploration power of the original FA, we introduced parameter AT with corresponding counters UIC_i ($i = 1, 2, \dots, SN$) that count how many times a particular firefly agent unsuccessfully tried

```

λ = 0
while λ ≤ 1 do
  SN = 20√N
  Set portfolio problem parameters K, ε and δ
  InitializationPhase()
  ArrangementAlgorithm()
  FitnessCalculation()
  Set UIC to 0 and calculate AT value according to (42)
  Set initial values for v and α
  IN =  $\frac{1000N}{SN}$ 
  for t = 1 to IN do
    Firefly movement
    Apply Selection between old and new solution using Deb rules
    Exploration phase (if necessary)
    ArrangementAlgorithm()
    Rank all fireflies and find the current best
    Recalculate values for v and α
    t ++
  end for
  λ = λ + Δλ
end while

```

ALGORITHM 4: Modified firefly with parameters.

improvement. When the value of UIC_i reaches predetermined abandonment threshold AT , corresponding agent is being replaced by a random agent. AT is determined by the values of SN and IN , like in [14]:

$$AT = \frac{IN}{SN} = \frac{(1000 * N) / SN}{20\sqrt{N}}. \quad (42)$$

Exploration breakpoint EBP controls whether or not the exploration will be triggered. According to our experimental tests, modified FA generates worse results if the exploration is triggered during late iterations. In most of the runs, the algorithm is able to find a proper part of the search space during early cycles, and exploration during late cycles is not useful. To the contrary, it just relaxes the exploitation. EBP is empirically set to $IN/2$.

FA search process parameter α is set to start at 0.5, but it is being gradually decreased from its initial value according to (39).

The promising approaches for handling equality constraints include dynamic, self-adaptive tolerance adjustment [59]. When this tolerance is included, the exploration is enhanced by exploring a larger space.

In modified FA implementation, besides the adoption of arrangement algorithm we used (4) and violation limit v for handling constraints. Good experimental results are obtained by starting with a relatively large v value, which is gradually decreasing through the algorithm's iterations. It is very important to chose the right value for v . If the chosen value is too small, the algorithm may not find feasible solutions, and otherwise the results may be far from the feasible region [14].

We used the following dynamic settings for the v :

$$v(t+1) = \frac{v(t)}{dec}, \quad (43)$$

where t is the current generation and dec is a value slightly larger than 1. For handling equality constraints, we set initial value for v to 1.0, dec to 1.001 and the threshold for v to 0.0001 like in [3].

For generating heuristics efficient frontier, we used $\xi = 51$ different λ values. Thus, we set $\Delta\lambda$ to 0.02 because λ in the first algorithm's run is 0 and in the last is 1.

We also set number of assets that will be included in portfolio K to 10, lower asset's weight ε to 0.01, and upper asset's weight δ to 1.

Since the entropy lower bound depends on the number of assets that will be included in portfolio, we set L_E in the range of $[0, \ln 10]$.

We present again short modified FA pseudocode as Algorithm 4, but this time with the emphasis on the parameter settings.

For making better distinction between parameters, we divided algorithm parameters into four groups: modified FA global control parameters, FA search parameters, portfolio parameters, and constraint-handling parameters.

Parameters are summarized in Table 1.

5.2. Experimental Results and Comparative Analysis. In this subsection, we show the results obtained when searching the general efficient frontier that provides the solution for the problem formulated in (21)–(25). The test data were downloaded from <http://people.brunel.ac.uk/~mastjib/jeb/orlib/portinfo.html>.

TABLE 1: Parameters.

Parameter	Value
Modified FA global control parameters	
Number of fireflies-solutions (SN)	Depends on N
Number of iterations (IN)	Depends on SN
Abandonment threshold (AT)	Depends on SN and IN
Exploration breakpoint (EBP)	Depends on IN
FA search parameters	
Initial value for randomization parameter α	0.5
Attractiveness at $r = 0$, β_0	0.2
Absorption coefficient γ	1.0
Portfolio parameters	
Number of potential securities (N)	Depends on the problem
Number of assets in portfolio (K)	10
Initial value of risk aversion (λ)	0
Different λ values (ξ)	51
Lower asset's weight (ϵ)	0.01
Upper asset's weight (δ)	1.0
Lower bound of entropy (L_E)	$[0, \ln K]$
Constraint-handling parameters	
Initial violation tolerance (v)	1.0
Decrement (dec)	1.002

Benchmark data refers to the weekly stock prices from March 1992 to September 1997 for the indexes: the Hong Kong Hang Seng with 31 assets, the German Dax 100 with 85 assets, the British FTSE 100 with 89 assets, the US S&P 100 with 98 assets, and the Japanese Nikkei with 225 assets.

We adapted test data and stored it in Excel spreadsheets. For all indexes, we used the following data: mean return, standard deviation of return for each asset, and correlation for each possible pair of assets. Also, for generating standard efficiency frontier, we used mean return and variance of return for each security.

Since SN , MCN , and AT parameters depend on the problem size N (number of securities in the test), we show exact values used for all indexes (tests) in Table 2. Formula results are rounded to the closest integer values.

Lower bound for entropy for all benchmarks set is in the range between 0 and $\ln 10$, because K is set to 10 for all test cases.

We conducted tests on Intel Core™ i7-4770 K processor @4 GHz with 16 GB of RAM memory, Windows 7 x64 Ultimate 64 operating system and Visual Studio 2012 with NET 4.5 Framework.

TABLE 2: Benchmark specific parameters.

Parameter	Value
Hang Seng index with 31 assets	
Number of fireflies-solutions (SN)	111
Number of iterations (IN)	279
Abandonment threshold (AT)	3
Exploration breakpoint (EBP)	140
DAX 100 index with 85 assets	
Number of fireflies-solutions (SN)	185
Number of iterations (IN)	459
Abandonment threshold (AT)	3
Exploration breakpoint (EBP)	230
FTSE 100 index with 89 assets	
Number of fireflies-solutions (SN)	189
Number of iterations (IN)	479
Abandonment threshold (AT)	3
Exploration breakpoint (EBP)	240
S&P 100 index with 98 assets	
Number of fireflies-solutions (SN)	198
Number of iterations (IN)	494
Abandonment threshold (AT)	3
Exploration breakpoint (EBP)	247
Nikkei index with 225 assets	
Number of fireflies-solutions (SN)	300
Number of iterations (IN)	750
Abandonment threshold (AT)	3
Exploration breakpoint (EBP)	375

When sets of Pareto optimal portfolios obtained with modified FA are taken, heuristic efficient frontier can be traced. In this paper, we compare the standard efficient frontiers for five real-world benchmark sets mentioned above with the heuristic efficient frontier for the same data set. For comparison of standard and heuristic efficiency frontier, we use mean Euclidean distance, variance of return error, and mean return error as in [1]. We also give the execution time of modified FA for each benchmark on our computer system platform.

For calculation purposes of mean Euclidean distance, let the pair $(v_i^s, r_i^s) = (i = 1, 2, 3, \dots, 2000)$ denote the variance and mean return of the point in the standard efficient frontier, and the pair $(v_j^h, r_j^h) = (j = 1, 2, 3, \dots, \xi)$ represents the variance and mean return of the point in the heuristic efficient frontier. Then, the index i_j of the closest standard efficiency frontier point to the heuristic efficiency frontier point, denoted as $(v_{i,j}^s, r_{i,j}^s)$, is calculated using Euclidean distance by

$$i_j = \arg \min_{i=1,2,\dots,2000} \left(\sqrt{(v_i^s - v_j^h)^2 + (r_i^s - r_j^h)^2} \right) \quad (44)$$

$$j = 1, 2, 3, \dots, \xi.$$

Using (44), mean Euclidean distance is defined as

$$\frac{\sum_{j=1}^{\xi} \sqrt{(v_{i,j}^s - v_j^h)^2 + (r_{i,j}^s - r_j^h)^2}}{\xi} \quad (45)$$

In addition to mean Euclidean distance, we used two other measures to test modified FA, variance of return error and mean return error.

Variance of return error is defined as

$$\left(\sum_{j=1}^{\xi} \frac{|v_{i,j}^s - v_j^h|}{v_j^h} \right) \frac{1}{\xi} \quad (46)$$

Mean return error is calculated as

$$\left(\sum_{j=1}^{\xi} \frac{|r_{i,j}^s - r_j^h|}{r_j^h} \right) \frac{1}{\xi} \quad (47)$$

For testing purposes, we conducted three experiments. In the first experiment, we compared mFA with the original FA for CCMV problem with entropy constraint. Second experiment refers to comparative analysis between mFA for CCMV problem with and without entropy constraint. Finally, in the third experiment, we perform comparative analysis between our modified mFA and other state-of-the-art metaheuristics. We compared our proposed algorithm to Cura's PSO [1] and also to GA, TS, and SA, indirectly from [23].

We first wanted to analyze how our mFA compares to the original FA when optimizing CCMV portfolio model with entropy constraint. Thus, we also implemented original FA for this purpose. We compared mean Euclidean distance, variance of return error, and mean return error. These performance indicators were described above. We also calculated computational time for both algorithms. This time is comparable since the same computer platform was used for testing both original FA and mFA. This comparison is shown in Table 3. For better distinction between indicator values, we marked better results in bold.

As can be seen from Table 3, mFA obtains better results for almost all benchmarks. Only for variance of return error and mean return error indicators for *FTSE100* index test, original FA managed to achieve better values. For this benchmark, exploration in early iterations is unnecessary because the algorithm quickly converges to the right part of the search space, and the firefly agents are being wasted on exploration.

All three indicators, mean Euclidean distance, variance of return error, and mean return error, are significantly better for mFA tests for *HangSeng*, *DAX100*, *S&P100*, and *Nikkei* indexes. Since mFA utilizes exploration at early cycles, computation time for all tests is worse (higher) than for the original FA implementation.

In the second experiment, we compared our mFA for CCMV problem with and without entropy constraint to show how the entropy constraint influences the results. CCMV formulation without entropy constraint is defined in (21)–(24).

According to the results presented in Table 4, it is clear that the entropy constraint slightly effects the portfolio's performance. In the CCMV optimization with entropy constraint, for *HangSeng* and *S&P* tests, mean Euclidean distance is slightly better, so the portfolio is better diversified. For other three tests, the results obtained for this indicator are the same. Also, for *HangSeng*, *DAX100*, *FTSE100*, and *S&P100* indexes, optimization of the model with entropy gains better variance of return error and mean return error values. Only for *Nikkei* tests, those indicators have better value for CCMV model optimization without entropy constraint. Since the algorithm takes extra time to calculate the entropy constraint, execution time for CCMV with entropy is higher for all tests except *HangSeng* because this benchmark incorporates less securities than the other benchmarks.

The implementation of metaheuristics for CCMV portfolio model with entropy constrained could not be found in the literature. Thus, in the third experiment, we compared our mFA approach with metaheuristics for CCMV portfolio formulation which did not employ entropy. This model is defined by (21)–(24). We note that this test is not objective indicator of mFA's effectiveness compared to the other algorithms.

We compared mFA with tabu search (TS), genetic algorithm (GA), simulated annealing (SA), from [23], and PSO from [1] for the same set of benchmark data. As in the first two experiments, for performance indicators, we used mean Euclidean distance, variance of return error, and mean return error. Parameter settings for our mFA are given in Tables 1 and 2 and are comparable to parameters for other four compared algorithms that can be found in [1, 23]. We also give computational time for mFA, but those results are incomparable with results for other metaheuristics because we used different computer platform and portfolio model. In experiments in [1], Pentium M 2.13 GHz computer with 1 GB RAM was used. In the results table, best obtained results of all five heuristics are printed bold.

Other metaheuristic implementations for CCMV portfolio problem, such as modified ABC [29] and hybrid ABC (HABC) [41] that have similar performance, can be found in the literature.

If we consider that the optimization of CCMV with entropy constraint obtains only slightly better results than optimization of CCMV model without entropy in Table 4, the experimental results in Table 5 could be used for comparison of the performance of mFA with other metaheuristics in some sense.

The experimental results presented in Table 5 prove that none of the four algorithms which we used for comparisons has distinct advantages but that on average, mFA is better approach than other four metaheuristics.

mFA obtains better (smaller) mean Euclidean distance for all five benchmark sets. In *HangSeng* and *FTSE100* benchmarks, mFA is better than all four algorithms for all three indicators, mean Euclidean distance, variance of return error, and mean return error. For those benchmarks, mFA was able to approximate the standard efficient frontier with the smallest mean return and variance of return error, and under the same risk values.

TABLE 3: Experimental results of FA and mFA for CCMV model.

Index	N	Performance indicators	FA	mFA
Hang Seng	31	Mean Euclidean distance	0.0006	0.0003
		Variance of return error (%)	1.7092	1.2387
		Mean return error (%)	0.7172	0.4715
		Execution time	18	20
DAX 100	85	Mean Euclidean distance	0.0032	0.0009
		Variance of return error (%)	7.3892	7.2569
		Mean return error (%)	1.4052	1.3786
		Execution time	67	71
FTSE 100	89	Mean Euclidean distance	0.0005	0.0004
		Variance of return error (%)	2.6391	2.7085
		Mean return error (%)	0.3025	0.3121
		Execution time	81	94
S&P 100	98	Mean Euclidean distance	0.0011	0.0003
		Variance of return error (%)	3.9829	3.6026
		Mean return error (%)	1.0025	0.8993
		Execution time	129	148
Nikkei	225	Mean Euclidean distance	0.0001	0.0000
		Variance of return error (%)	1.7834	1.2015
		Mean return error (%)	0.7283	0.4892
		Execution time	335	367

TABLE 4: Experimental results of mFA for CCMV model with and without entropy constraint.

Index	N	Performance indicators	mFA for CCMV	mFA for CCMV with entropy
Hang Seng	31	Mean Euclidean distance	0.0004	0.0003
		Variance of return error (%)	1.2452	1.2387
		Mean return error (%)	0.4897	0.4715
		Execution time	20	20
DAX 100	85	Mean Euclidean distance	0.0009	0.0009
		Variance of return error (%)	7.2708	7.2569
		Mean return error (%)	1.3801	1.3786
		Execution time	70	71
FTSE 100	89	Mean Euclidean distance	0.0004	0.0004
		Variance of return error (%)	2.7236	2.7085
		Mean return error (%)	0.3126	0.3121
		Execution time	92	94
S&P 100	98	Mean Euclidean distance	0.0004	0.0003
		Variance of return error (%)	3.6135	3.6026
		Mean return error (%)	0.8997	0.8993
		Execution time	146	148
Nikkei	225	Mean Euclidean distance	0.0000	0.0000
		Variance of return error (%)	1.1927	1.2015
		Mean return error (%)	0.464	0.4892
		Execution time	360	367

Second best algorithm shown in Table 5 is GA which obtains best mean return error and variance of return error in DAX100 and S&P100 tests, respectively. TS shows best performance for mean return error indicator in S&P100 benchmark, SA for mean return error in *Nikkei* test, while

PSO proves to be most robust for variance of return error in DAX100 index.

From the presented analysis it can be concluded that our approach obtained results for CCMV portfolio optimization problem that can be more valuable for the investors: mFA's

TABLE 5: Experimental results for five metaheuristics.

Index	N	Performance indicators	GA	TS	SA	PSO	mFA
Hang Seng	31	Mean Euclidean distance	0.0040	0.0040	0.0040	0.0049	0.0003
		Variance of return error (%)	1.6441	1.6578	1.6628	2.2421	1.2387
		Mean return error (%)	0.6072	0.6107	0.6238	0.7427	0.4715
		Execution time	18	9	10	34	20
DAX 100	85	Mean Euclidean distance	0.0076	0.0082	0.0078	0.0090	0.0009
		Variance of return error (%)	7.2180	9.0309	8.5485	6.8588	7.2569
		Mean return error (%)	1.2791	1.9078	1.2817	1.5885	1.3786
		Execution time	99	42	52	179	71
FTSE 100	89	Mean Euclidean distance	0.0020	0.0021	0.0021	0.0022	0.0004
		Variance of return error (%)	2.8660	4.0123	3.8205	3.0596	2.7085
		Mean return error (%)	0.3277	0.3298	0.3304	0.3640	0.3121
		Execution time	106	42	55	190	94
S&P 100	98	Mean Euclidean distance	0.0041	0.0041	0.0041	0.0052	0.0003
		Variance of return error (%)	3.4802	5.7139	5.4247	3.9136	3.6026
		Mean return error (%)	1.2258	0.7125	0.8416	1.4040	0.8993
		Execution time	126	51	66	214	148
Nikkei	225	Mean Euclidean distance	0.0093	0.0010	0.0010	0.0019	0.0000
		Variance of return error (%)	1.2056	1.2431	1.2017	2.4274	1.2015
		Mean return error (%)	5.3266	0.4207	0.4126	0.7997	0.4892
		Execution time	742	234	286	919	367

results are more accurate and the generated investment strategy is able to more efficiently diversify the risk of the portfolio.

6. Conclusions

In this paper we presented modified firefly algorithm (mFA) for cardinality constrained mean-variance portfolio optimization problem with entropy constraint. We adopted from the ABC algorithm *limit* parameter that controls and directs the exploration process. Original firefly algorithm suffers from low exploration power at early iterations of algorithm's execution for this type of problems. By introducing exploration into this phase of execution, we overcome this deficiency. However, during late cycles when the right part of the search space was reached, the exploration is no longer needed. To control whether the exploration will be triggered or not, we introduced exploration breakpoint *EBP* control parameter.

Since swarm intelligence implementations for the CCMV portfolio model with entropy constraint could not be found in the literature, we conducted three experiments. In the first experiment, to measure the enhancement gained by our modifications, we compared our proposed mFA with the original FA for CCMV model. Test results show that our modifications completely rectified original FA deficiencies. To show how the entropy constraint affects the CCMV portfolio model, in the second experiment we compared results of the mFA for CCMV models with and without entropy constraints. Test results proved that inclusion of the entropy constraint is justified since it ensures portfolio diversification and, consequently, quality of results enhancement. Finally,

to test the performance and robustness of our algorithm, we compared it with four other state-of-the-art algorithms from [1] (and indirectly [23]). Our proposed algorithm proved almost uniformly better compared to genetic algorithm, tabu search, simulated annealing, and particle swarm optimization. This all establishes modified firefly algorithm as a usable tool for cardinality constrained mean-variance portfolio optimization problem with entropy constraint.

Future research may include application of the proposed mFA to other portfolio optimization models and formulations with different constraints. Also, additional modifications of the FA algorithm can be investigated for possible further improvement of results.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This research was supported by the Ministry of Science of Republic of Serbia, Grant no. III-44006.

References

- [1] T. Cura, "Particle swarm optimization approach to portfolio optimization," *Nonlinear Analysis: Real World Applications*, vol. 10, no. 4, pp. 2396–2406, 2009.
- [2] R. Zhou, R. Cai, and G. Tong, "Applications of entropy in finance: a review," *Entropy*, vol. 15, no. 11, pp. 4909–4931, 2013.

- [3] E. Mezura-Montes, Ed., *Constraint-Handling in Evolutionary Optimization*, vol. 198 of *Studies in Computational Intelligence*, Springer, 2009.
- [4] T. El-Ghazali, *Metaheuristics: From Design To Implementation*, Wiley, 2009.
- [5] X.-S. Yang, *Nature-Inspired Optimization Algorithms*, Elsevier, 2014.
- [6] K. Tang, J. Yang, H. Chen, and S. Gao, "Improved genetic algorithm for nonlinear programming problems," *Journal of Systems Engineering and Electronics*, vol. 22, no. 3, pp. 540–546, 2011.
- [7] X.-S. Yang, "Swarm intelligence based algorithms: a critical analysis," *Evolutionary Intelligence*, vol. 7, no. 1, pp. 175–184, 2014.
- [8] M. Dorigo and L. M. Gambardella, "Ant colonies for the travelling salesman problem," *Biosystems*, vol. 43, no. 2, pp. 73–81, 1997.
- [9] R. Jovanovic and M. Tuba, "An ant colony optimization algorithm with improved pheromone correction strategy for the minimum weight vertex cover problem," *Applied Soft Computing Journal*, vol. 11, no. 8, pp. 5360–5366, 2011.
- [10] R. Jovanovic and M. Tuba, "Ant colony optimization algorithm with pheromone correction strategy for the minimum connected dominating set problem," *Computer Science and Information Systems*, vol. 10, no. 1, pp. 133–149, 2013.
- [11] M. Tuba and R. Jovanovic, "Improved ACO algorithm with pheromone correction strategy for the traveling salesman problem," *International Journal of Computers, Communications & Control*, vol. 8, no. 3, pp. 477–485, 2013.
- [12] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Tech. Rep., 2005.
- [13] I. Brajevic and M. Tuba, "An upgraded artificial bee colony (ABC) algorithm for constrained optimization problems," *Journal of Intelligent Manufacturing*, vol. 24, no. 4, pp. 729–740, 2013.
- [14] N. Bacanin and M. Tuba, "Artificial bee colony (ABC) algorithm for constrained optimization improved with genetic operators," *Studies in Informatics and Control*, vol. 21, no. 2, pp. 137–146, 2012.
- [15] M. Subotic and M. Tuba, "Parallelized multiple swarm artificial bee colony algorithm (MS-ABC) for global optimization," *Studies in Informatics and Control*, vol. 23, no. 1, pp. 117–126, 2014.
- [16] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, December 2009.
- [17] X.-S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343, 2010.
- [18] I. Brajevic and M. Tuba, "Cuckoo search and firefly algorithm applied to multilevel image thresholding," in *Cuckoo Search and Firey Algorithm: Theory and Applications*, X.-S. Yang, Ed., vol. 516 of *Studies in Computational Intelligence*, pp. 115–139, Springer, 2014.
- [19] C. Dai, W. Chen, Y. Song, and Y. Zhu, "Seeker optimization algorithm: a novel stochastic search algorithm for global numerical optimization," *Journal of Systems Engineering and Electronics*, vol. 21, no. 2, pp. 300–311, 2010.
- [20] M. Tuba, I. Brajevic, and R. Jovanovic, "Hybrid seeker optimization algorithm for global optimization," *Applied Mathematics & Information Sciences*, vol. 7, no. 3, pp. 867–875, 2013.
- [21] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary Computation*, vol. 2, no. 3, pp. 221–248, 1994.
- [22] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.
- [23] T.-J. Chang, N. Meade, J. E. Beasley, and Y. M. Sharaiha, "Heuristics for cardinality constrained portfolio optimisation," *Computers and Operations Research*, vol. 27, no. 13, pp. 1271–1302, 2000.
- [24] H. Soleimani, H. R. Golmakani, and M. H. Salimi, "Markowitz-based portfolio selection with minimum transaction lots, cardinality constraints and regarding sector capitalization using genetic algorithm," *Expert Systems with Applications*, vol. 36, no. 3, pp. 5058–5063, 2009.
- [25] G.-F. Deng and W.-T. Lin, "Ant colony optimization for Markowitz mean-variance portfolio model," *Swarm, Evolutionary, and Memetic Computing*, vol. 6466, pp. 238–245, 2010.
- [26] K. F. Haqiqi and T. Kazemi, "Ant colony optimization approach to portfolio optimization—a lingo companion," *International Journal of Trade, Economics and Finance*, vol. 3, no. 2, pp. 148–153, 2012.
- [27] Y. Li, B. Heng, S. Zhou, R. Chen, and S. Liu, "A novel ACO algorithm based on average entropy for real estate portfolio optimization," *Journal of Theoretical and Applied Information Technology*, vol. 45, no. 2, pp. 502–507, 2012.
- [28] H. Zhu, Y. Wang, K. Wang, and Y. Chen, "Particle Swarm Optimization (PSO) for the constrained portfolio optimization problem," *Expert Systems with Applications*, vol. 38, no. 8, pp. 10161–10169, 2011.
- [29] Z. Wang, S. Liu, and X. Kong, "Artificial bee colony algorithm for portfolio optimization problems," *International Journal of Advancements in Computing Technology*, vol. 4, no. 4, pp. 8–16, 2012.
- [30] M. Tuba, N. Bacanin, and B. Pelevic, "Artificial bee colony algorithm for portfolio optimization problems," *International Journal of Mathematical Models and Methods in Applied Sciences*, vol. 7, no. 10, pp. 888–896, 2013.
- [31] M. Tuba and N. Bacanin, "Artificial bee colony algorithm hybridized with firefly metaheuristic for cardinality constrained mean-variance portfolio problem," *Applied Mathematics & Information Sciences*, vol. 8, no. 6, pp. 2809–2822, 2014.
- [32] A. Fernández and S. Gómez, "Portfolio selection using neural networks," *Computers and Operations Research*, vol. 34, no. 4, pp. 1177–1191, 2007.
- [33] X.-S. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, 2008.
- [34] S. Yu, S. Yang, and S. Su, "Self-adaptive step firefly algorithm," *Journal of Applied Mathematics*, vol. 2013, Article ID 832718, 8 pages, 2013.
- [35] X.-S. Yang, "Multiobjective firefly algorithm for continuous optimization," *Engineering With Computers*, vol. 29, no. 2, pp. 175–184, 2013.
- [36] A. Galvez and A. Iglesias, "Firefly algorithm for explicit B-spline curve fitting to data point," *Mathematical Problems in Engineering*, vol. 2013, Article ID 528215, 12 pages, 2013.
- [37] R. K. Moghadas, A. Garakani, and M. Kalantarzadeh, "Optimum design of geometrically nonlinear double-layer domes by firefly metaheuristic algorithm," *Journal of Applied Mathematics*, vol. 2013, Article ID 169823, 13 pages, 2013.
- [38] K. P. Anagnostopoulos and G. Mamanis, "Multiobjective evolutionary algorithms for complex portfolio optimization problems," *Computational Management Science*, vol. 8, no. 3, pp. 259–279, 2011.

- [39] L. di Gaspero, G. di Tollo, A. Roli, and A. Schaerf, "Hybrid metaheuristics for constrained portfolio selection problems," *Quantitative Finance*, vol. 11, no. 10, pp. 1473–1487, 2011.
- [40] H. Markowitz, "Portfolio selection," *The Journal of Finance*, vol. 7, no. 1, pp. 77–91, 1952.
- [41] Z. Wang, R. Ouyang, and X. Kong, "A hybrid artificial bee colony for portfolio optimization," *Journal of Theoretical and Applied Information Technology*, vol. 49, no. 1, pp. 94–100, 2013.
- [42] K. P. Anagnostopoulos and G. Mamanis, "Multiobjective evolutionary algorithms for complex portfolio optimization problems," *Computational Management Science*, vol. 8, no. 3, pp. 259–279, 2011.
- [43] H. R. Golmakani and M. Fazel, "Constrained portfolio selection using particle swarm optimization," *Expert Systems with Applications*, vol. 38, no. 7, pp. 8327–8335, 2011.
- [44] W. F. Sharpe, "Mutual fund performance," *The Journal of Business*, vol. 39, no. 1, pp. 119–138, 1966.
- [45] M. Corazza and D. Favaretto, "On the existence of solutions to the quadratic mixed-integer mean-variance portfolio selection problem," *European Journal of Operational Research*, vol. 176, no. 3, pp. 1947–1960, 2007.
- [46] E. T. Jaynes, "Information theory and statistical mechanics," *Physical Review*, vol. 106, no. 4, pp. 620–630, 1957.
- [47] M. Tuba, "Asymptotic behavior of the maximum entropy routing in computer networks," *Entropy*, vol. 15, no. 1, pp. 361–371, 2013.
- [48] X. Huang, "An entropy method for diversified fuzzy portfolio selection," *International Journal of Fuzzy Systems*, vol. 14, no. 1, pp. 160–165, 2012.
- [49] J.-L. Lin, "On the diversity constraints for portfolio optimization," *Entropy*, vol. 15, no. 11, pp. 4607–4621, 2013.
- [50] I. Usta and Y. M. Kantar, "Mean-variance-skewness-entropy measures: a multi-objective approach for portfolio selection," *Entropy*, vol. 13, no. 1, pp. 117–133, 2011.
- [51] J. Xu, X. Zhou, and D. D. Wu, "Portfolio selection using λ mean and hybrid entropy," *Annals of Operations Research*, vol. 185, no. 1, pp. 213–229, 2011.
- [52] X.-S. Yang, "Firey algorithm, stochastic test functions and design optimisation," *International Journal of Bio-Inspired Computation*, vol. 2, no. 2, pp. 78–84, 2010.
- [53] X.-S. Yang, "Multiobjective firefly algorithm for continuous optimization," *Engineering With Computers*, vol. 29, no. 2, pp. 175–184, 2013.
- [54] M. K. Sayadia, A. Hafezalkotob, and S. G. J. Nainia, "Firefly-inspired algorithm for discrete optimization problems: an application to manufacturing cell formation," *Journal of Manufacturing Systems*, vol. 32, no. 1, pp. 78–84, 2013.
- [55] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Mixed variable structural optimization using firefly Algorithm," *Computers and Structures*, vol. 89, no. 23–24, pp. 2325–2336, 2011.
- [56] X.-S. Yang, "Firefly algorithms for multimodal optimization," *Stochastic Algorithms: Foundations and Applications*, vol. 5792, pp. 169–178, 2009.
- [57] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2–4, pp. 311–338, 2000.
- [58] K. Deb, *Optimization For Engineering Design: Algorithms and Examples*, PHI, 2012.
- [59] E. Mezura-Montes and C. A. Coello, "Constraint-handling in nature-inspired numerical optimization: past, present and future," *Swarm and Evolutionary Computation*, vol. 1, no. 4, pp. 173–194, 2011.

Research Article

Cuckoo Search with Lévy Flights for Weighted Bayesian Energy Functional Optimization in Global-Support Curve Data Fitting

Akemi Gálvez,¹ Andrés Iglesias,^{1,2} and Luis Cabellos³

¹ Department of Applied Mathematics and Computational Sciences, E.T.S.I. Caminos, Canales y Puertos, University of Cantabria, Avenida de los Castros s/n, 39005 Santander, Spain

² Department of Information Science, Faculty of Sciences, Toho University, 2-2-1 Miyama, Funabashi 274-8510, Japan

³ Institute of Physics of Cantabria (IFCA), Avenida de los Castros s/n, 39005 Santander, Spain

Correspondence should be addressed to Andrés Iglesias; iglesias@unican.es

Received 25 April 2014; Accepted 5 May 2014; Published 28 May 2014

Academic Editor: Xin-She Yang

Copyright © 2014 Akemi Gálvez et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The problem of data fitting is very important in many theoretical and applied fields. In this paper, we consider the problem of optimizing a weighted Bayesian energy functional for data fitting by using global-support approximating curves. By global-support curves we mean curves expressed as a linear combination of basis functions whose support is the whole domain of the problem, as opposed to other common approaches in CAD/CAM and computer graphics driven by piecewise functions (such as B-splines and NURBS) that provide local control of the shape of the curve. Our method applies a powerful nature-inspired metaheuristic algorithm called *cuckoo search*, introduced recently to solve optimization problems. A major advantage of this method is its simplicity: cuckoo search requires only two parameters, many fewer than other metaheuristic approaches, so the parameter tuning becomes a very simple task. The paper shows that this new approach can be successfully used to solve our optimization problem. To check the performance of our approach, it has been applied to five illustrative examples of different types, including open and closed 2D and 3D curves that exhibit challenging features, such as cusps and self-intersections. Our results show that the method performs pretty well, being able to solve our minimization problem in an astonishingly straightforward way.

1. Introduction

The problem of data fitting is very important in many theoretical and applied fields [1–4]. For instance, in computer design and manufacturing (CAD/CAM), data points are usually obtained from real measurements of an existing geometric entity, as it typically happens in the construction of car bodies, ship hulls, airplane fuselage, and other free-form objects [5–15]. This problem also appears in the shoes industry, archeology (reconstruction of archeological assets), medicine (computed tomography), computer graphics and animation, and many other fields. In all these cases, the primary goal is to convert the real data from a physical object into a fully usable digital model, a process commonly called *reverse engineering*. This allows significant savings in terms of storage capacity and processing and manufacturing time. Furthermore, the digital models are easier and cheaper to

modify than their real counterparts and are usually available anytime and anywhere.

Depending on the nature of these data points, two different approaches can be employed: *interpolation* and *approximation*. In the former, a parametric curve or surface is constrained to pass through all input data points. This approach is typically employed for sets of data points that come from smooth shapes and that are sufficiently accurate. On the contrary, approximation does not require the fitting curve or surface to pass through all input data points, but just close to them, according to some prescribed distance criteria. The approximation scheme is particularly well suited for the cases of highly irregular sampling and when data points are not exact, but subjected to measurement errors. In real-world problems the data points are usually acquired through laser scanning and other digitizing devices and are, therefore, subjected to some measurement noise, irregular

sampling, and other artifacts [12, 13]. Consequently, a good fitting of data should be generally based on approximation schemes rather than on interpolation [16–20].

There are two key components for a good approximation of data points with curves: a proper choice of the approximating function and a suitable parameter tuning. Due to their good mathematical properties regarding evaluation, continuity, and differentiability (among many others), the use of polynomial functions (especially splines) is a classical choice for the approximation function [16, 17, 23–27]. In general, the approximating curves can be classified as global-support and local-support. By global-support curves we mean curves expressed as a linear combination of basis functions whose support is the whole domain of the problem. As a consequence, these curves exhibit a global control, in the sense that any modification of the shape of the curve in a particular location is propagated throughout the whole curve. This is in clear contrast to the local-support approaches that have become prevalent in CAD/CAM and computer graphics, usually driven by piecewise functions (such as B-splines and NURBS) that provide local control of the shape of the curve [23, 28]. In this work we are particularly interested to explore the performance of the global-support approach by using different global-support basis functions for our approximating curves.

1.1. Main Contributions and Structure of the Paper. In this paper, we consider the problem of optimizing a weighted Bayesian energy functional for data fitting by using global-support approximating curves. In particular, our goal is to obtain the global-support approximating curve that fits the data points better while keeping the number of free parameters of the model as low as possible. To this aim, we formulate this problem as a minimization problem by using a weighted Bayesian energy functional for global-support curves. This is one of the major contributions of this paper. Our functional is comprised of two competing terms aimed at minimizing the fitting error between the original and the reconstructed data points while simultaneously minimizing the degrees of freedom of the problem. Furthermore, the functional can be modified and extended to include various additional constraints, such as the fairness and smoothness constraints typically required in many industrial operations in computer-aided manufacturing, such as CNC (computer numerically controlled) milling, drilling, and machining [4, 5, 12].

Unfortunately, our formulation in previous paragraph leads to a nonlinear continuous optimization problem that cannot be properly addressed by conventional mathematical optimization techniques. To overcome this limitation, in this paper we apply a powerful nature-inspired metaheuristic algorithm called cuckoo search, introduced in 2009 by Yang and Deb to solve optimization problems [21]. The algorithm is inspired by the obligate interspecific brood parasitism of some cuckoo species that lay their eggs in the nests of host birds of other species. Since its inception, the cuckoo search (specially its variant that uses Lévy flights) has been successfully applied in several papers reported recently in

the literature to difficult optimization problems from different domains. However, to the best of our knowledge, the method has never been used so far in the context of geometric modeling and data fitting. This is also one of the major contributions of this paper.

A critical problem when using metaheuristic approaches concerns the parameter tuning, which is well known to be time-consuming and problem-dependent. In this regard, a major advantage of the cuckoo search with Lévy flights is its simplicity: it only requires two parameters, many fewer than other metaheuristic approaches, so the parameter tuning becomes a very simple task. The paper shows that this new approach can be successfully applied to solve our optimization problem. To check the performance of our approach, it has been applied to five illustrative examples of different types, including open and closed 2D and 3D curves that exhibit challenging features, such as cusps and self-intersections. Our results show that the method performs pretty well, being able to solve our minimization problem in an astonishingly straightforward way.

The structure of this paper is as follows: in Section 2 some previous work in the field is briefly reported. Then, Section 3 introduces the basic concepts and definitions along with the description of the problem to be solved. The fundamentals and main features of the cuckoo search algorithm are discussed in Section 4. The proposed method for the optimization of our weighted Bayesian energy functional for data fitting with global-support curves is explained in Section 5. Some other issues such as the parameter tuning and some implementation details are also reported in that section. As the reader will see, the method requires a minimal number of control parameters. As a consequence, it is very simple to understand, easy to implement and can be applied to a broad variety of global-support basis functions. To check the performance of our approach, it has been applied to five illustrative examples for the cases of open and closed 2D and 3D curves exhibiting challenging features, such as cusps and self-intersections, as described in Section 6. The paper closes with the main conclusions of this contribution and our plans for future work in the field.

2. Previous Works

The problem of curve data fitting has been the subject of research for many years. First approaches in the field were mostly based on numerical procedures [1, 29, 30]. More recent approaches in this line use error bounds [31], curvature-based squared distance minimization [26], or dominant points [18]. A very interesting approach to this problem consists in exploiting minimization of the energy of the curve [32–36]. This leads to different functionals expressing the conditions of the problem, such as fairness, smoothness, and mixed conditions [37–40]. Generally, research in this area is based on the use of nonlinear optimization techniques that minimize an energy functional (often based on the variation of curvature and other high-order geometric constraints). Then, the problem is formulated as a multivariate nonlinear optimization problem in which

the desired form will be the one that satisfies various geometric constraints while minimizing (or maximizing) a measure of form quality. A variation of this formulation consists in optimizing an energy functional while simultaneously minimizing the number of free parameters of the problem and satisfying some additional constraints on the underlying model function. This is the approach we follow in this paper.

Unfortunately, the optimization problems given by those energy functionals and their constraints are very difficult and cannot be generally solved by conventional mathematical optimization techniques. On the other hand, some interesting research carried out during the last two decades has shown that the application of artificial intelligence techniques can achieve remarkable results regarding such optimization problems [6, 8, 10, 11, 14]. Most of these methods rely on some kind of neural networks, such as standard neural networks [8] and Kohonen’s SOM (self-organizing maps) nets [10]. In some other cases, the neural network approach is combined with partial differential equations [41] or other approaches [42]. The generalization of these methods to functional networks is also analyzed in [6, 11, 14]. Other approaches are based on the application of nature-inspired metaheuristic techniques, which have been intensively applied to solve difficult optimization problems that cannot be tackled through traditional optimization algorithms. Examples include artificial immune systems [43], bacterial foraging [44], honey bee algorithm [45], artificial bee colony [46], firefly algorithm [47, 48], and bat algorithm [49, 50]. A previous paper in [51] describes the application of genetic algorithms and functional networks yielding pretty good results. Genetic algorithms have also been applied to this problem in both the discrete version [52, 53] and the continuous version [7, 54]. Other metaheuristic approaches applied to this problem include the use of the popular particle swarm optimization technique [9, 24], artificial immune systems [55, 56], firefly algorithm [57, 58], estimation of distribution algorithms [59], memetic algorithms [60], and hybrid techniques [61].

3. Mathematical Preliminaries

In this paper we assume that the solution to our fitting problem is given by a model function $\Phi(\xi)$ defined on a finite interval domain $[\nu_1, \nu_2]$. Note that in this paper vectors are denoted in bold. We also assume that $\Phi(\xi)$ can be mathematically represented as a linear combination of the so-called blending functions:

$$\Phi(\xi) = \sum_{\alpha=1}^{\delta} \Theta_{\alpha} \psi_{\alpha}(\xi), \quad \xi \in [\nu_1, \nu_2]. \quad (1)$$

In this work, the family of blending functions $\{\psi_{\alpha}(\xi)\}_{\alpha}$ in (1) is assumed to be linearly independent and to form a basis of the vector space of functions of degree $\leq \delta - 1$ on $[\nu_1, \nu_2]$. In this paper we consider the case in which all functions $\{\psi_{\alpha}(\xi)\}_{\alpha}$ have their support on the whole domain $[\nu_1, \nu_2]$. Without loss of generality, we can also assume that $[\nu_1, \nu_2]$ is the unit interval $[0, 1]$. In practical terms, this means that the blending functions provide a global control of the shape of the approximating curve (these functions are usually referred

to as global-support functions), as opposed to the alternative case of local control given by the piecewise representation that is characteristic of popular curves such as B-splines and NURBS. Typical examples of global-support basis functions are

- (1) the canonical polynomial basis: $\psi_{\alpha}(\xi) = \xi^{\alpha-1}$;
- (2) the Bernstein basis: $\psi_{\alpha}(\xi) = \binom{\delta-1}{\alpha-1} \xi^{\alpha-1} (1-\xi)^{\delta-\alpha}$.

Other examples include the Hermite polynomial basis, the trigonometric basis, the hyperbolic basis, the radial basis, and the polyharmonic basis.

Let us suppose now that we are given a finite set of data points $\{\Delta_{\beta}\}_{\beta=1, \dots, \zeta}$ in a D -dimensional space (usually $D = 2$ or $D = 3$). Our goal is to obtain a global-support approximating curve that best fits these data points while keeping the number of degrees of freedom as low as possible. This leads to a difficult minimization problem involving two different (and competing) factors: the fitting error at the data points and the number of free parameters of the model function. In this paper, we consider the RMSE (root mean square error) as the fitting error criterion. The number of free parameters is computed by following a Bayesian approach (see [62] for further details). This is a very effective procedure to penalize fitting models with too many parameters, thus preventing data overfitting [63]. Therefore, our optimization problem consists in minimizing the following weighted Bayesian energy functional:

$$\mathcal{L} = \frac{\zeta}{2} \log \left(\sum_{\beta=1}^{\zeta} \Omega_{\beta} \left[\Delta_{\beta} - \sum_{\alpha=1}^{\delta} \Theta_{\alpha} \psi_{\alpha}(\rho_{\beta}) \right]^2 \right) + \frac{\zeta \cdot \gamma}{2} \left(\frac{2\delta - 1}{2} \right) \log(\zeta), \quad (2)$$

where we need a parameter value ρ_{β} to be associated with each data point Δ_{β} . Equation (2) is comprised of two terms: the first one computes the fitting error to the data points, while the second one plays the role of a penalty term in order to reduce the degrees of freedom of the model. The penalty term also includes a real positive multiplicative factor γ used to modulate how much this term will affect the whole energy functional.

This functional \mathcal{L} can be modified or expanded to include any additional constrain in our model. For instance, it is very common in many engineering domains such as computer-aided ship-hull design, car-body styling, and turbine-blade design to request conditions such as fairness or smoothness. In our approach, these conditions can readily be imposed by adding different energy functionals adapted to the particular needs. Suppose that instead of reducing the degrees of freedom of our problem, the smoothness of the fitting curve is required. This condition is simply

incorporated to our model by replacing the penalty term in (2) by the strain energy functional as follows:

$$\mathcal{L} = \frac{\zeta}{2} \log \left(\sum_{\beta=1}^{\zeta} \Omega_{\beta} \left[\Delta_{\beta} - \sum_{\alpha=1}^{\delta} \Theta_{\alpha} \psi_{\alpha}(\rho_{\beta}) \right]^2 \right) + \frac{\zeta \cdot \gamma}{2} \left(\lambda \int \|\Phi''(\xi)\|^2 d\xi \right). \quad (3)$$

Considering the vectors $\Xi_{\alpha} = (\psi_{\alpha}(\rho_1), \dots, \psi_{\alpha}(\rho_{\zeta}))^T$, with $\alpha = 1, \dots, \delta$, where $(\cdot)^T$ means transposition, $\Xi = (\Xi_1, \dots, \Xi_{\delta})$, $\Delta = (\Delta_1, \dots, \Delta_{\zeta})$, $\Omega = (\Omega_1, \dots, \Omega_{\zeta})$, and $\Theta = (\Theta_1, \dots, \Theta_{\delta})^T$, (2) can be written in matricial form as

$$\mathcal{L} = \Omega \cdot \Delta^T \cdot \Delta - \Omega \cdot \Theta^T \cdot \Xi^T \cdot \Delta - \Omega \cdot \Delta^T \cdot \Xi \cdot \Theta + \Omega \cdot \Theta^T \cdot \Xi^T \cdot \Xi \cdot \Theta. \quad (4)$$

Minimization of \mathcal{L} requires differentiating (4) with respect to Θ and equating to zero to satisfy the first-order conditions, leading to the following system of equations (called the *normal equations*):

$$\Xi^T \cdot \Xi \cdot \Theta = \Xi^T \cdot \Delta. \quad (5)$$

In general, the blending functions $\{\psi_{\alpha}(\xi)\}_{\alpha}$ are nonlinear in ξ , leading to a strongly nonlinear optimization problem, with a high number of unknowns for large sets of data points, a case that happens very often in practice. Our strategy for solving the problem consists in applying the cuckoo search method to determine suitable parameter values for the minimization of functional \mathcal{L} according to (2). The process is performed iteratively for a given number of iterations. Such a number is another parameter of the method that has to be calculated in order to run the algorithm until the convergence of the minimization of the error is achieved.

4. The Cuckoo Search Algorithm

Cuckoo search (CS) is a nature-inspired population-based metaheuristic algorithm originally proposed by Yang and Deb in 2009 to solve optimization problems [21]. The algorithm is inspired by the obligate interspecific brood parasitism of some cuckoo species that lay their eggs in the nests of host birds of other species with the aim of escaping from the parental investment in raising their offspring. This strategy is also useful to minimize the risk of egg loss to other species, as the cuckoos can distribute their eggs amongst a number of different nests. Of course, sometimes it happens that the host birds discover the alien eggs in their nests. In such cases, the host bird can take different responsive actions varying from throwing such eggs away to simply leaving the nest and build a new one elsewhere. However, the brood parasites have at their turn developed sophisticated strategies (such as shorter egg incubation periods, rapid nestling growth, and egg coloration or pattern mimicking their hosts) to ensure that the host birds will care for the nestlings of their parasites.

This interesting and surprising breeding behavioral pattern is the metaphor of the cuckoo search metaheuristic approach for solving optimization problems. In the cuckoo search algorithm, the eggs in the nest are interpreted as a pool of candidate solutions of an optimization problem, while the cuckoo egg represents a new coming solution. The ultimate goal of the method is to use these new (and potentially better) solutions associated with the parasitic cuckoo eggs to replace the current solution associated with the eggs in the nest. This replacement, carried out iteratively, will eventually lead to a very good solution of the problem.

In addition to this representation scheme, the CS algorithm is also based on three idealized rules [21, 22].

- (1) Each cuckoo lays one egg at a time and dumps it in a randomly chosen nest.
- (2) The best nests with high quality of eggs (solutions) will be carried over to the next generations.
- (3) The number of available host nests is fixed, and a host can discover an alien egg with a probability $p_a \in [0, 1]$. In this case, the host bird can either throw the egg away or abandon the nest so as to build a completely new nest in a new location.

For simplicity, the third assumption can be approximated by a fraction p_a of the n nests being replaced by new nests (with new random solutions at new locations). For a maximization problem, the quality or fitness of a solution can simply be proportional to the objective function. However, other (more sophisticated) expressions for the fitness function can also be defined.

Based on these three rules, the basic steps of the CS algorithm can be summarized as shown in the pseudocode reported in Algorithm 1. Basically, the CS algorithm starts with an initial population of n host nests and it is performed iteratively. In the original proposal, the initial values of the j th component of the i th nest are determined by the expression $x_i^j(0) = \text{rand} \cdot (\text{up}_i^j - \text{low}_i^j) + \text{low}_i^j$, where up_i^j and low_i^j represent the upper and lower bounds of that j th component, respectively, and rand represents a standard uniform random number on the open interval $(0, 1)$. Note that this choice ensures that the initial values of the variables are within the search space domain. These boundary conditions are also controlled in each iteration step.

For each iteration g , a cuckoo egg i is selected randomly and new solutions $\mathbf{x}_i(g+1)$ are generated by using the Lévy flight, a kind of random walk in which the steps are defined in terms of the step lengths, which have a certain probability distribution, with the directions of the steps being isotropic and random. According to the original creators of the method, the strategy of using Lévy flights is preferred over other simple random walks because it leads to better overall performance of the CS. The general equation for the Lévy flight is given by

$$\mathbf{x}_i(g+1) = \mathbf{x}_i(g) + \alpha \oplus \text{levy}(\lambda), \quad (6)$$

where g indicates the number of the current generation and $\alpha > 0$ indicates the step size, which should be related to

```

begin
  Objective function  $f(\mathbf{x})$ ,  $\mathbf{x} = (x_1, \dots, x_D)^T$ 
  Generate initial population of  $n$  host nests  $\mathbf{x}_i$  ( $i = 1, 2, \dots, n$ )
  While ( $t < \text{MaxGeneration}$ ) or (stop criterion)
    Get a cuckoo (say,  $i$ ) randomly by Lévy flights
    Evaluate its fitness  $F_i$ 
    Choose a nest among  $n$  (say,  $j$ ) randomly
    if ( $F_i > F_j$ )
      Replace  $j$  by the new solution
    end
    A fraction ( $p_a$ ) of worse nests are abandoned and new ones are built via Lévy flights
    Keep the best solutions (or nests with quality solutions)
    Rank the solutions and find the current best
  end while
  Postprocess results and visualization
end

```

ALGORITHM 1: Cuckoo search algorithm via Lévy flights as originally proposed by Yang and Deb in [21, 22].

the scale of the particular problem under study. The symbol \oplus is used in (6) to indicate the entrywise multiplication. Note that (6) is essentially a Markov chain, since next location at generation $g + 1$ only depends on the current location at generation g and a transition probability, given by the first and second terms of (6), respectively. This transition probability is modulated by the Lévy distribution as

$$\text{levy}(\lambda) \sim g^{-\lambda}, \quad (1 < \lambda \leq 3), \quad (7)$$

which has an infinite variance with an infinite mean. Here the steps essentially form a random walk process with a power-law step-length distribution with a heavy tail. From the computational standpoint, the generation of random numbers with Lévy flights is comprised of two steps: firstly, a random direction according to a uniform distribution is chosen; then, the generation of steps following the chosen Lévy distribution is carried out. The authors suggested using the so-called Mantegna’s algorithm for symmetric distributions, where “symmetric” means that both positive and negative steps are considered (see [64] for details). Their approach computes the factor

$$\hat{\phi} = \left(\frac{\Gamma(1 + \hat{\beta}) \cdot \sin((\pi \cdot \hat{\beta})/2)}{\Gamma(((1 + \hat{\beta})/2) \cdot \hat{\beta} \cdot 2^{(\hat{\beta}-1)/2})} \right)^{1/\hat{\beta}}, \quad (8)$$

where Γ denotes the Gamma function and $\hat{\beta} = 3/2$ in the original implementation by Yang and Deb [22]. This factor is used in Mantegna’s algorithm to compute the step length s as

$$\varsigma = \frac{u}{|v|^{1/\hat{\beta}}}, \quad (9)$$

where u and v follow the normal distribution of zero mean and deviation σ_u^2 and σ_v^2 , respectively, where σ_u obeys the Lévy distribution given by (8) and $\sigma_v = 1$. Then, the stepsize η is computed as

$$\eta = 0.01\varsigma(\mathbf{x} - \mathbf{x}_{\text{best}}), \quad (10)$$

where ς is computed according to (9). Finally, \mathbf{x} is modified as $\mathbf{x} \leftarrow \mathbf{x} + \eta \cdot \Upsilon$, where Υ is a random vector of the dimension of the solution \mathbf{x} and that follows the normal distribution $N(0, 1)$.

The CS method then evaluates the fitness of the new solution and compares it with the current one. In case the new solution brings better fitness, it replaces the current one. On the other hand, a fraction of the worse nests (according to the fitness) are abandoned and replaced by new solutions so as to increase the exploration of the search space looking for more promising solutions. The rate of replacement is given by the probability p_a , a parameter of the model that has to be tuned for better performance. Moreover, for each iteration step, all current solutions are ranked according to their fitness and the best solution reached so far is stored as the vector \mathbf{x}_{best} (used, e.g., in (10)).

This algorithm is applied in an iterative fashion until a stopping criterion is met. Common terminating criteria are that a solution is found that satisfies a lower threshold value, that a fixed number of generations have been reached, or that successive iterations no longer produce better results.

5. The Method

We have applied the cuckoo search algorithm discussed in previous section to our optimization problem described in Section 3. The problem consists in minimizing the weighted Bayesian energy functional given by (2) for a given family of global-support blending functions. To this aim, we firstly need a suitable representation of the variables of the problem. We consider an initial population of n nests, representing the potential solutions of the problem. Each solution consists of a real-valued vector of dimension $D \cdot \delta + 3\zeta + 2$ containing the parameters ρ_β , vector coefficients Θ_α , and weights Ω_β , δ , and γ . The structure of this vector is also highly constrained. On one hand, the set of parameters $\{\rho_\beta\}_\beta$ is constrained to lie within the unit interval $[0, 1]$. In computational terms, this means that different controls are to be set up in order

to check for this condition to hold. On the other hand, the ordered structure of data points means that those parameters must also be sorted. Finally, weights are assumed to be strictly positive real numbers.

Regarding the fitness function, it is given by either the weighted Bayesian energy functional in (2) or by the weighted strain energy functional in (3), where the former penalizes any unnecessarily large number of free parameters for the model, while the latter imposes additional constraints regarding the smoothness of the fitting curve. Note also that the strength of the functionals can be modulated by the parameter γ to satisfy additional constraints.

5.1. Parameter Tuning. A critical issue when working with metaheuristic approaches concerns the choice of suitable parameter values for the method. This issue is of paramount importance since the proper choice of such values will largely determine the good performance of the method. Unfortunately, it is also a very difficult task. On one hand, the field still lacks sufficient theoretical results to answer this question on a general basis. On the other hand, the choice of parameter values is strongly problem-dependent, meaning that good parameter values for a particular problem might be completely useless (even counterproductive) for any other problem. These facts explain why the choice of adequate parameter values is so troublesome and very often a bottleneck in the development and application of metaheuristic techniques.

The previous limitations have been traditionally overcome by following different strategies. Perhaps the most common one is to obtain good parameter values empirically. In this approach, several runs or executions of the method are carried out for different parameter values and a statistical analysis is performed to derive the values leading to the best performance. However, this approach is very time-consuming, especially when different parameters influence each other. This problem is aggravated when the metaheuristic depends on many different parameters, leading to an exponential growth in the number of executions. The cuckoo search method is particularly adequate in this regard because of its simplicity. In contrast to other methods that typically require a large number of parameters, the CS only requires two parameters, namely, the population size n and the probability p_a . This makes the parameter tuning much easier for CS than for other metaheuristic approaches.

Some previous works have addressed the issue of parameter tuning for CS. They showed that the method is relatively robust to the variation of parameters. For instance, authors in [21] tried different values for $n = 5, 10, 15, 20, 50, 100, 150, 250,$ and 500 and $p_a = 0, 0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.4,$ and 0.5 . They obtained that the convergence rate of the method is not very sensitive to the parameters used, implying that no fine adjustment is needed for the method to perform well. Our experimental results are in good agreement with these empirical observations. We performed several trials for the parameter values indicated above and found that our results do not differ significantly in any case. We noticed, however, that some parameter values are more

adequate in terms of the number of iterations required to reach convergence. In this paper, we set the parameters n and p_a to 100 and 0.25, respectively.

5.2. Implementation Issues. Regarding the implementation, all computations in this paper have been performed on a 2.6 GHz Intel Core i7 processor with 8 GB RAM. The source code has been implemented by the authors in the native programming language of the popular scientific program MATLAB, version 2012a. We remark that an implementation of the CS method has been described in [21]. Similarly, a vectorized implementation of CS in MATLAB is freely available in [65]. Our implementation is strongly based (although not exactly identical) on that efficient open-source version of the CS.

6. Experimental Results

We have applied the CS method described in previous sections to different examples of curve data fitting. To keep the paper in manageable size, in this section we describe only five of them, corresponding to different families of global-support basis functions and also to open and closed 2D and 3D curves. In order to replicate the conditions of real-world applications, we assume that our data are irregularly sampled and subjected to noise. Consequently, we consider a nonuniform sampling of data in all our examples. Data points are also perturbed by an additive Gaussian white noise of small intensity given by a SNR (signal-to-noise ratio) of 60 in all reported examples.

First example corresponds to a set of 100 noisy data points obtained by nonuniform sampling from the Agnesi curve. The curve is obtained by drawing a line OB from the origin through the circle of radius r and center $(0, r)$ and then picking the point with the y coordinate of the intersection with the circle and the x coordinate of the intersection of the extension of line OB with the line $y = 2r$. Then, they are fitted by using the Bernstein basis functions. Our results are depicted in Figure 1(a), where the original data points are displayed as red emptied circles, whereas the reconstructed points appear as blue plus symbols. Note the good matching between the original and the reconstructed data points. In fact, we got a fitness value of 1.98646×10^{-3} , indicating that the reconstructed curve fits the noisy data points with high accuracy. The average CPU time for this example is 3.01563 seconds. We also computed the absolute mean value of the difference between the original and the reconstructed data points for each coordinate and obtained good results: $(9.569738 \times 10^{-4}, 1.776091 \times 10^{-3})$. This good performance is also reflected in Figure 1(b), where the original data points and the reconstructed Bézier fitting curve are displayed as black plus symbols and a blue solid line, respectively.

Second example corresponds to the Archimedean spiral curve (also known as the arithmetic spiral curve). This curve is the locus of points corresponding to the locations over time of a point moving away from a fixed point with a constant speed along a line which rotates with constant angular velocity. In this example, we consider a set of 100

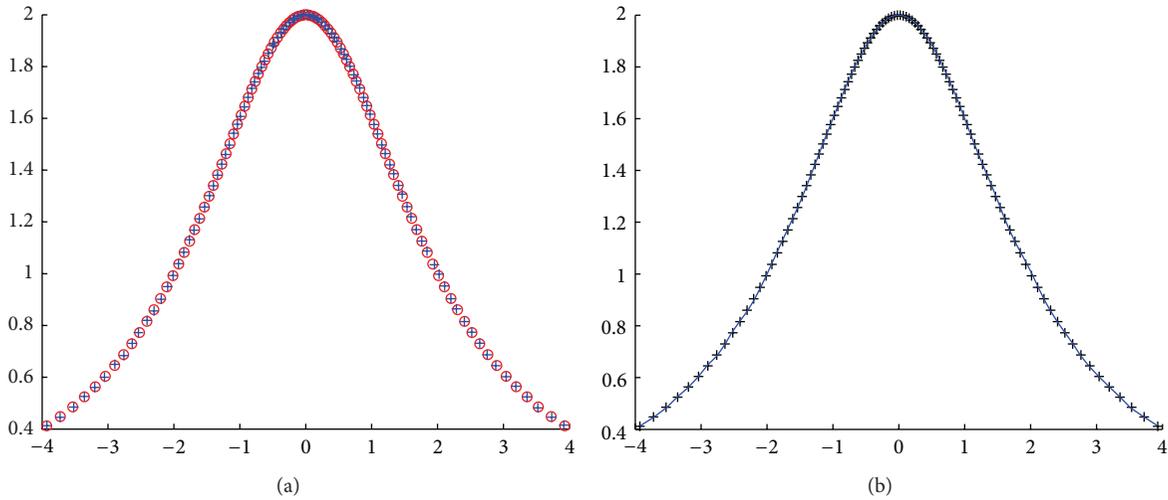


FIGURE 1: Application of the cuckoo search algorithm to the Agnesi curve: (a) original data points (red emptied circles) and reconstructed points (in blue plus symbol); (b) data points (black plus symbol) and fitting curve (solid blue line).

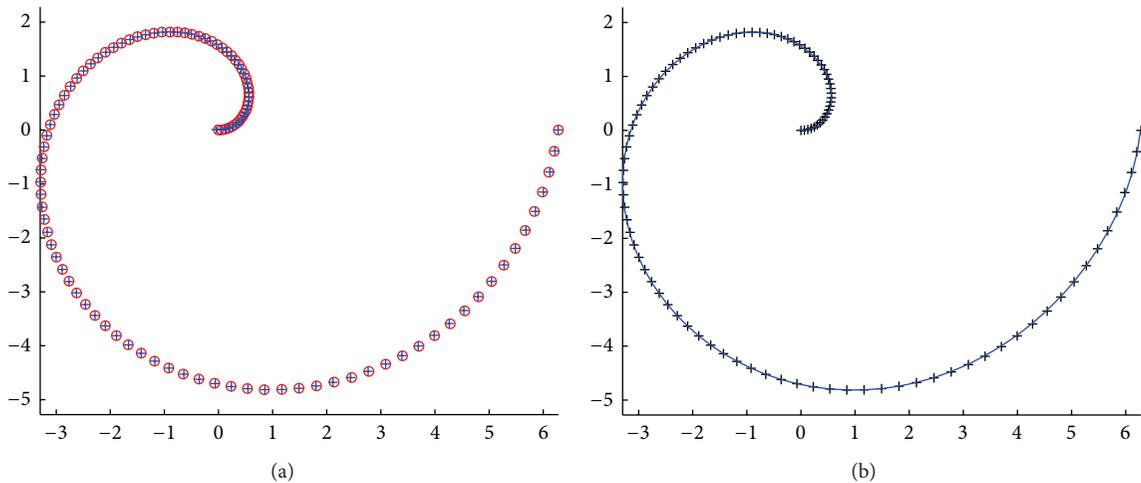


FIGURE 2: Application of the cuckoo search algorithm to the Archimedean spiral curve: (a) original data points (red emptied circles) and reconstructed points (in blue plus symbol); (b) data points (black plus symbol) and fitting curve (solid blue line).

noisy data points from such a curve that are subsequently fitted by using the canonical polynomial basis functions. Our results for this example are depicted in Figure 2. We omit the interpretation of this figure because it is similar to the previous one. Once again, note the good matching between the original and the reconstructed data points. In this case we obtained a fitness value of 1.12398×10^{-2} for these data points, while the absolute mean value of the difference between the original and the reconstructed data points for each coordinate is $(1.137795 \times 10^{-2}, 6.429596 \times 10^{-3})$. The average CPU time for this example is 4.68752 seconds. We conclude that the CS method is able to obtain a global-support curve that fits the data points pretty well.

Third example corresponds to a hypocycloid curve. This curve belongs to a set of a much larger family of curves called the roulettes. Roughly speaking, a roulette is a curve generated by tracing the path of a point attached to a curve

that is rolling upon another fixed curve without slippage. In principle, they can be any two curves. The particular case of a hypocycloid corresponds to a roulette traced by a point attached to a circle of radius r rolling around the inside of a fixed circle of radius R , where it is assumed that $R = k \cdot r$. If $k = R/r$ is a rational number, then the curve eventually closes on itself and has R cusps (i.e., sharp corners, where the curve is not differentiable). In this example, we consider a set of 100 noisy data points from the hypocycloid curve with 5 cusps. They are subsequently fitted by using the Bernstein basis functions. Figure 3 shows our results graphically. In this case, the best fitness value is 2.00706×10^{-3} , while the absolute mean value of the difference between the original and the reconstructed data points for each coordinate is $(1.661867 \times 10^{-3}, 1.521872 \times 10^{-3})$. The average CPU time for this example is 9.82813 seconds. In this case, the complex geometry of the curve, involving several cusps and self-intersections, leads

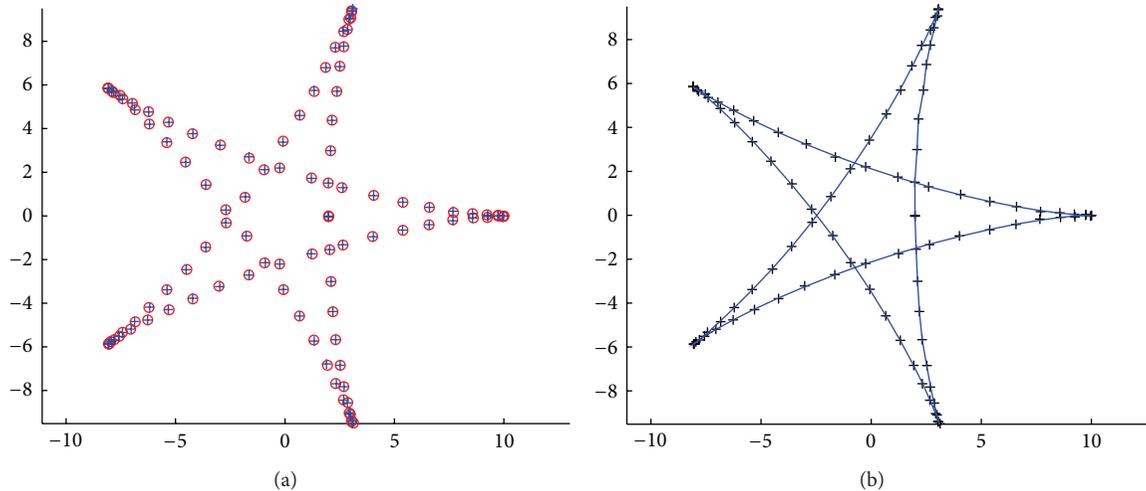


FIGURE 3: Application of the cuckoo search algorithm to the hypocycloid curve example: (a) original data points (red emptied circles) and reconstructed points (in blue plus symbol); (b) data points (black plus symbol) and fitting curve (solid blue line).

to this relatively large CPU time in comparison with the previous (much simpler) examples. In fact, this example is very illustrative about the ability of the method to perform well even in case of nonsmooth self-intersecting curves.

Fourth example corresponds to the so-called piriform curve, which can be defined procedurally in a rather complex way. Once again, we consider a set of 100 noisy data points fitted by using the Bernstein basis functions. Our results are shown in Figure 4. The best fitness value in this case is 1.17915×10^{-3} , while the absolute mean value of the difference between the original and the reconstructed data points for each coordinate is $(8.64616 \times 10^{-4}, 5.873391 \times 10^{-4})$. The average CPU time for this example is 3.276563 seconds. Note that this curve has a cusp in the leftmost part; moreover, the data points tend to concentrate around the cusp, meaning that the data parameterization is far from uniform. However, the method is still able to recover the shape of the curve with great detail.

The last example corresponds to a 3D closed curve called Eight Knot curve. Two images of the curve from different viewpoints are shown in Figure 5. The CS method is applied to a set of 100 noisy data points for the Bernstein basis functions. Our results are shown in Figure 6. The best fitness value in this case is 3.193634×10^{-2} , while the absolute mean value of the difference between the original and the reconstructed data points for each coordinate is $(2.7699870 \times 10^{-2}, 2.863125 \times 10^{-2}, 1.3710703 \times 10^{-2})$. The average CPU time for this example is 8.75938 seconds.

7. Conclusions and Future Work

This paper addresses the problem of approximating a set of data points by using global-support curves while simultaneously minimizing the degrees of freedom of the model function and satisfying other additional constraints. This problem is formulated in terms of a weighted Bayesian

energy functional that encapsulates all these constraints into a single mathematical expression. In this way, the original problem is converted into a continuous nonlinear multivariate optimization problem, which is solved by using a metaheuristic approach. Our method is based on the cuckoo search, a powerful nature-inspired metaheuristic algorithm introduced recently to solve optimization problems. Cuckoo search (especially its variant that uses Lévy flights) has been successfully applied to difficult optimization problems in different fields. However, to the best of our knowledge, this is the first paper applying the cuckoo search methodology in the context of geometric modeling and data fitting.

Our approach based on the cuckoo search method has been tested on five illustrative examples of different types, including open and closed 2D and 3D curves. Some examples also exhibit challenging features, such as cusps and self-intersections. They have been fitted by using two different families of global-support functions (Bernstein basis functions and the canonical polynomial basis) with satisfactory results in all cases. The experimental results show that the method performs pretty well, being able to solve our difficult minimization problem in an astonishingly straightforward way. We conclude that this new approach can be successfully applied to solve our optimization problem.

A major advantage of this method is its simplicity: cuckoo search requires only two parameters, many fewer than other metaheuristic approaches, so the parameter tuning becomes a very simple task. This simplicity is also reflected in the CPU runtime of our examples. Even though we are dealing with a constrained continuous multivariate nonlinear optimization problem and with curves exhibiting challenging features such as cusps and self-intersections, a typical single execution takes less than 10 seconds of CPU time for all the examples reported in this paper. In addition, the method is simple to understand, easy to implement and does not require any further pre-/postprocessing.

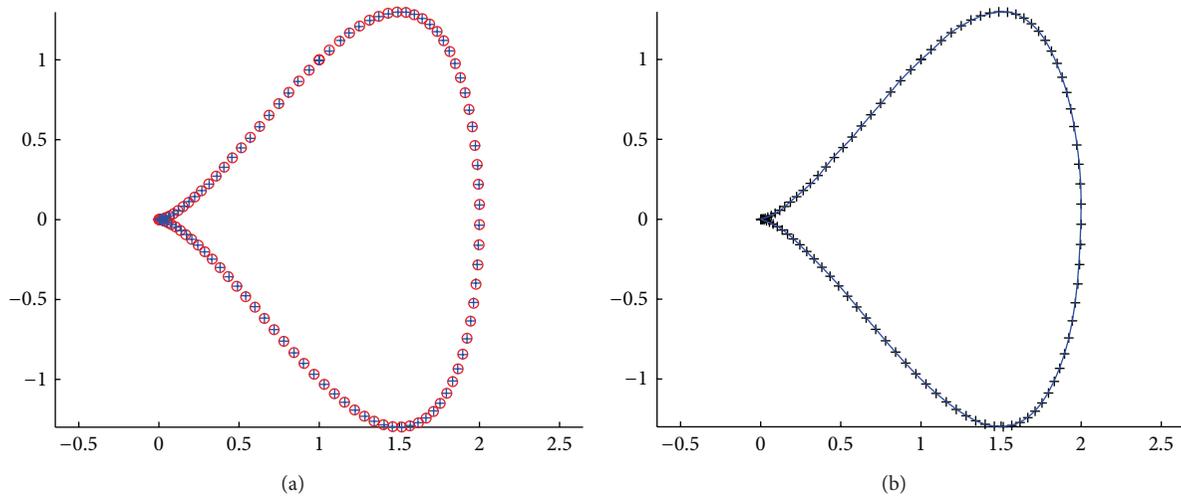


FIGURE 4: Application of the cuckoo search algorithm to the piriform curve example: (a) original data points (red emptied circles) and reconstructed points (in blue plus symbol); (b) data points (black plus symbol) and fitting curve (solid blue line).

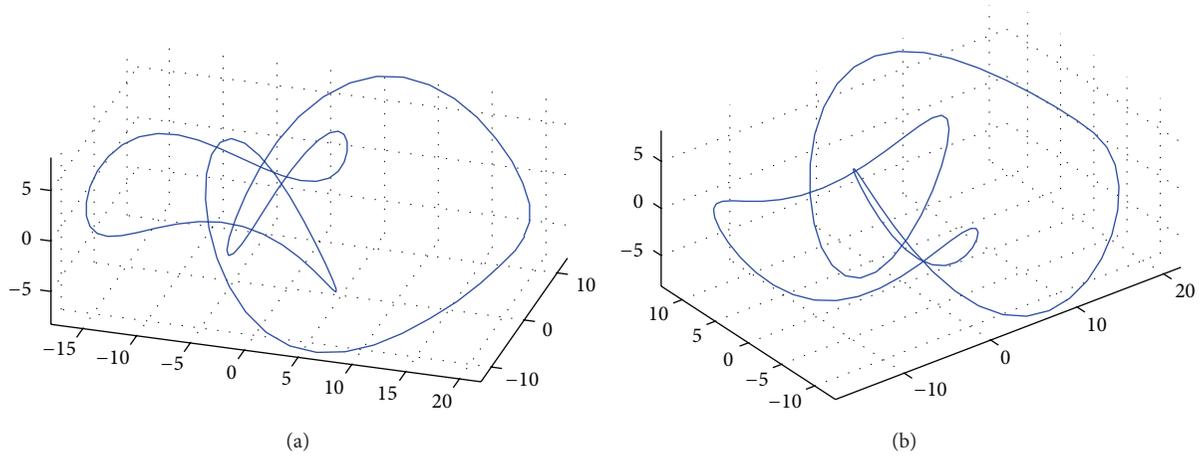


FIGURE 5: Two different viewpoints of the 3D Eight Knot curve.

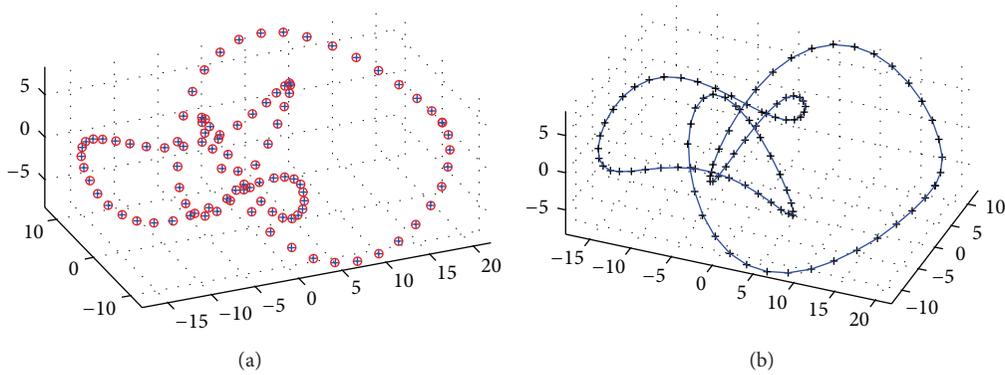


FIGURE 6: Application of the cuckoo search algorithm to the 3D Eight Knot curve example: (a) original data points (red emptied circles) and reconstructed points (in blue plus symbol); (b) data points (black plus symbol) and fitting curve (solid blue line).

In spite of these encouraging results, further research is still needed to determine the advantages and limitations of the present method at full extent. On the other hand, some modifications of the original cuckoo search have been claimed to outperform the initial method on some benchmarks. Our implementation has been designed according to the specifications of the original method and we did not test any of its subsequent modifications yet. We are currently interested in exploring these issues as part of our future work. The hybridization of this approach with other competitive methods for better performance is also part of our future work.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper. Any commercial identity mentioned in this paper is cited solely for scientific purposes.

Acknowledgments

This research has been kindly supported by the Computer Science National Program of the Spanish Ministry of Economy and Competitiveness, Project Ref. no. TIN2012-30768, Toho University (Funabashi, Japan), and the University of Cantabria (Santander, Spain). The authors are particularly grateful to the Department of Information Science of Toho University for all the facilities given to carry out this work.

References

- [1] J. R. Rice, *The Approximation of Functions*, vol. 2, Addison-Wesley, Reading, Mass, USA, 1969.
- [2] P. Dierckx, *Curve and Surface Fitting with Splines*, Oxford University Press, Oxford, UK, 1993.
- [3] J. R. Rice, *Numerical Methods, Software and Analysis*, Academic Press, New York, NY, USA, 2nd edition, 1993.
- [4] L. Piegl and W. Tiller, *The NURBS Book*, Springer, Berlin, Germany, 1997.
- [5] R. E. Barnhill, *Geometric Processing for Design and Manufacturing*, SIAM, Philadelphia, Pa, USA, 1992.
- [6] G. Echevarría, A. Iglesias, and A. Gálvez, "Extending neural networks for B-spline surface reconstruction," in *Computational Science—ICCS 2002*, vol. 2330 of *Lectures Notes in Computer Science*, pp. 305–314, Springer, Berlin, Germany, 2002.
- [7] A. Gálvez, A. Iglesias, and J. Puig-Pey, "Iterative two-step genetic-algorithm-based method for efficient polynomial B-spline surface reconstruction," *Information Sciences*, vol. 182, no. 1, pp. 56–76, 2012.
- [8] P. Gu and X. Yan, "Neural network approach to the reconstruction of freeform surfaces for reverse engineering," *Computer-Aided Design*, vol. 27, no. 1, pp. 59–64, 1995.
- [9] A. Gálvez and A. Iglesias, "Particle swarm optimization for non-uniform rational B-spline surface reconstruction from clouds of 3D data points," *Information Sciences*, vol. 192, pp. 174–192, 2012.
- [10] M. Hoffmann, "Numerical control of Kohonen neural network for scattered data approximation," *Numerical Algorithms*, vol. 39, no. 1–3, pp. 175–186, 2005.
- [11] A. Iglesias, G. Echevarría, and A. Gálvez, "Functional networks for B-spline surface reconstruction," *Future Generation Computer Systems*, vol. 20, no. 8, pp. 1337–1353, 2004.
- [12] N. M. Patrikalakis and T. Maekawa, *Shape Interrogation for Computer Aided Design and Manufacturing*, Springer, Heidelberg, Germany, 2002.
- [13] H. Pottmann, S. Leopoldseder, M. Hofer, T. Steiner, and W. Wang, "Industrial geometry: recent advances and applications in CAD," *Computer Aided Design*, vol. 37, no. 7, pp. 751–766, 2005.
- [14] A. Iglesias and A. Gálvez, "A new artificial intelligence paradigm for computer aided geometric design," in *Artificial Intelligence and Symbolic Computation*, vol. 1930 of *Lectures Notes in Artificial Intelligence*, pp. 200–213, Springer, Berlin, Germany, 2001.
- [15] T. Varady and R. Martin, "Reverse engineering," in *Handbook of Computer Aided Geometric Design*, G. Farin, J. Hoschek, and M. Kim, Eds., Elsevier Science, Amsterdam, The Netherlands, 2002.
- [16] T. C. M. Lee, "On algorithms for ordinary least squares regression spline fitting: a comparative study," *Journal of Statistical Computation and Simulation*, vol. 72, no. 8, pp. 647–663, 2002.
- [17] W. Y. Ma and J. P. Kruth, "Parameterization of randomly measured points for least squares fitting of B-spline curves and surfaces," *Computer-Aided Design*, vol. 27, no. 9, pp. 663–675, 1995.
- [18] H. Park and J. H. Lee, "B-spline curve fitting based on adaptive curve refinement using dominant points," *Computer Aided Design*, vol. 39, no. 6, pp. 439–451, 2007.
- [19] L. A. Piegl and W. Tiller, "Least-squares B-spline curve approximation with arbitrary end derivatives," *Engineering with Computers*, vol. 16, no. 2, pp. 109–116, 2000.
- [20] T. Varady, R. R. Martin, and J. Cox, "Reverse engineering of geometric models—an introduction," *Computer Aided Design*, vol. 29, no. 4, pp. 255–268, 1997.
- [21] X. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, IEEE, Coimbatore, India, December 2009.
- [22] X. S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343, 2010.
- [23] G. Farin, *Curves and Surfaces for CAGD*, Morgan Kaufmann, San Francisco, Calif, USA, 5th edition, 2002.
- [24] A. Gálvez and A. Iglesias, "Efficient particle swarm optimization approach for data fitting with free knot B-splines," *Computer Aided Design*, vol. 43, no. 12, pp. 1683–1692, 2011.
- [25] L. Jing and L. Sun, "Fitting B-spline curves by least squares support vector machines," in *Proceedings of the International Conference on Neural Networks and Brain (ICNNB '05)*, vol. 2, pp. 905–909, IEEE Press, Beijing, China, October 2005.
- [26] W. P. Wang, H. Pottmann, and Y. Liu, "Fitting B-spline curves to point clouds by curvature-based squared distance minimization," *ACM Transactions on Graphics*, vol. 25, no. 2, pp. 214–238, 2006.
- [27] H. P. Yang, W. P. Wang, and J. G. Sun, "Control point adjustment for B-spline curve approximation," *Computer Aided Design*, vol. 36, no. 7, pp. 639–652, 2004.
- [28] J. Hoschek and D. Lasser, *Fundamentals of Computer Aided Geometric Design*, A. K. Peters, Wellesley, Mass, USA, 1993.
- [29] D. L. B. Jupp, "Approximation to data by splines with free knots," *SIAM Journal of Numerical Analysis*, vol. 15, no. 2, pp. 328–343, 1978.

- [30] M. J. D. Powell, "Curve fitting by splines in one variable," in *Numerical Approximation to Functions and Data*, J. G. Hayes, Ed., Athlone Press, London, UK, 1970.
- [31] H. Park, "An error-bounded approximate method for representing planar curves in B-splines," *Computer Aided Geometric Design*, vol. 21, no. 5, pp. 479–497, 2004.
- [32] Y. J. Ahn, C. Hoffmann, and P. Rosen, "Geometric constraints on quadratic Bezier curves using minimal length and energy," *Journal of Computational and Applied Mathematics*, vol. 255, pp. 887–897, 2014.
- [33] L. Fang and D. C. Gossard, "Multidimensional curve fitting to unorganized data points by nonlinear minimization," *Computer-Aided Design*, vol. 27, no. 1, pp. 48–58, 1995.
- [34] H. Martinsson, F. Gaspard, A. Bartoli, and J. M. Lavest, "Energy-based reconstruction of 3D curves for quality control," in *Energy Minimization Methods in Computer Vision and Pattern Recognition*, vol. 4679 of *Lecture Notes in Computer Science*, pp. 414–428, Springer, Berlin, Germany, 2007.
- [35] T. I. Vassilev, "Fair interpolation and approximation of B-splines by energy minimization and points insertion," *Computer-Aided Design*, vol. 28, no. 9, pp. 753–760, 1996.
- [36] C. Zhang, P. Zhang, and F. Cheng, "Fairing spline curves and surfaces by minimizing energy," *Computer Aided Design*, vol. 33, no. 13, pp. 913–923, 2001.
- [37] G. Brunnett and J. Kiefer, "Interpolation with minimal-energy splines," *Computer-Aided Design*, vol. 26, no. 2, pp. 137–144, 1994.
- [38] H. P. Moreton and C. H. Sequin, "Functional optimization for fair surface design," *Computer Graphics*, vol. 26, no. 2, pp. 167–176, 1992.
- [39] E. Sariöz, "An optimization approach for fairing of ship hull forms," *Ocean Engineering*, vol. 33, no. 16, pp. 2105–2118, 2006.
- [40] R. C. Veltkamp and W. Wesselink, "Modeling 3D curves of minimal energy," *Computer Graphics Forum*, vol. 14, no. 3, pp. 97–110, 1995.
- [41] J. Barhak and A. Fischer, "Parameterization and reconstruction from 3D scattered points based on neural network and PDE techniques," *IEEE Transactions on Visualization and Computer Graphics*, vol. 7, no. 1, pp. 1–16, 2001.
- [42] A. Iglesias and A. Gálvez, "Hybrid functional-neural approach for surface reconstruction," *Mathematical Problems in Engineering*, vol. 2014, Article ID 351648, 13 pages, 2014.
- [43] J. D. Farmer, N. H. Packard, and A. S. Perelson, "The immune system, adaptation, and machine learning," *Physica D: Nonlinear Phenomena*, vol. 22, no. 1–3, pp. 187–204, 1986.
- [44] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 52–67, 2002.
- [45] S. Nakrani and C. Tovey, "On honey bees and dynamic server allocation in internet hosting centers," *Adaptive Behavior*, vol. 12, no. 3–4, pp. 223–240, 2004.
- [46] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.
- [47] X. S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications*, vol. 5792 of *Lecture Notes in Computer Science*, pp. 169–178, Springer, Berlin, Germany, 2009.
- [48] X. S. Yang, "Firefly algorithm, stochastic test functions and design optimization," *International Journal of Bio-Inspired Computation*, vol. 2, no. 2, pp. 78–84, 2010.
- [49] X. S. Yang, "A new metaheuristic Bat-inspired Algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, vol. 284 of *Studies in Computational Intelligence*, pp. 65–74, Springer, Berlin, Germany, 2010.
- [50] X. S. Yang, "Bat algorithm for multi-objective optimisation," *International Journal of Bio-Inspired Computation*, vol. 3, no. 5, pp. 267–274, 2011.
- [51] A. Gálvez, A. Iglesias, A. Cobo, J. Puig-Pey, and J. Espinola, "Bézier curve and surface fitting of 3D point clouds through genetic algorithms, functional networks and least-squares approximation," in *Computational Science and Its Applications—ICCSA 2007*, vol. 4706 of *Lecture Notes in Computer Science*, pp. 680–693, Springer, Berlin, Germany, 2007.
- [52] M. Sarfraz and S. A. Raza, "Capturing outline of fonts using genetic algorithms and splines," in *Proceedings of the 5th International Conference on Information Visualisation (IV '01)*, pp. 738–743, IEEE Computer Society Press, London, UK, 2001.
- [53] F. Yoshimoto, M. Moriyama, and T. Harada, "Automatic knot adjustment by a genetic algorithm for data fitting with a spline," in *Proceedings of the International Conference on Shape Modeling and Applications*, pp. 162–169, IEEE Computer Society Press, 1999.
- [54] F. Yoshimoto, T. Harada, and Y. Yoshimoto, "Data fitting with a spline using a real-coded genetic algorithm," *Computer Aided Design*, vol. 35, no. 8, pp. 751–760, 2003.
- [55] A. Gálvez and A. Iglesias, "Firefly algorithm for polynomial Bézier surface parameterization," *Journal of Applied Mathematics*, vol. 2013, Article ID 237984, 9 pages, 2013.
- [56] E. Ülker and A. Arslan, "Automatic knot adjustment using an artificial immune system for B-spline curve approximation," *Information Sciences*, vol. 179, no. 10, pp. 1483–1494, 2009.
- [57] A. Gálvez and A. Iglesias, "Firefly algorithm for explicit B-spline curve fitting to data points," *Mathematical Problems in Engineering*, vol. 2013, Article ID 528215, 12 pages, 2013.
- [58] A. Gálvez and A. Iglesias, "From nonlinear optimization to convex optimization through firefly algorithm and indirect approach with applications to CAD/CAM," *The Scientific World Journal*, vol. 2013, Article ID 283919, 10 pages, 2013.
- [59] X. Zhao, C. Zhang, B. Yang, and P. Li, "Adaptive knot placement using a GMM-based continuous optimization algorithm in B-spline curve approximation," *Computer Aided Design*, vol. 43, no. 6, pp. 598–604, 2011.
- [60] A. Galvez and A. Iglesias, "New memetic self-adaptive firefly algorithm for continuous optimization," *International Journal of Bio-Inspired Computation*. In press.
- [61] A. Gálvez and A. Iglesias, "A new iterative mutually coupled hybrid GA-PSO approach for curve fitting in manufacturing," *Applied Soft Computing Journal*, vol. 13, no. 3, pp. 1491–1504, 2013.
- [62] G. E. Schwarz, "Estimating the dimension of a model," *Annals of Statistics*, vol. 6, no. 2, pp. 461–464, 1978.
- [63] E. Castillo and A. Iglesias, "Some characterizations of families of surfaces using functional equations," *ACM Transactions on Graphics*, vol. 16, no. 3, pp. 296–318, 1997.
- [64] X. S. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, Frome, UK, 2nd edition, 2010.
- [65] "Vectorized cuckoo search implementation in Matlab freely," <http://www.mathworks.com/matlabcentral/fileexchange/29809-cuckoo-search-csalgorithm>.

Research Article

PSO-Based Support Vector Machine with Cuckoo Search Technique for Clinical Disease Diagnoses

Xiaoyong Liu^{1,2} and Hui Fu¹

¹ Department of Computer Science, Guangdong Polytechnic Normal University, Guangzhou, Guangdong 510665, China

² School of Business Administration, South China University of Technology, Guangzhou, Guangdong 510640, China

Correspondence should be addressed to Xiaoyong Liu; liugucas@gmail.com

Received 16 April 2014; Accepted 4 May 2014; Published 25 May 2014

Academic Editor: Xin-She Yang

Copyright © 2014 X. Liu and H. Fu. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Disease diagnosis is conducted with a machine learning method. We have proposed a novel machine learning method that hybridizes support vector machine (SVM), particle swarm optimization (PSO), and cuckoo search (CS). The new method consists of two stages: firstly, a CS based approach for parameter optimization of SVM is developed to find the better initial parameters of kernel function, and then PSO is applied to continue SVM training and find the best parameters of SVM. Experimental results indicate that the proposed CS-PSO-SVM model achieves better classification accuracy and F-measure than PSO-SVM and GA-SVM. Therefore, we can conclude that our proposed method is very efficient compared to the previously reported algorithms.

1. Introduction

Accurate diagnosis and effective treatment of disease are important issues in life science research and have a positive meaning for human health. Recently, medical experts pay more attention to early diagnosis of disease and propose many new methods to deal with disease diagnosis problem. Using machine learning methods to diagnose disease is rapid development of a novel research branch of machine learning. Researchers have applied artificial intelligence and computer technology to develop some medical diagnostic systems, which improve the efficiency of diagnosis and become practical tools.

It is shown that support vector machine has good generalization ability and has been widely used in many research areas, such as signal classification [1], image processing [2], and disease diagnosis [3–6]. Illán et al. [3] showed a fully automatic computer-aided diagnosis (CAD) system for improving the accuracy in the early diagnosis of the AD. The proposed approach is based firstly on an automatic feature selection and secondly on a combination of component-based support vector machine (SVM) classification and a pasting votes technique of assembling SVM classifiers.

Sartakhti et al. [4] proposed a novel machine learning method that hybridized support vector machine (SVM) and simulated annealing (SA) for hepatitis disease diagnosis. The obtained classification accuracy of SVM-SA method was 96.25% and it was very promising with regard to the other classification methods in the literature for this problem. The approach proposed by Ramírez et al. [5] was based on image parameter selection and support vector machine (SVM) classification. The proposed system yielded a 90.38% accuracy in the early diagnosis of Alzheimer's disease and outperformed existing techniques including the voxel-as-features (VAF) approach. Abdi and Giveki [6] developed a diagnosis model based on particle swarm optimization (PSO), support vector machines (SVMs), and association rules (ARs) to diagnose erythematousquamous diseases. The proposed model consists of two stages: first, AR is used to select the optimal feature subset from the original feature set. Then, a PSO-based approach for parameter determination of SVM is developed to find the best parameters of kernel function (based on the fact that kernel parameter setting in the SVM training procedure significantly influences the classification accuracy and PSO is a promising tool for global searching).

Support vector machine is a machine learning algorithm based on statistical learning theory and has the strong predictive ability for nonlinear problems. However, SVM prediction performance is closely related to the quality of the selected parameters. Parameter optimization algorithms currently used are particle swarm optimization and genetic algorithms, but these algorithms have their shortcomings and affect the accuracy of disease prediction.

Cuckoo search (CS) is a new swarm intelligent optimization algorithm. Preliminary studies show that cuckoo search algorithm is simple and efficient, easy to implement and has less parameters [7]. Cuckoo search algorithm is able to provide a new method for the SVM parameter optimization. This paper proposes a disease diagnosis model based on cuckoo search, particle swarm optimization (PSO), and support vector machine.

The structure of this paper is as the following. Section 2 firstly introduces related algorithms, such as support vector machine and cuckoo search, and then presents the novel models, CS-PSO-SVM. Section 3 gives results of different models in two real disease diagnoses datasets from University of California Irvine Machine Learning Repository. Finally, conclusions are presented in Section 4.

2. Methods and Materials

2.1. Methods

2.1.1. SVM. Support vector machines [8–10] (SVMs) are a set of related supervised learning methods used for classification and regression. A support vector machine constructs a hyperplane or set of hyperplanes in a high-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data points of any class (so-called functional margin), since, in general, the larger the margin, the lower the generalization error of the classifier.

In order to extend the SVM methodology to handle data that is not fully linearly separable, we relax the constraints slightly to allow for misclassified points; the formulation is following (1). This is done by introducing a positive slack variable ξ_i , $i = 1, 2, \dots, L$:

$$\begin{aligned} x_i \cdot w + b &\geq +1 - \xi_i \quad (y_i = +1), \\ x_i \cdot w + b &\leq -1 + \xi_i \quad (y_i = -1) \end{aligned} \quad (1)$$

which can be combined into

$$y_i (x_i \cdot w + b) - 1 + \xi_i \geq 0, \quad (2)$$

where $\xi_i \geq 0$.

In this soft margin SVM, data points on the incorrect side of the margin boundary have a penalty that increases with the distance from it. As we are trying to reduce the number

of misclassifications, a sensible way to adapt our objective function from the previously mentioned is to find

$$\begin{aligned} \min \quad & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^L \xi_i \\ \text{s.t.} \quad & y_i (x_i \cdot w + b) - \varepsilon + \xi_i \geq 0, \end{aligned} \quad (3)$$

where the parameter C controls the trade-off between the slack variable penalty and the size of the margin. Reformulating as a Lagrange, which as before we need to minimize with respect to w , b , and ξ_i and maximize with respect to α (where $\alpha_i \geq 0$, $u_i \geq 0$),

$$\begin{aligned} L_P \equiv & \frac{1}{2} \|w\|^2 + C \sum_{i=1}^L \xi_i \\ & - \sum_{i=1}^L \alpha_i [y_i (x_i \cdot w + b) - 1 + \xi_i] - \sum_{i=1}^L \mu_i \xi_i. \end{aligned} \quad (4)$$

Differentiating with respect to w , b , and ξ_i and setting the derivatives to zero,

$$\begin{aligned} \frac{\partial L_P}{\partial w} = 0 &\implies w = \sum_{i=1}^L \alpha_i y_i x_i, \\ \frac{\partial L_P}{\partial b} = 0 &\implies \sum_{i=1}^L \alpha_i y_i = 0, \\ \frac{\partial L_P}{\partial \xi_i} = 0 &\implies C = \alpha_i + \mu_i. \end{aligned} \quad (5)$$

So, we need to find

$$\begin{aligned} \max_{\alpha} \quad & \left[\sum_{i=1}^L \alpha_i - \frac{1}{2} \alpha^T H \alpha \right] \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C, \quad \sum_{i=1}^L \alpha_i y_i = 0. \end{aligned} \quad (6)$$

When applying SVM to nonlinear dataset, we need to define a feature mapping function $x \mapsto \phi(x)$. The feature mapping function is called kernel function. In the feature space, optimal hyperplane (Figure 1) can be gotten.

There are three common kernel functions:

polynomial kernel

$$k(x_i, x_j) = (x_i \cdot x_j + a)^b, \quad (7)$$

radial basis kernel

$$k(x_i, x_j) = e^{-\|x_i - x_j\|^2 / 2\sigma^2}, \quad (8)$$

sigmoidal kernel

$$k(x_i, x_j) = \tanh(ax_i \cdot x_j - b), \quad (9)$$

where a and b are parameters defining the kernel's behavior.

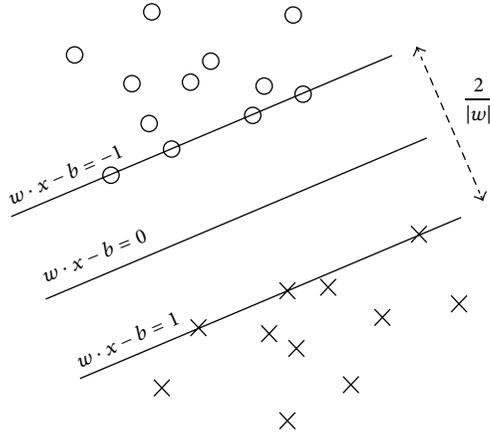


FIGURE 1: Optimal hyperplane.

In order to use SVM to solve a classification or regression problem on dataset that is nonlinearly separable, we need to first choose a kernel and relevant parameters which you expect they might map the nonlinearly separable data into a feature space where it is linearly separable. This is more of an art than an exact science and can be achieved empirically, for example, by trial and error. Sensible kernels to start with are polynomial, radial basis, and sigmoid kernels.

For classification, we need the following.

Create H , where $H_{ij} = y_i y_j \phi(x_i) \phi(x_j)$.

Choose how significantly misclassifications should be treated, by selecting a suitable value for the parameter C .

Find α so that

$$\begin{aligned} \max_{\alpha} \quad & \left[\sum_{i=1}^L \alpha_i - \frac{1}{2} \alpha^T H \alpha \right] \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq C, \quad \sum_{i=1}^L \alpha_i y_i = 0. \end{aligned} \tag{10}$$

Calculate

$$w = \sum_{i=1}^L \alpha_i y_i \phi(x_i). \tag{11}$$

Determine the set of support vectors V , by finding the indices such that $0 \leq \alpha_i \leq C$.

Calculate

$$b = \frac{1}{N_v} \sum_{v \in V} \left(y_v - \sum_{i \in V} \alpha_i y_i \phi(x_i) \phi(x_v) \right). \tag{12}$$

Each new point x' is classified by evaluating $y' = \text{sgn}(w\phi(x') + b)$.

2.1.2. PSO-SVM. Particle swarm optimization is an evolutionary computation technique proposed by Kennedy and Eberhart. It is a population-based stochastic search process, modeled after the social behavior of a bird flock [11, 12]. It is similar in spirit to birds migrating in a flock toward some

destination, where the intelligence and efficiency lie in the cooperation of an entire flock [13]. PSO algorithms make use of particles moving in an n -dimensional space to search for solutions for n -variable function optimization problem. All particles have fitness values which are evaluated by the fitness function to be optimized and have velocities which direct the flying of the particles. The particles fly through the problem space by following the particles with the best solutions so far. PSO is initialized with a group of random particles (solutions) and then searches for optima by updating each generation [14].

SVM also has a drawback that limits the use of SVM on academic and industrial platforms: there are free parameters (SVM hyperparameters and SVM kernel parameters) that need to be defined by the user. Since the quality of SVM regression models depends on a proper setting of these parameters, the main issue for practitioners trying to apply SVM is how to set these parameter values (to ensure good generalization performance) for a given training dataset.

SVM based on PSO optimizes two important hyperparameters C and ϵ using PSO. The hyperparameter C determines the trade-off between the model complexity and the degree to which deviations larger than ϵ are tolerated. A poor choice of C will lead to an imbalance between model complexity minimization (MCM) and empirical risk minimization (ERM). The hyperparameter ϵ controls the width of the ϵ -insensitive zone, and its value affects the number of SVs used to construct the regression function. If ϵ is set too large, the insensitive zone will have ample margin to include data points; this would result in too few SVs selected and lead to unacceptable “flat” regression estimates [15].

2.1.3. Cuckoo Search [16]. Cuckoo search is an optimization algorithm developed by Xin-she Yang and Suash Deb [16–18]. It was inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds (of other species). Some host birds can engage direct conflict with the intruding cuckoos. For example, if a host bird discovers that the eggs are not their own, it will either throw these alien eggs away or simply abandon its nest and build a new nest elsewhere. Some cuckoo species such as the New World brood-parasitic *Tapera* have evolved in such a way that female parasitic cuckoos are often very specialized in the mimicry in colors and pattern of the eggs of a few chosen host species [19].

Cuckoo search idealized such breeding behavior and thus can be applied for various optimization problems. It seems that it can outperform other metaheuristic algorithms in applications [20].

Cuckoo search (CS) uses the following representations.

Each egg in a nest represents a solution, and a cuckoo egg represents a new solution. The aim is to use the new and potentially better solutions (cuckoos) to replace a not-so-good solution in the nests. In the simplest form, each nest has one egg. The algorithm can be extended to more complicated cases in which each nest has multiple eggs representing a set of solutions.

```

Objective function:
 $f(x), X = (x_1, x_2, x_3, \dots, x_d)$ 
Generate an initial population of  $n$  host nests;
While ( $t < \text{MaxGeneration}$ ) or (stop criterion)
  Get a cuckoo's nest  $i$  randomly and replace its solution by performing Lévy flights;
  Evaluate its quality/fitness  $F_i$ 
  Choose a nest  $j$  among  $n$  randomly;
  if ( $F_i > F_j$ ),
    Replace  $j$  by the new solution;
  end if
  A fraction ( $P_a$ ) of the worse nests are abandoned and new ones are built;
  Keep the best solutions/nests;
  Rank the solutions/nests and find the current best;
  Pass the current best solutions to the next generation;
end while

```

ALGORITHM 1

CS is based on three idealized rules:

- (1) each cuckoo lays one egg at a time and dumps its egg in a randomly chosen nest;
- (2) the best nests with high quality of eggs will be carried over to the next generation;
- (3) the number of available hosts nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with a probability $P_a \in (0, 1)$. Discovering operate on some set of worst nests, and discovered solutions dumped from farther calculations.

In addition, Yang and Deb discovered that the random-walk style search is better performed by Lévy flights rather than simple random walk.

The pseudocode can be summarized as in Algorithm 1.

An important advantage of this algorithm is its simplicity. In fact, compared with other population- or agent-based metaheuristic algorithms such as particle swarm optimization and harmony search, there is essentially only a single parameter P_a in CS (apart from the population size n). Therefore, it is very easy to be implemented.

2.1.4. CS-PSO-SVM. CS-PSO-SVM consists of two stages: firstly, a CS based approach for parameter optimization of SVM is developed to find the better initial parameters of kernel function and then PSO is applied to continue SVM training and find the best parameters of SVM. CS-PSO-SVM algorithm can be shown as follows in detail.

Step 1. Initializing cuckoo and PSO with population size, inertia weight, generations, and the range of hyperparameters C and ϵ .

Step 2. Applying CS to find the better initial parameters C and ϵ .

Step 3. Evaluating the fitness of each particle.

Step 4. Comparing the fitness values and determining the local best and global best particle.

Step 5. Updating the velocity and position of each particle till the value of the fitness function converges.

Step 6. After converging, the global best particle in the swarm is fed to SVM classifier for training.

Step 7. Training and testing the SVM classifier.

The flowchart of CS-PSO-SVM algorithm is shown in Figure 2 in detail.

2.2. Materials. In this study, numerical experiments use two datasets, heart disease dataset, and breast cancer dataset from UCI Machine Learning Repository [21].

Statlog (heart) dataset has 270 instances. In this dataset, there are thirteen numerical attributes, including age, sex, chest, blood (Rbp), serum (mg/dL), sugar (Fbs), electrocardiographic (ECG), heart rate, angina, old peak, slope, vessels (0-3), and THAL. Number of the presence of heart disease in the patient is 150, and number of the absence of heart disease in the patient is 120. Breast cancer dataset has 699 instances. Number of attributes of each instance is nine. There are thirteen numerical attributes including radius (mean of distances from center to points on the perimeter), texture (standard deviation of gray-scale values), smoothness (local variation in radius lengths), compactness, concavity (severity of concave portions of the contour), concave points (number of concave portions of the contour), symmetry, and fractal dimension. There are 458 as "benign" and 241 as "malignant." In the above two datasets, "1" denotes one people as "benign" and "2" denotes one people as "malignant." Table 1 shows the detail of two datasets. Heart disease dataset and breast cancer dataset choose 190 and 549 instances as train datasets, respectively. The rest of two datasets are test datasets.

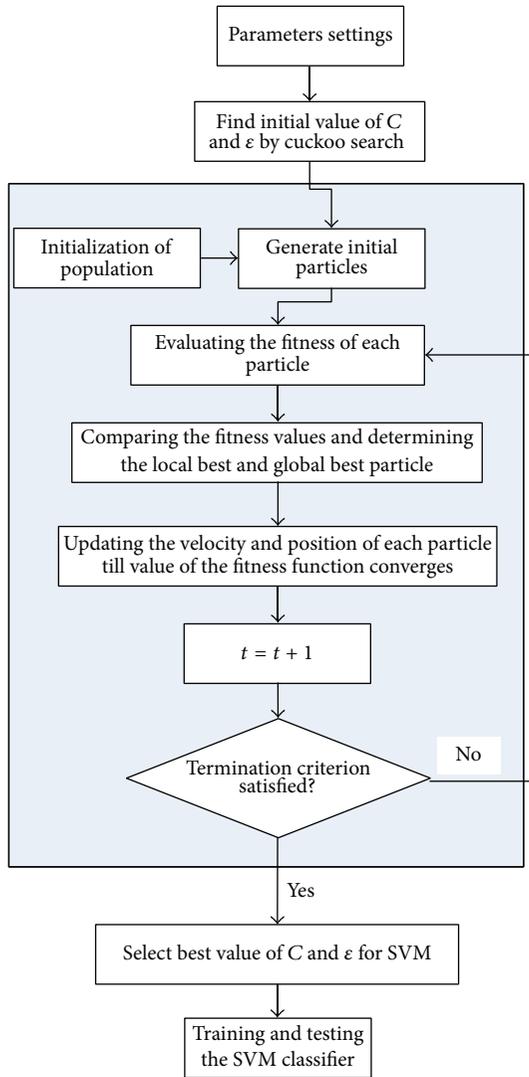


FIGURE 2: The flowchart of CS-PSO-SVM.

TABLE 1: Credit dataset.

Dataset	Number of instances	Number of attributes	Benign	Malignant
Heart disease dataset	270	13	150	120
Breast cancer dataset	699	9	458	241

Continuous attributes are normalized firstly and then used to train and test.

3. Results and Discussion

For the comparison of performance between traditional GA-SVM, PSO-SVM, and CS-PSO-SVM, these models are run several times. The program of the new algorithm is written by MATLAB 2012b and run on a computer with 2.0 GHz CPU, 1 GB DDR RAM. Figures 3 and 5 show the different curves

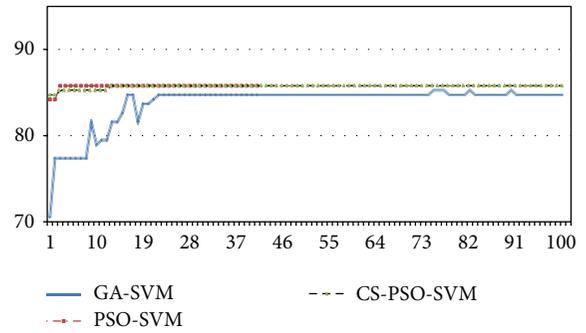


FIGURE 3: Population best fitness value in heart disease dataset.

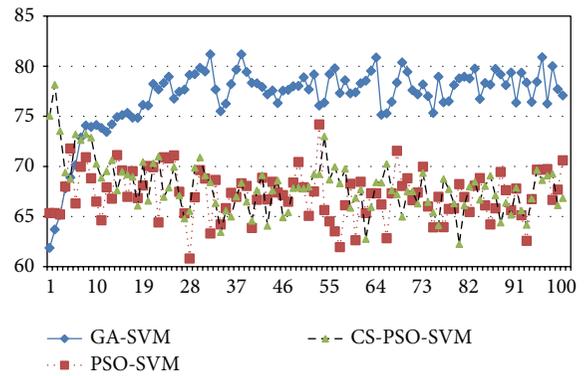


FIGURE 4: Population average fitness value in heart disease dataset.

TABLE 2: (a) Parameters setting of GA-SVM algorithm. (b) Parameters setting of PSO-SVM algorithm. (c) Parameters setting of CS-PSO-SVM algorithm.

(a)					
Parameter	PopSize	Iteration	P_c	P_m	
Value	20	100	0.5	0.005	
(b)					
Parameter	PopSize	Iteration	c_1	c_2	
Value	20	100	1.5	1.7	
(c)					
Parameter	PopSize	Iteration	P_a	PSO- c_1	PSO- c_2
Value	20	100	0.25	1.5	1.7

of population best fitness value in heart disease dataset and breast cancer dataset by GA-SVM, PSO-SVM, and CS-PSO-SVM. Figures 4 and 6 show the different curves of population average fitness value in heart disease dataset and breast cancer dataset by the above three algorithms. Table 2 lists the appropriate values of these parameters in three algorithms.

Table 3 shows the accuracy comparison of GA-SVM, PSO-SVM, and CS-PSO-SVM. For the heart disease dataset, in the test subset, the accuracy of CS-PSO-SVM is 85% and PSO-SVM and GA-SVM are 80%. In the training subset, the accuracy of CS-PSO-SVM is 100%.

TABLE 3: Comparison of models.

Dataset	Algorithm	Forecasting accuracy total in training set	Forecasting accuracy total in test set	Precision	Recall	F-measure
Heart disease dataset	GA-SVM	85.7895%	80%	76.09%	87.50%	81.40%
	PSO-SVM	86%	80%	76.09%	87.50%	81.40%
	CS-PSO-SVM	100%	85%	81.82%	90.00%	85.71%
Breast cancer dataset	GA-SVM	99.8179%	90.0%	92.08%	62.00%	74.10%
	PSO-SVM	97.4499%	90.0%	88.99%	64.67%	74.90%
	CS-PSO-SVM	98.3607%	91.3333%	91.43%	64.00%	75.29%

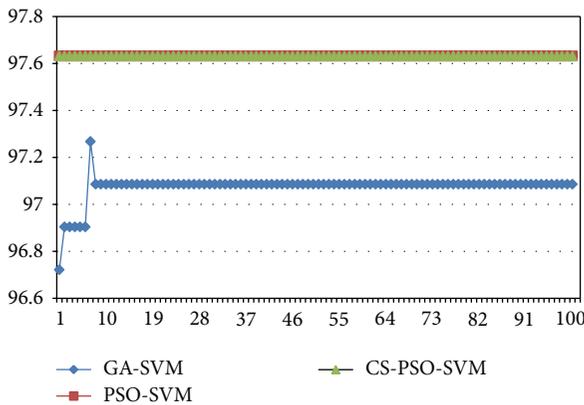


FIGURE 5: Population best fitness value in breast cancer dataset.

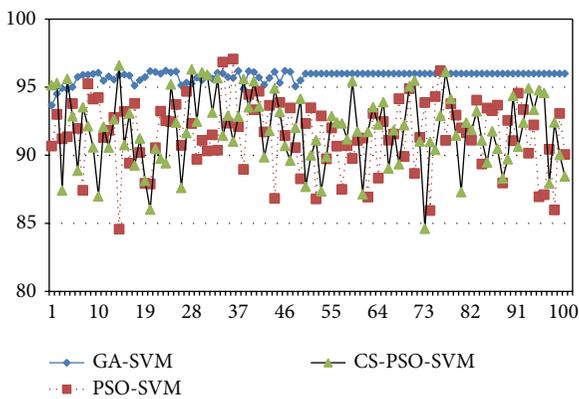


FIGURE 6: Population average fitness value in breast cancer dataset.

For the breast cancer dataset, in the test subset, the accuracy of CS-PSO-SVM is 91.333% and PSO-SVM and GA-SVM are 90%. The results of empirical analysis showed that the predictive ability of all the models is acceptable. However, the CS-PSO-SVM results outperformed the other methods. Therefore, CS-PSO-SVM is a more effective model to predict disease in two datasets.

4. Conclusion

In the last few decades, several disease diagnosis models have been developed for the disease prediction. The objective

of disease diagnosis models is to make a definite diagnosis from patients' laboratory sheet early as soon as possible and initiate timely treatment. Accurate diagnosis has an important meaning for human health. In this paper, we design a new disease diagnosis model, CS-PSO-SVM. The experimental research results show that the novel algorithm is better than GA-SVM and PSO-SVM.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors would like to thank anonymous reviewers for their constructive and enlightening comments, which improved the paper. This work has been supported by Grants from Program for Excellent Youth Scholars in Universities of Guangdong Province (Yq2013108). The authors are partly supported by the Key Grant Project from Guangdong Provincial Party Committee Propaganda Department, China (LLYJ1311).

References

- [1] A. Subasi and M. Gursay, "EEG signal classification using PCA, ICA, LDA and support vector machines," *Expert Systems with Applications*, vol. 37, no. 12, pp. 8659–8666, 2010.
- [2] A. Plaza, J. A. Benediktsson, J. W. Boardman et al., "Recent advances in techniques for hyperspectral image processing," *Remote Sensing of Environment*, vol. 113, no. 1, pp. S110–S122, 2009.
- [3] I. A. Illán, J. M. Górriz, M. M. López et al., "Computer aided diagnosis of Alzheimer's disease using component based SVM," *Applied Soft Computing*, vol. 11, no. 2, pp. 2376–2382, 2011.
- [4] J. S. Sartakhti, M. H. Zangoeei, and K. Mozafari, "Hepatitis disease diagnosis using a novel hybrid method based on support vector machine and simulated annealing (SVM-SA)," *Computer Methods and Programs in Biomedicine*, vol. 108, no. 2, pp. 570–579, 2012.
- [5] J. Ramírez, J. M. Górriz, D. Salas-Gonzalez et al., "Computer-aided diagnosis of Alzheimer's type dementia combining support vector machines and discriminant set of features," *Information Sciences*, vol. 237, pp. 59–72, 2013.

- [6] M. J. Abdi and D. Giveki, "Automatic detection of erythematous-squamous diseases using PSO-SVM based on association rules," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 1, pp. 603–608, 2013.
- [7] X.-S. Yang and S. Deb, "Cuckoo search: recent advances and applications," *Neural Computing and Applications*, vol. 24, no. 1, pp. 169–174, 2014.
- [8] Tristan Fletcher, "Support Vector Machines Explained," 2014, <http://www.tristanfletcher.co.uk/SVM%20Explained.pdf>.
- [9] A. Reyaz-Ahmed, Y.-Q. Zhang, and R. W. Harrison, "Granular decision tree and evolutionary neural SVM for protein secondary structure prediction," *International Journal of Computational Intelligence Systems*, vol. 2, no. 4, pp. 343–352, 2009.
- [10] W. An, C. Angulo, and Y. Sun, "Support vector regression with interval-input interval-output," *International Journal of Computational Intelligence Systems*, vol. 1, no. 4, pp. 299–303, 2008.
- [11] V. J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Joint Conference on Neural Networks*, vol. 4, pp. 1942–1948, December 1995.
- [12] J. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm Intelligence*, Morgan Kaufmann, 2002.
- [13] Y. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 69–73, May 1998.
- [14] F. Ardjani and K. Sadouni, "Optimization of SVM multiclass by particle swarm (PSO-SVM)," *International Journal of Modern Education and Computer Science*, vol. 2, no. 2, pp. 32–38, 2010.
- [15] Y. Ren and G. Bai, "Determination of optimal SVM parameters by using GA/PSO," *Journal of Computers*, vol. 5, no. 8, pp. 1160–1168, 2010.
- [16] http://en.wikipedia.org/wiki/Cuckoo_search.
- [17] I. Fister Jr., D. Fister, and I. Fister, "A comprehensive review of cuckoo search: variants and hybrids," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 4, no. 4, pp. 387–409, 2013.
- [18] I. Fister Jr., X.-S. Yang, D. Fister, and I. Fister, "Cuckoo search: a brief literature review," in *Cuckoo Search and Firefly Algorithm*, X.-S. Yang, Ed., vol. 516 of *Studies in Computational Intelligence*, pp. 49–62, 2014.
- [19] R. B. Payne, M. D. Sorenson, and K. Klitz, *The Cuckoos*, Oxford University Press, 2005.
- [20] <http://www.scientificcomputing.com/news-DA-Novel-Cuckoo-Search-Algorithm-Beats-Particle-Swarm-Optimization-060110.aspx>.
- [21] UCI Machine Learning Repository, <http://archive.ics.uci.edu/ml>.

Research Article

An Island Grouping Genetic Algorithm for Fuzzy Partitioning Problems

S. Salcedo-Sanz,¹ J. Del Ser,² and Z. W. Geem³

¹ Department of Signal Processing and Communications, Universidad de Alcalá, 28871 Madrid, Spain

² OPTIMA Area, Tecnalia Research & Innovation, 48170 Bizkaia, Spain

³ Department of Energy IT, Gachon University, Seongnam 461-701, Republic of Korea

Correspondence should be addressed to Z. W. Geem; geem@gachon.ac.kr

Received 1 March 2014; Accepted 2 May 2014; Published 22 May 2014

Academic Editor: Xin-She Yang

Copyright © 2014 S. Salcedo-Sanz et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper presents a novel fuzzy clustering technique based on grouping genetic algorithms (GGAs), which are a class of evolutionary algorithms especially modified to tackle grouping problems. Our approach hinges on a GGA devised for fuzzy clustering by means of a novel encoding of individuals (containing elements and clusters sections), a new fitness function (a superior modification of the Davies Bouldin index), specially tailored crossover and mutation operators, and the use of a scheme based on a local search and a parallelization process, inspired from an island-based model of evolution. The overall performance of our approach has been assessed over a number of synthetic and real fuzzy clustering problems with different objective functions and distance measures, from which it is concluded that the proposed approach shows excellent performance in all cases.

1. Introduction

Clustering (also known as partitioning) is an important subgroup of unsupervised learning techniques which hinges on grouping data objects into groups of *disjoint* (“crisp”) clusters [1–3]. A huge amount of key problems in science, engineering, and economics (e.g., bioengineering, telecommunications, energy, and risk assessment) can be formulated as clustering problems [4–8]. In this context, an important line of research related to clustering stems from the fact that, in some problems, the clusters intrinsically overlap with each other and, consequently, conventional crisp clustering algorithms are *not* suitable for dealing with this overlap [9, 10]. In these cases when an object can “partially” belong to different groups, *fuzzy clustering* algorithms have been proposed as a powerful methodology in recent years, more flexible than traditional crisp approaches and with excellent results in different real problems [11, 12].

To be specific, “fuzzy clustering” is the class of clustering problems where the boundary between clusters is ill-defined, in the sense that a given sample is allowed to belong to different clusters. As such, the notion of *fuzziness* becomes relevant since any object of the data set is assigned to a given cluster with some *membership grade*, usually set between 0

and 1 (low and high membership grade, resp.). Formally, if $X = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ is a set of N data vectors in a given observation space \mathcal{S} , the goal of a fuzzy clustering algorithm is to find a partition of X in a finite number of k clusters, so that a data vector \mathbf{x}_j can belong to a cluster $C_i \in \{C_1, \dots, C_k\}$ with a degree of membership $u_{ij} \in [0, 1]$. This is equivalent to finding a “partition matrix” \mathbf{U} whose elements $u_{ij} \in [0, 1]$ (with $1 \leq i \leq k$ and $1 \leq j \leq N$) fulfill

$$\begin{aligned} \sum_{i=1}^k u_{ij} &= 1, \quad \forall j \in \{1, \dots, N\}, \\ 0 &< \sum_{j=1}^N u_{ij} < N, \quad \forall i \in \{1, \dots, k\}. \end{aligned} \quad (1)$$

Among the different techniques applied to fuzzy clustering that can be found in the literature, we focus on those based on the fuzzy C-means (FCM) algorithm [13], kernel methods [14, 15], statistical methods [16], clonal selection theory [17], rule-based clustering [18–20], and many different heuristic and metaheuristic approaches [21–25]. Metaheuristic algorithms have been thoroughly applied to fuzzy clustering in the last years due to their superior properties of robustness and convergence to near-optimal solutions at a moderate

computational cost. Many of these approaches are based on evolutionary variants of the C-means algorithm [26, 27] or simply on direct fuzzy clustering algorithms based on genetic and evolutionary approaches [28–31], multiobjective algorithms [32], differential evolution [33], particle swarm metaheuristics [34], or evolutionary programming approaches [35].

However, despite the research activity invested on different metaheuristic approaches applied to fuzzy clustering, several avant-garde algorithms have not been explored yet in their entirety for fuzzy clustering problem. Specifically, this paper proposes a grouping genetic algorithm for fuzzy clustering. The grouping genetic algorithm (GGA) [36, 37] is a class of evolutionary algorithms whose encoding procedure is especially designed to deal with grouping problems. It has been successfully applied to a variety of problems involving grouping of items but, surprisingly, its performance has not been assessed yet when tackling fuzzy clustering problems. For this purpose, this paper builds upon preliminary work in [38] by presenting a novel grouping encoding, a modified objective function, and crossover and mutation operators specifically adapted to fuzzy clustering problems tackled via GGA heuristics. In order to further enhance the performance of the grouping genetic approach the proposed scheme also incorporates a local search stage and a parallelization of the GGA using the well-known *island* model, which can be both considered as additional novel ingredients with respect to [38]. Simulation results are presented so as to assess the performance of the proposed scheme in a number of application scenarios, based on which it is concluded that the GGA-based procedure here presented outperforms conventional fuzzy C-means methods.

The rest of this paper is structured as follows. For keeping the paper self-contained, Section 2 summarizes some key mathematical concepts to define clustering algorithms, such as different definitions of distance and objective functions. Section 3 presents the aforementioned proposed GGA to tackle fuzzy clustering problems, along with a full description of its novel encoding, objective function, operators, local search, and parallelization approach. Section 4 discusses the performance of the proposed approach in a variety of different synthetic and real problems. Finally, Section 5 completes the paper by discussing some concluding remarks.

2. Background: Fuzzy Clustering Concepts

The classification of objects into clusters aims at grouping those that are similar. The extent to which two objects are similar to each other must be quantified by using an appropriate distance measure. In this regard, Section 2.1 discusses some different definitions for distances in fuzzy clustering. The second key concept, strongly related to the first one and outlined in Section 2.2, aims at evaluating the quality of a candidate solution under test in a fuzzy clustering problem and plays a key role in the GGA described in Section 3.

2.1. Distances in Fuzzy Clustering. The adequate definition of the aforementioned distances plays a central role in fuzzy clustering. For instance, a norm based on Mahalanobis

distance can be used with a similar definition compared to that for the crisp clustering case; namely,

$$d_M^2(\mathbf{x}_j, \boldsymbol{\mu}_i) = \|\mathbf{x}_j - \boldsymbol{\mu}_i\|_{\Sigma_i^{-1}} = (\mathbf{x}_j - \boldsymbol{\mu}_i) \cdot \Sigma_i^{-1} \cdot (\mathbf{x}_j - \boldsymbol{\mu}_i)^T, \quad (2)$$

though, in this case, the definition of the inverse of the covariance matrix of any cluster Σ_i is slightly different and is given by

$$\Sigma_i^{-1} = \frac{\sum_{j=1}^N u_{ij}^\alpha \cdot (\mathbf{x}_j - \boldsymbol{\mu}_i) \cdot (\mathbf{x}_j - \boldsymbol{\mu}_i)^T}{\sum_{j=1}^N u_{ij}^\alpha}. \quad (3)$$

An alternative to Mahalanobis distance, more suitable for fuzzy clustering, is the Gustafson-Kessel (GK) distance [39]. This distance metric is defined as

$$d_{\text{GK}}^2(\mathbf{x}_j, \boldsymbol{\mu}_i) = \|\mathbf{x}_j - \boldsymbol{\mu}_i\|_{|\Sigma_i|^{1/d} \Sigma_i^{-1}} = (\mathbf{x}_j - \boldsymbol{\mu}_i) \cdot |\Sigma_i|^{1/d} \cdot \Sigma_i^{-1} \cdot (\mathbf{x}_j - \boldsymbol{\mu}_i)^T \quad (4)$$

and allows for the consideration of elliptic clusters with different orientations. However, this distance is not able to distinguish between different cluster sizes. To circumvent this drawback, a modification of this distance was proposed in [39] in the context of the adaptive fuzzy clustering (AFC) algorithm presented therein; that is,

$$d_{\text{AFC}}^2(\mathbf{x}_j, \boldsymbol{\mu}_i) = (\mathbf{x}_j - \boldsymbol{\mu}_i) \cdot \lambda_{i,d} \cdot \Sigma_i^{-1} \cdot (\mathbf{x}_j - \boldsymbol{\mu}_i)^T, \quad (5)$$

where $\lambda_{i,d}$ is a novel adaptive term associated with the smallest eigenvalue of the i th cluster's covariance matrix Σ_i^{-1} and $\boldsymbol{\mu}_i$ is the centroid of those objects the centroid of those objects belonging to cluster C_i . By using this definition, any clustering algorithm will have the chance of locating clusters with different orientation and also with different volumes.

Using a proper definition for distance plays a key role when evaluating to what extent an algorithm solves accurately the problem at hand. Exploring different functions for fuzzy clustering evaluation is thus the goal of the following section.

2.2. Fuzzy Clustering Evaluation. The evaluation of a given solution in a fuzzy clustering problem can be carried out using two different strategies. First, it is possible to directly evaluate the fuzzy clusters produced by the algorithm at hand by using the membership functions of the different observations of the problem. A second strategy consists of using a defuzzification process, prior to the clustering evaluation, followed by the application of any of the crisp clustering evaluation measures described below. As in the crisp clustering case, evaluation measures can be unsupervised or supervised. In the first case, direct evaluation is usually applied, whereas in the second one a defuzzification is often required, since existing labeled data are usually crisp.

2.2.1. Unsupervised Evaluation. For comparison purposes with the objective (fitness) function later proposed for evaluating the performance of the algorithm, we summarize herein

some of the most used unsupervised measures in the related literature.

(i) Fuzzy sum of quadratic errors (fSSE): consider

$$fSSE(\mathbf{U}) = \sum_{i=1}^k \sum_{x \in C_i} u_{ij}^\alpha d^2(\mathbf{x}_j, \boldsymbol{\mu}_i), \quad (6)$$

where $\alpha \in [1, \infty)$ controls the fuzziness degree of the solution; that is, values of α close to 1 lead the solution to a disjoint partition, whereas large values of α lead to more fuzzy clusters. Usually a value of $\alpha = 2$ is selected.

(ii) Xie-Beni index (XB): defined in [40], this measure combines the sum of square errors with a term for measuring clusters separation:

$$XB(\mathbf{U}) = \frac{\sum_{i=1}^k \sum_{j=1}^N u_{ij}^\alpha \cdot d^2(\mathbf{x}_j, \boldsymbol{\mu}_i)}{N \cdot \min_{1 \leq i, j \leq k, i \neq j} \{d^2(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j)\}}. \quad (7)$$

(iii) Fukuyama-Sugeno index (FS): the FS index [41] is similar to the XB index but, in this case, the separation between clusters is evaluated with respect to the average centroid of the k clusters, $\boldsymbol{\mu}^* = (1/k) \sum_{i=1}^k \boldsymbol{\mu}_i$, instead of the centroid of the rest of clusters. Based on this rationale we obtain

$$FS(\mathbf{U}) = \sum_{i=1}^k \sum_{j=1}^N u_{ij}^\alpha \cdot d^2(\mathbf{x}_j, \boldsymbol{\mu}_i) - \sum_{i=1}^k \sum_{j=1}^N u_{ij}^\alpha \cdot d^2(\boldsymbol{\mu}_i, \boldsymbol{\mu}^*). \quad (8)$$

The aforementioned unsupervised measures are useful in those problems in which there is no additional information to check the quality of the generated clusters. However, there are some clustering problems in which such information is indeed available, hence allowing for supervised measures.

2.2.2. Utilized Supervised Measurement: Rand Index. Among the supervised measures—sometimes called *external* measures—in this work the well-known Rand index (R) [42] will be utilized after defuzzification of the samples. It computes the similarity between the obtained partition and the *known* optimal solution as follows:

$$R(\mathbf{U}) = \frac{TP + FN}{TP + FP + TN + FN}, \quad (9)$$

where TP and FP are the number of correct and incorrect assignments, respectively, when the decision consists of assigning two elements to the *same* cluster; and TN and FN are the number of correct and incorrect assignments, respectively, when the decision consists of assigning two elements

to *different* clusters. In other words, it is a measure of the percentage of correct decisions taken by the algorithm. Note that the value of R lies on the interval $[0, 1]$: values of R closer to 1 indicate a better quality of the solution tested.

3. Proposed Grouping Genetic Algorithm for Fuzzy Clustering

As mentioned in Section 1, the grouping genetic algorithm is a class of evolutionary algorithms whose encoding strategy is especially designed to tackle grouping problems. It was first proposed by Falkenauer [36, 37], who realized that traditional genetic algorithms had difficulties when applied to grouping problems. In GGA the encoding procedure and crossover and mutation operators of traditional GAs are modified to yield a compact algorithm, with improved performance in grouping-based problems. In light of their outperforming behavior with respect to its traditional counterparts, grouping genetic algorithms have so far been successfully applied to diverse problems [43–51], including crisp clustering [52]. This paper joins the upsurge of research gravitating on GGAs by adapting this heuristic to *fuzzy* clustering problems. This section discusses several modifications we have devised towards further enhancing the performance of GGAs in fuzzy clustering, including our modifications in the encoding process, the objective function, and the crossover and mutation operators (Sections 3.1, 3.2, 3.4, and 3.5, resp.).

3.1. Problem Encoding. The proposed GGA for fuzzy clustering is a variable-length genetic algorithm, with a novel encoding to deal with this specific problem. The encoding is carried out by splitting each chromosome in the algorithm (or equivalently, its corresponding individual or candidate solution) into two parts: $\mathbf{c} = [\mathbf{U} \mid \mathbf{g}]$. The first part is the *element* section composed by the partition matrix \mathbf{U} , whereas the second part is denoted as the *group* section of the individual. Following this notation, a certain individual for a fuzzy clustering problem with N elements (objects or observations) and k clusters can be expressed as

$$\left[\begin{array}{cccc|cccc} u_{1,1}, \dots, u_{1,N} & & & & g_1, g_2, \dots, g_k \\ \vdots, \dots, \vdots & & & & & & & \\ u_{k,1}, \dots, u_{k,N} & & & & & & & \end{array} \right], \quad (10)$$

where it is important to note that each element $u_{i,j}$ represents the degree of membership of j th observation to i th cluster, whereas the group section keeps a list of tags associated with each of the clusters of the solution. Also observe that in this encoding, both the group and the element section have a variable length, since the number of clusters is also a variable of the problem. For the sake of clarity, let us assume the following individual:

$$\left[\begin{array}{cccccccccccc|cccc} 0.6 & 0.0 & 0.0 & 0.8 & 0.0 & 1.0 & 0.6 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.4 & 1.0 \\ 0.0 & 0.0 & 1.0 & 0.2 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.2 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.0 & 0.9 & 0.0 & 0.0 & 0.8 & 0.0 & 0.4 & 0.0 \\ 0.4 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 0.3 & 0.0 & 0.1 & 0.0 & 0.0 & 0.2 & 1.0 & 0.0 & 0.0 \end{array} \right] | 1, 2, 3, 4. \quad (11)$$

This chromosome encodes an individual (candidate solution) for a simple clustering problem with $N = 15$ objects: $X = \{\mathbf{x}_1, \dots, \mathbf{x}_{15}\}$. Note that the *group section* encodes a solution with 4 clusters, labeled “1,” “2,” “3,” and “4,” respectively. Any of the columns in the element section indicates to what extent any object \mathbf{x}_j belongs to a cluster C_i , that is, the partition matrix element u_{ij} . For instance, the first column in the element section encodes a candidate fuzzy solution in which the object \mathbf{x}_1 belongs to cluster C_1 with a degree of membership $u_{1,1} = 0.6$ and belongs to C_4 with $u_{4,1} = 0.4$. Keeping this in mind, the aforementioned chromosome encodes an individual that represents a solution with 4 clusters, where observations $x_2, x_3, x_5, x_6, x_8, x_{10}, x_{11}, x_{13}$, and x_{15} belong to a single cluster, observations x_1, x_4, x_9 , and x_{12} belong to two different clusters with different degrees of membership, and finally observations x_7 and x_{14} belong to three different clusters.

3.2. Objective Function. The proposed GGA will be run with different objective (fitness) functions to lead the search. Specifically, and for comparative purposes, we will use some of the classical objective functions for fuzzy clustering summarized in Section 2.2. In addition, in this paper we propose an adaptation of the well-known Davis-Bouldin index (used in crisp clustering problems) to the fuzzy case which, to the best of our knowledge, is novel in fuzzy clustering. We will show that the use of this modified index renders better results for the GGA than the other existing evaluation indices. The idea of the Davis-Bouldin index [53] for crisp clustering problems is to minimize the intracluster distances while simultaneously maximizing the distances among the different clusters, yielding

$$DB(\mathbf{U}) = \frac{1}{k} \sum_{i=1}^k \max_{i \neq j} \left\{ \frac{\sum_{x \in C_i} d^2(\mathbf{x}, \boldsymbol{\mu}_i) + \sum_{x \in C_j} d^2(\mathbf{x}, \boldsymbol{\mu}_j)}{d^2(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j)} \right\}. \tag{12}$$

In the above expression note that small values of the conventional DB index correspond to compact and well-separated clusters. The adaptation of the DB index for fuzzy clustering proposed in this work is expressed as

$$MDB(\mathbf{U}, d) = \frac{1}{k} \sum_{i=1}^k \max_{i \neq j} \left\{ \frac{\sum_{t=1}^N u_{i,t}^\alpha d^2(\mathbf{x}_t, \boldsymbol{\mu}_i) + \sum_{t=1}^N u_{j,t}^\alpha d^2(\mathbf{x}_t, \boldsymbol{\mu}_j)}{d^2(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j)} \right\}, \tag{13}$$

where $\boldsymbol{\mu}_i$ stands for the centroid associated with cluster C_i , calculated by considering the average of each observation weighted by the degree of membership to cluster C_i . Note in expression (13) that the proposed MDB index explicitly depends on the particular definition considered for the distance d . For example, if we consider the GK distance and

based on the covariance matrices of the clusters, the DB index for fuzzy clustering problems will be given by

$$MDB(\mathbf{U}, d_{GK}) = \frac{1}{k} \sum_{i=1}^k \max_{i \neq j} \left\{ \frac{\sum_{t=1}^N u_{i,t}^\alpha d_{|\Sigma_i|^{1/d} \Sigma_i^{-1}}^2(\mathbf{x}_t, \boldsymbol{\mu}_i) + \sum_{t=1}^N u_{j,t}^\alpha d_{|\Sigma_j|^{1/d} \Sigma_j^{-1}}^2(\mathbf{x}_t, \boldsymbol{\mu}_j)}{\min \left\{ d_{|\Sigma_i|^{1/d} \Sigma_i^{-1}}^2(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j), d_{|\Sigma_j|^{1/d} \Sigma_j^{-1}}^2(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j) \right\}} \right\}. \tag{14}$$

3.3. Selection Operator. In this paper we use a rank-based wheel selection mechanism, similar to the one described in [44]. First, the individuals are sorted in a list based on their quality. The position of the individuals in the list is called *rank of the individual* and is denoted as R_i ($i = 1, \dots, \xi$, with ξ standing for the number of individuals in the population of the GGA). A rank to which the best individual x is assigned will be $R_x = \xi$, whereas the second best will be y , $R_y = \xi - 1$, and so forth. A *fitness* value associated with each individual is then defined as

$$f_i = \frac{2 \cdot R_i}{\xi \cdot (\xi + 1)}. \tag{15}$$

Note that these values are normalized between 0 and 1, depending on the position of the individual in the ranking list. It is important to note that this rank-based selection mechanism is *static*, in the sense that probabilities of survival (given by f_i) do not depend on the generation but on the position of the individual in the list. As a toy example, consider a population formed by 5 individuals, in which individual 1 is the best quality one ($R_1 = 5$), individual 2 is the second best ($R_2 = 4$), and so on. In this case, the fitness associated with the individuals is $\{0.33, 0.26, 0.2, 0.13, 0.06\}$, and the associated intervals for the roulette wheel are $\{0-0.33, 0.34-0.6, 0.61-0.8, 0.81-0.93, 0.94-1\}$.

The process carried out in our algorithm consists of selecting the *parents* for crossover by using this selection mechanism. This procedure is performed *with* replacement; that is, a given individual can be selected several times as one of the parents. However, individuals in the crossover operator must be different.

3.4. Crossover Operator. The crossover operator implemented in the grouping genetic algorithm used in this paper is a modified version of the one initially proposed by Falkenauer in [36], but with the added bonus of being adapted to the fuzzy clustering problem. These are the main steps followed in the crossover operation.

- (1) Select two individuals at random and choose two crossing points in their group part.
- (2) Insert the elements belonging to the selected groups of the first individual into the offspring.
- (3) Assign the degree of membership of the inserted elements equal to the first individual.
- (4) Insert the elements belonging to the selected groups of the second individual into the offspring.

- (5) Assign the degree of membership of the inserted elements in the following way. First, the remaining degree membership after the assignment of the elements of the first individual is calculated. This remaining degree membership is then proportionally shared among the elements of the second individual.
- (6) Remove empty clusters, if any.

- (7) Modify the labels of the current groups in the offspring in order to numerate them from 1 to k .

A simple yet illustrative enough example follows. Let us consider two different individuals ξ_1 and ξ_2 that have been randomly chosen among all individuals in a given GGA population so as to perform crossover on them. The groups selected to carry out the procedure are marked in boldface:

$$\xi_1 = \begin{bmatrix} 0.6 & 0.0 & 0.0 & 0.8 & 0.0 & 1.0 & 0.6 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.4 & 1.0 \\ 0.0 & 0.0 & 1.0 & 0.2 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.2 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.0 & 0.9 & 0.0 & 0.0 & 0.8 & 0.0 & 0.4 & 0.0 \\ 0.4 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 0.3 & 0.0 & 0.1 & 0.0 & 0.0 & 0.2 & 1.0 & 0.0 & 0.0 \end{bmatrix} \mid 1, 2, 3, 4, \tag{16}$$

$$\xi_2 = \begin{bmatrix} 0.8 & 0.4 & 0.0 & 0.0 & 0.0 & 0.3 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.6 & 1.0 & 0.7 & 0.0 \\ 0.0 & 0.6 & 0.0 & 1.0 & 0.1 & 0.3 & 0.0 & 0.5 & 0.0 & 0.0 & 1.0 & 0.4 & 0.0 & 0.1 & 1.0 \\ 0.2 & 0.0 & 1.0 & 0.0 & 0.9 & 0.4 & 0.0 & 0.5 & 0.9 & 1.0 & 0.0 & 0.0 & 0.0 & 0.2 & 0.0 \end{bmatrix} \mid 1, 2, 3.$$

After steps 2 and 3 of the proposed crossover procedure (insertion of the group elements of the first individual and

assignment of the degree of membership), the offspring results in

$$O = \begin{bmatrix} 0.0 & 0.0 & 1.0 & 0.2 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.2 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.0 & 0.9 & 0.0 & 0.0 & 0.8 & 0.0 & 0.4 & 0.0 \end{bmatrix} \mid 2, 3. \tag{17}$$

Then the group elements of the second individual are inserted, and the membership degree is modified considering the previous existing degrees from individual 1:

$$O = \begin{bmatrix} 0.0 & 0.0 & 1.0 & 0.2 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.2 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.0 & 0.9 & 0.0 & 0.0 & 0.8 & 0.0 & 0.4 & 0.0 \\ 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5 & 0.9 & 0.0 & 0.0 & 0.0 & 0.0 & 0.12 & 1.0 & 0.35 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.8 & 1.0 & 0.5 & 0.0 & 0.0 & 0.1 & 0.0 & 1.0 & 0.08 & 0.0 & 0.05 & 1.0 \end{bmatrix} \mid 2, 3, 1^*, 2^*. \tag{18}$$

There are no empty clusters. Therefore, we pass on to the final step of the crossover approach: modify the labels of current

groups in the offspring in order to numerate them from 1 to k (4 in this case):

$$O = \begin{bmatrix} 0.0 & 0.0 & 1.0 & 0.2 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.2 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.1 & 0.0 & 0.9 & 0.0 & 0.0 & 0.8 & 0.0 & 0.4 & 0.0 \\ 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5 & 0.9 & 0.9 & 0.0 & 0.0 & 0.0 & 0.0 & 0.12 & 1.0 & 0.35 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.8 & 1.0 & 0.5 & 0.0 & 0.0 & 0.0 & 0.1 & 0.0 & 1.0 & 0.08 & 0.0 & 0.05 & 1.0 \end{bmatrix} \mid 1, 2, 3, 4. \tag{19}$$

An example of the reassignment of the degree of membership in the final offspring is shown in Figure 1, where the evolution of the degrees of membership is shown for observation x_{14} along the crossover operation. Intuitively the crossover should be high in the first stages of the algorithm and more moderate in the last ones in order to favor the explorative behavior of the algorithm through the search space. Thus, we

have implemented an adaptive crossover probability defined as

$$P_c(j) = P_{ci} + \frac{j}{TG} (P_{ci} - P_{cf}), \tag{20}$$

where $P_c(j)$ is the crossover probability used in a given generation j , TG stands for the total number of generations of

the algorithm, and P_{ci} and P_{cf} are the initial and final values of probability, respectively, which are set as inputs for the proposed algorithm.

3.5. *Mutation Operator.* Mutation operators include modifications in each individual of the population with a low probability in order to explore new regions of the search space and also to escape from local optima when the algorithm is near convergence. In this case, we have implemented two different mutation operators adapted to the fuzzy clustering problems.

(i) Mutation by *cluster splitting*: this operator consists of splitting a selected cluster into two different parts. The

degrees of membership are also randomly split between the two new clusters. The samples belonging to the original cluster are assigned to the new clusters with equal probability. Note that one of the new generated clusters will keep its label in the group section of the individual, whereas the other will be assigned a new label ($k + 1$). The selection for the initial cluster to be split is carried out depending on the clusters' size, with more probability of splitting imposed on clusters of larger size. As an example, we illustrate an application of this operator in the final offspring individual of the previous example:

$$O = \begin{bmatrix} 0.0 & 0.0 & 1.0 & 0.2 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.2 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.0 & 0.9 & 0.0 & 0.0 & 0.8 & 0.0 & 0.4 & 0.0 \\ 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5 & 0.9 & 0.0 & 0.0 & 0.0 & 0.0 & 0.12 & 1.0 & 0.35 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.8 & 1.0 & 0.5 & 0.0 & 0.0 & 0.1 & 0.0 & 1.0 & 0.08 & 0.0 & 0.05 & 1.0 \end{bmatrix} | 1, 2, 3, 4. \tag{21}$$

Let us suppose that the cluster chosen to be split is cluster 1. A possible cluster splitting mutation operation would be

$$O_m = \begin{bmatrix} 0.0 & 0.0 & 0.4 & 0.08 & 0.0 & 0.0 & 0.0 & 0.4 & 0.0 & 0.4 & 0.0 & 0.0 & 0.0 & 0.08 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.1 & 0.0 & 0.9 & 0.0 & 0.0 & 0.8 & 0.0 & 0.4 & 0.0 \\ 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5 & 0.9 & 0.0 & 0.0 & 0.0 & 0.0 & 0.12 & 1.0 & 0.35 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.8 & 1.0 & 0.5 & 0.0 & 0.0 & 0.1 & 0.0 & 1.0 & 0.08 & 0.0 & 0.05 & 1.0 \\ 0.0 & 0.0 & 0.6 & 0.12 & 0.0 & 0.0 & 0.0 & 0.6 & 0.0 & 0.6 & 0.0 & 0.0 & 0.0 & 0.12 & 0.0 \end{bmatrix} | 1, 2, 3, 4, 5. \tag{22}$$

(ii) Mutation by *clusters merging*: this mutation consists of randomly selecting two existing clusters and merging them into just one single cluster. The degree of membership of the new cluster is the sum of the degrees of the previous ones. As in mutation by cluster splitting, the probability of choosing

the clusters depends on their size. In order to illustrate this mutation, we use again the final offspring from the crossover operator example. In this case, let us consider that the selected clusters to be merged are clusters 2 and 4, resulting in

$$O_m = \begin{bmatrix} 0.0 & 0.0 & 1.0 & 0.2 & 0.0 & 0.0 & 0.0 & 1.0 & 0.0 & 1.0 & 0.0 & 0.0 & 0.0 & 0.2 & 0.0 \\ 0.0 & 1.0 & 0.0 & 0.8 & 1.0 & 0.5 & 0.1 & 0.0 & 1.0 & 0.0 & 1.0 & 0.88 & 0.0 & 0.45 & 1.0 \\ 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.5 & 0.9 & 0.0 & 0.0 & 0.0 & 0.0 & 0.12 & 1.0 & 0.35 & 0.0 \end{bmatrix} | 1, 2, 3. \tag{23}$$

Analogously to the crossover operator, we also consider an adaptive version of the probability of applying the mutation operators described above. Note that we apply the two mutation operators in a serial fashion (one after the other), with independent probabilities of application. In this case, probability of mutation is made smaller in the first generations of the algorithm and larger in the last ones in order to have more opportunities to escape from local minima in the last stages of the evolutionary process; that is,

$$P_m(j) = P_{mi} + \frac{j}{TG} (P_{mf} - P_{mi}), \tag{24}$$

where $P_m(j)$ is the probability of mutation used in a given generation j , TG stands for the total number of generations of the algorithm, and P_{mf} and P_{mi} are the final and initial values of probability considered, respectively.

3.6. *Replacement and Elitism.* In the proposed GGA, the population at a given generation $j + 1$ is obtained by replacement of the individuals in the population at generation j , through the application of the selection, crossover, and mutation operators described above. An elitist scheme is also applied: the best individual in generation j is automatically

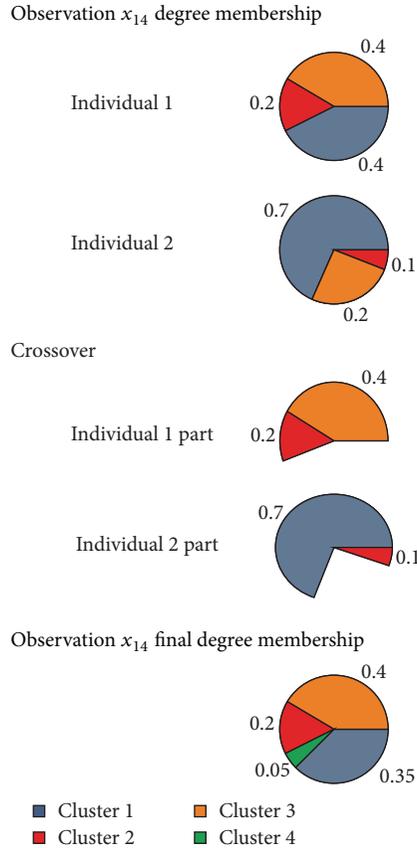


FIGURE 1: Example of the crossover operator implemented in the proposed grouping genetic algorithm for fuzzy clustering problems.

passed on to the population of generation $j + 1$, ensuring that the best solution encountered so far in the evolution is always kept by the algorithm.

3.7. Local Search. We use a local search procedure to try to find local optima in a close neighborhood of a given individual. The proposed local search is based on minor modifications of the current individual, as far as they produce an increase of the associated objective function: the local search changes the degree of membership of the observations, starting by one randomly chosen. The changes in the degree of membership are randomly generated. We finally keep the assignment with the largest objective function. Since this local search procedure is a time-consuming operation, it is applied to a given individual with a small probability, p_b , that is modified between an initial and final value in the algorithm in the same way that the crossover probability is modified.

3.8. An Island Model to Improve the Algorithm's Performance. In order to improve the performance of the proposed GGA, an island model is considered for its parallelization. In this context, \mathcal{S} subpopulations (islands) are set in such a way that the evolution in each island is forced to be independent but the migration of good individuals is allowed between islands. We consider an elitist migration model, in which only

TABLE 1: GGA parameters values used in the experiments of the paper.

Parameter	Meaning	Value
P_s	Population size	20
S	Number of subpopulations	4
TG	Maximum number of generations	400
P_{ci}	Initial crossover probability	0.8
P_{cf}	Final crossover probability	0.6
P_{mi}	Initial mutation probability	0.05
P_{mf}	Final crossover probability	0.1
P_{bi}	Initial local search probability	0.1
P_{bf}	Final local search probability	0.05
P_e	Probability of migrating (islands model)	0.03
α	Fuzziness degree	2

the best individual in each island migrates and substitutes a randomly chosen individual in one of the other islands. There is a probability of migration p_e predefined in the algorithm. The migration process is summarized in the following steps.

- (1) Choose the best individual in each island.
- (2) Randomly choose the island toward which each individual will migrate.
- (3) Randomly choose an individual in the destiny island and change it by the migrating individual.

4. Experiments and Results

This section summarizes and discusses the experimental work we have carried out in order to assess the performance of our proposed GGA approach. We have explored a number of variations of the proposed GGA (by combining different distances and/or objective functions) in a variety of fuzzy clustering scenarios (which, as will be shown later, exhibit an increasing degree of complexity). Table 1 lists the values of the GGA parameters used in all the simulations carried out in this paper. These values have been found to be the most appropriate after a number of side experiments, not shown for the sake of brevity. The algorithm presented here is compared with the fuzzy C-means (FCM) [13] algorithm because it has been successfully applied to many real clustering problems and applications characterized by different levels of complexity [26, 27].

For reasons made clearer in what follows, the experimental setup for comparing the considered algorithms will be divided into two different parts, characterized by using synthetic and real data (Sections 4.1 and 4.2, resp.).

4.1. Synthetic Data

4.1.1. Experiment 1 with Synthetic Data: Spherical Clusters. In this first experiment, we test the performance of the proposed GGA in a two-dimensional clustering problem, defined by 300 observations randomly generated using a Gaussian distribution from 8 equiprobable classes, with mean values $\mu_1 = (-1, 1)$, $\mu_2 = (2, -2)$, $\mu_3 = (1, 0)$, $\mu_4 = (3, -1)$,

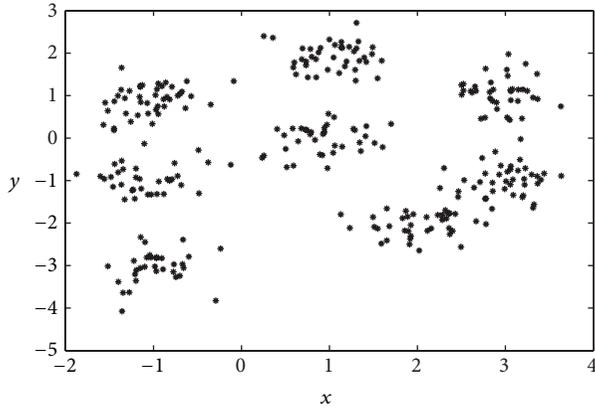


FIGURE 2: Two-dimensional representation of data for the first synthetic clustering example (spherical data). See the main text for further details.

$\mu_5 = (-1, -1)$, $\mu_6 = (-1, -3)$, $\mu_7 = (1, 2)$, and $\mu_8 = (3, 1)$ and covariance matrices:

$$\Sigma_1 = \Sigma_2 = \dots = \Sigma_8 = \begin{bmatrix} 0.35^2 & 0 \\ 0 & 0.35^2 \end{bmatrix}. \quad (25)$$

Note that this procedure results in a problem characterized by spherical clusters. Figure 2 illustrates the two-dimensional distribution of the observations following the above statistical distribution.

We have applied to this problem a number of configurations of the proposed GGA—with MDB, XB, and FS objective (fitness) functions—and the FCM algorithm fed with the real number of clusters as a priori information. At this point it is important to emphasize that the proposed GGA is able to infer the number of clusters within the problem, whereas the FCM requires this parameter to be set before execution (namely, C in the above description of FCM). To circumvent this issue, side simulations have been run for FCM and the considered scenario by varying C over a wide range of integer values, from which the value rendering the best metric value has been selected for comparison. Also included is the GGA approach from [38] in order to assess the impact of the novel aspects of the island-based GGA proposed here.

Having said this, Table 2 lists the supervised evaluation of the results obtained by the aforementioned algorithms. Note that the proposed GGA with the three different objective functions obtains better results than the FCM algorithm. In particular, our GGA with the MDB index exhibits the best behavior ($R = 0.9937$), higher than that of the conventional FCM algorithm ($R = 0.9712$) and the GGA with MDB index from [38] ($R = 0.9918$). In addition, note that the GGA with MDB and XB indexes achieves the solution with the optimal number of clusters (i.e., 8). In order to better describe the behavior of the best algorithm (the GGA with our MBD index), it would be very interesting to have a closer look at Figures 3 to 5.

- (i) Figure 3 represents the two-dimensional distributions of the 8 clusters found. The color of each observation has been obtained as a combination of those colors

TABLE 2: Comparison of the results (in terms of the number of clusters finally found and as a function of the Rand index) obtained by the proposed GGA algorithm with MDB, XB, and FS indexes, respectively, with the previous GGA in [38] and the FCM algorithm in the first synthetic clustering problem considered.

Algorithm	Number of clusters	Rand index
Proposed GGA (MDB index)	8	0.9937
Proposed GGA (XB index)	8	0.9805
Proposed GGA (FS index)	9	0.9874
GGA from [38] (MDB index)	8	0.9918
GGA from [38] (XB index)	8	0.9785
GGA from [38] (FS index)	9	0.9847
FCM	8	0.9712

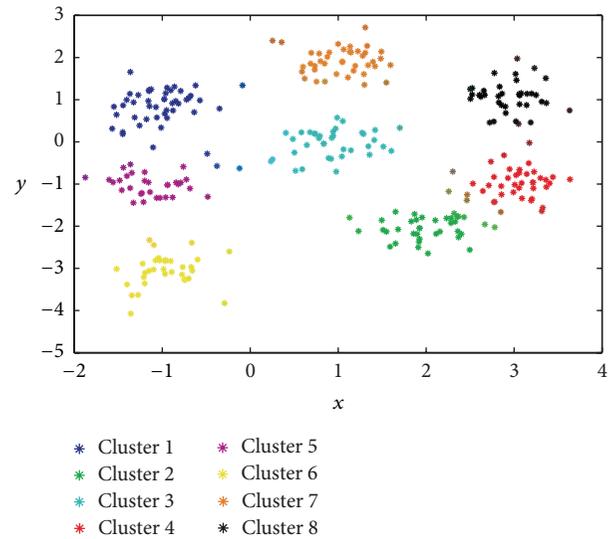


FIGURE 3: Representation of the best result obtained by the proposed GGA with MDB fitness function in the first synthetic clustering example.

representing each cluster, weighted by the degree of membership of each observation.

- (ii) Figures 4(a) and 4(b) depict, as a function of the number of generations considered, the evolution of the objective function and that of the number of clusters, respectively, in what is the best solution found for this problem. It is worth noting that the algorithm is able to isolate the 8 clusters of the data set with a value of the objective function of 9.7688.
- (iii) Finally, Figure 5 shows the final solution after the defuzzification process, illustrating the ability of the proposed algorithm to find the 8 clusters.

The question arising from this first experiment lies on how the proposed fuzzy clustering approach works when facing clusters that are not spherical or exhibiting different distributions. This is the rationale behind the following second synthetic experiment.

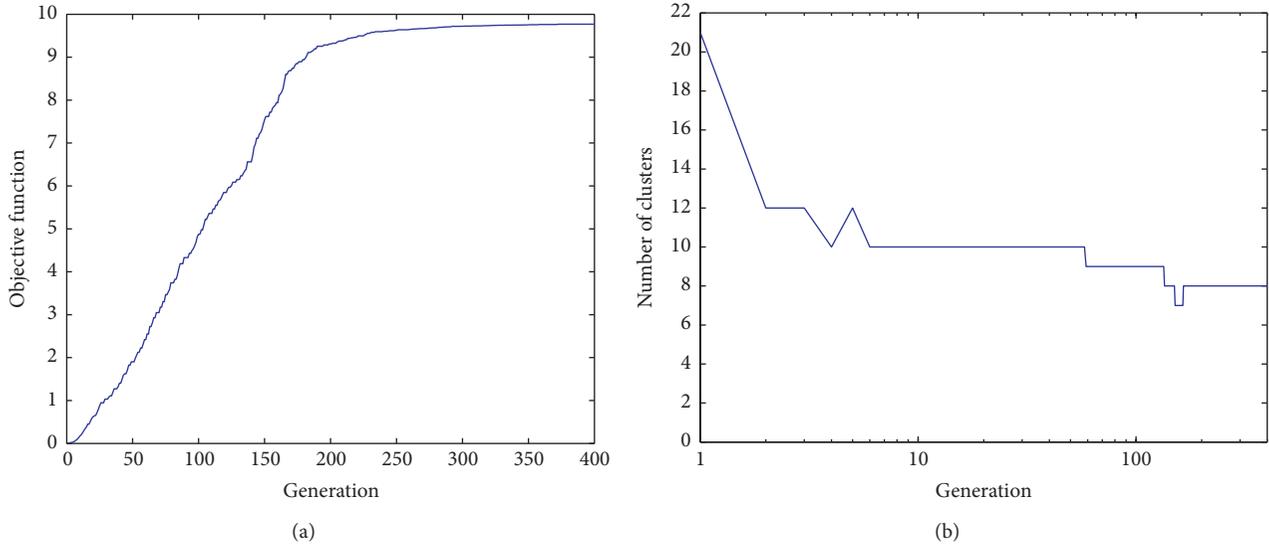


FIGURE 4: Evolution, as a function of the number of generations involved, of (a) the objective (fitness) function (MDB) and (b) the number of clusters obtained.

4.1.2. *Experiment 2 with Synthetic Data: Unbalanced Data.* We now test the performance of the proposed GGA in a different two-dimensional clustering problem, defined by 400 randomly generated objects following a distribution drawn from 3 Gaussian classes with probabilities $p_1 = 0.5$, $p_2 = 0.33$, and $p_3 = 0.17$. The mean values of each of such classes are $\mu_1 = (0, 2)$, $\mu_2 = (-1, -1)$, and $\mu_3 = (2, -1)$, whereas their covariance matrices are given by

$$\begin{aligned} \Sigma_1 &= \begin{bmatrix} 1^2 & 0 \\ 0 & 0.8^2 \end{bmatrix}, \\ \Sigma_2 &= \begin{bmatrix} 0.6^2 & 0 \\ 0 & 0.4^2 \end{bmatrix}, \\ \Sigma_3 &= \begin{bmatrix} 0.3^2 & 0 \\ 0 & 0.5^2 \end{bmatrix}. \end{aligned} \tag{26}$$

Note that, in this case, the classes are not spherical and have different distributions. Figure 6 displays the observations generated for this instance.

Table 3 shows, in terms of the Rand index, the results obtained by the proposed GGA with MDB, XB, and FS indexes and the previous scheme from [38] with the same set of indexes and those achieved by the FCM algorithm. As shown in this table, the GGA with MDB and XB indexes obtains similar results (better than the FCM), whereas the result of the GGA with FS index is slightly worse than the result of the FCM algorithm. The best results correspond to the here proposed GGA algorithm with the MDB index, rendering a value of $R = 0.9284$ (higher than that of the GGA approach from [38] with the same index and the FCM algorithm) and, what is very important, finding the 3 clusters hidden in the data. Finally, Figure 7 illustrates, in a more

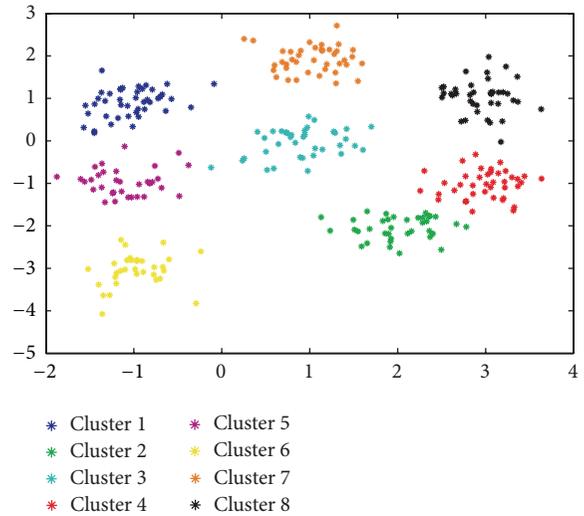


FIGURE 5: Final solution obtained by the proposed GGA (MDB index) after the defuzzification process in the first synthetic clustering problem considered.

intuitive way, the fuzzy clustering reached by the proposed GGA using the MDB index as objective function.

4.1.3. *Experiment 3 with Synthetic Data: Heterogeneous Clusters.* The goal of this final synthetic experiment consists of exploring the effects of using different distances in the MDB objective function rendering the best results obtained by the proposed GGA. We again set up another two-dimensional clustering problem defined by 300 Gaussian-distributed objects, but in this case the Gaussian distribution is randomly drawn from 6 classes with probabilities $p_1 = 0.1$, $p_2 = 0.1$, $p_3 = 0.1$, $p_4 = 0.25$, $p_5 = 0.25$, and $p_6 = 0.2$.

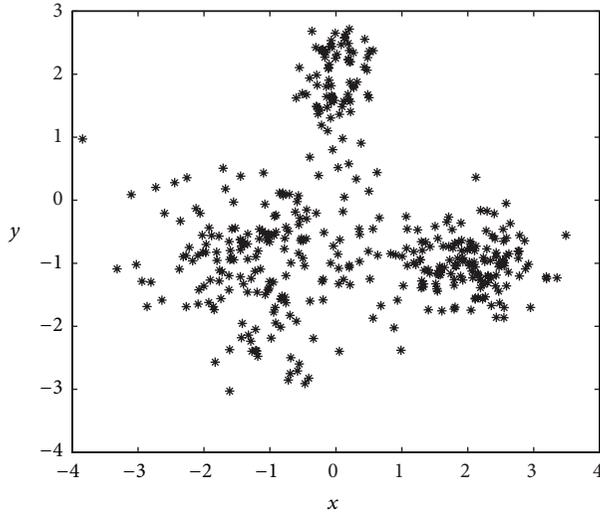


FIGURE 6: Data for the second synthetic clustering example (unbalanced data). See the main text for further details.

Means of the classes are set as $\mu_1 = (-2, -2)$, $\mu_2 = (0, -2)$, $\mu_3 = (2, -2)$, $\mu_4 = (0, 0.5)$, $\mu_5 = (-1.5, 2)$, and $\mu_6 = (2, 2.5)$, whereas the covariance matrices are selected to be

$$\begin{aligned}
 \Sigma_1 &= \begin{bmatrix} 0.3 & 0.28 \\ 0.28 & 0.3 \end{bmatrix}, \\
 \Sigma_2 &= \begin{bmatrix} 0.02 & 0 \\ 0 & 0.02 \end{bmatrix}, \\
 \Sigma_3 &= \begin{bmatrix} 0.3 & -0.28 \\ -0.28 & 0.3 \end{bmatrix}, \\
 \Sigma_4 &= \begin{bmatrix} 0.46 & 0 \\ 0 & 0.46 \end{bmatrix}, \\
 \Sigma_5 &= \begin{bmatrix} 0.2 & 0 \\ 0 & 0.2 \end{bmatrix}, \\
 \Sigma_6 &= \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}.
 \end{aligned} \tag{27}$$

For illustrative purposes, Figure 8 displays the observations generated for this instance.

The analysis we have carried out in this case consists of comparing the GGA with the MDB index as objective function (which has obtained the best results in previous experiments), but using *different* distance within MDB metrics. Specifically, we will show the effect of including Euclidean, GK, and AFC distances within the proposed GGA. Figure 9(a) represents the solution found by the GGA with MDB index and Euclidean distance. Note that the algorithm is not able to distinguish *nonspheric clusters*. By contrast, Figure 9(b) shows the result obtained by the proposed GGA with the MDB index and the GK distance. In this case, the algorithm is able to detect the structure of the problem, as can be checked out in the detection of the elliptic clusters at the bottom of the figure. Finally, Figure 9(c) shows the result obtained by the proposed GGA with MDB index and the AFC

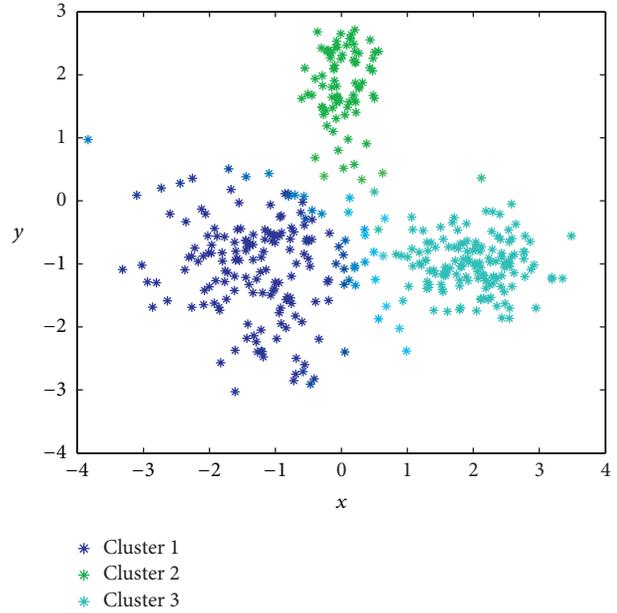


FIGURE 7: Best result obtained by the proposed GGA (MDB index) in the second synthetic clustering example.

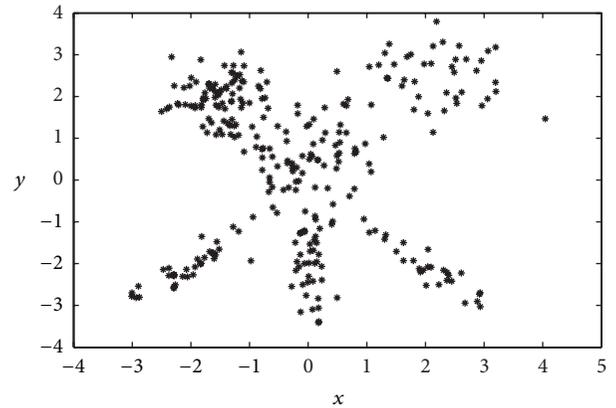


FIGURE 8: Data for the third synthetic clustering example (heterogeneous data).

TABLE 3: Comparison of the results (in terms of the number of clusters finally found and as a function of the Rand index) obtained by the proposed GGA algorithm with MDB, XB, and FS indexes, respectively, with the previous GGA in [38] and the FCM algorithm in the second synthetic clustering problem considered.

Algorithm	Number of clusters	Rand index
Proposed GGA (MDB index)	3	0.9284
Proposed GGA (XB index)	3	0.9203
Proposed GGA (FS index)	7	0.7998
GGA from [38] (MDB index)	3	0.9177
GGA from [38] (XB index)	3	0.9128
GGA from [38] (FS index)	7	0.7606
FCM	4	0.8561

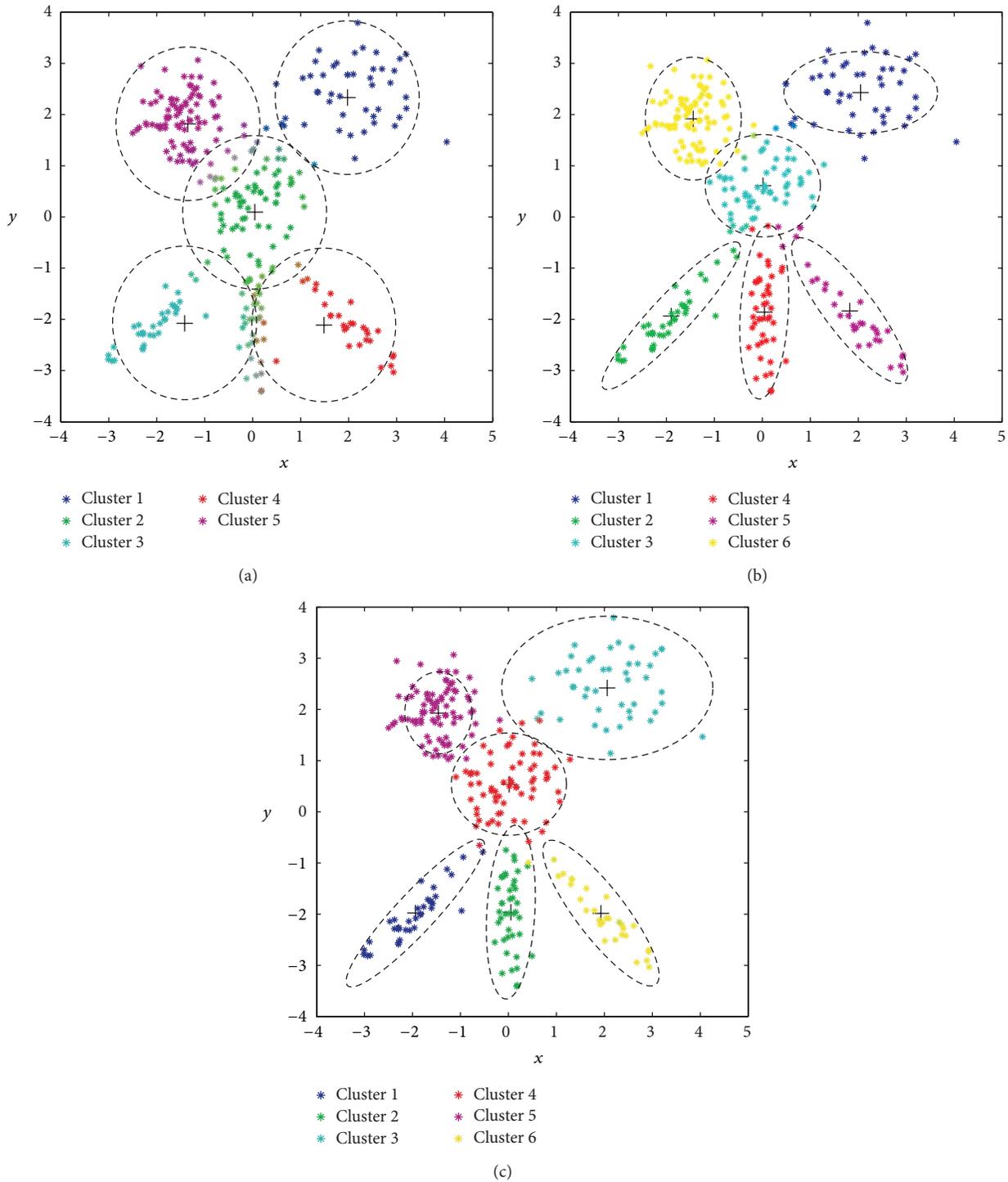


FIGURE 9: Best solutions found by the GGA with MDB index and different distances: (a) Euclidean distance; (b) GK distance; (c) AFC distance.

distance. Note that in this case the adaptive distance measure allows detecting clusters of *different sizes*, as the large ones at the topmost part of the figure.

The analysis of the GGA performance in this problem proceeds by comparing the results obtained in terms of the Rand index (supervised measure). Table 4 lists the results computed by the proposed GGA, with MDB index and

the different distances considered, compared to the results achieved by the FCM approach (with Euclidean distance, which has been found to be the best for the FCM algorithm). Note that the strategy using the proposed GGA with our MDB fitness function and the AFC distance exhibits the best performance, not only because it reaches the highest Rand index ($R = 0.9670$), but also because it properly detects the 6

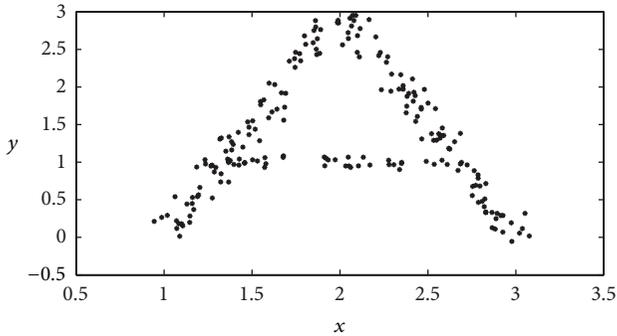


FIGURE 10: Data for the character recognition problem considered.

TABLE 4: Comparison of the results (in terms of the number of clusters finally found and as a function of the Rand index) obtained by the proposed GGA algorithm with MDB index and different distances and the FCM algorithm (with Euclidean distance, which has been found to be the best for this algorithm) in the third synthetic clustering problem considered. See the main text for further details.

Algorithm	Distance	Number of clusters	Rand index
GGA (MDB index)	Euclidean	5	0.8989
GGA (MDB index)	GK	6	0.9475
GGA (MDB index)	AFC	6	0.9670
FCM	Euclidean	6	0.9416

clusters hidden in the data. Furthermore, only the proposed GGA approach with MDB index and AFC and GK distances is able to locate the correct number of clusters in the final solution.

4.2. Real Data

4.2.1. Real Problem 1: Character Recognition. This problem can be stated as follows: let \mathcal{F} be a character, a two-dimensional image, in which each pixel, I_{ij} , has been converted to black and white, with black pixels forming the character image. The goal is to optimally segment all the black pixels into clusters, in such a way that a final step of comparison with a reference set can be carried out, with the aim of recognizing the character of the image.

To illustrate the feasibility of our procedure, we have made use of an example, given by the character “A” depicted by means of the different samples in the image represented in Figure 10. The performance of the proposed GGA in the recognition of this character is given in Figures 11(a) and 11(b), which display the results achieved by the GGA using our MDB index as objective function and the Euclidean and GK distances, respectively. It is important to note how the proposed GGA approach using the GK distance is able to correctly allocate the three segments that form the A character. The GGA with the Euclidean distance does not provide, however, as good results as those depicted in Figure 11(a). To further assess the feasibility of our proposal, Table 5 summarizes a quantitative comparison in terms of the Rand index. The GGA with MDB index and GK distance is the

TABLE 5: Comparison of the results obtained by the proposed GGA algorithm with MDB index and different distances and the FCM algorithm in the character recognition problem.

Algorithm	Distance	Number of clusters	Rand index
GGA (MDB index)	Euclidean	5	0.6606
GGA (MDB index)	GK	3	0.9380
GGA (MDB index)	AFC	3	0.6906
FCM	Euclidean	5	0.6781

TABLE 6: Comparison of the results obtained by the proposed GGA algorithm with MDB index and different distances and the FCM algorithm in the diabetes problem. P_C (%) stands for the probability of correct classification.

Algorithm	Distance	P_C (%)
GGA (MDB index)	Euclidean	0.7246
GGA (MDB index)	GK	0.8348
GGA (MDB index)	AFC	0.7406
FCM	Euclidean	0.6601

best among all the algorithms compared, whereas the GGA using the MDB index and either Euclidean or AFC distances obtains similar results to those of the FCM approach. The approach that leads to the best solution of this problem is the proposed GGA by using our MDB fitness function along with the GK distance: it is able to correctly find the 3 segments (clusters of points) with the highest Rand index ($R = 0.9380$).

4.2.2. Real Problem 2: Diabetes Data Set. The data set called “diabetes” (UCI machine learning repository, see [54]) is a well-known problem in classification and clustering involving the diagnosis of diabetes patients, as defined by the World Health Organization. This data base is formed by 768 data vectors, containing, in turn, 8 features that represent medical conditions of the patients, such as age, arterial pressure, or body mass index. The observations belong to two classes, 500 of which belong to a negative diabetes diagnosis and 268 to a positive one. The results obtained by the proposed GGA assisted by the MDB index (which has been found to be the best) are shown in Table 6, in terms of percentage of correct classification. Note that the GGA-MDB with GK distance is the best algorithm among all compared, with a percentage of correct classification over 83%.

5. Conclusions

In this paper we have presented a grouping genetic algorithm for fuzzy clustering problems. The main contributions of this work are (1) a novel encoding approach of the individuals involved in the evolutionary process, containing information not only of the partition matrix elements, but also of the clusters being obtained; (2) a novel fitness function based on a modification of the Davis-Bouldin index for its efficient use in fuzzy clustering problems and that enables the chance of introducing norms adapted to any problem; (3) novel crossover and mutation operators particularly derived to achieve the effective evolution of the individuals; and

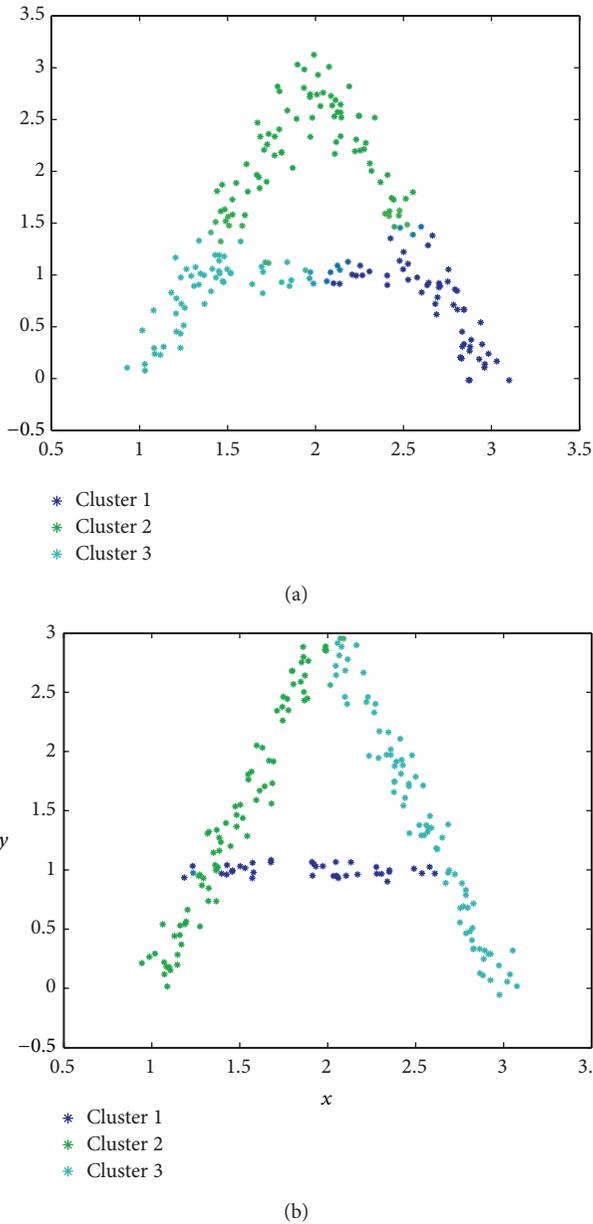


FIGURE 11: Results obtained in the character recognition problem by the proposed GGA: (a) Euclidean distance; (b) GK distance.

(4) a local search and parallelization-based scheme of the algorithm aimed at improving its overall performance.

Indeed, such performance has been explored in a variety of experiments, both synthetically generated and based on practical problems. The experimental work devised—based on different fuzzy problems characterized by an increasing degree of complexity (clusters with different distribution, volume, and orientation)—proves that our algorithm (using our proposed fitness function with distances such as the Gustafson-Kessel distance or the one established for the adaptive fuzzy clustering) exhibits a significantly better performance than that achieved by the fuzzy C-means algorithm.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported by the Gachon University research fund of 2014.

References

- [1] A. K. Jain, M. N. Murty, and P. J. Flynn, “Data clustering: a review,” *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.
- [2] R. Xu and D. Wunsch II, “Survey of clustering algorithms,” *IEEE Transactions on Neural Networks*, vol. 16, no. 3, pp. 645–678, 2005.
- [3] T. W. Liao, “Clustering of time series data—a survey,” *Pattern Recognition*, vol. 38, no. 11, pp. 1857–1874, 2005.
- [4] P. Lingras and X. Huang, “Statistical, evolutionary, and neuro-computing clustering techniques: cluster-based vs object-based approaches,” *Artificial Intelligence Review*, vol. 23, no. 1, pp. 3–29, 2005.
- [5] J. Ji, W. Pang, C. Zhou, X. Han, and Z. Wang, “A fuzzy k-prototype clustering algorithm for mixed numeric and categorical data,” *Knowledge-Based Systems*, vol. 30, pp. 129–135, 2012.
- [6] S. Mitra and H. Banka, “Multi-objective evolutionary biclustering of gene expression data,” *Pattern Recognition*, vol. 39, no. 12, pp. 2464–2477, 2006.
- [7] P. Scheunders, “A genetic C-means clustering algorithm applied to color image quantization,” *Pattern Recognition*, vol. 30, no. 6, pp. 859–866, 1997.
- [8] V. M. Gomez-Muñoz and M. A. Porta-Gándara, “Local wind patterns for modeling renewable energy systems by means of cluster analysis techniques,” *Renewable Energy*, vol. 25, no. 2, pp. 171–182, 2002.
- [9] A. Baraldi and P. Blonda, “A survey of fuzzy clustering algorithms for pattern recognition, part I,” *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 29, no. 6, pp. 778–785, 1999.
- [10] A. Baraldi and P. Blonda, “A survey of fuzzy clustering algorithms for pattern recognition, part II,” *IEEE Transactions on Systems, Man, and Cybernetics B: Cybernetics*, vol. 29, no. 6, pp. 786–801, 1999.
- [11] U. Maulik and S. Bandyopadhyay, “Fuzzy partitioning using a real-coded variable-length genetic algorithm for pixel classification,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 41, no. 5, pp. 1075–1081, 2003.
- [12] X. Guo, Z. Zhu, and J. Shi, “A corporate credit rating model using support vector domain combined with fuzzy clustering algorithm,” *Mathematical Problems in Engineering*, vol. 2012, Article ID 302624, 20 pages, 2012.
- [13] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, IEEE Press, 1981.
- [14] D. Graves and W. Pedrycz, “Kernel-based fuzzy clustering and fuzzy clustering: a comparative experimental study,” *Fuzzy Sets and Systems*, vol. 161, no. 4, pp. 522–543, 2010.
- [15] H. Zhang and J. Lu, “Semi-supervised fuzzy clustering: a kernel-based approach,” *Knowledge-Based Systems*, vol. 22, no. 6, pp. 477–481, 2009.

- [16] S. Chatzis and T. Varvarigou, "Factor analysis latent subspace modeling and robust fuzzy clustering using t-distributions," *IEEE Transactions on Fuzzy Systems*, vol. 17, no. 3, pp. 505–517, 2009.
- [17] Y. Zhong and L. Zhang, "A new fuzzy clustering algorithm based on clonal selection for land cover classification," *Mathematical Problems in Engineering*, vol. 2011, Article ID 708459, 21 pages, 2011.
- [18] A. Celikyilmaz and I. Burhan-Turksen, "Enhanced fuzzy system models with improved fuzzy clustering algorithm," *IEEE Transactions on Fuzzy Systems*, vol. 16, no. 3, pp. 779–794, 2008.
- [19] E. G. Mansoori, M. J. Zolghadri, and S. D. Katebi, "SGERD: a steady-state genetic algorithm for extracting fuzzy classification rules from data," *IEEE Transactions on Fuzzy Systems*, vol. 16, no. 4, pp. 1061–1071, 2008.
- [20] E. G. Mansoori, "FRBC: a fuzzy rule-based clustering algorithm," *IEEE Transactions on Fuzzy Systems*, vol. 19, no. 5, pp. 960–971, 2011.
- [21] S. Eschrich, J. Ke, L. O. Hall, and D. B. Goldgof, "Fast accurate fuzzy clustering through data reduction," *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 2, pp. 262–270, 2003.
- [22] E. N. Nasibov and G. Ulutagay, "A new unsupervised approach for fuzzy clustering," *Fuzzy Sets and Systems*, vol. 158, no. 19, pp. 2118–2133, 2007.
- [23] D. Horta, I. C. de Andrade, and R. J. Campello, "Evolutionary fuzzy clustering of relational data," *Theoretical Computer Science*, vol. 412, no. 42, pp. 5854–5870, 2011.
- [24] J. Yu and M.-S. Yang, "A generalized fuzzy clustering regularization model with optimality tests and model complexity analysis," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 5, pp. 904–915, 2007.
- [25] W. Yaonan, L. Chunsheng, and Z. Yi, "A selection model for optimal fuzzy clustering algorithm and number of clusters based on competitive comprehensive fuzzy evaluation," *IEEE Transactions on Fuzzy Systems*, vol. 17, no. 3, pp. 568–577, 2009.
- [26] C. Hwang and F. C. Rhee, "Uncertain fuzzy clustering: interval type-2 fuzzy approach to C-means," *IEEE Transactions on Fuzzy Systems*, vol. 15, no. 1, pp. 107–120, 2007.
- [27] M. Lee and W. Pedrycz, "The fuzzy C-means algorithm with fuzzy P-mode prototypes for clustering objects having mixed features," *Fuzzy Sets and Systems*, vol. 160, no. 24, pp. 3590–3600, 2009.
- [28] V. S. Alves, R. J. Campello, and E. R. Hruschka, "A fuzzy variant of an evolutionary algorithm for clustering," in *Proceedings of the IEEE International Conference on Fuzzy Systems*, pp. 375–380, Imperial College, London, UK, July 2007.
- [29] C. Chun-Hao, V. S. Tseng, and H. Tzung-Pei, "Cluster-based evaluation in fuzzy-genetic data mining," *IEEE Transactions on Fuzzy Systems*, vol. 16, no. 1, pp. 249–262, 2008.
- [30] M. K. Pakhira, S. Bandyopadhyay, and U. Maulik, "A study of some fuzzy cluster validity indices, genetic clustering and application to pixel classification," *Fuzzy Sets and Systems*, vol. 155, no. 2, pp. 191–214, 2005.
- [31] G. Gan, J. Wu, and Z. Yang, "A genetic fuzzy k-Modes algorithm for clustering categorical data," *Expert Systems with Applications*, vol. 36, no. 2, pp. 1615–1620, 2009.
- [32] A. Mukhopadhyay, U. Maulik, and S. Bandyopadhyay, "Multi-objective genetic algorithm-based fuzzy clustering of categorical attributes," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 991–1005, 2009.
- [33] U. Maulik and I. Saha, "Automatic fuzzy clustering using modified differential evolution for image classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 9, pp. 3503–3510, 2010.
- [34] H. Izakian and A. Abraham, "Fuzzy C-means and fuzzy swarm for fuzzy clustering problem," *Expert Systems with Applications*, vol. 38, no. 3, pp. 1835–1838, 2011.
- [35] H. Dong, Y. Dong, C. Zhou, G. Yin, and W. Hou, "A fuzzy clustering algorithm based on evolutionary programming," *Expert Systems with Applications*, vol. 36, no. 9, pp. 11792–11800, 2009.
- [36] E. Falkenauer, "The grouping genetic algorithm—widening the scope of the GAs," *Proceedings of the Belgian Journal of Operations Research, Statistics and Computer Science*, vol. 33, pp. 79–102, 1992.
- [37] E. Falkenauer, *Genetic Algorithms for Grouping Problems*, John Wiley & Sons, New York, NY, USA, 1998.
- [38] S. Salcedo-Sanz, L. Carro-Calvo, J. A. Portilla-Figueras, L. Cuadra, and D. Camacho, "Fuzzy clustering with grouping genetic algorithms," in *Intelligent Data Engineering and Automated Learning*, vol. 8206 of *Lecture Notes in Computer Science*, pp. 334–341, Springer, Berlin, Germany, 2013.
- [39] R. Krishnapuram and J. Kim, "A note on the Gustafson-Kessel and adaptive fuzzy clustering algorithms," *IEEE Transactions on Fuzzy Systems*, vol. 7, no. 4, pp. 453–461, 1999.
- [40] X. L. Xie and G. Beni, "A validity measure for fuzzy clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 8, pp. 841–847, 1991.
- [41] Y. Fukuyama and M. Sugeno, "A new method of choosing the number of clusters for the fuzzy C-means method," in *Proceedings of the 5th Fuzzy Systems Symposium*, pp. 247–250, Kobe, Japan, 1989.
- [42] W. M. Rand, "Objective criteria for the evaluation of clustering methods," *Journal of the American Statistical Society*, vol. 66, pp. 846–850, 1971.
- [43] L. E. Agustín-Blas, S. Salcedo-Sanz, P. Vidales, G. Urueta, and J. A. Portilla-Figueras, "Near optimal citywide WiFi network deployment using a hybrid grouping genetic algorithm," *Expert Systems with Applications*, vol. 38, no. 8, pp. 9543–9556, 2011.
- [44] T. L. James, E. C. Brown, and K. B. Keeling, "A hybrid grouping genetic algorithm for the cell formation problem," *Computers & Operations Research*, vol. 34, no. 7, pp. 2059–2079, 2007.
- [45] T. James, M. Vroblefski, and Q. Nottingham, "A hybrid grouping genetic algorithm for the registration area planning problem," *Computer Communications*, vol. 30, no. 10, pp. 2180–2190, 2007.
- [46] E. C. Brown and M. Vroblefski, "A grouping genetic algorithm for the microcell sectorization problem," *Engineering Applications of Artificial Intelligence*, vol. 17, no. 6, pp. 589–598, 2004.
- [47] E. C. Brown and R. T. Sumichrast, "Evaluating performance advantages of grouping genetic algorithms," *Engineering Applications of Artificial Intelligence*, vol. 18, no. 1, pp. 1–12, 2005.
- [48] C. Hung, R. T. Sumichrast, and E. C. Brown, "CPGEA: a grouping genetic algorithm for material cutting plan generation," *Computers & Industrial Engineering*, vol. 44, no. 4, pp. 651–672, 2003.
- [49] P. de Lit, E. Falkenauer, and A. Delchambre, "Grouping genetic algorithms: an efficient method to solve the cell formation problem," *Mathematics and Computers in Simulation*, vol. 51, no. 3–4, pp. 257–271, 2000.
- [50] E. C. Brown and R. T. Sumichrast, "Impact of the replacement heuristic in a grouping genetic algorithm," *Computers & Operations Research*, vol. 30, no. 11, pp. 1575–1593, 2003.

- [51] V. B. Kreng and T. Lee, "Modular product design with grouping genetic algorithm—a case study," *Computers & Industrial Engineering*, vol. 46, no. 3, pp. 443–460, 2004.
- [52] L. E. Agustn-Blas, S. Salcedo-Sanz, S. Jiménez-Fernández, L. Carro-Calvo, J. del Ser, and J. A. Portilla-Figueras, "A new grouping genetic algorithm for clustering problems," *Expert Systems with Applications*, vol. 39, no. 10, pp. 9695–9703, 2012.
- [53] D. Davies and D. Bouldin, "A cluster sparation measure," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, pp. 224–227, 1979.
- [54] A. Asuncion and D. J. Newman, "UCI Machine Learning Repository," School of Information and Computer Science, University of California, Irvine, Calif, USA, 2007, <http://www.ics.uci.edu/~mllearn/MLRepository.html>.

Research Article

On the Effectiveness of Nature-Inspired Metaheuristic Algorithms for Performing Phase Equilibrium Thermodynamic Calculations

Seif-Eddeen K. Fateen¹ and Adrian Bonilla-Petriciolet²

¹ Department of Chemical Engineering, Cairo University, Giza 12316, Egypt

² Department of Chemical Engineering, Aguascalientes Institute of Technology, 20256 Aguascalientes, AGS, Mexico

Correspondence should be addressed to Seif-Eddeen K. Fateen; fateen@eng1.cu.edu.eg

Received 27 March 2014; Accepted 30 April 2014; Published 20 May 2014

Academic Editor: Xin-She Yang

Copyright © 2014 S.-E. K. Fateen and A. Bonilla-Petriciolet. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The search for reliable and efficient global optimization algorithms for solving phase stability and phase equilibrium problems in applied thermodynamics is an ongoing area of research. In this study, we evaluated and compared the reliability and efficiency of eight selected nature-inspired metaheuristic algorithms for solving difficult phase stability and phase equilibrium problems. These algorithms are the cuckoo search (CS), intelligent firefly (IFA), bat (BA), artificial bee colony (ABC), MAKHA, a hybrid between monkey algorithm and krill herd algorithm, covariance matrix adaptation evolution strategy (CMAES), magnetic charged system search (MCSS), and bare bones particle swarm optimization (BBPSO). The results clearly showed that CS is the most reliable of all methods as it successfully solved all thermodynamic problems tested in this study. CS proved to be a promising nature-inspired optimization method to perform applied thermodynamic calculations for process design.

1. Introduction

Applied thermodynamic calculations in chemical engineering often involve the repeated solution of phase stability and phase equilibrium problems as their solutions are needed during the design of several equipment and separation processes. These problems can be formulated as minimization problems, for which the global minimum represents the required result. These calculations are challenging due to the high nonlinearity of thermodynamic models used to describe the equilibrium phases, the potential nonconvexity of the thermodynamic functions used as objective, and the presence of trivial solutions in the feasible search space. Thus, the solution of this type of problems via global optimization algorithms remains to be an active area of research. These problems generally feature local minima that are comparable to the global minimum, which accentuates the need for reliable global optimizers [1, 2]. For example, the features of reactive phase equilibrium calculations increase the dimensionality and complexity of the optimization problem because

the objective functions are required to satisfy the chemical equilibrium constraints [1, 2].

The global stochastic optimization methods show high probabilities to locate the global minimum within reasonable computational costs, and thus they offer a desirable balance between reliability and efficiency for finding the global optimum solution. Moreover, stochastic methods do not require any assumptions for the optimization problem at hand, are more capable of addressing the nonlinearity and nonconvexity of the objective function, and are relatively easier to program and implement, among other advantages [3].

The application of stochastic global optimization methods for solving phase equilibrium thermodynamic problems has grown considerably during last years. To date, the most popular stochastic global optimization methods have been used and applied for solving phase equilibrium thermodynamic problems, for example, simulated annealing, genetic algorithms, tabu search, differential evolution, particle swarm optimization, and ant colony optimization (ACO) [4–15].

For example, a variant of ACO was tested in the global optimization of thermodynamic problems and was found to be robust in solving vapor-liquid equilibrium parameter estimation problems [4]. Zhu et al. [5] used an enhanced simulated annealing algorithm to solve multicomponent phase stability problems. Bonilla-Petriciolet and his coworkers compared different variants of PSO [6] and different variants of simulated annealing [14] for solving phase equilibrium problems. Repulsive particle swarm optimization was also studied by Rahman et al. [8]. Rangaiah and his co-workers studied the differential evolution [9, 10], tabu search [11], and genetic algorithms [12] for solving phase stability and phase equilibrium problems.

The above studies have analyzed the capabilities and limitations of stochastic optimizers. But there exists no conclusive evaluation of those methods in comparison to one another for the solution of phase stability and phase equilibrium problems. Typically, each algorithm is introduced and compared with some of the other algorithms in a research publication. However, to the best of our knowledge, there exists no study that presents to the scientific community a ranking of the efficiency and reliability of those algorithms for the purpose of solving phase equilibrium and stability problems.

The aim of this study is provide a definitive ranking of the performance of a set of nature-inspired metaheuristic algorithms. To do so, we have selected eight of the most promising nature-inspired optimization methods based on the performance reported in the literature or obtained from our previous studies. These algorithms are cuckoo search (CS), intelligent firefly (IFA), bat (BA), artificial bee colony (ABC), monkey and krill herd hybrid (MAKHA), covariance matrix adaptation evolution strategy (CMAES), magnetic charged system search (MCSS), and bare bones particle swarm optimization (BBPSO). We systematically used those methods on some of the difficult phase stability and phase equilibrium problems reported in the literature and then analyzed their performance in terms of clear reliability and efficiency metrics.

The remainder of this paper is organized as follows. The eight optimization methods and the rationale for their selection are briefly presented in Section 2. A brief description of the phase stability and equilibrium problems is given in Section 3, including the implementation details of the eight algorithms. Section 4 presents the results and discussion of their performance in solving these thermodynamic calculations. Finally, the conclusions of this study are summarized in Section 5.

2. Selection and Description of the Nature-Inspired Metaheuristic Algorithms

Each of the eight selected metaheuristics is presented below. Only brief introductions are made here. Interested readers are referred to the primary sources of those algorithms for more information.

Cuckoo search (CS) is an optimization algorithm inspired by the obligate brood parasitism of some cuckoo species

by laying their eggs in the nests of other host birds [16]. Intelligent firefly algorithm (IFA) [17] is a variant of firefly algorithm [18], a metaheuristic algorithm, inspired by the flashing behavior of fireflies to attract other fireflies. MAKHA is a hybrid between monkey algorithm (MA) [19], which is inspired by the simulation of the climbing processes of monkeys to find the highest mountaintop, and krill-herd algorithm (KHA) [20], which is based on the simulation of the herding behavior of krill individuals. Covariance matrix adaptation evolution strategy (CMAES) [21] is a stochastic and derivative free method for numerical optimization of nonlinear nonconvex problems. Artificial bee colony (ABC) [22] is an optimization algorithm based on the intelligent foraging behavior of honey bee swarm. Bat algorithm (BA) [23] is another bioinspired optimization algorithm based on the echolocation behavior of microbats with varying pulse rates of emission and loudness. Magnetic charged system search (MCSS) [24] is a variant of charged system search [25], which is based on the application of physics principles such as Coulomb law and Newtonian laws of mechanics to model how charged particles affect one another during their move towards the largest bodies. In MCSS, magnetic forces are also considered in addition to electrical forces. Finally, a variant of bare bones particle swarm optimization (BBPSO) [26] is based on the original particle swarm optimization [27], but without parameters and with the incorporation of mutation and crossover operators of DE to enhance the global search capability.

Since it was not possible to include all global stochastic optimization methods available in the literature for this comparative study, a screening process was performed to select the most promising ones. This process depended mainly on the results of solving phase stability and phase equilibrium problems using global optimization methods as reported in the literature. In several publications, limited comparisons were reported between some GSO methods. For example, CMAES was selected as it was shown to perform better than shuffled complex evolution in solving phase equilibrium and phase stability problems [28]; IFA performed better than FA in general [17], CS better than integrated differential evolution [29], MCSS better than CSS for phase equilibrium and phase stability problems [30], and BBPSO better than PSO [26]. In addition, our preliminary calculations showed that MAKHA performed better than MA and KHA, and ABC and BA performed better than FA.

One approach to solving phase stability and phase equilibrium problems is to start the optimization process with a stochastic global optimizer, as the methods studied in this work. Once a certain stopping criterion is satisfied, we follow with a local optimizer, such as sequential quadratic programming, to close down to the minimum within the vicinity of the best value found by the global optimizer. This approach has been proven successful in previous studies [28–30] and it would complement any of the methods studied above. However, we restricted this study to the performance of the stochastic global optimizers without the use of a local optimizer to focus on the strength and weakness of the studied methods free from any artificial enhancement of their results.

3. Description of Phase Stability and Phase Equilibrium Problems Used for the Evaluation

3.1. Objective Functions. In this study, the phase stability and equilibrium problems are stated as a global optimization problem. Therefore, the global optimization problem to be solved is as follows: minimize $F(\mathbf{X})$ with respect to D decision variables: $\mathbf{X} = (X^1, \dots, X^D)$. The upper and lower bounds of these variables are $(X_{\max}^1, \dots, X_{\max}^D)$ and $(X_{\min}^1, \dots, X_{\min}^D)$, respectively.

The phase stability, phase equilibrium, and reactive phase equilibrium calculations for testing the performance of global optimization methods are explained briefly in Table 1, which shows the problem formulation, objective function, decision variables, and constraints used for those thermodynamic calculations. Specifically, the phase stability analysis was performed using the global minimization of the tangent plane distance function (TPDF) [31], while the global optimization of the Gibbs free energy was used for phase equilibrium calculations with or without chemical reactions [2]. The mathematical formulation for phase stability and phase equilibrium calculations for nonreactive systems is an unconstrained minimization of the objective function, while the constrained Gibbs free energy minimization in reactive systems was performed using the penalty function method according to the approach reported by Bonilla-Petriciolet et al. [1]. For interested readers, several references provide a detailed description of these thermodynamic calculations [1, 2, 4, 10, 12].

Previous work reported the evaluation of global optimization methods for solving twenty-four problems [4, 28, 30]. In this work, we focused on the nine most difficult ones. The basis for the selection was the relatively lower success rates that optimization methods obtained when solving them in the previous studies. These problems are presented in Table 2.

3.2. Details of Numerical Implementation and Performance Metrics Used for Testing the Algorithms. All thermodynamic problems and the different optimization algorithms were coded in the MATLAB technical computing environment. The codes for CS and BA were obtained from MATLAB file exchange server as uploaded by their developers and used without change. The code for IFA was developed by the authors through minor modifications of the FA code that was obtained from the MATLAB file exchange server as well. The codes for CMAES and ABC were obtained from the developers' web sites and used without change. The code for MCSS was written by the authors based on the developer's published work [24, 25]. MAKHA was developed and coded by the authors. The code for BBPSO was obtained from its developer [26]. Each problem was solved 30 times independently and with different random initial seeds to determine the reliability of the optimization algorithms. Calculations were performed for a certain number of iterations and then stopped. This maximum value for the number of iterations was different for different algorithms. The maximum values were selected to give the same number of function evaluations at the end of

the run. Table 3 shows the values selected for the parameters of the eight optimization algorithms, which were determined using preliminary calculations.

The methods were evaluated according to the reliability and efficiency for finding the global optimum. The efficiency is determined by recording the number of function evaluations NFE for each optimization algorithm, where a low value of NFE means a higher efficiency. Note that NFE is an unbiased indicator of the computational costs required by a certain algorithm and is independent of the host hardware. In previous studies [1, 4, 6, 26, 28, 30], reliability was measured by the success rate at certain number of iterations. The success rate is defined as the ratio of number of runs in which the global minimum was attained within a tolerance at this iteration number to the total number of runs. In this work, we present a different reliability metric: a plot of the average best value against the number of function evaluations. The best values are averaged over all the runs and plotted against NFE, which is calculated at each iteration. Since the NFE needed for each iteration differs amongst the optimization methods, the plot of average best value against NFE is a better indication of reliability versus efficiency of the optimization method.

For a comparative evaluation of the global optimization methods, we have employed performance profile (PP) reported by Dolan and Moré [32], who introduced PP as a tool for evaluating and comparing the performance of optimization software. In particular, PP has been proposed to represent compactly and comprehensively the data collected from a set of solvers for a specified performance metric such as the computing time or the number of function evaluations. The PP plot allows visualization of the expected performance differences among several solvers and comparing the quality of their solutions by eliminating the bias of failures obtained in a small number of problems.

Consider n_s solvers (i.e., optimization methods) to be tested over a set of n_p problems. For each problem p and solver s , the performance metric t_{ps} must be defined. In our study, reliability of the stochastic method in accurately finding the global minimum of the objective function is considered as the principal goal, and hence the reliability performance metric is defined as

$$t_{ps} = f_{\text{calc}} - f^*, \quad (1)$$

where f^* is the known global optimum of the objective function and f_{calc} is the mean value of that objective function calculated by the metaheuristic over several runs. We have also used another performance metric for the evaluation of the efficiency of the method in obtaining the global minimum. This metric is the minimum number of NFE needed to reach with 10^{-5} of the global minimum.

For the performance metric of interest, the performance ratio, r_{ps} , is used to compare the performance on problem p by solver s with the best performance by any solver on this problem. This performance ratio is given by

$$r_{ps} = \frac{t_{ps}}{\min \{t_{ps} : 1 \leq s \leq n_s\}}. \quad (2)$$

TABLE 1: Description of thermodynamic functions and optimization problems for phase stability analysis and equilibrium calculations in reactive and nonreactive systems.

Calculation	Description	Thermodynamic function	Optimization problem
Phase stability analysis	It involves the determination of whether a system will remain in one phase at the given conditions or split into two or more phases	<p>Tangent plane distance function</p> $\text{TPDF} = \sum_{i=1}^c \gamma_i \left(\mu_{i,y} - \mu_{i,z} \right)$ <p>where c is the number of components of the mixture and $\mu_{i,y}$ and $\mu_{i,z}$ are the chemical potentials calculated at trial composition y and feed composition z</p>	<p>minTPDF</p> $0 \leq \beta_i \leq 1 \quad i = 1, \dots, c.$ <p>The decision variables are $\beta_i \in (0, 1)$ using the following relationships:</p> $n_{i,y} = \beta_i z_i n_F \quad i = 1, \dots, c$ $\gamma_i = \frac{n_{i,y}}{\sum_{j=1}^c n_{j,y}} \quad i = 1, \dots, c$ <p>where $n_{i,y}$ are the mole numbers of component i in phase y and n_F is the total moles in the mixture under analysis</p>
Phase equilibrium calculation	It involves the determination of the number, type, and composition of the phases at equilibrium at the given operating conditions	<p>Gibbs free energy of mixing (g)</p> $g = \sum_{j=1}^{\pi} \sum_{i=1}^c n_{ij} \ln(x_{ij} \gamma_{ij})$ $= \sum_{j=1}^{\pi} \sum_{i=1}^c n_{ij} \ln \left(\frac{x_{ij} \hat{\phi}_{ij}}{\phi_i} \right)$ <p>where π is the number of phases at equilibrium and θ_{ij} denotes the composition (i.e., x or n) or thermodynamic property of component i in phase j</p>	<p>ming</p> $0 \leq \beta_{ij} \leq 1$ $i = 1, \dots, c$ $j = 1, \dots, \pi - 1.$ <p>The decision variables are $\beta_{ij} \in (0, 1)$ using the following relationships:</p> $n_{i1} = \beta_{i1} z_i n_F \quad i = 1, \dots, c$ $n_{ij} = \beta_{ij} \left(z_i n_F - \sum_{m=1}^{j-1} n_{im} \right)$ $i = 1, \dots, c; \quad j = 2, \dots, \pi - 1$ $n_{i\pi} = z_i n_F - \sum_{m=1}^{\pi-1} n_{im} \quad i = 1, \dots, c$
Reactive phase equilibrium Calculations	It involves the determination of the number, type and composition of the phases at equilibrium at the given operating conditions and subject to element/mass balances and chemical equilibrium constraints.	<p>Gibbs free energy of mixing defined using reaction equilibrium constants [2]</p> $G_K = g - \sum_{j=1}^{\pi} \ln \mathbf{K}_{\text{eq}} \mathbf{N}^{-1} \mathbf{n}_{\text{ref},j}$ <p>where g is the Gibbs free energy of mixing, $\ln \mathbf{K}_{\text{eq}}$ is a row vector of logarithms of chemical equilibrium constants for r independent reactions, \mathbf{N} is an invertible, square matrix formed from the stoichiometric coefficients of a set of reference components chosen from the r reactions, and \mathbf{n}_{ref} is a column vector of moles of each of the reference components</p>	<p>minG_K</p> <p>subject to</p> $\sum_{j=1}^{\pi} \left(n_{ij} - \mathbf{v}_i \mathbf{N}^{-1} \mathbf{n}_{\text{ref},j} \right) = n_{i,F} - \mathbf{v}_i \mathbf{N}^{-1} \mathbf{n}_{\text{ref},F}$ $i = 1, \dots, c - r$ $n_{ij} > 0 \quad i = 1, \dots, c \quad j = 1, \dots, \pi$ <p>where $n_{i,F}$ is the initial moles of component i in the feed, \mathbf{v}_i is the row vector (of dimension r) of stoichiometric coefficients of component i in r reactions, and $n_{i,j}$ is the number of moles of component i in phase j. The constrained global optimization problem can be solved by minimizing G_K with respect to $c(\pi - 1) + r$ decision variables $n_{i,j}$. In this formulation, the mass balance equations are rearranged to reduce the number of decision variables of the optimization problem and to eliminate equality constraints</p>

TABLE 2: Details of the phase stability, phase equilibrium, and reactive phase equilibrium problems used in this study.

Code	System	Feed conditions	Thermodynamic models	Global optimum
T7	$C_1 + C_2 + C_3 + C_4 + C_5 + C_6 + C_{7-16} + C_{17+}$	$n_F = (0.7212, 0.09205, 0.04455, 0.03123, 0.01273, 0.01361, 0.07215, 0.01248)$ at 353 K and 38500 kPa	Phase stability problem with SRK EoS with classical mixing rules	-0.002688
T8	$C_1 + C_2 + C_3 + iC_4 + C_4 + iC_5 + C_5 + C_6 + iC_{15}$	$n_F = (0.614, 0.10259, 0.04985, 0.008989, 0.02116, 0.00722, 0.01187, 0.01435, 0.16998)$ at 314 K and 2010.288 kPa	Phase stability problem with SRK EoS with classical mixing rules	-1.486205
T9	$C_1 + C_2 + C_3 + C_4 + C_5 + C_6 + C_7 + C_8 + C_9 + C_{10}$	$n_F = (0.6436, 0.0752, 0.0474, 0.0412, 0.0297, 0.0138, 0.0303, 0.0371, 0.0415, 0.0402)$ at 435.35 K and 19150 kPa	Phase stability problem with SRK EoS with classical mixing rules	-0.0000205
G4	$C_1 + H_2S$	$n_F = (0.9813, 0.0187)$ at 190 K and 4053 kPa	Phase equilibrium problem with SRK EoS with classical mixing rules	-0.019892
G6	$C_2 + C_3 + C_4 + C_5 + C_6$	$n_F = (0.401, 0.293, 0.199, 0.0707, 0.0363)$ at 390 K and 5583 kPa	Phase equilibrium problem with SRK EoS with classical mixing rules	-1.183653
G7	$C_1 + C_2 + C_3 + C_4 + C_5 + C_6 + C_{7-16} + C_{17+}$	$n_F = (0.7212, 0.09205, 0.04455, 0.03123, 0.01273, 0.01361, 0.07215, 0.01248)$ at 353 K and 38500 kPa	Phase equilibrium problem with SRK EoS with classical mixing rules	-0.838783
G8	$C_1 + C_2 + C_3 + iC_4 + C_4 + iC_5 + C_5 + C_6 + iC_{15}$	$n_F = (0.614, 0.10259, 0.04985, 0.008989, 0.02116, 0.00722, 0.01187, 0.01435, 0.16998)$ at 314 K and 2010.288 kPa	Phase equilibrium problem with SRK EoS with classical mixing rules	-0.769772
R4	A1 + A2 ↔ A3 + A4 (1) Acetic acid (2) n-Butanol (3) Water (4) n-Butyl acetate	$n_F = (0.3, 0.4, 0.3, 0.0)$ at 298.15 K and 101.325 kPa	Reactive phase equilibrium problem with UNIQUAC model and ideal gas: $\ln K_{eq} = \frac{450}{T} + 0.8$	-1.10630
R7	A1 + A2 ↔ A3	$n_F = (0.52, 0.48, 0.0)$ at 323.15 K and 101.325 kPa	Reactive phase equilibrium problem with Margules solution model: $K_{eq} = 3.5$	-0.653756

TABLE 3: Selected values of the parameters used in the implementation of the eight nature-inspired metaheuristic algorithms.

Metaheuristic	Parameter	Selected value
MAKHA	n	$40D$
	B	0.5
	C	-0.1
	D	0.1
	D_{\max}	0
	C_t	0.5
	V_f	0.2
	W_f	0.1
IFA	α_o	0.5
	β_{\min}	0.2
	γ	1
	n	$20D$
CMAES	ϕ	0.05
	σ	0.2
ABC	n	$20D$
	Food number	$n/2$
BA	limit	100
	n	$20D$
MCSS	A	0.25
	r	0.5
	CMCR	0.95
CS	PAR	0.1
	N	$20D$
BBPSO	CMS	$n/4$, if integer $n/2$, if $n/4$ is not integer
	n	$20D$
BBPSO	p	0.25
	n	$20D$

The value of r_{ps} is 1 for the solver that performs the best on a specific problem p . To obtain an overall assessment of the performance of solvers on n_p problems, the following cumulative function for r_{ps} is used:

$$\rho_s(\zeta) = \frac{1}{n_p} \text{size} \{p : r_{ps} \leq \zeta\}, \quad (3)$$

where $\rho_s(\zeta)$ is the fraction of the total number of problems, for which solver s has a performance ratio r_{ps} within a factor of ζ of the best possible ratio. The PP of a solver is a plot of $\rho_s(\zeta)$ versus ζ ; it is a nondecreasing, piecewise constant function, continuous from the right at each of the breakpoints [32]. To identify the best solver, it is only necessary to compare the values of $\rho_s(\zeta)$ for all solvers and to select the highest one, which is the probability that a specific solver will “win” over the rest of solvers used.

In our case, one PP plot compares how accurately the stochastic methods can find the global optimum value relative

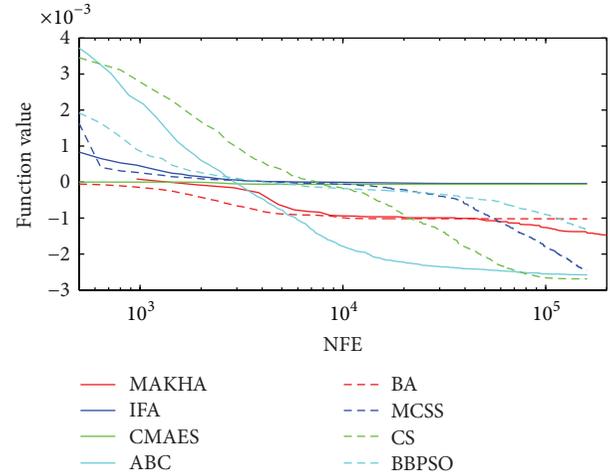


FIGURE 1: The evolution of the mean best value calculated via the eight metaheuristics versus NFE for problem T7.

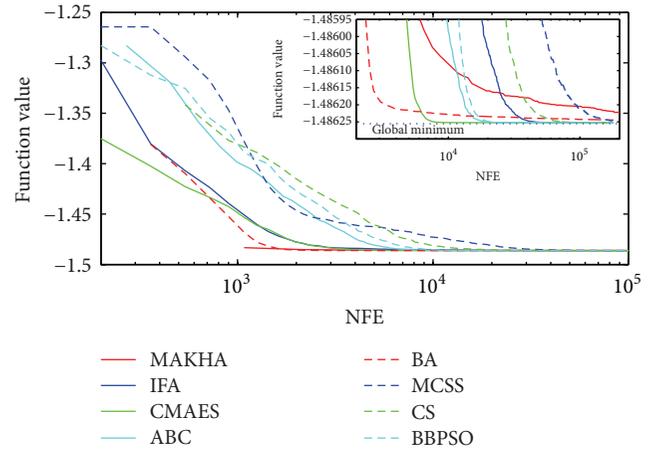


FIGURE 2: The evolution of the mean best value calculated via the eight metaheuristics versus NFE for problem T8.

to one another, and so the term “win” refers to the stochastic method that provides the most accurate value of the global minimum in the benchmark problems used. The other PP plot compares how fast the stochastic methods can find the global minimum with a tolerance level of 10^{-5} , so the term “win”, in this case, refers to the method that reaches the solution fastest for the problems used.

4. Results and Discussion

The results are presented in three different ways. For each problem, the mean best values are plotted versus NFE for each of the eight algorithms. These plots are found in Figures 1–9. The minimum NFE required to reach a certain tolerance from the known global minimum for each problem was calculated and presented in Table 4. The performance profiles for the reliability and efficiency metrics are shown in Figures 10 and 11, respectively. A detailed discussion of the results follows.

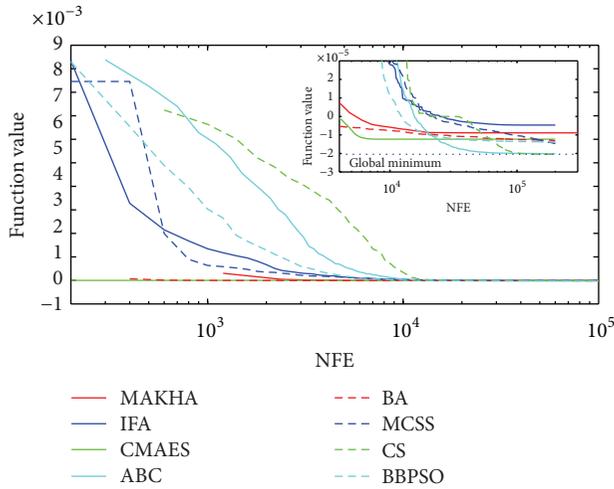


FIGURE 3: The evolution of the mean best value calculated via the eight metaheuristics versus NFE for problem T9.

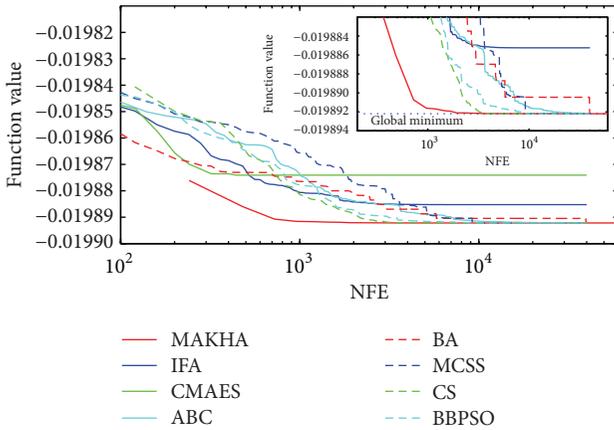


FIGURE 4: The evolution of the mean best value calculated via the eight metaheuristics versus NFE for problem G4.

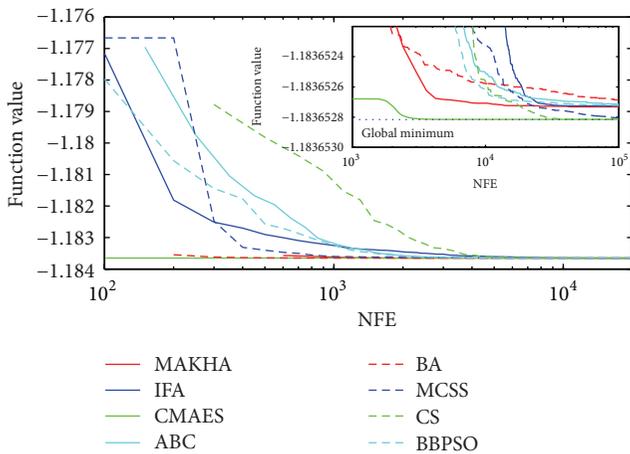


FIGURE 5: The evolution of the mean best value calculated via the eight metaheuristics versus NFE for problem G6.

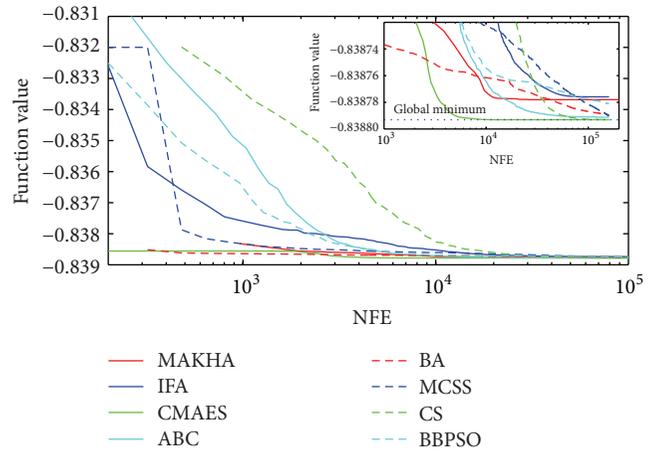


FIGURE 6: The evolution of the mean best value calculated via the eight metaheuristics versus NFE for problem G7.

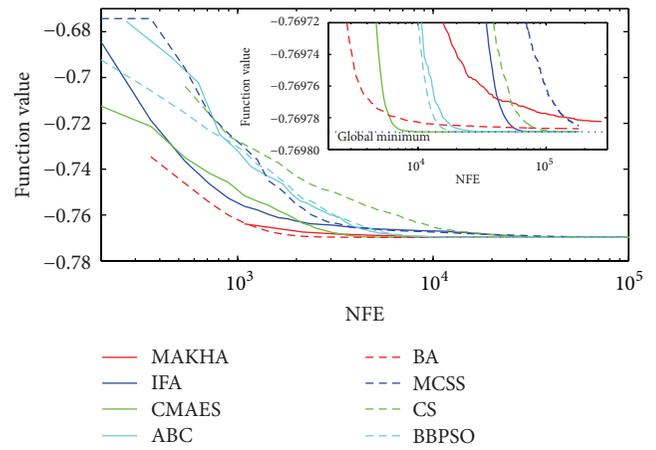


FIGURE 7: The evolution of the mean best value calculated via the eight metaheuristics versus NFE for problem G8.

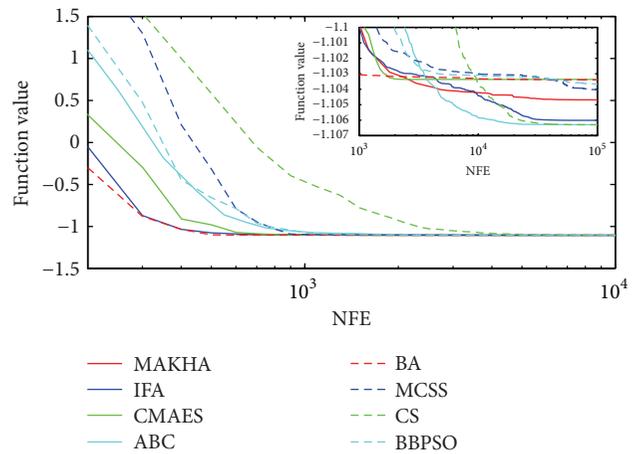


FIGURE 8: The evolution of the mean best value calculated via the eight metaheuristics versus NFE for problem RG4.

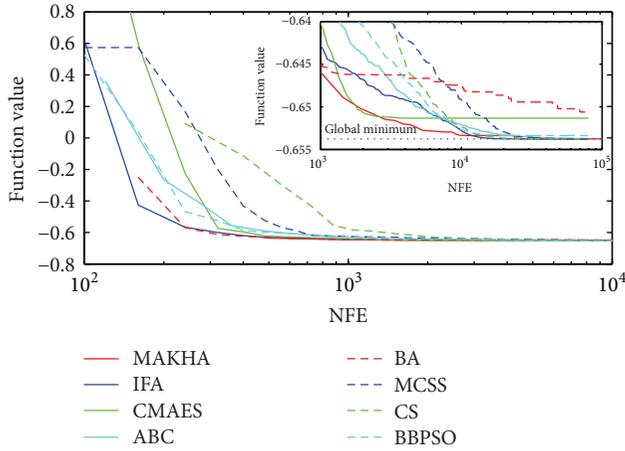


FIGURE 9: The evolution of the mean best value calculated via the eight metaheuristics versus NFE for problem RG7.

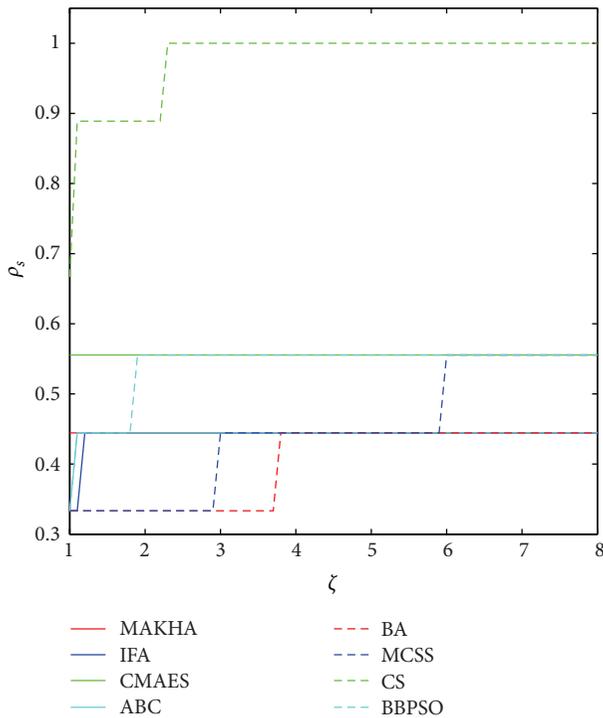


FIGURE 10: Performance profile (PP) of the reliability metric of the eight metaheuristics for the 9 phase stability and equilibrium problems.

4.1. Phase Stability Problems. Problem T7 is a nine-variable phase-stability problem that is extremely difficult to solve. The means of the minimum values obtained by all methods were not close enough to the global minimum except for CS. As shown in Figure 1 and Table 4, ABC and MCSS were able to get to within 10^{-3} of the global minimum. On the other hand, CS was able to find the global minimum down to a tolerance of 10^{-7} . To reach the global minimum within a tolerance of 10^{-5} , it required 109280 function evaluations.

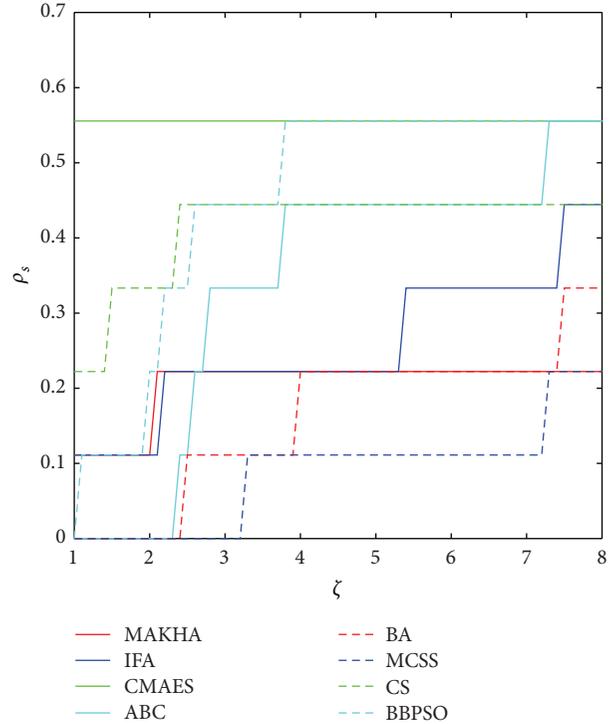


FIGURE 11: Performance profile (PP) of the efficiency metric of the eight metaheuristics for the 9 phase stability and equilibrium problems.

Problem T8 is also a difficult phase stability problem. Figure 2 shows how all problems were able to reach values close to the global optimum. However, close analysis at the vicinity of the global minimum, as depicted in the inset of Figure 2, at the level of 10^{-5} , revealed that MAKHA and BA failed to find the global minimum up to the end of the runs. CMAES was the most efficient as it converged to the global minimum in the least NFE by at least one order of magnitude. None of the methods was able to reach within 10^{-6} of the global minimum, as shown in Table 4.

Problem T9 is the last of the three phase stability problems. Even though, MAKHA was quite fast in approaching the global minimum, as depicted in Figure 3, it failed at converging to within 10^{-5} of the global minimum. IFA was also not able to find the global minimum. CMAES was the most efficient method in getting down to 10^{-5} distance from the global minimum but was not able to get any closer. CS, again, was the only method to converge reliably down to 10^{-7} of the global minimum.

For the phase stability problems, CS is clearly the most reliable method. It may not be as efficient in its initial approach to the global minimum as other methods such as BA or CMAES, but it outperforms the rest in terms of finding the global minimum. An open area of development for CS would be to make it more efficient via hybridization with some of the other methods in their initial approach to the global minimum.

TABLE 4: Minimum NFE for the average best value to reach 1E-3, 1E-4, 1E-5, 1E-6, and 1E-7 from the known global minimum.

Metaheuristic	ϵ	Phase equilibrium thermodynamic problem								
		T7	T8	T9	G4	G6	G7	G8	R4	R7
MAKHA	1E-3	∞	2164	601	121	301	481	4328	∞	10845
	1E-4	∞	45444	1803	121	602	4329	39493	∞	40006
	1E-5	∞	∞	∞	484	903	∞	∞	∞	40488
	1E-6	∞	∞	∞	2299	1806	∞	∞	∞	40729
	1E-7	∞	∞	∞	2783	∞	∞	∞	∞	41211
IFA	1E-3	∞	8820	1400	40	400	1600	19980	17300	9440
	1E-4	∞	24840	5200	40	3500	18880	36720	∞	14480
	1E-5	∞	39600	∞	1040	9000	∞	52560	∞	20080
	1E-6	∞	∞	∞	∞	15400	∞	73260	∞	25040
	1E-7	∞	∞	∞	∞	∞	∞	∞	∞	30160
CMAES	1E-3	∞	3961	201	41	101	161	3961	∞	∞
	1E-4	∞	5761	201	41	101	2721	5401	∞	∞
	1E-5	∞	7381	5801	∞	101	4801	7021	∞	∞
	1E-6	∞	∞	∞	∞	101	8641	8821	∞	∞
	1E-7	∞	∞	∞	∞	2401	11681	10621	∞	∞
ABC	1E-3	9840	8010	4100	60	850	2480	6570	9750	9560
	1E-4	∞	12150	9700	60	1950	7920	11790	24959	48236
	1E-5	∞	17010	21500	1220	3850	34644	19530	∞	∞
	1E-6	∞	∞	58104	12460	8650	∞	27990	∞	∞
	1E-7	∞	∞	∞	33092	∞	∞	41850	∞	∞
BA	1E-3	∞	1980	400	80	200	320	1800	∞	∞
	1E-4	∞	2340	400	80	200	5760	3240	∞	∞
	1E-5	∞	∞	43200	6960	400	∞	16920	∞	∞
	1E-6	∞	∞	∞	8400	3700	∞	∞	∞	∞
	1E-7	∞	∞	∞	8440	∞	∞	∞	∞	∞
MCSS	1E-3	87520	35640	800	40	400	640	26460	∞	21200
	1E-4	∞	80820	6200	40	1100	36800	104400	∞	34160
	1E-5	∞	178920	78000	3520	3300	157600	∞	∞	64320
	1E-6	∞	∞	∞	7880	11000	∞	∞	∞	∞
	1E-7	∞	∞	∞	9120	49600	∞	∞	∞	∞
CS	1E-3	41440	17460	7400	120	1700	7840	22140	16100	10640
	1E-4	79200	35100	11800	120	4100	26400	44820	30900	17680
	1E-5	109280	61380	57400	1160	5500	52960	74700	50300	30000
	1E-6	132640	∞	93400	3480	6700	93280	112500	70300	39760
	1E-7	155040	∞	131400	4440	24100	140000	152820	87300	53840
BBPSO	1E-3	∞	9900	2800	40	600	1760	7380	∞	9440
	1E-4	∞	14040	7400	40	1700	12640	11340	∞	17920
	1E-5	∞	18720	21600	920	3300	∞	15300	∞	21440
	1E-6	∞	∞	∞	2200	6000	∞	19080	∞	23360
	1E-7	∞	∞	∞	5120	∞	∞	22500	∞	28400

4.2. Phase Equilibrium Problems. Problem G4 is a two-variable phase equilibrium problem that is relatively easy to solve. However, CMAES seemed to have been trapped in a local minimum and was unable to find its global minimum, within a tolerance of 10^{-5} , as shown in Figure 4. IFA did slightly better than CMAES, but was unable to reach the global minimum within a tolerance of 10^{-6} . MAKHA was the most efficient in finding the global minimum within 10^{-6} and 10^{-7} , with BBPSO and CS performing quite well.

Despite the fact that CMAES was not able to solve problem G4, it was superior in solving problem G6. With only 101 NFE, CMAES reached down to 10^{-6} of the global minimum, as is shown in Figure 5. All methods converged to 10^{-6} from the global minimum, but only CMAES, CS, and MCSS converged to 10^{-7} , with CMAES being ten times more efficient. This convergence pattern was repeated in problem G7. Only CMAES and CS solved the problem down to the 10^{-6} and 10^{-7} levels, with CMAES being one order of magnitude

more efficient, as is clear in Figure 6 and Table 4. MAKHA, BA, and BBPSO were not able to converge at the 10^{-5} level.

Problem G8 was successfully solved at the 10^{-5} level by IFA, CMAES, ABC, BA, CS, and BBPSO, as shown in Figure 7. Only CMAES and CS solved the problem down to the 10^{-7} levels, with CMAES being one order of magnitude more efficient. In fact, CMAES was quite efficient at all tolerance levels, as shown by the NFE numbers in Table 4.

The convergence profiles of the four phase equilibrium problems (G4, G6, G7, and G8) indicated that CS is the most reliable of all algorithms as it was the only one to be able to solve all problems down to the 10^{-7} tolerance level. CMAES was the most efficient as it required one order of magnitude less NFE to solve three of the four problems down to the same tolerance level. However, CMAES failed to solve the two-variable problem that was successfully solved by all other methods, except IFA, down to the 10^{-7} level.

4.3. Reactive Phase Equilibrium Problems. Regardless of the number of variables, the reactive phase equilibrium problems are more difficult than the nonreactive phase equilibrium problems because the chemical reaction equilibria constraints must be satisfied. Problem R4, see Figure 8, was successfully solved down to the 10^{-5} tolerance level by CS, which was also able to converge to the global minimum at the 10^{-6} and 10^{-7} levels. MAKHA, CMAES, BA, MCSS, and BBPSO were not able to arrive even at a level of 10^{-3} from the global minimum. Similarly, CMAES and BA were not able to reach the 10^{-3} level for Problem R7. However, MAKHA, IFA, CS, and BBPSO converged down to 10^{-7} distance from the global minimum, with IFA being the most efficient down to the 10^{-5} level and BBPSO at the 10^{-6} and 10^{-7} levels.

The complete failure of CMAES to solve reactive phase equilibrium problems is remarkable. CMAES functions extremely well in certain types of problems and extremely bad in others. On the other hand, CS solved the reactive phase equilibrium problems just as it reliably solved all other problems in this study. Since CS uses Lévy walk, instead of random walk, in its global search, it can explore the search space more efficiently and avoid entrapment in local minima, as was demonstrated by our results. However, CS requires significantly large NFE to allow it to converge to the global minimum. Any attempt to improve CS performance should target its slow convergence behavior.

Our results are summarized in the PP plots of Figures 9 and 10. The reliability ranking, as extracted from the reliability PP plot of Figure 9, is as follows. CS is the most reliable, followed by CMAES, BBPSO, and MCSS, on the second level. The third level contains MAKHA, ABC, IFA, and BA, in that order. The efficiency ranking starts with CMAES, BBPSO, and ABC. The second level contains CS and IFA. The third level contains BA, MAKHA, and MCSS.

5. Conclusions

In this study, we have selected eight promising nature-inspired metaheuristic algorithms for the solution of nine

difficult phase stability and phase equilibrium problems. These thermodynamic problems were systematically solved by the different metaheuristics and the results were tracked and compared. The results clearly show that CS is the most reliable of all tested optimization methods as it successfully solved all problems down to the 10^{-5} tolerance from the global minima. Any attempt to improve the performance of CS should target its slow convergence behavior. Recently developed CS variants [33] could provide more efficient performance for the solution of phase stability and phase equilibrium problems. These variants could be evaluated in a future study in an attempt to find the most reliable and efficient algorithm for this application. On the other hand, CMAES was the most efficient in finding the solution for the problems it was able to solve. However, it was not able to converge to the global minimum for some of the tested thermodynamic problems.

Nomenclature

A:	Parameter used in BA
ABC:	Artificial bee colony
B:	Parameter used in MAKHA algorithm
BA:	Bat algorithm
BBPSO:	Bare bones particle swarm optimization
C:	Parameter used in MAKHA algorithm
CMAES:	Covariance matrix adaptation evolution strategy
CMCR:	Parameter used in MCSS
CMS:	Parameter used in MCSS
CS:	Cuckoo search
C_i :	Empirical and experimental constants—used in MAKHA
D:	Dimension of the problem, Parameter used in MAKHA Algorithm
D_{max} :	Maximum diffusion speed—used in MAKHA
F, F_{obj} :	Objective function
FA:	Firefly algorithm
G:	Gibbs free energy
g^* :	Global minimum
I:	A counter
i, j :	Index of the component, or index used in an algorithm
IFA:	Intelligent firefly algorithm
K_{eq} :	Equilibrium constant
m :	Dimension of the problem, that is, number of variables
MAKHA:	Monkey Algorithm-Krill Herd Algorithm hybrid
MCSS:	Magnetic charged system search
N:	The number of iterations (criterion maximum number)
n :	Population number
NFE:	Number of function evaluations
N_{max} :	Maximum induced speed
N_p :	Population size (number of points)
n_p :	Number of problems

\mathbf{n}_{ref} :	Column vector of moles of each of the reference components
n_s :	Number of solvers
NV:	Dimension of the problem, that is, number of variables
p :	Parameter used in CS
PP:	Performance profile
R :	The half-range of boundaries between the lower boundary and the upper boundary of the decision variables (X)—used in MAKHA
r :	Parameter used in BA
r_{ps} :	The performance ratio
TPDF:	Tangent plane distance function
t_{ps} :	Performance metric
V_f :	Foraging speed—used in MAKHA
w_f :	Inertia weight—used in MAKHA
X :	Decision variables
x :	Decision variables
Y :	Decision variables
y :	Decision variables
c :	Number of components.

Greek Letters

ζ :	The simulating value of r_{ps}
ρ :	The counter of ρ points
ζ_{max} :	The maximum assumed value of r_{ps}
β :	Transformed decision variables used instead of mole fractions
ε :	Vector of random numbers in FA
ϕ :	Fugacity coefficient, Fraction of top fireflies to be utilized in the move—Parameter used in IFA
γ :	Activity coefficient, Parameter used in IFA
π :	Number of phases
μ :	Chemical potential
α_o :	Parameter used in IFA
β_o :	Parameter used in IFA
ρ :	The cumulative probabilistic function of r_{ps} and the fraction of the total number of problems
σ :	Parameter used in CMAES.

Subscripts

F :	Feed
I :	Index for the components in the mixture
min:	Minimum value
O :	Initial value of the parameter
y :	At composition y
z :	At composition z .

Superscripts

0:	Pure component.
----	-----------------

Conflict of Interests

The authors of this paper do not have a direct financial relation that might lead to a conflict of interests with regard to this paper.

References

- [1] A. Bonilla-Petriciolet, G. P. Rangaiah, and J. G. Segovia-Hernández, "Constrained and unconstrained Gibbs free energy minimization in reactive systems using genetic algorithm and differential evolution with tabu list," *Fluid Phase Equilibria*, vol. 300, no. 1-2, pp. 120–134, 2011.
- [2] H. Zhang, A. Bonilla-Petriciolet, and G. P. Rangaiah, "A review on global optimization methods for phase equilibrium modeling and calculations," *The Open Thermodynamics Journal*, vol. 5, pp. 71–92, 2011.
- [3] G. P. Rangaiah, *Stochastic Global Optimization: Techniques and Applications in Chemical Engineering*, World Scientific, Singapore, 1st edition, 2010.
- [4] J. A. Fernández-Vargas, A. Bonilla-Petriciolet, and J. G. Segovia-Hernández, "An improved Ant Colony Optimization method and its application for the thermodynamic modeling of phase equilibrium," *Fluid Phase Equilibria*, vol. 353, pp. 121–131, 2013.
- [5] Y. Zhu, H. Wen, and Z. Xu, "Global stability analysis and phase equilibrium calculations at high pressures using the enhanced simulated annealing algorithm," *Chemical Engineering Science*, vol. 55, no. 17, pp. 3451–3459, 2000.
- [6] A. Bonilla-Petriciolet and J. G. Segovia-Hernández, "A comparative study of particle swarm optimization and its variants for phase stability and equilibrium calculations in multicomponent reactive and non-reactive systems," *Fluid Phase Equilibria*, vol. 289, no. 2, pp. 110–121, 2010.
- [7] A. Bonilla-Petriciolet, G. P. Rangaiah, J. G. Segovia-Hernández, and J. E. Jaime-Leal, "Stochastic global optimization: techniques and applications in chemical engineering," *World Scientific*, vol. 2, pp. 413–463, 2010.
- [8] I. Rahman, A. K. Das, R. B. Mankar, and B. D. Kulkarni, "Evaluation of repulsive particle swarm method for phase equilibrium and phase stability problems," *Fluid Phase Equilibria*, vol. 282, no. 2, pp. 65–67, 2009.
- [9] M. Srinivas and G. P. Rangaiah, "Differential evolution with tabu list for global optimization and its application to phase equilibrium and parameter estimation problems," *Industrial and Engineering Chemistry Research*, vol. 46, no. 10, pp. 3410–3421, 2007.
- [10] M. Srinivas and G. P. Rangaiah, "A study of differential evolution and tabu search for benchmark, phase equilibrium and phase stability problems," *Computers and Chemical Engineering*, vol. 31, no. 7, pp. 760–772, 2007.
- [11] Y. S. Teh and G. P. Rangaiah, "Tabu search for global optimization of continuous functions with application to phase equilibrium calculations," *Computers and Chemical Engineering*, vol. 27, no. 11, pp. 1665–1679, 2003.
- [12] G. P. Rangaiah, "Evaluation of genetic algorithms and simulated annealing for phase equilibrium and stability problems," *Fluid Phase Equilibria*, vol. 187–188, pp. 83–109, 2001.
- [13] G. Nagatani, J. Ferrari, L. Cardozo Filho et al., "Phase stability analysis of liquid-liquid equilibrium with stochastic methods," *Brazilian Journal of Chemical Engineering*, vol. 25, no. 3, pp. 571–583, 2008.

- [14] A. Bonilla-Petriciolet, R. Vázquez-Román, G. A. Iglesias-Silva, and K. R. Hall, "Performance of stochastic global optimization methods in the calculation of phase stability analyses for nonreactive and reactive mixtures," *Industrial and Engineering Chemistry Research*, vol. 45, no. 13, pp. 4764–4772, 2006.
- [15] M. L. Corazza, L. Cardozo Filho, J. Vladimir Oliveira, and C. Dariva, "A robust strategy for SVL equilibrium calculations at high pressures," *Fluid Phase Equilibria*, vol. 221, no. 1-2, pp. 113–126, 2004.
- [16] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, December 2009.
- [17] S. E. K. Fateen and A. Bonilla-Petriciolet, "Intelligent firefly algorithm for global optimization," *Studies in Computational Intelligence*, vol. 516, pp. 315–330, 2014.
- [18] X. S. Yang, "Firefly algorithm," in *Nature-Inspired Metaheuristic Algorithms*, pp. 79–90, 2007.
- [19] R. Zhao and W. Tang, "Monkey algorithm for global numerical optimization," *Journal of Uncertain Systems*, vol. 2, pp. 165–176, 2008.
- [20] A. H. Gandomi and A. H. Alavi, "Krill herd: a new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, pp. 4831–4845, 2012.
- [21] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003.
- [22] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.
- [23] X.-S. Yang, "A new metaheuristic Bat-inspired algorithm," *Studies in Computational Intelligence*, vol. 284, pp. 65–74, 2010.
- [24] A. Kaveh, M. A. M. Share, and M. Moslehi, "Magnetic charged system search: a new meta-heuristic algorithm for optimization," *Acta Mechanica*, vol. 224, pp. 85–107, 2013.
- [25] A. Kaveh and S. Talatahari, "A novel heuristic optimization method: charged system search," *Acta Mechanica*, vol. 213, no. 3, pp. 267–289, 2010.
- [26] H. Zhang, D. D. Kennedy, G. P. Rangaiah, and A. Bonilla-Petriciolet, "Novel bare-bones particle swarm optimization and its performance for modeling vapor-liquid equilibrium data," *Fluid Phase Equilibria*, vol. 301, no. 1, pp. 33–45, 2011.
- [27] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.
- [28] S.-E. K. Fateen, A. Bonilla-Petriciolet, and G. P. Rangaiah, "Evaluation of covariance matrix adaptation evolution strategy, shuffled complex evolution and firefly algorithms for phase stability, phase equilibrium and chemical equilibrium problems," *Chemical Engineering Research and Design*, vol. 90, pp. 2051–2071, 2012.
- [29] V. Bhargava, S. E. K. Fateen, and A. Bonilla-Petriciolet, "Cuckoo Search: a new nature-inspired optimization method for phase equilibrium calculations," *Fluid Phase Equilibria*, vol. 337, pp. 191–200, 2013.
- [30] A. O. Elnabawy, S. E. K. Fateen, and A. Bonilla-Petriciolet, "Phase stability analysis and phase equilibrium calculations in reactive and non-reactive systems using Charged System Search algorithms," *Industrial & Engineering Chemistry Research*, vol. 53, pp. 2382–2395, 2014.
- [31] M. L. Michelsen, "The isothermal flash problem. Part I. Stability," *Fluid Phase Equilibria*, vol. 9, no. 1, pp. 1–19, 1982.
- [32] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical Programming B*, vol. 91, no. 2, pp. 201–213, 2002.
- [33] I. F. D. Fister Jr. and I. Fister, "A comprehensive review of cuckoo search: variants and hybrids," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 4, pp. 387–409, 2013.

Research Article

Cloud Model Bat Algorithm

Yongquan Zhou, Jian Xie, Liangliang Li, and Mingzhi Ma

College of Information Science and Engineering, Guangxi University for Nationalities, Nanning, Guangxi 530006, China

Correspondence should be addressed to Yongquan Zhou; yongquanzhou@126.com

Received 18 March 2014; Accepted 22 April 2014; Published 19 May 2014

Academic Editor: Xin-She Yang

Copyright © 2014 Yongquan Zhou et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Bat algorithm (BA) is a novel stochastic global optimization algorithm. Cloud model is an effective tool in transforming between qualitative concepts and their quantitative representation. Based on the bat echolocation mechanism and excellent characteristics of cloud model on uncertainty knowledge representation, a new cloud model bat algorithm (CBA) is proposed. This paper focuses on remodeling echolocation model based on living and preying characteristics of bats, utilizing the transformation theory of cloud model to depict the qualitative concept: “bats approach their prey.” Furthermore, Lévy flight mode and population information communication mechanism of bats are introduced to balance the advantage between exploration and exploitation. The simulation results show that the cloud model bat algorithm has good performance on functions optimization.

1. Introduction

Metaheuristics is a new method for stochastic optimization; in recent years, more and more different metaheuristic algorithms have been proposed, such as particle swarm optimization (PSO) [1], differential evolution (DE) [2], and bat algorithm (BA) [3], and some novel metaheuristic algorithms are proposed. The bat algorithm was proposed by Xin-She Yang in 2010, which is inspired by the echolocation behaviour of microbats. The bat algorithm controls the size and orientation of bats moving speed through adjusting the frequency of each bat and then moves to a new location; the intensive local search is controlled by the loudness and pulse emission rate. To some extent, PSO is a special case of suitably simplified BA. Due to the fact that BA combines with the advantages of swarm intelligence, which utilizes a balanced combination of the advantages of the standard PSO and the intensive local search controlled by the loudness and pulse rate, BA is widely researched in different field applications. BA has some advantages over other algorithms, and the number of adjustable parameters is fewer. Consequently, BA has been used for solving engineering design optimization [4–6], classifications [7], fuzzy cluster [8], prediction [9], neural networks, and other applications.

The cloud model is proposed by Li et al. in 1995, which is a model of the uncertain transition between a linguistic

term of qualitative concept and its numerical representation [10]. In recent years, the cloud model is applied in the field of metaheuristics, such as cloud model based genetic algorithm (CGA) [11] and cloud model based evolutionary algorithm (CBEA) [12, 13]. In this paper, the bat algorithm was used for reference, the echolocation mechanism based on cloud model was remodeled, and two mechanisms were introduced: population information communicating of each individual and random Lévy flight; a cloud model bat algorithm (CBA) was proposed, and the purpose is to improve the convergence rate and precision of bat algorithm. At the end of this paper, combination strategies and parameter settings of CBA are discussed, several appropriate parameters are selected, and eight typical benchmark functions are tested, and the test results show that the proposed algorithm is feasible and effective.

2. Behaviors of Bats and Cloud Model

2.1. Flight and Echolocation of Bats. Bats are the only volant mammals in the world; after tens of millions of years of evolution, there are nearly 1,000 species of bats. Bats have powered flight ability, which is much more complex than glide; their flight can generate complex aerodynamic tracks, and the flight is accompanied with local self-similarity [14]. Many microbats have amazing echolocation; these bats can

```

Objective function  $f(x), x = [x_1, x_2, \dots, x_d]^T$ 
Initialize the bat population  $x_i (i = 1, 2, \dots, n)$  and  $v_i$ 
Define pulse frequency  $f_i$  at  $x_i$  Initialize pulse rates  $r_i$  and the loudness  $A_i$ 
While ( $t < \text{Max number of iterations}$ )
    Generate new solutions by adjusting frequency,
    and updating velocities and locations/solutions [(1)]
    if ( $\text{rand} > r_i$ )
        Select a solution among the best solutions
        Generate a local solution around the selected best solution
    end if
        Generate a new solution by flying randomly
    if ( $\text{rand} < A_i \ \& \ f(x_i) < f(x_*)$ )
        Accept the new solutions
        Increase  $r_i$  and reduce  $A_i$ 
    end if
        Rank the bats and find the current best  $x_*$ 
    end while
    Postprocess results and visualization.

```

ALGORITHM 1: Pseudocode of the bat algorithm (BA) [3].

emit a very loud and short sound pulse and receive the echo that reflects back from the surrounding objects by their extraordinary big auricle. Then, they analyze this feedback information of echo in their subtle brain. They not only can discriminate direction for their own flight pathway according to the echo but also can distinguish different insects and obstacles, to hunt prey and avoid collision effectively in the day or night. Bats minimize the conspicuousness of their echolocation call to potential insect prey by reducing call intensity and by changing the frequencies in the call [15]. Furthermore, the echolocation signal that one individual bat uses to collect information can simultaneously serve as a communication function, allowing, for example, group members to remain in contact with one another. Echolocation call plays a crucial and hitherto underestimated role for social communication in a highly mobile and gregarious nocturnal mammal and thus facilitates social communication in bats population [16].

2.2. Bat Algorithm. In simulations, they use virtual bats naturally, to define the updated rules of their positions x_i and velocities v_i in a D -dimensional search space. The new solutions x_i^t and velocities v_i^t at time step t are given by

$$\begin{aligned}
 f_i &= f_{\min} + (f_{\max} - f_{\min})\beta, \\
 v_i^t &= v_i^{t-1} + (x_i^t - x_*)f_i, \\
 x_i^t &= x_i^{t-1} + v_i^t,
 \end{aligned} \tag{1}$$

where $\beta \in [0, 1]$ is a random vector drawn from a uniform distribution. Here, x_* is the current global best location (solution) which is located after comparing all the solutions among all the n bats.

For the local search part, once a solution is selected among the current best solutions, a new solution for each bat is generated locally using random walk:

$$x_{\text{new}} = x_{\text{old}} + \varepsilon A_t, \tag{2}$$

where $\varepsilon \in [-1, 1]$ is a random number, while $A_t = \langle A_i^t \rangle$ is the average loudness of all the bats at this time step.

Furthermore, the loudness A_i and the rate r_i of pulse emission have to be updated accordingly as the iterations proceed. These formulas are

$$\begin{aligned}
 A_i^{t+1} &= \alpha A_i^t, \\
 r_i^{t+1} &= r_i^0 [1 - \exp(-\gamma t)],
 \end{aligned} \tag{3}$$

where α and γ are constants.

Based on these approximations and idealization, the basic steps of the bat algorithm [3] can be summarized as the pseudocode shown in Algorithm 1.

2.3. Lévy Flight. Lévy flight is a random walk in which the step-lengths have a probability distribution that is heavy-tailed. Lévy flight has several properties: "heavy tails," statistical self-similarity, random fractal characteristics, and infinite variance with an infinite mean value [17]. Lévy distribution, Gaussian distribution, and Cauchy distribution, which is a α stable distribution; however, probability density function (PDF) curves of Gaussian distribution and the Cauchy distribution are symmetrical; Lévy distribution is not symmetrical [18]. Probability density function of Lévy distribution on $x > \mu$ is

$$\text{Lévy} \sim f(x) = \sqrt{\frac{C}{2\pi}} \frac{e^{-C/2(x-\mu)}}{(x-\mu)^{3/2}}, \tag{4}$$

where μ is the location parameter and C is the scale parameter. PDF curve of the three distributions is presented in Figure 1.

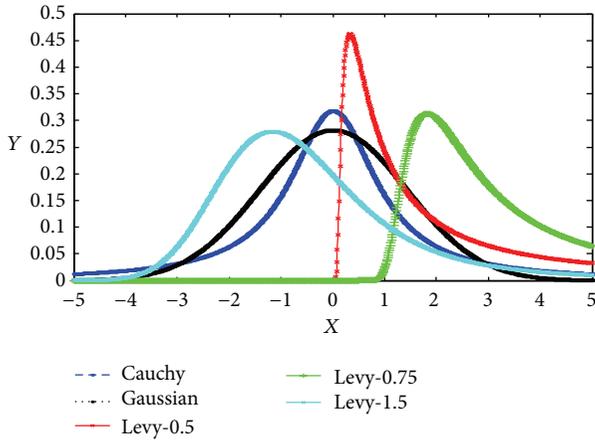


FIGURE 1: The PDF curve of the three distributions.

Studies have shown that flight behaviour of many animals and insects has demonstrated the typical characteristics of Lévy flights. A recent study by Reynolds and Frye shows that fruit flies explore their landscape using a series of straight flight paths punctuated by a sudden 90° turn, leading to a Lévy flight-style intermittent scale free search pattern [19]. Studies on human behaviour such as the Ju/'hoansi hunter-gatherer foraging patterns also show the typical feature of Lévy flights [20]. Subsequently, due to the remarkable properties of stable Lévy distribution, Lévy flight has been applied to optimization and optimal search [21], and preliminary results show its promising capability.

2.4. Cloud Model. Cloud model build a transformational bridge between a linguistic term of qualitative concept and quantitative representation, which reflects randomness, fuzziness, and the relationship between randomness and fuzziness of uncertainty in knowledge representation [22, 23]. The cloud and cloud droplets are defined as follows.

Let U be the set $U = \{x\}$, as the universe of discourse, and let C be a linguistic term associated with U . The membership degree of x in U to the linguistic term C , $\mu(x)$, is a random number with a stable tendency. $\mu(x)$ takes the values in $[0, 1]$. A membership cloud, or compatibility cloud, is a mapping from the universe of discourse U to the unit interval $[0, 1]$. That is,

$$\begin{aligned} \mu(x) : U &\longrightarrow [0, 1], \\ \forall x \in U, \quad x &\longrightarrow \mu(x). \end{aligned} \tag{5}$$

The distribution of x in universe of discourse U is called cloud and each x is called a drop of cloud [22].

A normal cloud is defined with three digital characteristics, expected value Ex , entropy En , and hyper entropy He and a cloud, namely, $C(Ex, En, He)$. Expectation Ex is the position at U corresponding to the center of gravity of the cloud. In other words, the element Ex in the universe of discourse is fully compatible with the linguistic term. The entropy En is a measure of the coverage of the concept within the universe of discourse. In other words, En is defined by

the bandwidth of the mathematical expected curve (MEC) of the normal cloud showing how many elements in the universe of discourse could be accepted to the linguistic term, the greater En , and the broader coverage. It can be also considered as a measure of fuzziness of the concept, representing the scope of the universe of discourse that can be accepted by the concept. The hyper entropy He is the entropy of the entropy En . It is a measure of dispersion of the cloud drops; it can be used a measure of thickness of the cloud, which not only reflects the randomness of samples appearing that represent qualitative concepts value but also reveals the relatedness between fuzziness and randomness.

Normal cloud model makes full use of the universality of the normal distribution and normal membership function, which not only broaden the formation conditions of the normal distribution but also make the normal membership function be the expectation of the random membership degree; the randomness and fuzziness are represented uniformly by entropy and then the theoretical basis of universality of the normal cloud model is established [24]. Cloud model has the 3σ characteristics; there are 99.7% drops of cloud located in $[Ex - 3En, Ex + 3En]$. These drops of cloud are generated by the normal cloud generator. Atomized feature of the cloud model: the drops of cloud spread around while the hyper entropy is increasing, but many drops still stand in the central area of the cloud, which can be used to adjust the strategies of the evolution and help to escaping from local optima [11]. The clouds with different digital characteristics are depicted in Figure 2.

3. Cloud Model Bat Algorithm

Bats prey by emitting pulse with a certain frequency and detection of the echo; they communicate with each other using echolocation call. This paper assimilates its principle to idealize some of the echolocation characteristics of micro-bats. Based on the excellent characteristics of cloud model on uncertainty knowledge representation, a bat algorithm based on cloud model was proposed (cloud model bat algorithm, CBA).

3.1. Knowledge Representation of Bat Cloud. In order to depict the CBA, the habits of bats are used for reference, taking advantage of the excellent properties of cloud model. Firstly, representation of relevant knowledge needs to be described; several concepts certain about CBA were given as follows.

(1) *Optimizing Generation.* Optimizing generation indicates the number of iteration circles in the algorithm; each iteration circle may include several times replacement of population, simply, namely, t .

(2) *Individual.* In CBA, each bat is treated as an individual; when it is in flight, the position of each bat x_i^t signifies a candidate solution of optimization problem, where i is the number of individuals and t is the optimizing generation. For the high-dimensional optimization, x_i^t represent a vector under high-dimensional space, correspondingly, where each

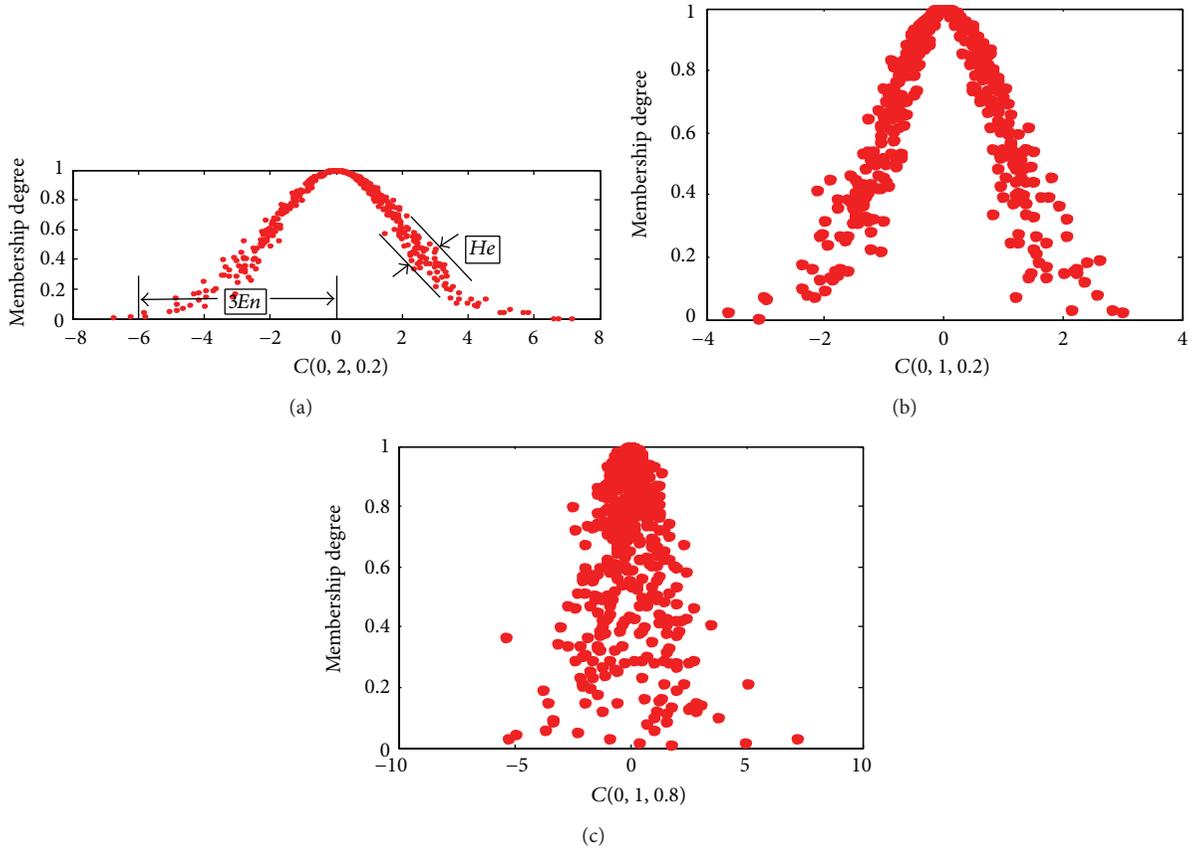


FIGURE 2: The cloud with different digital characteristics. $C(0, 2, 0.2)$ denote a cloud, 0 is expectation, 2 is entropy, and 0.2 is hyper entropy.

dimension denotes an attribute of the solution of optimization problem.

(3) *Fitness Function.* Fitness function denotes adaptation degree of each individual aiming at their located environment in the community. It is used to evaluate the individual and decides which individual to retain or eliminate. Fitness function usually is the expression of costs, profitability, variance, and so on.

(4) *Population Bats Cloud.* The bats live and prey together, and many bats constitute a community. A cloud was generated, which is to represent the distribution characteristics of the same dimension of all individuals, called population bats cloud, namely, $Pbc_j(Ex, En, He)$, where j represents the j th dimension of population and Ex, En , and He are three digital characteristics of cloud model.

(5) *Individual Experience.* It denotes these individuals that are able to remember their own history during the process of optimization. In the proposed algorithm, bats can memorize their own best location x_{pbest} during moving. Its main purpose is to guide the flight of bat and to promote the communication among the population.

(6) *Population Elite.* In this proposed algorithm, population elite denotes the position of the optimal individual, namely,

x_{gbest} ; the population elite x_{gbest} will be saved and be used in swarm information communication.

3.2. *Cloud Model Bat Search Algorithm.* In this paper, at the basis of original BA and the habits of bats, based on the cloud model and Lévy flights, cloud model bat algorithm is proposed under idealized simulation of echolocation of bats. For simplicity, some idealized rules are as follows.

- (1) Using the echolocation, bats not only can identify the direction, measure the distance, and determine the current status of their prey but also can avoid collision, distinguish obstacles, and prey from background clutter. This paper only simulates that bats search for a prey using echolocation mechanism in a search space under ideal environment, where the position of prey means an optimal solution of the problem; each position of bats indicates a candidate solution of optimization problem. Bats may not prey their target, but they gradually approach the target, close to the prey, approximately regard as successful preying under a certain tolerance.
- (2) Each bat flies randomly with frequency f_i^t ; the position x_i^t moves under the adjustment of frequency f_i^t . The frequency f_i^t resembles an adjustment coefficient

of step length, and frequency f_i^t of sound pulse is changeable, where t is the optimizing generation.

- (3) The adjustment of frequency f caused the change of the wavelength λ (this is because of the fact that $\lambda f = v$ is a constant and v is the speed of sound in air; typically, $v = 340$ m/s); such wavelengths λ are in the same order of their prey sizes and help to locate the target. Generally, the frequency f is in a range $[f_{\min}, f_{\max}]$, and each individual can communicate information with others by echolocation call in a population.
- (4) Each bat emits sonic pulse with emission rate $R_i \in [0, 1]$ and loudness Ld_i . At the beginning of prey, bats have a smaller R_i and larger Ld_i . During the process of locating prey, the pulse emission rate increases and loudness reduces once the bat searches for target traces, which is figuratively indicated by “bat is approaching the target.”
- (5) In exploration of bats for prey, their flight features are accompanied by typical Lévy flight characteristic; many insects and animals have it as well. Exploration and traces its of to detect potential prey traces of a random flight.

On the basis of the above mentioned idealized rules, the properties of the cloud model are utilized which represents the membership degree of qualitative concept and reveals the relationship between randomness and fuzziness in uncertainty knowledge representation. According to normal cloud model generator with expectations, entropy, and hyper entropy, many drops of the cloud with quantitative transformation value corresponding to qualitative concept are produced. In this paper, updating the position of bats by cloud model, swarm information communication in each individual and random Lévy flights are introduced, a cloud model bat algorithm is proposed, and the steps of CMBA algorithm can be summarized as follows:

- (1) *Initialization*. Randomly initialize the position of each bat in the population and relevant parameters.
- (2) *Initial Evaluation*. Evaluate these initial positions using fitness function, and find out a population elite x_{gbest} .
- (3) *Bats Cloud Updating*. Generate bats cloud based on cloud model, and update the position of bats.
- (4) *Swarm Information Communication*. Information communication of bat population adopts a differential operator that is similar to mutation strategy “DE/best/2” in differential algorithm.
- (5) *Bats Random Lévy Flight*. Each bat randomly flights using Lévy flight.
- (6) *Population Evaluation*. For each population in steps (3)–(5), evaluate each individual by fitness function and find out

and update the individual experience x_{pbest} for each bat and population elite x_{gbest} in each step.

- (7) *Pulse Emission Rate and Loudness Update*. The rate of pulse emission R_i and loudness Ld_i for each bat need to update when the achieved optimal solution after steps (3)–(5) is better than the optimal solution of last generation.
- (8) *Termination Judgment*. $t = t + 1$; execute steps (3)–(7) until t reaches a predefined maximum number of optimizing generation.

In this algorithm framework, three problems need to be solved: first of all, the formation of bat cloud model, second information communication of bat population, and third the updating of R, Ld .

3.2.1. Formation of Population Bats Cloud Model. This paper simulates the moving of bats when several bats pursue and capture prey. Each bat expects to move toward the direction of prey (the optimal solution). In the search space, the entire population is trying to approximate the optimal solution; the position x_i^t of each individual should move toward the optimum position. Consequently, the same dimensions of population have stable tendency. However, each individual has their own feature, and the implementations of position updating are random for each individual. Therefore, the characteristics of approximated process that bats have to approximate prey are simulated by cloud model. Sequentially, they adapt bat cloud model to depict the qualitative concept: “bats approach their prey.”

Normal cloud model of bat approach process utilizes the characteristics of cloud model that are the uncertain transition between qualitative and quantitative. The populations bats cloud $Pbc_j(Ex, En, He)$ analogize to cloud $C(Ex, En, He)$, where the expected value Ex is the j th dimension of population elite x_{gbest} , the entropy En is the average loudness of all bats, and the hyper entropy He is the average pulse emission rate of all bats. The population bats cloud $Pbc_i(Ex, En, He)$ is a 1-dimensional normal cloud. In order to update the position of each individual, each dimension of new individual is generated by randomly selecting several drops of the cloud from cloud cluster, and then calculate the result which is mean of the membership degree of each selected drop multiplied by expected value Ex . The membership degree of each drop is the certainty degree of approximation expectation Ex . The computational formula is described as follows:

$$x_{ii}^{t+1} = \text{AVG} \left(\sum Ex \times RS(pbc_i(Ex, En, He)) \right), \quad (6)$$

where i denotes i th individual, j denotes j th dimension, $Ex = x_{gbest,j}$, $En = \text{AVG}(\sum \text{Rate}_i)$, $He = \text{AVG}(\sum \text{Loudness}_i)$, $RS(\cdot)$ denotes a function of the randomly selected several records, and $\text{AVG}(\cdot)$ denotes averaging function.

The pulse emission rate R_i increases and loudness Ld_i decreases while the iteration is increasing, and the entropy En and hyper entropy He therewith update. Consequently, different cloud clusters are generated, so those individuals

gradually approach the target. Position of the bat is updated by population bats cloud $Pbc_j(Ex, En, He)$, which quantitatively represents the qualitative concept that bat approaches target. Sequentially, which reflect the determine tendency of swarm optimizing, meanwhile show the fuzziness and randomness of uncertainty knowledge representation.

The proposed algorithm introduces information communication in bat population under idealized conditions, which assure that the entire bat colony gets helpful information by communicating experience among individuals in a bat population. This mechanism guides these bats that are approaching prey fast.

Bats can emit sound pulses with certain frequency range. Generally, the frequency f is in a range $[f_{\min}, f_{\max}]$ and the typical range is $[0, 1]$ in implementation. This paper defines the frequency f updating formula as follows:

$$f_{1i}^t = \left((f_{1,\min} - f_{1,\max}) \frac{t}{n_t} + f_{1,\max} \right) \beta_1, \quad (7)$$

$$f_{2i}^t = \left((f_{2,\max} - f_{2,\min}) \frac{t}{n_t} + f_{2,\min} \right) \beta_2, \quad (8)$$

where $\beta_1, \beta_2 \in [0, 1]$ is a random vector drawn from a uniform distribution, $f_{1,\max} = f_{2,\max} = f_{\max}$, $f_{1,\min} = f_{2,\min} = f_{\min}$, and n_t is a constant. The frequency f would be analogous to an adjustable parameter. The step length of individual moving is adjusted by adjusting frequency f . Meanwhile, it can be interpreted as bats adjusting their own position by adjusting their own frequency and communicating with other bats. Information communication of bat population adopts a differential operator that is similar to mutation strategy "DE/best/2" in differential algorithm, which is described as follows:

$$x_i^{t+1} = x_{g_{\text{best}}}^t + f_{1i}^t (x_{r1}^t - x_{r2}^t) + f_{2i}^t (x_{r3}^t - x_{r4}^t), \quad (9)$$

where $x_{g_{\text{best}}}^t$ represents the current population elite after updating by bat cloud updating and x_{ri}^t is i th individual randomly selected in the population after bat cloud updating.

In addition, the above mentioned mechanism accelerates the convergence rate, while the Lévy flight behavior is introduced to greatly ensure the swarm diversity against the premature convergence. Random Lévy flights are manipulated at the basis of individual experience $x_{p_{\text{best}}}^t$, where $x_{p_{\text{best}}}^t$ represents the current individual experience after swarm information communication. The random Lévy flights are used to improve the individual ability to escape from the local optima; simultaneously, this mechanism also assures the intensification. The detailed description is as follows:

$$x_i^{t+1} = x_{p_{\text{best}}}^t + \mu \times \text{sign} [\text{rand} - 0.5] \oplus \text{Lévy}, \quad (10)$$

where μ is a random parameter drawn from a uniform distribution, $\text{sign} \oplus$ means entry-wise multiplications, $\text{rand} \in [0, 1]$, and random step length Lévy obeys Lévy distribution.

3.2.2. Method of Pulse Emission Rate and Loudness Updating. The pulse emission rate R_i and loudness Ld_i of each bat will

be adjusted suitably when it moves to a better position than last generation $t - 1$. In this paper, the updating formulas adopt (11). It is worth noting that the loudness and emission rates will be updated only if the final population elite $x_{g_{\text{best}}}^t$ in current generation are better than the final population elite $x_{g_{\text{best}}}^{t-1}$ in last generation:

$$Ld_i^{t+1} = \alpha Ld_i^t, \quad (11)$$

$$R_i^{t+1} = \frac{1}{1 + e^{-(10/t_{\max}) \times (t - (t_{\max}/2) + R_i^t)}},$$

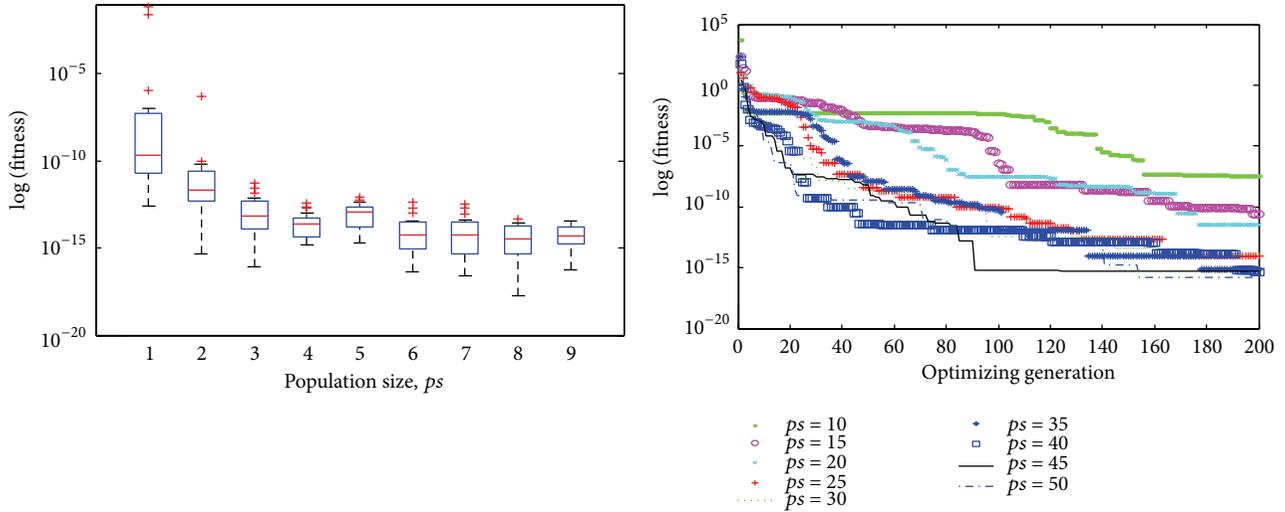
where $\alpha \in [0, 1]$ is a constant, t is optimizing generation, t_{\max} denotes maximum optimizing generation, and R_i^1 denotes initial pulse emission rate of each bat.

Cloud model bat algorithm is inspired by the behavior of bat; original BA and cloud model are used for reference, based on the properties of cloud model and echolocation of bat in foraging behavior, to remodel the algorithm framework, defining several idealized rules and constructing optimizing mechanism; a cloud model bat algorithm is proposed. The cloud model bat algorithm is different from the original bat algorithm, which uses echolocation predation mechanism of bats as the starting point and uses the universality of the normal cloud model as the basis. Several predominant mechanisms are integrated organically in the CBA.

For the performance of the proposed algorithm, the populations bats cloud $Pbc_j(Ex, En, He)$ utilizes the information provided by the current optimal solution to generate the drops of cloud. The loudness and pulse emission rate are regarded as entropy En and hyper entropy He , respectively, to control the measure of the coverage and randomness of optimal solution structure. Reduction of the loudness and increasing of the pulse rate emission show that bats approach their target. To quantitatively represent this qualitative concept by population, bats cloud model makes many individual clusters around the current optimal solution and forms a bat cloud, thus exploring much better solutions. This proposed algorithm has strong stability with bat cloud updating, which can gradually approach the optimal solution.

The swarm information communication guides the whole population moving toward the optimal solution. The increase or decrease of frequency f controls the scale of the individual moving forward or backward. Each individual can communicate information with others and ultimately move toward the common goal or direction. On the one hand, (7) implements on the basis of population elite x_i^t , which can accelerate the convergence speed of proposed algorithm; however, it may lead to premature convergence. On the other hand, the mechanism also reflects the importance of Lévy flight.

From (7), we know that premature convergence may take place; from (8), Lévy flight is implemented on the individual experience of population. This randomness of Lévy flight can ensure the diversity of the population against premature convergence. Lévy flight has a certain role in escaping from local optima; meanwhile, based on individual experience of population it can accelerate the convergence rate to some extent.



(a) Box-and-whisker after 50 independent runs with different population size ps , where 1-9 in abscissa axis correspond to $ps = 10, 15, \dots, 50$, respectively
 (b) Iterative curve after one independent run with different population size ps , where 1-9 in abscissa axis correspond to $ps = 10, 15, \dots, 50$, respectively

FIGURE 3: Box-and-whisker diagram and iterative curve about the impact of different population size ps .

The range of loudness Ld_i and pulse emission rate R_i may have influence on the performance of the proposed algorithm; meanwhile, the number of the drops of cloud is no exception. In [13], for the drops of cloud are generated by the normal cloud generator, there are 99.7% located in the interval $[Ex - 3En, Ex + 3En]$. Consequently, initial value of Ld_i initializes $0.5(x_{max} - x_{min})/3$, where x_{max} and x_{min} denote the upper and lower limit of the search space. The range of the drops of cloud that are located around the expectation Ex will reduce while the loudness Ld_i reduces gradually. In addition, the thickness of cloud cluster will increase while the pulse emission rate increases gradually; even the cloud cluster is excessively discrete and represents atomized feature. In [25], the expectation Ex can be approximated by reverse clouds generator and the error is less than 0.01 if only the number of the drops of cloud is more than 10. Similarly, to approximate entropy En and assure relative error less than 0.01, the number of the drops of cloud is more than 100. Consequently, this paper produces 100 drops of cloud in the implementation, as a cloud cluster, and randomly selects a larger sample with 50 drops of cloud from the cloud cluster to fit the structure individual.

4. Simulations and Result Analysis

In order to validate the validity of bat cloud model algorithm, several unconstrained high-dimensional benchmark test functions are selected (simulation platform: Microsoft Windows XP Professional SP3, AMD Athlon (tm) II X4 640 3.00 GHz, 4.00 GB; programming tools: Matlab R2012a).

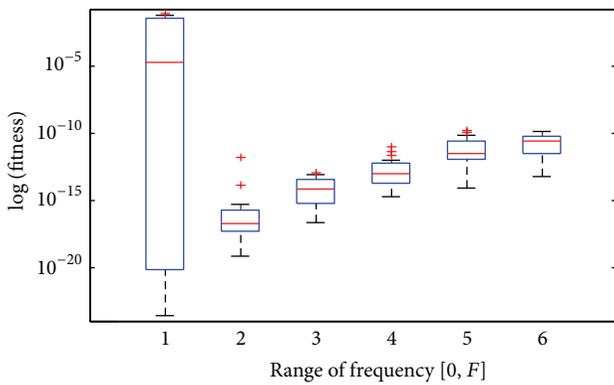
4.1. Parameter Settings and Analysis. In this section, in order to test the sensibility of parameter settings, the 2-dimensional Rosenbrock function was selected. And its global minimum

value is 0 at $(1, 1, \dots, 1)$; the global minimum is inside a long, narrow, and parabolic shaped flat valley. To find the valley is trivial. To converge to the global minimum, however, is difficult. Statistical result of minimum fitness after 50 independent runs was represented by box-and-whisker diagram, and the iterative curve was depicted for once independent run.

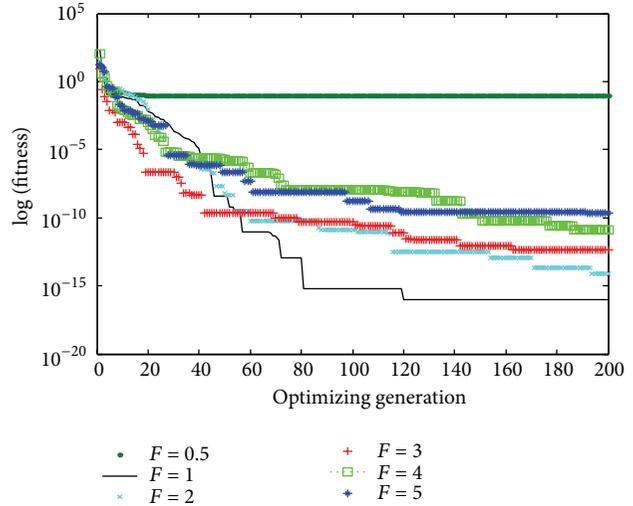
To initialize the parameter frequency $f \in [0, 2]$, $nt = 4000$, select different population size ps for experiment. The purpose is to investigate the influence of the population size for the proposed algorithm. The descriptive statistics of the results are plotted in Figure 3(a), where 1-9 in abscissa axis correspond to $ps = 10, 15, \dots, 50$, respectively. As shown as Figure 3(a), the precision of the optimum value gradually increases while the population size ps increases, and the increment of precision gradually decreases. The precision of the optimum value is low and the extreme outliers will appear when population only includes 10 individuals, which show that the population size is insufficient and the exploring ability is poor. The precision of the optimal value increases properly and the outliers are mild when population size reaches 20. After the population size reaches 30, performance of the proposed algorithm gradually stabilizes, and the incremental extent of the precision is inapparent.

Figure 3(b) is the iterative curves for one independent run of CBA. As shown in the figure, the difference of the optimum value is not outstanding after the population size increases to 30. Considering the precision and calculation, $ps = 45$ is a preferable balance, which has high precision and less calculation; in addition, the convergence rate is relatively fast.

After setting the parameter $ps = 45$, $nt = 4000$ to test the proposed algorithm with different upper limit of frequency $f \in [0, F]$. The purpose is to investigate the impact of the

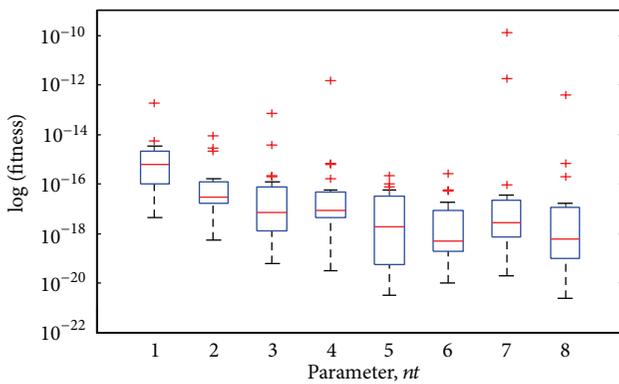


(a) Box-and-whisker after 50 independent runs with different upper limit of frequency $f \in [0, F]$, where 1-6 in abscissa axis correspond to $F = 0.5, 1, 2, \dots, 5$, respectively

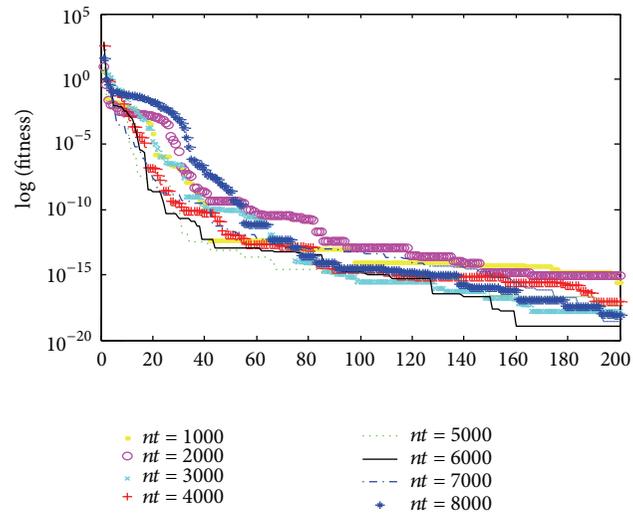


(b) Iterative curve after one independent run with different upper limit of frequency $f \in [0, F]$, where 1-6 in abscissa axis correspond to $F = 0.5, 1, 2, \dots, 5$, respectively

FIGURE 4: Box-and-whisker diagram and iterative curve about the impact of different upper limit of frequency $f \in [0, F]$.



(a) Box-and-whisker after 50 independent runs with different parameter nt , where 1-8 in abscissa axis correspond to $nt = 1000, 2000, \dots, 8000$, respectively



(b) Iterative curve after one independent run with different parameter nt , where 1-8 in abscissa axis correspond to $nt = 1000, 2000, \dots, 8000$, respectively

FIGURE 5: Box-and-whisker diagram and iterative curve about the impact of different parameter nt .

frequency range for the proposed algorithm. The descriptive statistics of the results are plotted in Figure 4(a), where 1-6 in abscissa axis correspond to $F = 0.5, 1, 2, \dots, 5$, respectively. As shown in Figure 4(a), the algorithm is very sensitive to the initial values, while $f \in [0, 0.5]$, the exploring ability is weak and cannot avoid the premature convergence and escape from local minima, and the stability is poor. The performance of algorithm is relatively good when the range is $[0, 1]$; the performance of CBA gradually reduces while the upper limit of frequency increases. Figure 4(b) is the iterative

curves for one independent run of CBA. Figure 4(b) shows that the performance of algorithm and convergence speed are preferable to the other condition.

Confirm the parameter $ps = 45$, $f \in [0, 1]$, and then investigate the impact of the parameter nt for the proposed algorithm. The descriptive statistics of the results are plotted in Figure 5(a), where 1-8 in abscissa axis correspond to $nt = 1000, 2000, \dots, 8000$, respectively. Figure 5(b) is the iterative curves for one independent run of the proposed algorithm with different parameter nt . As shown in Figure 5,

TABLE 1: Benchmarking test functions.

Benchmarks functions	Functions expression	Exact value	x_*	Search space
f_1 : Sphere	$f(x) = \sum_{i=1}^n x_i^2$	$f_{\min} = 0$	$(0, 0, \dots, 0)$	$[-10, 10]$
f_2 : Schwefel	$f(x) = \sum_{i=1}^n x_i + \prod_{i=1}^n x_i $	$f_{\min} = 0$	$(0, 0, \dots, 0)$	$[-10, 10]$
f_3 : Rosenbrock	$f(x) = \sum_{i=1}^n [(x_i - 1)^2 + 100(x_{i+1} - x_i^2)^2]$	$f_{\min} = 0$	$(1, 1, \dots, 1)$	$[-2.408, 2.408]$
f_4 : Ackley	$f(x) = 20 + e - 20 \exp \left[-0.2 \sqrt{\left(\frac{1}{n}\right) \times \sum_{i=1}^n x_i^2} \right] - \exp \left[-0.2 \sqrt{\left(\frac{1}{n}\right) \times \sum_{i=1}^n \cos(2\pi x_i)} \right]$	$f_{\min} = 0$	$(0, 0, \dots, 0)$	$[-30, 30]$
f_5 : Griewangk	$f(x) = \frac{1}{4000} \times \sum_{i=1}^n x_i^2 - \prod_{i=1}^n \cos \frac{x_i}{\sqrt{i}} + 1$	$f_{\min} = 0$	$(0, 0, \dots, 0)$	$[-600, 600]$
f_6 : Rastrigin	$f(x) = 10n + \sum_{i=1}^n [x_i^2 - 10 \cos(2\pi x_i)]$	$f_{\min} = 0$	$(0, 0, \dots, 0)$	$[-5.12, 5.12]$
f_7 : Shubert	$f(x, y) = \left[\sum_{i=1}^5 i \cos(i + (i + 1)x) \right] \cdot \left[\sum_{i=1}^5 i \cos(i + (i + 1)y) \right]$	$f_{\min} \approx -186.7309$	—	$[-10, 10]$
f_8 : Easom	$f(x, y) = -\cos(x) \cos(y) \exp[-(x - \pi)^2 + (y - \pi)^2]$	$f_{\min} = -1$	(π, π)	$[-10, 10]$

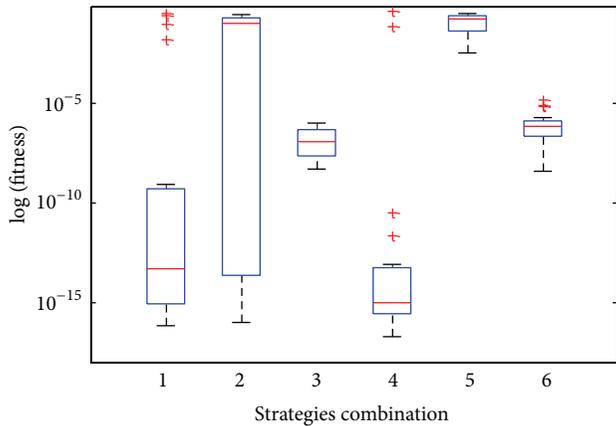


FIGURE 6: Box-and-whisker after 20 runs independently about different strategies combination.

the performance of the algorithm has improved to some extent with gradually increasing parameter nt . In general, the algorithm parameter is not sensitive for parameter nt . Considering the precision of optimal value, convergence speed, and stability, parameter nt around 5000 or 6000 is preferable. In this paper, $nt = 6000$.

4.2. Combination Strategies Analysis. In order to discuss the impact of three strategies (bats cloud updating, swarm information communication, and bats random Lévy flight) after selecting proper parameter, firstly, set bats cloud updating as **C**, swarm information communication as **G**, and bats random Lévy flight as **L** and then select Rosenbrock as test function. The statistical results after 20 times run independently are shown in Figure 6, where 1-6 in abscissa axis correspond to

LGC, **CLG**, **CGL**, **GL**, **CL**, and **CG**, respectively, where the combination of letters represents the combination of different strategies. As shown as Figure 6, **LGC** sometimes can find a better solution, but it is unstable, and several extreme outliers appear sometimes, which represent that the algorithm may be premature convergence. The **CLG** is the most unstable, which is sensitive to the initial position. **GL** can repeatedly find a better solution; however, several extreme outliers appear likewise, which represent that the algorithm lacks stability. **CL** optimizes difficultly and its performance is poor. With **CG** several mild outliers will appear and the performance of **CG** is somewhat less than **CGL**, and the reason is lack of diversity without **L**. Nevertheless, the **CGL** loses trifling precision of the optimum value, but the overall performance is the most stable, and it can always find better solution.

4.3. Experimental Results and Analysis. In order to compare the performance with other algorithms, eight test functions are selected to test CBA convergence. In Table 1, the values listed in the search space column are used to specify the range of the initial random particles' position; the x_* denotes the global optimum, and the f_{\min} is the corresponding fitness value.

In [11], a cloud model based genetic algorithm (CGA) was proposed; CGA is based on both the idea of GA and the properties of randomness and stable tendency of a normal cloud model. In [12], a cloud model based evolutionary algorithm (CBEA) was proposed by Zhang et al., which is based on the outstanding characteristics of the cloud model on the process of transforming a qualitative concept to a set of quantitative numerical values and integrates with the basic principle of evolutionary computation. In [13], cloud based evolutionary algorithm (CBEA) was proposed by Liu

TABLE 2: Experimental results comparison between CGA, CBA, and SAPSO.

Algorithms	f_2	f_3	f_5	f_6	f_7
S-S	[-10, 10]	[-2.048, 2.048]	[-512, 512]	[-10, 10]	[-10, 10]
F-D	5	2	10	15	2
CGA	0.00013	$2.5749e - 08$	0.01170	$1.8529e - 06$	-186.6267
SAPSO	—	$3.5383e - 003$	$5.7773e - 008$	$1.9425e - 004$	—
CBA	$1.9850e - 93$	$1.2683e - 12$	0	0	-186.7309

TABLE 3: Experimental results comparison between CBEA08, CBA, and SAPSO.

Algorithms	f_1	f_5	f_6	f_7	f_8
S-S	[-100, 100]	[-600, 600]	[-5.12, 5.12]	[-10, 10]	[-100, 100]
F-D	10	10	10	2	2
CBEA08	0	0	0	-186.7309088310227	-1
SAPSO	0.04860173	$5.7773e - 008$	$1.9425e - 004$	—	—
CBA	0	0	0	-186.7309088310230	-1

TABLE 4: Experimental results comparison between CBEA09, CBA, and SAPSO.

Algorithms	f_1	f_2	f_3	f_4	f_5
S-S	[-5.12, 5.12]	[-10, 10]	[-30, 30]	[-32.768, 32.768]	[-32.768, 32.768]
CBEA09	$1.1696e - 166$	$5.2927e - 97$	26.178	0	0
SAPSO	0.04860173	—	$3.5383e - 003$	$1.5684e - 003$	$5.7773e - 008$
CBA	0	$2.4707e - 223$	28.879	0	0

et al., who discuss the atomized feature of cloud model; the selection pressure of evolution is adjusted by changing the hyper entropy that is the main factor in atomized feature.

There are many ways to carry out the comparison of algorithm performance with different termination criteria; the preferable approaches is to compare their accuracies for a fixed number of fitness function evaluations FEs . This paper adopts fixed FEs as the termination criterion ($FEs = ps \times t \times N$, where ps is population size, t is optimizing generation, and N is the number of fitness function evaluations in each optimizing generation). In order to compare with different algorithms, search space (SS) and function dimension (FD) of selected benchmark functions are consistent with corresponding algorithm. CBA is run 50 times independently for each function and the mean of the function values found in 50 runs was described in experimental result.

In [11], ps was set as 100, selected different test functions have different optimizing generation t , the minimum FEs is 10^6 , and the maximum FEs is 2×10^7 . In this paper, optimizing generation t is set as 200 in each run for corresponding function. $FEs = 27000$. The experimental results are presented in Table 2, where results of CGA in 30 runs are derived from [11] and results of SAPSO (simulated annealing particle swarm optimization) are derived from [6].

Table 3 shows the results of comparison between CBEA08 and CBA, where results of CBEA08 in 50 runs are derived from [12], and results of SAPSO are derived from [6]. Here $ps = 1000$, $t = 100$, and $FEs = 10^5$. In this paper, for these functions, t is set as 500, corresponding to $FEs = 67500$.

Table 4 shows the results of comparison between CBEA09 and CBA, where results of CBEA09 in 50 runs are derived from [13], and results of SAPSO are derived from [6]. All the selected functions are 30-dimensional function, and the optimizing generation t is set as 500 for all algorithms.

Tables 2, 3, and 4 not only show that the proposed algorithm is feasible and effective but also demonstrate the superior approximation capabilities in high-dimensional space. The proposed algorithm can perform better performance while the optimizing generation t increases gradually. As shown as Table 2, CBA has higher precision than CGA both unimodal and multimodal functions. From Table 3, CBA has similar precision than CBEA08 both unimodal and multimodal function, and several theoretical values of benchmark function can be achieved. Table 4 shows the results that are tested under high-dimensional condition. The CBA can perform with better precision, except for Rosenbrock function which has no best performance. Rosenbrock function has a narrow valley from the perceived local optima to the global optimum.

5. Conclusions

In this paper, the bat algorithm is used for reference; two mechanisms were introduced, population information communicating of each individual and random Lévy flights, to propose a cloud model bat algorithm based on normal cloud model and echolocation mechanism. Cloud model builds a transformational bridge between a linguistic term of qualitative concept and quantitative representation, which

reflects randomness, fuzziness, and the relationship between randomness and fuzziness of uncertainty in knowledge representation. A population bats cloud model $Pbc_j(Ex, En, He)$ was built, which analogizes to cloud $C(Ex, En, He)$, which depicts the qualitative concept: “bats approach their prey.” CBA mainly considers the balance between the global random search and local search and the balance between intensification and diversification. In addition, we discuss the mechanism and parameter set of CBA; several appropriate parameters are set. The simulation results show that the proposed algorithm is feasible, effective, and robust.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This work is supported by the National Science Foundation of China under Grant no. 61165015, Key Project of Guangxi Science Foundation under Grant no. 2012GXNSFDA053028, and Key Project of Guangxi High School Science Foundation under Grant nos. 20121ZD008 and 201203YB072.

References

- [1] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of the 1995 IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, December 1995.
- [2] R. Storn and K. Price, “Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces,” *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
- [3] X. S. Yang, “A new metaheuristic Bat-inspired algorithm,” in *Nature Inspired Cooperative Strategies for Optimization (NICSO '10)*, vol. 284 of *Studies in Computational Intelligence*, pp. 65–74, Springer, 2010.
- [4] X. S. Yang, “Bat algorithm for multi-objective optimisation,” *International Journal of Bio-Inspired Computation*, vol. 3, pp. 267–274, 2011.
- [5] A. H. Gandomi, X. S. Yang, S. Talatahari, A. H. Alavi, and S. Talatahari, “Bat algorithm for constrained optimization,” *Neural Computing and Applications*, vol. 22, pp. 1239–1255, 2013.
- [6] X. S. He, W. J. Ding, and X. S. Yang, “Bat algorithm based on simulated annealing and gaussian perturbations,” *Neural Computing and Applications*, 2013.
- [7] S. Mishra, K. Shaw, and D. Mishra, “A new meta-heuristic bat inspired classification approach for microarray data,” *Procedia Technology*, vol. 4, pp. 802–806, 2012.
- [8] K. Khan, A. Nikov, and A. Sahai, “A fuzzy bat clustering method for ergonomic screening of office workplaces,” in *Proceedings of the 3rd International Conference on Software, Services and Semantic Technologies (S3T '11)*, vol. 101 of *Advances in Intelligent and Soft Computing*, pp. 59–66, Springer, 2011.
- [9] K. Khan and A. Sahai, “A comparison of BA, GA, PSO, BP and LM for training feed forward neural networks in e-learning context,” *International Journal of Intelligent Systems Technologies and Applications*, vol. 4, no. 7, pp. 23–29, 2012.
- [10] D. Y. Li, H. J. Meng, and X. M. Shi, “Membership clouds and membership cloud generations,” *Journal of Computer Research and Development*, vol. 32, no. 6, pp. 15–20, 1995.
- [11] C. Dai, Y. Zhu, W. Chen, and J. Lin, “Cloud model based genetic algorithm and its applications,” *Acta Electronica Sinica*, vol. 35, no. 7, pp. 1419–1424, 2007.
- [12] G. Zhang, R. He, Y. Liu, D. Li, and G. Chen, “Evolutionary algorithm based on cloud model,” *Chinese Journal of Computers*, vol. 31, no. 7, pp. 1082–1091, 2008.
- [13] Y. Liu, D. Li, G. Zhang, and G. Chen, “Atomized feature in cloud based evolutionary algorithm,” *Acta Electronica Sinica*, vol. 37, no. 8, pp. 1651–1658, 2009.
- [14] A. Hedenström, L. C. Johansson, M. Wolf, R. von Busse, Y. Winter, and G. R. Spedding, “Bat flight generates complex aerodynamic tracks,” *Science*, vol. 316, no. 5826, pp. 894–897, 2011.
- [15] M. E. Bates, J. A. Simmons, and T. V. Zorikov, “Bats use echo harmonic structure to distinguish their targets from background clutter,” *Science*, vol. 333, no. 6042, pp. 627–630, 2011.
- [16] M. Knörnschild, K. Jung, M. Nagy, M. Metz, and E. Kalko, “Bat echolocation call facilitate social communication,” *Proceedings of the Royal Society B*, vol. 279, no. 1748, pp. 4827–4835.
- [17] A. V. Chechkin, R. Metzler, J. Klafter, and V. Y. Gonchar, “Introduction to the theory of Lévy flights,” in *Anomalous Transport: Foundations and Applications*, pp. 129–162, Wiley-VCH, 2008.
- [18] “Alpha-Stable distributions in MATLAB,” <http://math.bu.edu/people/mveillet/html/alphastablepub.html>.
- [19] A. M. Reynolds, M. A. Frye, and S. Rands, “Free-flight odor tracking in *Drosophila* is consistent with an optimal intermittent scale-free search,” *PLoS ONE*, vol. 2, no. 4, article e354, 2007.
- [20] C. T. Brown, L. S. Liebovitch, and R. Glendon, “Lévy flights in dove *Ju'hoansi* foraging patterns,” *Human Ecology*, vol. 35, no. 1, pp. 129–138, 2007.
- [21] X. Yang and S. Deb, “Cuckoo search via Lévy flights,” in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, December 2009.
- [22] D. Y. Li, C. Y. Liu, and W. Y. Gan, “Proof of the heavy-tailed property of normal cloud model,” *Engineer and Science of China*, vol. 13, no. 4, pp. 20–23, 2011.
- [23] D. Y. Li and Y. Du, *Artificial Intelligence with Uncertainty*, National Defense Industry Press, Beijing, China, 2005.
- [24] D. Y. Li and C. Y. Liu, “Study on the universality of the normal cloud model,” *Engineer and Science of China*, vol. 6, no. 8, pp. 28–33, 2004.
- [25] H. Lu, Y. Wang, D. Li, and C. Liu, “Application of backward cloud in qualitative evaluation,” *Chinese Journal of Computers*, vol. 26, no. 8, pp. 1009–1014, 2003.

Research Article

Evolutionary Multiobjective Query Workload Optimization of Cloud Data Warehouses

Tansel Dokeroglu,¹ Seyyit Alper Sert,¹ and Muhammet Serkan Cinar²

¹ METU Computer Engineering Department, Cankaya, 06800 Ankara, Turkey

² Hacettepe University Computer Engineering Department, Beytepe, 06800 Ankara, Turkey

Correspondence should be addressed to Tansel Dokeroglu; tansel@ceng.metu.edu.tr

Received 11 December 2013; Accepted 20 February 2014; Published 29 April 2014

Academic Editors: S.-F. Chien, T. O. Ting, and X.-S. Yang

Copyright © 2014 Tansel Dokeroglu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the advent of Cloud databases, query optimizers need to find pareto optimal solutions in terms of response time and monetary cost. Our novel approach minimizes both objectives by deploying alternative virtual resources and query plans making use of the virtual resource elasticity of the Cloud. We propose an exact multiobjective branch-and-bound and a robust multiobjective genetic algorithm for the optimization of distributed data warehouse query workloads on the Cloud. In order to investigate the effectiveness of our approach, we incorporate the devised algorithms into a prototype system. Finally, through several experiments that we have conducted with different workloads and virtual resource configurations, we conclude remarkable findings of alternative deployments as well as the advantages and disadvantages of the multiobjective algorithms we propose.

1. Introduction

Cloud computing has emerged as a new computation paradigm that builds elastic and scalable software systems. Vendors such as Amazon, Google, Microsoft, and Salesforce offer several options for computing infrastructures, platforms, and software systems [1–4] and supply highly scalable database services with simplified interfaces and the goal of reducing the total cost of ownership [5–7]. Users pay all costs associated with hosting and querying their data where database-as-a-service providers present different choices to tradeoff price and performance to increase the satisfaction of the customers and optimize the overall performance [8, 9]. Recently, extensive academic and commercial research is being done to construct self-tuned, efficient, and resource-economic Cloud database services that protect the benefits of both the customers and the vendors [10–13].

Virtualization is being exploited to simplify the administration of physical machines and accomplish efficient systems and it is the main enabling technology of Cloud computing that provides the illusion of infinite resources in many respects [14]. The perception of hardware and software resources is decoupled from the actual implementation and

the virtual resources perceived by applications are mapped to real physical resources. Through mapping virtual resources to physical ones as needed, the virtualization can be used by several databases that are located on physical servers to share and change the allocation of resources according to query workloads [15] (see Figure 1). This capability of virtualization provides efficient Cloud databases where each virtual machine (VM) has its own operating system and thinks that it is using dedicated resources (CPU, main memory, network bandwidth, I/O, etc.), whereas in reality the physical resources are shared among by using a VM monitor (VMM) that controls the allocation of resources [16–19].

In addition to providing efficient queries in accordance with the service level agreements, contemporary relational Cloud database management systems need to optimize a multicriteria problem that the overall cost of hardware ownership price is to be minimized. More specifically, the problem can be stated as follows.

Given a budget constraint and a query workload, how can the virtual resources of the Cloud (CPU, main memory, network bandwidth, etc.) be allocated to virtual machines, each having a part of a distributed database, that the best

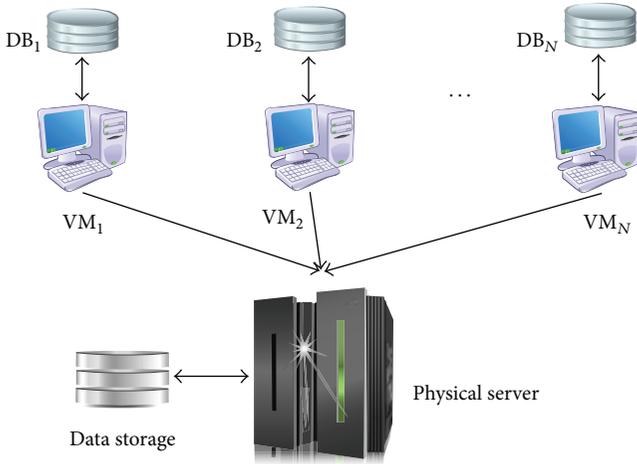


FIGURE 1: Virtualization of resources on a server.

overall query performance can be achieved with minimum pricing?

In this study, we have developed a framework to provide (near-)optimal virtual resource allocations with respect to the overall cost of hardware ownership price and a good tradeoff between the efficiency and the overall cost of a database is ensured. Our framework produces cost-efficient design alternatives (virtual resource configuration and query plans) and recommends them to decision makers. A budgetary constraint can be a more important criterion for a consumer, whereas the response time of the queries is more crucial for another [20]. Therefore, in order to fully realize the potential of the Cloud, alternative query plans are executed with well configured virtual resources instead of only optimizing single query plans on statically designed virtual resources [21]. This means that instead of designing the database over standard VMs, we have configured the virtual resources, which indicates that CPU usage and RAM can be a crucial point for a data warehouse workload, whereas network or I/O bandwidth is more important for another.

In this part, we give a scenario to explain the multiobjective problem in more detail. Consider a distributed TPC-H decision support database where all of its tables are located on different VMs. When we execute TPC-H query 3 with two different query plans (QP₁ and QP₂ given in Appendix) and with alternative virtual resource allocations, we obtain different results. The results of this experiment are presented in Tables 1 and 2. As it can be seen, the configuration of VMs with 4 × 2 Ghz CPU, 8 GB RAM, and 300 Mbps network bandwidth and with QP₁ is observed to be the best performing platform; however, its monetary price is one of the most expensive alternatives. The configuration of 1 × 2 Ghz CPU, 768 MB RAM, and 100 Mbps network bandwidth and with QP₁ has a response time, that is, only 25.9% slower but 72.0% cheaper. The paretooptimal visualization of the solutions can be seen in Figure 2. Looking at the results, the cheapest VM configuration is not the worst and the most expensive configuration is not the best solution in accordance with both

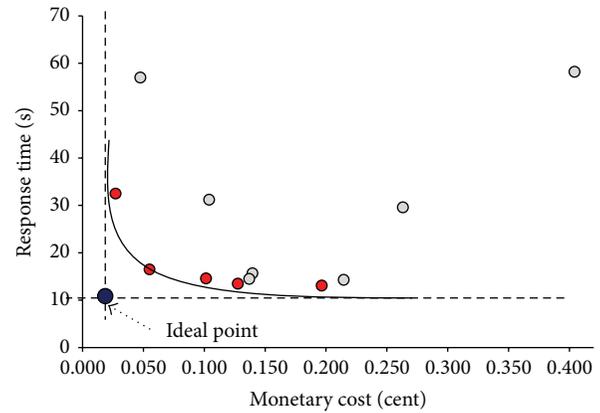


FIGURE 2: Paretooptimal curve for the response time and monetary cost of TPC-H Q3 with different virtual resource configurations and query plans.

TABLE 1: Response time and monetary costs of TPC-H query 3 (QP₁) with 6 different virtual resource configurations.

Virtual machine configuration	Network bandwidth	Response time (sec.)	Price ¢
1 × 2 Ghz CPU, 768 MB RAM	10 Mbps	57.0	¢0.048
1 × 2 Ghz CPU, 768 MB RAM	100 Mbps	16.5	¢0.055
1 × 2 Ghz CPU, 768 MB RAM	300 Mbps	15.7	¢0.140
4 × 2 Ghz CPU, 8 GB RAM	10 Mbps	58.2	¢0.404
4 × 2 Ghz CPU, 8 GB RAM	100 Mbps	13.5	¢0.128
4 × 2 Ghz CPU, 8 GB RAM	300 Mbps	13.1	¢0.197

objectives. Instead, alternatives chosen according to virtual resource demands provide better paretooptimal solutions. For example, QP₁ requests more network resources, whereas QP₂ spends more main memory. In this case, QP₁ can be a better way to execute a query where the network bandwidth is high and cheap.

In order to investigate the effectiveness of our approach, we incorporate the devised framework into a prototype system for evaluation and instantiate it with a simple heuristic algorithm (SHA), an exact solution method, branch-and-bound (MOBB), and a soft computing method multiobjective genetic algorithm (MOGA). Finally, through several experiments that we have conducted with the prototype elastic virtual resource deployment optimizer on TPC-H query workloads, we conclude remarkable results of the space of alternative deployments as well as the advantages and disadvantages of the multiobjective optimization algorithms.

The remainder of this paper is organized as follows. In Section 2, we provide information about the related studies. In Section 3, we give mathematical multiobjective query

TABLE 2: Response time and monetary costs of TPC-H query 3 (QP₂) with 6 different virtual resource configurations.

Virtual machine configuration	Network bandwidth	Response time (sec.)	Price €
1 × 2 Ghz CPU, 768 MB RAM	10 Mbps	32.5	€0.027
1 × 2 Ghz CPU, 768 MB RAM	100 Mbps	31.2	€0.104
1 × 2 Ghz CPU, 768 MB RAM	300 Mbps	29.6	€0.263
4 × 2 Ghz CPU, 8 GB RAM	10 Mbps	14.6	€0.101
4 × 2 Ghz CPU, 8 GB RAM	100 Mbps	14.5	€0.137
4 × 2 Ghz CPU, 8 GB RAM	300 Mbps	14.3	€0.215

optimization problem formulation. In Section 4, infrastructure and pricing scheme parameters of the Cloud are given. Section 5 proposes our simple heuristic (SHA), branch-and-bound (MOBB), and genetic algorithms (MOGA). Section 6 defines our experimental environment and the setup of the selected TPC-H query workloads and presents the results of the experiments. In Section 7, we give our concluding remarks.

2. Related Work

In this section, we summarize some of the studies related to our work. There has been a lot of research related to the Cloud, but relatively there is no approach like ours that is concerned with both the optimization of the total ownership price and the performance of the queries by taking into account alternative virtual resource allocation and query plans.

Distributed databases are considered to be the first representatives of the Cloud databases. Therefore, we first analyzed Mariposa, an early distributed database system that implements an economic paradigm to solve many drawbacks of wide-area network cost-based optimizers [22]. In Mariposa, clients and servers have an account in a network bank and users allocate a budget to each of their queries. The processing mechanism aims to service the query in the limited budget by executing portions of it on various sites. The latter place bids for the execution of query parts and the bids are collected in query brokers. The decision of selecting the most appropriate bids is delegated to the user. A series of similar works have been proposed for the solution of the problem [23, 24]. Papadimitriou and Yannakakis [25] showed that Mariposa's greedy heuristic can be far from the optimum solution and proposed that the optimum cost-delay tradeoff (Pareto) curve in Mariposa's framework can be approximated fast within any desired accuracy. They also present a polynomial algorithm for the general multiobjective query optimization problem, which approximates the optimum cost-delay tradeoff.

An advisor automatically configures a set of VMs for database workloads where the advisor requests domain

knowledge in Soror's study [15]. Although his approach concerns with the efficient allocation of the VMs, it does not optimize the total ownership price of the system. Recently, efficient cost models have been proposed in the Cloud for scheduling of dataflows with regard to monetary cost and/or completion time [12] and cost amortization of data structures to ensure the economic viability of the provider [13], particularly for self-tuned caching [11] and for a real-life astronomy application using the Amazon Cloud [26].

New cost models that fit into the pay-as-you-go paradigm of Cloud computing are introduced in [10]. These cost models achieve a multiobjective optimization of the view materialization versus CPU power consumption problem under budget constraints. It is shown that Cloud view materialization is always desirable. Koutris et al. [27] built a theoretical foundation, the first one towards a practical design and implementation of a pricing system. They present a formal framework for query-based pricing. Central to this framework are the notions of arbitrage-free and discount-free pricing.

In [28], the cost performance tradeoffs of different execution and resource provisioning plans have been simulated, showing that by provisioning the right amount of storage and computing resources, cost can be reduced significantly. The performance of three workflow applications with different I/O, memory, and CPU requirements has also been compared on Amazon EC2 and a typical high-performance cluster (HPC) to identify what applications achieve the best performance in the Cloud at the lowest cost [26].

Recent research takes interest in various aspects of database and decision support technologies in the Cloud. Different studies investigate the storage and processing of structured data [29], the optimization of join queries, and how to support analysis operations such as aggregation [30]. Cloud data warehousing and OLAP systems also raise various problems related to storage and query performance [31]. Adaptations of these technologies to the Cloud are addressed in [32] or the calculation of OLAP cuboids using the MapReduce runtime environment [33].

In [34], a virtual-machine provisioning policy based on marginal cost and revenue functions is proposed. For each Cloud customer there exists a budget as a function of the execution time of the tasks that are submitted. Knowledge of this function, combined with the machine-hour cost, allows for educated decisions regarding the amount of virtual resources allocated per customer in the context of an IaaS-Cloud, an answer to the question of exactly how many VMs a consumer should request from a Cloud within a budget.

In [35], a cost-aware provisioning system for Cloud applications that can optimize either the rental cost for provisioning a certain capacity or the transition cost of reconfiguring an application's current capacity is proposed. The system exploits both replication and migration to dynamically provision capacity and uses an integer linear program formulation to optimize cost.

There has been a substantial amount of work on the problem of tuning database system configurations for specific workloads or execution environments [36] and on the problem of making database systems more flexible and adaptive in their use of computing resources [37, 38]. In our study

we are tuning the virtual resources to a database system, rather than tuning the database system for a given resource setting. Also our study optimizes the objectives of minimum money consumption and maximum benefit from the virtual resources and optimizes this with an efficient multiobjective genetic algorithm. In summary, our study focuses on the elasticity of Cloud resources and produces multiple resource deployment plans with alternative query plans for a set of queries in a batch, enabling the user to select the desired tradeoff with efficient cost models. To the best of our knowledge, no resource deployment processing system deals with the concept of elasticity and cost-efficiency on relational Cloud databases like our system.

3. Multiobjective Query Optimization Problem Formulation

In this part, we first give the definitions of some main terms that are used in the study to provide the reader a better understanding and later we present the mathematical representation of our multiobjective query optimization problem of a data warehouse workload on the Cloud.

Virtual Machine (VM). VM is software that emulates the architecture and functions of a real computer. The number of its processors and main memory can be changed through virtualization.

Distributed Data Warehouse. Distributed data warehouse is a TPC-H decision support data warehouse that is distributed over a set of VMs on a network. VMs communicate with each other by paying a cost required by Cloud vendor.

Workload (W). Workload is a set of queries that are submitted as a batch to a distributed data warehouse. Our data warehouse is a distributed database so that the tasks of the queries are sent to the related VM and its results are received by other VMs to join the data and give a result.

Response Time. Response time is the time that has elapsed between the submission of the queries and obtaining of the results.

Total Execution Time. Total execution time is the sum of the time spent by the CPUs of the VMs and the time period during the data transmission over the network.

Monetary Cost. Monetary cost of a Cloud data warehouse workload (C_{total}) includes renting the resources to run the database. These resources are mainly data storage ($C_{storage}$) processing time of the VMs (C_{comp}) and the sum of data transfer cost (C_{comm}) [10]:

$$C_{total} = C_{storage} + C_{comp} + C_{comm}. \quad (1)$$

Data storage cost, $C_{storage}$, depends on the size of the data (including the structures such as indexes and replications) and its storage time. Processing time of the VMs, C_{comp} , is the total price for CPU usage. During the execution of the queries, different VM configurations can be used and the configuration of a VM (RAM, number of CPUs, etc.) is flexible in accordance with the resources used. Micro, small, large, and extra-large are some of the configurations provided by the Cloud vendors at various prices [3]. Data transfer cost, C_{comm} , is related to the amount of data migrated between sites and the pricing model applied by the Cloud provider.

Alternative Query Plans (QPs). QPs provide different ways for executing a query. In Figure 3, we can see two different QPs of TPC-H query Q3. QP_1 first joins the customer and orders relations, whereas QP_2 joins the orders and lineitem relations first. Alternative QPs can take advantage of different ways of executing the same query; thus, cheaper resources can reduce the total price of a query while increasing the response time. This elasticity provides new opportunities for the solution of our multiobjective problem.

The formulation of the problem consists of two parts: the monetary cost and the response time of the query workloads that will work on the selected VMs with the alternative QPs of the queries.

The monetary cost is calculated in accordance with (1) and the response time of the query workloads is calculated with the parameters and statistics used by query optimizers. The main goal of the problem is to minimize

$$F(x) = \{\text{Resp_time}(x), \text{Total_cost}(x)\}, \quad (2)$$

where x denotes a solution vector consisting of a set of VMs, QPs of m queries in workload W_k that will be executed on selected VMs, and the following network:

$$x = \{\{VM_1, \dots, VM_n\}, \{QP_1, \dots, QP_m\}, \text{Network}_i\}. \quad (3)$$

There are n VMs with independent DBMS and each VM has a set of processors and a main memory. Each DBMS has a workload that consists of a set of queries. W_i represents the i th workload. The resources to be deployed to VMs are CPU and main memory.

$M_Cost(W_k, x_i)$ is the total monetary cost of workload W_k for the solution vector x_i .

$\text{Resp_Time}(W_k, x_i)$ is the response time of workload W_k with the virtual resource and QP settings given in solution x_i .

The main goal is to obtain the paretooptimal set of solutions such that the overall workload cost is minimized. The overall multiobjective objective function (finding solutions closer to the ideal point given in Figure 2) is represented in the following:

$$F(x_i) = \min \sqrt{(\text{Resp_time}(x_i) - \text{Best_time}(x))^2 + (M_cost(x_i) - \text{Best_cost}(x))^2}, \quad (4)$$

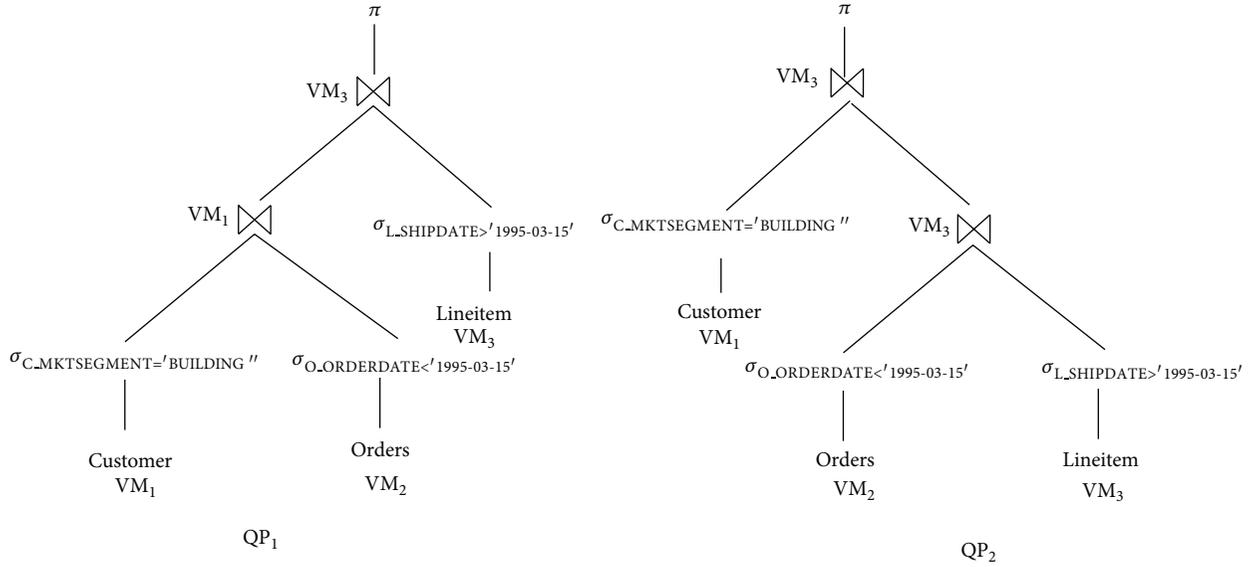


FIGURE 3: Alternative QPs for TPC-H Q3 query.

where $Best_time(x)$ is the best response time and $Best_cost(x)$ is the minimal monetary cost for workload W_k with solution vector x .

Response Time Cost Model. In order to measure the response time of a workload with a configuration of VMs, we developed a response time based cost model [21, 39] that uses bushy-plan trees [40]. The cost model depends on the statistics of the database catalog. The main parameters used in the cost model are shown in Table 3.

Using the number of pages as a parameter, the response time taken by a query is calculated as

$$\begin{aligned}
 Resp_time = & (T_{CPU} * seq_#insts) + (T_{I/O} * seq_#I/Os) \\
 & + (T_{MSG} * seq_#msgs) + (T_{TR} * seq_#bytes). \tag{5}
 \end{aligned}$$

The network communication time of transferring an intermediate query result from one site to another is calculated as

$$CT(\#pages) = T_{MSG} + (T_{TR} * \#pages). \tag{6}$$

4. Infrastructure and Pricing Scheme Parameters of the Cloud

In this section, we describe the infrastructure details of the Cloud database and the pricing scheme that we have used during the optimizations. Each customer requests queries from the Cloud by using Internet and contacts with the aggregate node. The aggregate node distributes the query to the appropriate VMs. The Cloud infrastructure provides unlimited amount of storage space, CPU nodes, RAM, and very high speed intra-Cloud networking. All the resources of the Cloud are assumed to be on a network. The CPU nodes,

TABLE 3: Parameters used in the cost model.

Symbol	Definition
$T_{I/O}$	I/O time for a page
$\#I/O$	Number of page I/O operations
$seq_#I/O$	Max. number of sequential pages I/O
T_{CPU}	Time for a CPU instruction
$seq_#insts$	Max. number of sequential instructions
T_{MSG}	Time to initiate and receive a message
$seq_#msgs$	Max. number of sequential messages
T_{TR}	Time to transmit a page
$seq_#pages$	Max. number of sequential pages
$\#insts$	Number of instructions

RAM, and I/O bandwidth of each VM are different and can be deployed by using VM monitors (VMM) in milliseconds [14]. The storage system is based on a clustered file system where the disk blocks are stored close to the CPU nodes accessing them. I/O bandwidth of the storage is divided evenly to the VMs (that may have multiple cores up to 8).

There are many Cloud service providers (CSP) in the market and they offer different pricing schema for the services they provide. Different pricing schema of Cloud server providers can be opportunities for customers in accordance with the tasks they want to complete. In our study, we will use a pricing scheme that is similar to Windows Azure [3]. VM configurations such as extra small (XS), small (S), and medium (M) are provided by the Cloud service provider. The detailed information of VM configurations can be seen in Table 4. The cost for a small VM (1 GHz CPU, 768 MB RAM) is \$0.02/hr, whereas for A7 (8 × 1.6 GHz CPU, 56 GB RAM) is \$1.64/hr.

TABLE 4: Virtual machine prices.

Symbol	Virtual machine configuration	Price
XS	1 GHz CPU, 768 MB RAM	\$0.02/hr
S	1.6 GHz CPU, 1.75 GB RAM	\$0.06/hr
M	2 × 1.6 GHz CPU, 3.5 GB RAM	\$0.12/hr
L	4 × 1.6 GHz CPU, 7 GB RAM	\$0.24/hr
XL	8 × 1.6 GHz CPU, 14 GB RAM	\$0.48/hr
A ₆	4 × 1.6 GHz CPU, 28 GB RAM	\$0.82/hr
A ₇	8 × 1.6 GHz CPU, 56 GB RAM	\$1.64/hr

TABLE 5: Cloud database storage prices.

Database size	Price
100 MB	\$5.00/mo
1 GB	\$9.99/mo
2 GB	\$13.99/mo
5 GB	\$25.98/mo
10 GB	\$45.96/mo
50 GB	\$125.88/mo
150 GB	\$225.78/mo

TABLE 6: Network bandwidth prices.

Bandwidth	Price
10 Mbps	\$0.05/hr
100 Mbps	\$0.50/hr
200 Mbps	\$1.00/hr

Data storage is also billed by the Cloud service providers. In our model, monthly storage price is used. During our experiments, the data storage price was constant for all the queries. Therefore, we do not add this parameter to our overall cost. The detailed information of database storage prices can be seen in Table 5.

Most of the Cloud providers do not charge for the data transfers in a private Cloud but the data that leaves the Cloud and the bandwidth of the intra-Cloud network can reach up to 10 Gbps. In order to make our problem more interesting and handle this dimension of the optimization, we have located our VMs on a virtual switch. Different bandwidth networks can be chosen and the pricing scheme changes in this communication infrastructure. The pricing we have used for the network layer is given in Table 6. The bandwidth of the network is increased from 10 Mbps up to 200 Mbps during the experiments.

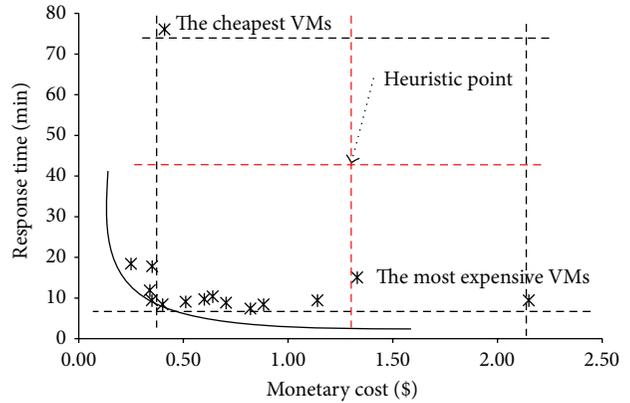


FIGURE 4: Proposed heuristic value for MOBB algorithm.

5. Proposed Algorithms

In this section, we propose three algorithms for the solution of the problem: simple heuristic algorithm (SHA), an exact algorithm branch-and-bound which finds the pareto optimal solutions, and a multiobjective genetic algorithm (MOGA).

5.1. Simple Heuristic Algorithm (SHA). In SHA, we have ranked the VMs in accordance with the frequency of the join operations. For example, with the database configuration we have studied, the first one was the VM that most of the join operations took place and we have assigned the best configuration to this VM. For the rest of the VMs we have assigned configurations with decreasing prices. The effects of all network types and QPs are evaluated on these configurations of the VMs. This algorithm is developed to provide us a test environment to evaluate the performance of other proposed algorithms, MOBB and MOGA.

5.2. Multiobjective Optimization of Cloud Database Configuration Using Branch-and-Bound Algorithm (MOBB). Multiobjective branch-and-bound algorithm (MOBB) is an exhaustive optimization algorithm. It enumerates all candidate solutions, where fruitless subsets of candidates are discarded, by using upper and lower estimated bounds of the problem instance being optimized [41]. MOBB starts searching with *null* initial values indicating that no QP has yet been selected for any queries with the current VM configuration. Later, QPs are assigned to selected VM configuration. At each level of the tree, one additional QP is assigned to the query workload. This procedure is repeated for every VM configuration.

We define two initial upper bounds for MOBB. The minimum monetary cost is the running time of VMs that execute the queries in a workload of queries. The response time is the finishing time of the workload with the given VM configuration. In order to estimate a lower bound, different heuristic functions can be used. The heuristic we proposed here is reasonable and performs well during the optimization process. We will explain the heuristic with a scenario. In Figure 4 we can see the results of a sample multiobjective

query workload optimization. The best response time and the minimum monetary cost values are defined and marked on the Figure. We can obtain these values with the most expensive and the cheapest VM configurations easily. Hereby, we propose a heuristic point (marked as *heuristic point* on the Figure), that is, the center of the square constructed by the response time and monetary costs of the most expensive and the cheapest VM configurations. If the response time of a workload falls above heuristic point or if the monetary cost is at the right-hand side of heuristic point on the Figure then it is pruned according to our heuristic.

Table 7 gives us an execution order of a sample workload W . QP_1 denotes the first query execution plan of a query and $\langle QP_1, QP_1, \dots, QP_2 \rangle$ is the sequence of submitted queries in a workload. The first query is executed with query plan QP_1 , the second query is executed with query plan QP_1 , and the last query is executed with query plan QP_2 . The executions in the table start with query execution plan of query 1 and two null queries. The final solution is the state with no *null* values. After finding the best and worst response times with the most expensive and the cheapest VM configurations, we set our heuristic point as monetary cost = \$1.2 and response time = 50 min.

The execution starts by assigning the queries to the current VM configuration. The response time and the monetary cost of the first query are calculated with its three different QPs. The first QP is in the acceptable bounds (monetary cost = \$0.5 and response time = 10 min.) but the other two QPs exceed the limits. The second one is more expensive than the heuristic value (\$1.3) and the third one is slower than the heuristic value (55 min.). Therefore, they are pruned. In the second phase, we assign the QPs of the second query. They are within the limits of the heuristic value and at the last phase we add the third query. They do not exceed the limits of the heuristic value and they become solutions. Pseudocode of our MOBB algorithm is given in Algorithm 1.

5.3. Multiobjective Optimization of Cloud Database Configuration Using Genetic Algorithm (MOGA). The principles of applying natural evolution to optimization problems were first described in [42, 43]. The GA theory has been further developed and GAs have become very powerful tools for solving search and optimization problems [44–46]. GAs are based on the principle of genetics and evolution and have been frequently used to solve many NP-complete problems. GAs use a computational model that simulates the natural processes of selection and evolution. Individuals with better quality have more chance to survive, to reproduce, and to pass their genetic characteristics to future generations. Each potential solution in the search space is considered as an individual and is represented by strings called chromosomes. Genes are the atomic parts of chromosomes and codify a specific characteristic. Chromosomes are encoded in different ways for each application. A random population is generated in the first step of the algorithm and by applying selection, crossover, and mutation operations iteratively new generations are created [47]. The individual having the best fitness value in the population is returned as the solution of

the problem. Algorithm 2 gives the details of GA used in MOGA system.

Multiobjective query optimization problem can be modeled by evolutionary methods. A chromosome corresponds to a solution instance including a set of relational Cloud database QPs. Figure 5 shows the chromosome structure of a solution instance. The chromosome consists of three parts; leftmost segment represents the configuration of the VMs. Middle segment is the set of QPs for the queries in the workload. Rightmost part gene represents the selected network layer of the solution vector.

We have defined three operators for the solution of GA model.

Crossover Operator. The operator uses two parents that are selected from the population by a selection method. We have proposed two types of crossover operators, *global* and *local*. Global crossover operator swaps VM, QP, or network part of two selected chromosomes with the same counter chromosome. Below we can see two parents and their VM parts are exchanged to provide two new chromosomes. Details can be seen in Figure 6. Consider

$$\begin{aligned} \text{par}_1 &= \{\{\mathbf{VM}_3, \mathbf{VM}_2, \mathbf{VM}_1, \mathbf{VM}_2\}, \\ &\quad \{\mathbf{QP}_2, \mathbf{QP}_1, \mathbf{QP}_2, \mathbf{QP}_1\}, \text{Network}_1\} \\ \text{par}_2 &= \{\{\mathbf{VM}_1, \mathbf{VM}_1, \mathbf{VM}_2, \mathbf{VM}_1\}, \\ &\quad \{\mathbf{QP}_1, \mathbf{QP}_2, \mathbf{QP}_1, \mathbf{QP}_1\}, \text{Network}_2\}. \end{aligned} \quad (7)$$

New offspring (offs) are

$$\begin{aligned} \text{offs}_1 &= \{\{\mathbf{VM}_1, \mathbf{VM}_1, \mathbf{VM}_2, \mathbf{VM}_1\}, \\ &\quad \{\mathbf{QP}_2, \mathbf{QP}_1, \mathbf{QP}_2, \mathbf{QP}_1\}, \text{Network}_1\} \\ \text{offs}_2 &= \{\{\mathbf{VM}_3, \mathbf{VM}_2, \mathbf{VM}_1, \mathbf{VM}_2\}, \\ &\quad \{\mathbf{QP}_1, \mathbf{QP}_2, \mathbf{QP}_1, \mathbf{QP}_1\}, \text{Network}_2\}. \end{aligned} \quad (8)$$

The local crossover operator works on the VM and QP segments of the chromosome by dividing the parents and exchanging the segments with each other. Figure 7 gives an example of local crossover that divides the QPs of the chromosomes and exchanges to generate new offspring.

Mutation Operator. Mutation operator changes a randomly selected gene of a chromosome. In our chromosome structure this operator can act on any of the segments. Only a gene is replaced at every mutation process. Figure 8 shows how a mutation operator changes a QP in a chromosome.

Fitness Calculation. Multiobjective fitness evaluation does not produce a single solution vector. Therefore, we have selected the nondominant individuals in the population as the resulting solution set. The fitness of the individuals is evaluated in accordance with (4).

Parameters of MOGA are as follows:

- (i) *population size*: total number of chromosomes (individuals) in each generation;

TABLE 7: Execution of multiobjective branch-and-bound algorithm with heuristic values (monetary cost = \$1.2 and response time = 50 min.).

State	Monetary cost (\$)	Response time (sec.)	Action
$\langle QP_1, -, - \rangle$	\$0.5	10 min.	Expanded
$\langle QP_2, -, - \rangle$	\$1.3	25 min.	Pruned
$\langle QP_3, -, - \rangle$	\$0.5	55 min.	Pruned
$\langle QP_1, QP_1, - \rangle$	\$0.9	37 min.	Expanded
$\langle QP_1, QP_2, - \rangle$	\$0.9	37 min.	Expanded
$\langle QP_1, QP_1, QP_1 \rangle$	\$1.1	42 min.	Solution
$\langle QP_1, QP_2, QP_1 \rangle$	\$1.2	45 min.	Solution

Input: Set of VM types (VMs), Query workload (W)
Output: A set of pareto-optimal solutions

- (1) QP: Query Plan
- (2) Q: Queue of QPs
- (3) *Calculated_value*: Multiobjective cost of optimization
- (4) S: Set of solutions
- (5) **for** ($i = 1$ to all configurations of (VMs)) **do**
- (6) $B_response_time \leftarrow$ Find_the_best_response_time (VM_i, W)
- (7) $Cheapest_cost \leftarrow$ Find_the_cheapest_cost (VM_i, W)
- (8) $Heuristic_value \leftarrow$ Calculate_heuristic_value ($B_response_time, Cheapest_cost$)
- (9) $Q = null;$
- (10) $S = \{ \};$
- (11) **for** (Each Query Plan QP in the workload W) **do**
- (12) $Q.Enqueue$ QP_in(W);
- (13) $Calculated_value =$ Calculate_with (VM_i, Q);
- (14) **if** ($Calculated_value$ is worse than $Heuristic_value$) **then**
- (15) Break the loop and start with the next VM configuration;
- (16) **if** (W is empty and the $Calculated_value$ is better than $heuristic_value$) **then**
- (17) Add (VM_i) to the solution set of S;
- (18) **return** S;

ALGORITHM 1: Multiobjective optimization of Cloud database configuration using branch-and-bound.

- (ii) *number of generations*: each iteration of a GA that a number of crossovers and mutations are applied;
 - (iii) *maximum number of genes to transfer*: maximum length of the crossed segment in segmented crossover operation and maximum number of genes transferred in a multiple-point crossover operation;
 - (iv) *minimum number of genes to transfer*: minimum length of the crossed segment in segmented crossover operation and minimum number of genes transferred in a multiple-point crossover operation.
 - (v) *selection type (tournament)*: r chromosomes (r is the tournament size) are selected from the population, and the chromosome with the best fitness value is chosen for the next generation from the r -element group; this process is repeated as many times as the population size of the next generation;
 - (vi) *tournament size*: number of individuals entering a selection in tournament selection technique;
 - (vii) *truncate ratio*: ratio of the best individuals, which are sorted according to their fitness values, used for producing the next generation;
 - (viii) *mutation ratio*: probability of mutations in a single gene.
- The results of the experiments with SHA, MOBB, and MOGA are presented in Section 6.

6. Experimental Evaluation

In this section, we describe our experimental environment, the setup of the selected TPC-H query workloads, VMM Hyper-V, and parameter settings for MOGA and present the results of the experiments we have obtained with multiobjective simple heuristic algorithm (SHA), branch-and-bound (MOBB), and multiobjective genetic algorithm (MOGA). The VM and network configurations are first optimized by the algorithms and later we have run the workloads on a real Cloud database to see the real results and measure the correctness of our algorithms.

Input: Set of VM types (VMs), Query workload (W)
Output: A set of pareto-optimal solutions

- (1) VM: Set of Virtual Machine types
- (2) QP: Set of alternative query plans
- (3) N : Set of alternative network bandwidths
- (4) p : Population
- (5) par_1, par_2 : Individuals (parent) selected for crossover or mutation
- (6) s : Generated individual
- (7) $p \leftarrow$ Generate random individuals(VM, QP, N)
- (8) Calculate fitness of individuals(p)
- (9) $p \leftarrow$ truncate(p)
- (10) $B_response_time \leftarrow$ Find_best_response_time(VMs, W)
- (11) $Cheapest_cost \leftarrow$ Find_cheapest_cost(VMs, W)
- (12) **for** $k := 1$ to generations **do**
- (13) (par_1, par_2) \leftarrow Select pair of parents(p)
- (14) $s \leftarrow$ Crossover(par_1, par_2)
- (15) Replace with least-fit in the population(p, s)
- (16) $s \leftarrow$ Mutation(p, s)
- (17) Replace with least-fit in the population(p, s)
- (18) Replace duplicate_chromosomes(p)
- (19) Update $B_response_time$
- (20) Update $Cheapest_cost$

ALGORITHM 2: Multiobjective optimization of cloud database configuration using genetic algorithm.

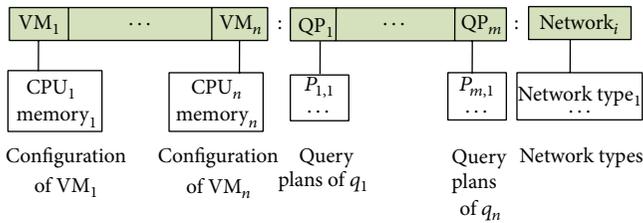


FIGURE 5: Chromosome structure for the proposed multiobjective genetic algorithm that consists of the virtual machines, the query plans, and the network layer.

6.1. Experimental Environment. We have performed our experiments on a private Cloud server: 4U DELL PowerEdge R910 having 32 (64 with Hyper Threading) cores and each core is Intel Xeon E7-4820 with a total of 2.00 Ghz processing power. Server has 128 GB DDR3 1600 Mhz virtualised memory and Broadcom Nextreme II 5709 1Gbps NICs. Operating system is Windows Server 2012 Standard Edition and as guest operating systems Windows Server 2008 R2 SP2 Enterprise Edition is used and on top of guest operating system, SQL Server 2012 Enterprise Edition Service Pack 1 is implemented as the database server. Windows Hyper-V 3.0 is used as virtualization platform. Network page size was 4 KByte during the experiments. The configuration of distributed data warehouse infrastructure we have used during the experiments is given in Figure 9. The resources (CPUs, main memory, and network bandwidth) of the VMs are changed according to the optimized solutions. VM aggregate is used to submit the workloads and obtain the results.

Hyper-V, known as Windows Server Virtualization, is a native hypervisor that enables platform virtualization on x86-64 systems. Hyper-V implements isolation of VMs in terms of a partition which is a logical unit of isolation, supported by the hypervisor, where each guest operating system executes. A hypervisor instance has to have at least one parent partition, running a supported version of Windows Server (2008, 2008 R2, or 2012). The virtualization stack runs in the parent partition and has direct access to the hardware devices. The parent partition later creates the child partitions which host the guest OSs.

6.2. TPC-H Workloads. A TPC-H database of size 10 GB is used during the experiments. The TPC-H database has 8 relations: lineitem (8,145 MB), orders (1,757 MB), partsupp (1,236 MB), part (290 MB), customer (256 MB), supplier (2 MB), region (0,008 MB), and nation (0,008 MB). The tables are assumed to locate at 5 different VMs. By replicating the small tables, nation, supplier, and region, we aimed to obtain a better performance.

We have used three different workloads that consist of TPC-H queries. Our purpose was to test the proposed algorithms under diverse workloads, in terms of the response time of query execution times and total cost of ownership. Each workload consists of 10 to 15 different TPC-H queries with predicates. Workload 1 has first 10 TPC-H queries, workload 2 has queries with smaller relations, and workload 3 has queries with larger relations and joined operations. For each query we have selected two QPs (including the best QP) on the average during the experiments. QPs can have more than a single task and these tasks can have dependencies with

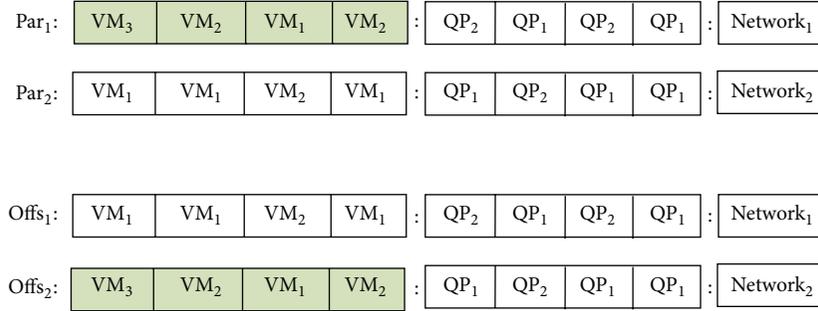


FIGURE 6: Global crossover operator for the multiobjective optimization of query workloads.

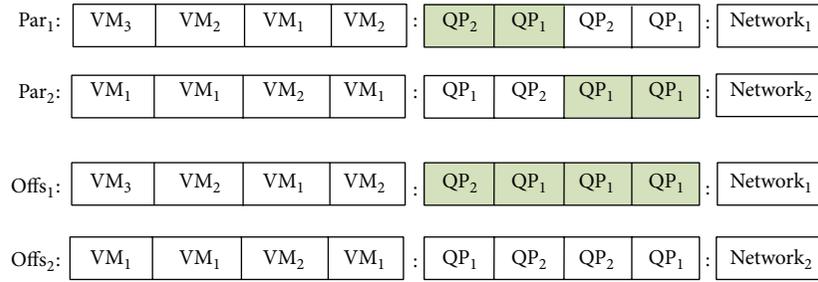


FIGURE 7: Local crossover operator for the multiobjective optimization of query workloads.

TABLE 8: TPC-H queries used in the workloads.

Workload	TPC queries
W ₁ (10 queries)	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
W ₂ (10 queries)	2, 3, 4, 9, 10, 11, 12, 13, 14, 16
W ₃ (15 queries)	1, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 18, 20, 22

the other tasks to complete a query. Selected queries for the workloads are given in Table 8.

In Figure 10, we have presented the response time of the selected TPC-H queries we have used during the experiments. These response times are obtained with the highest configuration of VMs (XL) and network bandwidth (200 Mbps).

6.3. Parameter Settings for Multiobjective Genetic Algorithm.

Population size and the number of generations of a genetic algorithm are the most important parameters that must be well tuned to obtain (near-)optimal solutions during the optimization. Larger number of individuals and generations explore the search space more effectively. On the other hand, this may bring very long optimization times. In order to diminish the effect of this drawback we have performed some experiments with changing number of population sizes and generations. In Figure 11, we give the performance details of MOGA with different population sizes (10 to 100) and number of generations (10 to 100) for workload 1. The figure gives the average fitness value of populations during the generations. As it can be seen, MOGA almost converges after 100 generations and continues to improve itself slightly after this point. Although population size 10 seems to perform as

the best option, population size 40 produces individuals that are more close to the ideal point that we aim to find.

Figure 12 gives the optimization times of MOGA with increasing number of populations. The optimization time of MOGA increases in accordance with the number of individuals in the population. For 10 individuals optimization time is 3 seconds and for 100 individuals it is 60 seconds. We have selected 40 individuals and 100 generations as our (near-)optimal parameters for the optimizations. These values provide good solutions for moderate size workloads such as ours.

In Figure 13, we have analyzed the effect of increasing number of submitted queries for MOGA with 40 individuals and 100 generations. There are three sets of queries (10, 20, and 40). It can be seen that increasing the number of submitted queries decreases the average fitness quality of the population. With 10 queries, we can obtain solutions below 0.01 fitness value. For 20 and 40 queries the solution quality increases to 0.04 and 0.11, respectively. Although the average fitness values get worse as the number of submitted queries increases, the values are still not more than 0.11, which is very efficient. MOGA improves the quality of the individuals in accordance with the objective response time and monetary costs. Table 9 shows our parameters settings used for MOGA. Crossover and mutation ratio are the values proposed by Holland [42]. Tournament is a very effective selection mechanism we have applied in our previous studies [44].

6.4. Experiments with the Workloads. In this part, we have performed experiments with the three TPC-H workloads we have defined previously. These workloads are first optimized

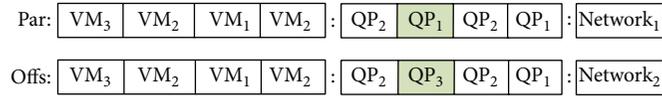


FIGURE 8: Mutation operator for the multiobjective optimization of query workloads.

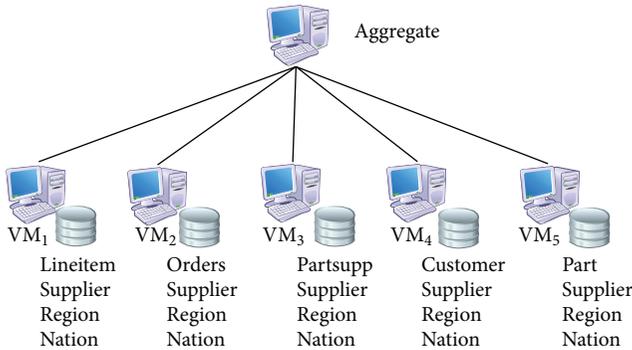


FIGURE 9: Architecture of the proposed system.

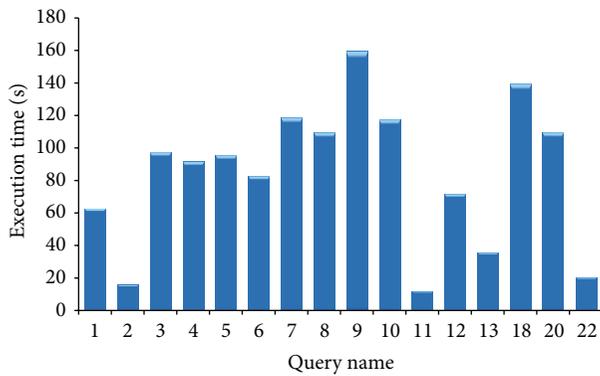


FIGURE 10: Response time of sample queries with the highest performance configuration VM settings and network bandwidths.

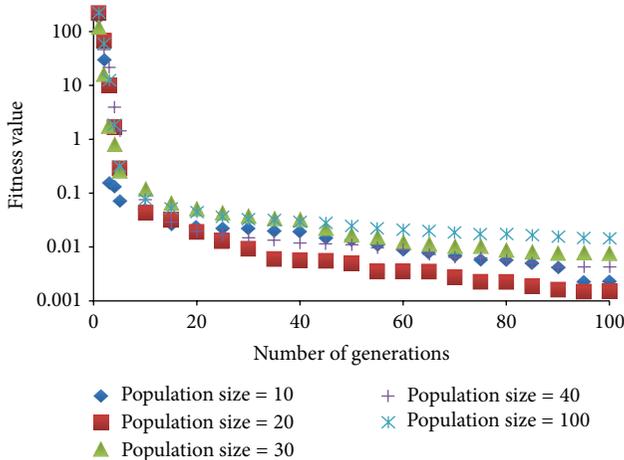


FIGURE 11: Average fitness value of different populations during generations.

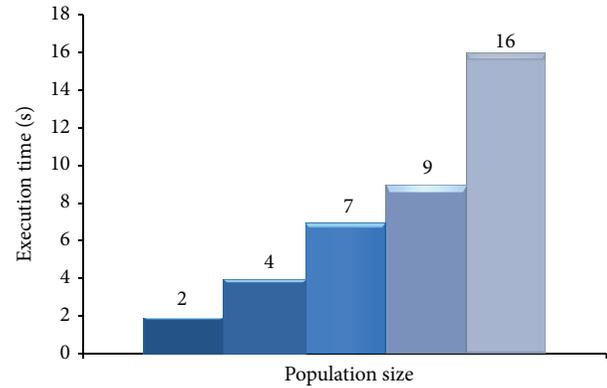


FIGURE 12: Average fitness value of different populations during generations.

FIGURE 13: Average fitness values of 40 individuals and 100 generations for increasing number of submitted queries.

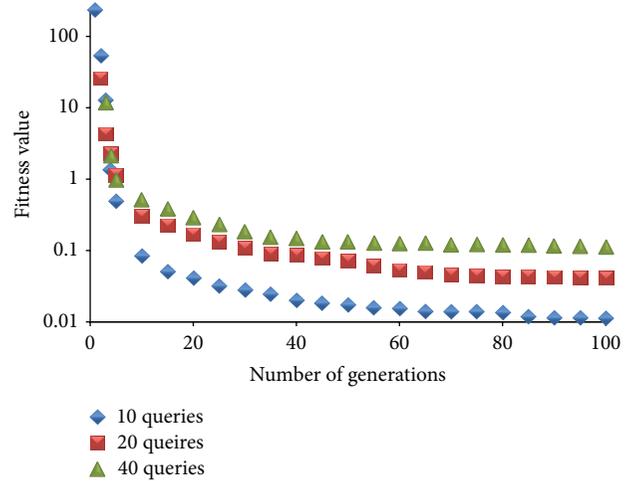


FIGURE 13: Average fitness values of 40 individuals and 100 generations for increasing number of submitted queries.

TABLE 9: Parameter settings for multiobjective genetic algorithm.

MOGA parameter	Value
Population size	40
Number of generations	100
Maximum number of genes to transfer	50%
Minimum number of genes to transfer	10%
Tournament size	10
Mutation ratio	1%

TABLE 10: Proposed paretooptimal VM and network configurations for workload 1.

Solution	Virtual machines	Network type
The most expensive	$VM_0 = A_7; VM_1 = A_7; VM_2 = A_7; VM_3 = A_7; VM_4 = A_7; VM_5 = A_7$	200 (Mbps)
The cheapest	$VM_0 = XS; VM_1 = XS; VM_2 = XS; VM_3 = XS; VM_4 = XS; VM_5 = XS$	10 (Mbps)
SHA-1	$VM_0 = L; VM_1 = A_7; VM_2 = A_6; VM_3 = XL; VM_4 = L; VM_5 = M$	10 (Mbps)
SHA-2	$VM_0 = L; VM_1 = A_7; VM_2 = A_6; VM_3 = XL; VM_4 = L; VM_5 = M$	100 (Mbps)
SHA-3	$VM_0 = L; VM_1 = A_7; VM_2 = A_6; VM_3 = XL; VM_4 = L; VM_5 = M$	200 (Mbps)
MOBB-1	$VM_0 = S; VM_1 = A_7; VM_2 = XS; VM_3 = L; VM_4 = S; VM_5 = L$	100 (Mbps)
MOBB-2	$VM_0 = XS; VM_1 = XL; VM_2 = XL; VM_3 = S; VM_4 = S; VM_5 = XS$	100 (Mbps)
MOBB-3	$VM_0 = S; VM_1 = L; VM_2 = A_7; VM_3 = A_7; VM_4 = XS; VM_5 = L$	100 (Mbps)
MOBB-4	$VM_0 = XL; VM_1 = A_7; VM_2 = XL; VM_3 = A_7; VM_4 = XS; VM_5 = A_6$	100 (Mbps)
MOBB-5	$VM_0 = L; VM_1 = L; VM_2 = M; VM_3 = L; VM_4 = A_6; VM_5 = A_6$	100 (Mbps)
MOGA-1	$VM_0 = XS; VM_1 = A_7; VM_2 = S; VM_3 = XL; VM_4 = L; VM_5 = A_6$	100 (Mbps)
MOGA-2	$VM_0 = S; VM_1 = XL; VM_2 = XL; VM_3 = XL; VM_4 = S; VM_5 = XS$	100 (Mbps)
MOGA-3	$VM_0 = XS; VM_1 = XL; VM_2 = XL; VM_3 = XL; VM_4 = S; VM_5 = S$	100 (Mbps)
MOGA-4	$VM_0 = L; VM_1 = S; VM_2 = XS; VM_3 = XL; VM_4 = A_7; VM_5 = L$	100 (Mbps)
MOGA-5	$VM_0 = XS; VM_1 = S; VM_2 = XS; VM_3 = A_7; VM_4 = A_6; VM_5 = XS$	100 (Mbps)

TABLE 11: Proposed paretooptimal VM and network configurations for workload 2.

Solution	Virtual machines	Network type
The most expensive	$VM_0 = A_7; VM_1 = A_7; VM_2 = A_7; VM_3 = A_7; VM_4 = A_7; VM_5 = A_7$	200 (Mbps)
The cheapest	$VM_0 = XS; VM_1 = XS; VM_2 = XS; VM_3 = XS; VM_4 = XS; VM_5 = XS$	10 (Mbps)
SHA-1	$VM_0 = L; VM_1 = A_7; VM_2 = A_6; VM_3 = XL; VM_4 = L; VM_5 = M$	10 (Mbps)
SHA-2	$VM_0 = L; VM_1 = A_7; VM_2 = A_6; VM_3 = XL; VM_4 = L; VM_5 = M$	100 (Mbps)
SHA-3	$VM_0 = L; VM_1 = A_7; VM_2 = A_6; VM_3 = XL; VM_4 = L; VM_5 = M$	200 (Mbps)
MOBB-1	$VM_0 = XS; VM_1 = XL; VM_2 = S; VM_3 = XL; VM_4 = XL; VM_5 = XS$	100 (Mbps)
MOBB-2	$VM_0 = XS; VM_1 = XL; VM_2 = XL; VM_3 = XL; VM_4 = XL; VM_5 = XS$	100 (Mbps)
MOBB-3	$VM_0 = S; VM_1 = XL; VM_2 = S; VM_3 = XL; VM_4 = XL; VM_5 = L$	100 (Mbps)
MOBB-4	$VM_0 = XL; VM_1 = M; VM_2 = L; VM_3 = XL; VM_4 = XS; VM_5 = A_7$	100 (Mbps)
MOBB-5	$VM_0 = XS; VM_1 = XL; VM_2 = S; VM_3 = XL; VM_4 = XL; VM_5 = S$	100 (Mbps)
MOGA-1	$VM_0 = XS; VM_1 = XL; VM_2 = XS; VM_3 = XL; VM_4 = XL; VM_5 = S$	100 (Mbps)
MOGA-2	$VM_0 = L; VM_1 = XL; VM_2 = L; VM_3 = XL; VM_4 = M; VM_5 = S$	100 (Mbps)
MOGA-3	$VM_0 = M; VM_1 = A_7; VM_2 = S; VM_3 = A_7; VM_4 = XL; VM_5 = M$	100 (Mbps)
MOGA-4	$VM_0 = S; VM_1 = XL; VM_2 = M; VM_3 = A_7; VM_4 = XL; VM_5 = S$	100 (Mbps)
MOGA-5	$VM_0 = S; VM_1 = A_7; VM_2 = XS; VM_3 = XL; VM_4 = XL; VM_5 = S$	100 (Mbps)

with SHA, MOBB, and MOGA algorithms. Later, selected solutions are executed in our Cloud database environment to verify the correctness of our approach. There are 15 alternative virtual resource configurations in these tests: 3 SHA, 5 MOBB, 5 MOGA, the highest performance VM configuration, and the cheapest VM configuration. The last three solutions are used to measure the effectiveness of other solutions. The workloads are executed 10 times with the selected VM configurations and the average values are used.

Figure 14 shows snapshots of CPU, network, and memory consumptions of VMs during the execution of workload W_1 , respectively. As it can be seen VM_1 demands the largest CPU resource and memory usage and VM_2 and VM_4 ship larger amounts of data. These snapshots are provided to give an idea about the resource demands of VMs during the execution of a workload.

The results of experiments with workloads 1, 2, and 3 are as follows.

In Figures 15, 16, and 17 and Tables 10, 11, and 12 we have presented the solutions produced by SHA, MOBB, and MOGA algorithms and the set of proposed VM and network bandwidths, respectively. The solutions with the highest and the cheapest performance VMs are also added to define upper and lower bounds. VMs with the highest configuration capabilities (A_7) give the best response time and VMs with the cheapest configurations (XS) give the longest execution time. In this sense, they provide meaningful results to evaluate the quality of solutions provided by MOBB and MOGA.

In the figures, a hypothetical *ideal point* is defined to show the optimal fitness value that can be achieved within the given minimum response time and minimum pricing. The

TABLE 12: Proposed paretooptimal VM and network configurations for workload 3.

Solution	Virtual machines	Network type
The most expensive	VM ₀ = A ₇ ; VM ₁ = A ₇ ; VM ₂ = A ₇ ; VM ₃ = A ₇ ; VM ₄ = A ₇ ; VM ₅ = A ₇	200 (Mbps)
The cheapest	VM ₀ = XS; VM ₁ = XS; VM ₂ = XS; VM ₃ = XS; VM ₄ = XS; VM ₅ = XS	10 (Mbps)
SHA-1	VM ₀ = L; VM ₁ = A ₇ ; VM ₂ = A ₆ ; VM ₃ = XL; VM ₄ = L; VM ₅ = M	10 (Mbps)
SHA-2	VM ₀ = L; VM ₁ = A ₇ ; VM ₂ = A ₆ ; VM ₃ = XL; VM ₄ = L; VM ₅ = M	100 (Mbps)
SHA-3	VM ₀ = L; VM ₁ = A ₇ ; VM ₂ = A ₆ ; VM ₃ = XL; VM ₄ = L; VM ₅ = M	200 (Mbps)
MOBB-1	VM ₀ = S; VM ₁ = XL; VM ₂ = L; VM ₃ = XL; VM ₄ = M; VM ₅ = M	200 (Mbps)
MOBB-2	VM ₀ = XS; VM ₁ = XL; VM ₂ = XL; VM ₃ = XL; VM ₄ = XL; VM ₅ = S	200 (Mbps)
MOBB-3	VM ₀ = L; VM ₁ = A ₆ ; VM ₂ = L; VM ₃ = M; VM ₄ = M; VM ₅ = XS	200 (Mbps)
MOBB-4	VM ₀ = A ₇ ; VM ₁ = M; VM ₂ = A ₆ ; VM ₃ = XL; VM ₄ = A ₆ ; VM ₅ = A ₆	200 (Mbps)
MOBB-5	VM ₀ = A ₆ ; VM ₁ = A ₇ ; VM ₂ = L; VM ₃ = S; VM ₄ = L; VM ₅ = A ₇	200 (Mbps)
MOGA-1	VM ₀ = M; VM ₁ = M; VM ₂ = A ₆ ; VM ₃ = A ₇ ; VM ₄ = A ₆ ; VM ₅ = L	200 (Mbps)
MOGA-2	VM ₀ = A ₆ ; VM ₁ = A ₇ ; VM ₂ = XS; VM ₃ = M; VM ₄ = A ₆ ; VM ₅ = XL	200 (Mbps)
MOGA-3	VM ₀ = XS; VM ₁ = XL; VM ₂ = L; VM ₃ = XS; VM ₄ = L; VM ₅ = L	200 (Mbps)
MOGA-4	VM ₀ = S; VM ₁ = L; VM ₂ = S; VM ₃ = A ₇ ; VM ₄ = A ₇ ; VM ₅ = XS	200 (Mbps)
MOGA-5	VM ₀ = M; VM ₁ = A ₇ ; VM ₂ = M; VM ₃ = A ₇ ; VM ₄ = L; VM ₅ = S	200 (Mbps)

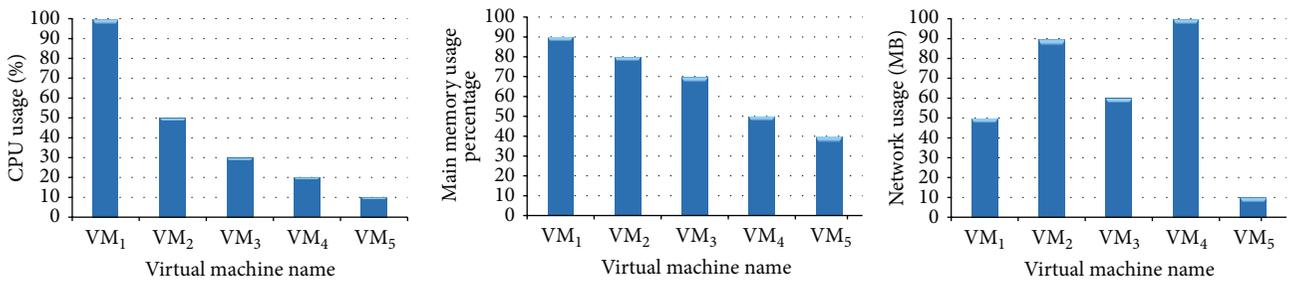


FIGURE 14: CPU, network, and memory consumption of the virtual machines.

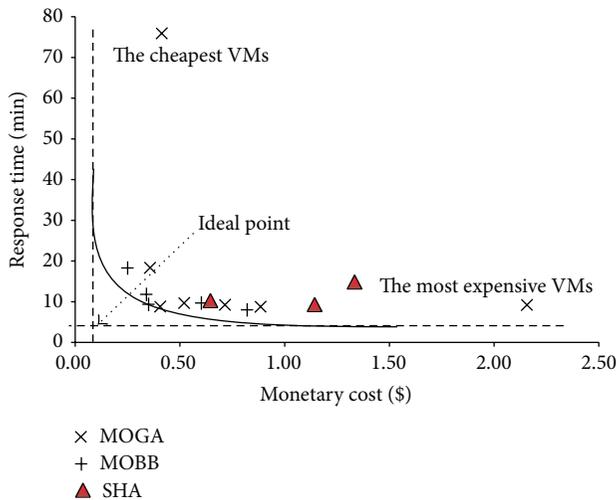


FIGURE 15: Proposed paretooptimal solutions for workload 1 by SHA, MOBB, and MOGA algorithms.

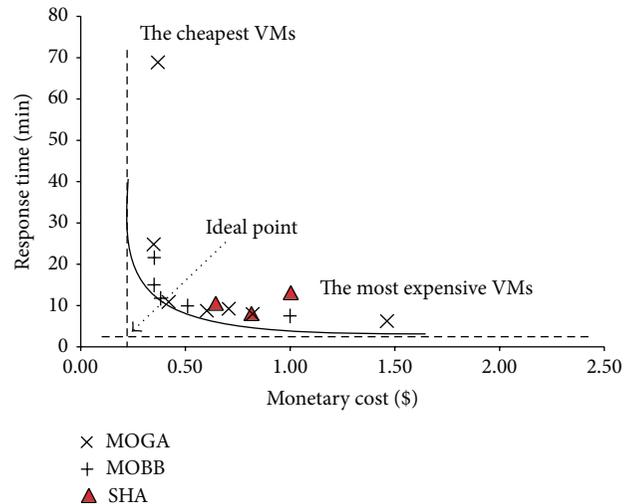


FIGURE 16: Proposed paretooptimal solutions for workload 2 by SHA, MOBB, and MOGA algorithms.

solutions that are chosen from the set of solutions produced by MOBB and MOGA algorithms construct a paretooptimal convex curve where a decision maker can choose any of

the solutions according to his/her requirements. *The most expensive VMs* option gives the fastest response time and *the cheapest VMs* option is the most time consuming.

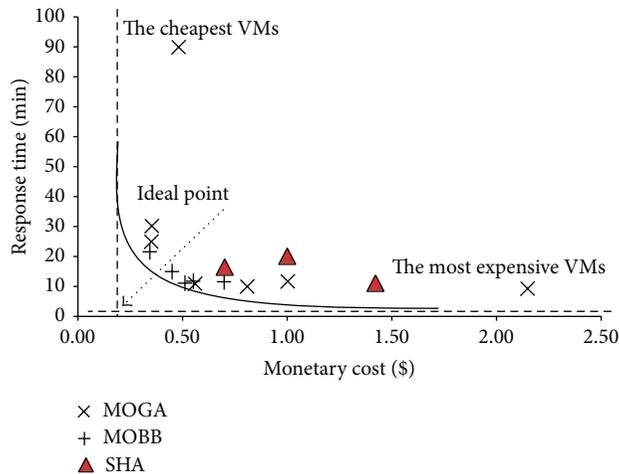


FIGURE 17: Proposed paretooptimal solutions for workload 3 by SHA, MOBB, and MOGA algorithms.

The optimal solutions are produced by MOBB algorithm. MOGA also gives almost the same solutions with faster optimization times than MOBB. The optimization time of MOBB is the longest and it can be prohibitive with workloads having more than 20 queries. In workload 3, MOBB algorithm was 20 times longer than MOGA. The solutions of SHA are slightly above the paretooptimal curve but they are far from the ideal point. Mostly, the most expensive VMs are assigned to VM1 that executes much of the join operations. Workloads 1 and 2 used 100 Mbps network but workload 3 that needs more communication between the VMs tends to use 200 Mbps network.

The solution sets produced by MOBB and MOGA construct a paretooptimal curve and decision makers can choose any of these solutions depending on their needs. The best solutions are near the ideal point. They have fast response times and cheaper monetary costs.

7. Conclusions and Future Work

In this paper, we solve the multiobjective optimization problem of Cloud data warehouse query workloads by making use of the elasticity of the virtual resources and alternative query execution plans. We minimize the monetary cost as well as providing fast response times. We formulate the problem and propose three algorithms, namely, simple heuristic (SHA), multiobjective branch-and-bound (MOBB), and multiobjective robust genetic algorithm (MOGA) for the optimization of the problem. To the best of our knowledge, the multiobjective query optimization problem is being solved for the first time with such a method. There are studies that are concerned with the best virtual resource deployment or the minimal monetary cost of workloads in static hardware resources; however, we combine both of these optimization techniques together with alternative query plans and obtain remarkable results as they are presented in our study. It is possible to design and expand the study with additional elastic resources such as I/O bandwidth and dynamic RAMs.

Appendix

A. Alternative Query Execution Plans for TPC-H Query Q3

TPC-H Q3 statement in accordance with the query execution plan 1 where all of the tables are shipped to query issuing node:

```
select top 10 l_orderkey, . . . , o_shippriority
from [vm2].customer c, [vm3].orders o,
[vm1].lineitem l,
where c.c.mktsegment = "building"
and c.c.custkey = o.o.custkey
and l.l_orderkey = o.o_orderkey
and o.o_orderdate < "1995-03-15"
and l.l_shipdate > "1995-03-15"
group by l.l_orderkey, o.o_orderdate, o.o_shippriority
order by revenue desc, o.o_orderdate.
```

TPC-H Q3 statement in accordance with query execution plan 2 where customer and orders tables are joined at virtual machine 3 and the resulting tuples are shipped to virtual machine 2 to join with lineitem table:

```
select top 10 l_orderkey, . . . , o_shippriority
from
openquery([vm3], "select o_orderdate, o_shippriority,
o_orderkey
from [vm2].customer c, [vm3].orders o
where c.c.mktsegment = "building"
and c.c.custkey = o.o.custkey
and o.o_orderdate < "1995-03-15"
group by o.o_orderdate, o.o_shippriority, o_orderkey
order by o.o_orderdate) remotel, [vm1].lineitem l
where and l.l_orderkey = remotel.o_orderkey
and l.l_shipdate > "1995-03-15"
group by l.l_orderkey, remotel.o_orderdate,
remotel.o_shippriority
order by revenue desc".
```

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

- [1] Amazon Web Services (AWS), <http://aws.amazon.com>.
- [2] Google App Engine, <http://code.google.com/appengine/>.
- [3] Windows Azure Platform, <http://microsoft.com/windowsazure/>.

- [4] Salesforce, <http://www.salesforce.com/>.
- [5] Windows Azure Storage Services REST API Ref, <http://msdn.microsoft.com/en-us/library/dd179355.aspx>.
- [6] Amazon Elastic MapReduce, <http://aws.amazon.com/elastic-mapreduce>.
- [7] Amazon Relational Database Service, <http://aws.amazon.com/rds/>.
- [8] P. Upadhyaya, M. Balazinska, and D. Suciu, "How to price shared optimizations in the cloud," *Proceedings of the VLDB Endowment*, vol. 5, no. 6, pp. 562–573, 2012.
- [9] M. Balazinska, B. Howe, and D. Suciu, "Data markets in the cloud: an opportunity for the database community," *PVLDB*, vol. 4, no. 12, pp. 1482–1485, 2011.
- [10] T. V. A. Nguyen, S. Bimonte, L. D'Orazio, and J. Darmont, "Cost models for view materialization in the cloud," in *Proceedings of EDBT/ICDT Workshops*, pp. 47–54, 2012.
- [11] D. Dash, V. Kantere, and A. Ailamaki, "An economic model for self-tuned cloud caching," in *Proceedings of the 25th IEEE International Conference on Data Engineering (ICDE '09)*, pp. 1687–1693, Shanghai, China, April 2009.
- [12] H. Kllapi, E. Sitaridi, M. M. Tsangaris, and Y. Ioannidis, "Schedule optimization for data processing flows on the cloud," in *Proceedings of the ACM SIGMOD International Conference on Management of Data (SIGMOD '11)*, pp. 289–300, June 2011.
- [13] V. Kantere, D. Dash, G. François, S. Kyriakopoulou, and A. Ailamaki, "Optimal service pricing for a cloud cache," *IEEE Transactions on Knowledge and Data Engineering*, vol. 23, no. 9, pp. 1345–1358, 2011.
- [14] P. Barham, B. Dragovic, K. Fraser et al., "Xen and the art of virtualization," *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5, pp. 164–177, 2003.
- [15] A. A. Soror, A. Aboulnaga, and K. Salem, "Database virtualization: a new frontier for database tuning and physical design," in *Proceedings of the Workshops in Conjunction with the 23rd International Conference on Data Engineering (ICDE '07)*, pp. 388–394, Istanbul, Turkey, April 2007.
- [16] M. Rosenblum and T. Garfinkel, "Virtual machine monitors: current technology and future trends," *Computer*, vol. 38, no. 5, pp. 39–47, 2005.
- [17] J. E. Smith and R. Nair, "The architecture of virtual machines," *Computer*, vol. 38, no. 5, pp. 32–38, 2005.
- [18] A. Aboulnaga, C. Amza, and K. Salem, "Virtualization and databases: state of the art and research challenges," in *Proceedings of the 11th International Conference on Extending Database Technology (EDBT '08)*, pp. 746–747, March 2008.
- [19] <http://www.xen.org/>.
- [20] L. D'Orazio, S. Bimonte, and J. Darmont, "Cost models for view materialization in the cloud," in *Proceedings of the Workshop on Data Analytics in the Cloud (EDBT-ICDT/DanaC)*, 2012.
- [21] M. T. Ozsu and P. Valduriez, *Principles of Distributed Database Systems*, 3rd edition, 2011.
- [22] M. Stonebraker, P. M. Aoki, W. Litwin et al., "Mariposa: a wide-area distributed database system," *VLDB Journal*, vol. 5, no. 1, pp. 48–63, 1996.
- [23] V. Marbukh and K. Mills, "Demand pricing & resource allocation in market-based compute grids: a model and initial results," in *Proceedings of the 7th International Conference on Networking (ICN '08)*, pp. 752–757, Cancún, Mexico, April 2008.
- [24] R. Moreno and A. B. Alonso-Conde, "Job scheduling and resource management techniques in economic grid environments," in *Grid Computing*, pp. 25–32, Springer, Berlin, Germany, 2004.
- [25] C. H. Papadimitriou and M. Yannakakis, "Multiobjective query optimization," in *Proceedings of the 20th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, pp. 52–58, May 2001.
- [26] G. B. Berriman, G. Juve, E. Deelman, M. Regelson, and P. Plavchan, "The application of cloud computing to astronomy: a study of cost and performance," in *Proceedings of the 6th IEEE International Conference on e-Science Workshops (e-ScienceW '10)*, pp. 1–7, Brisbane, Australia, December 2010.
- [27] P. Koutris, P. Upadhyaya, M. Balazinska, B. Howe, and D. Suciu, "Toward practical query pricing with query market," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data (SIGMOD '13)*, pp. 613–624, 2013.
- [28] E. Deelman, G. Singh, M. Livny, B. Berriman, and J. Good, "The cost of doing science on the cloud: the montage example," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis (SC '08)*, p. 50, Austin, Tex, USA, November 2008.
- [29] D. Chatziantoniou and E. Tzortzakakis, "ASSET queries: a declarative alternative to MapReduce," *SIGMOD Record*, vol. 38, no. 2, pp. 35–41, 2009.
- [30] T. Condie, N. Conway, P. Alvaro et al., "Online aggregation and continuous query support in MapReduce," in *Proceedings of the International Conference on Management of Data (SIGMOD '10)*, pp. 1115–1118, June 2010.
- [31] H. Mahboubi and J. Darmont, "Enhancing XML data warehouse query performance by fragmentation," in *Proceedings of the 24th Annual ACM Symposium on Applied Computing (SAC '09)*, pp. 1555–1562, March 2009.
- [32] M. Armbrust, A. Fox, R. Griffith et al., "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.
- [33] J. Zhou, P. A. Larson, and H. G. Elmongui, "Lazy maintenance of materialized views," in *Proceedings of the 33rd International Conference on Very Large Data Bases*, pp. 231–242, 2007.
- [34] K. Tsakalozos, H. Kllapi, E. Sitaridi, M. Roussopoulos, D. Paparas, and A. Delis, "Flexible use of cloud resources through profit maximization and price discrimination," in *Proceedings of the IEEE 27th International Conference on Data Engineering (ICDE '11)*, pp. 75–86, Hanover, Germany, April 2011.
- [35] U. Sharma, P. Shenoy, S. Sahu, and A. Shaikh, "A cost-aware elasticity provisioning system for the cloud," in *Proceedings of the 31st International Conference on Distributed Computing Systems (ICDCS '11)*, pp. 559–570, Minneapolis, Minn, USA, July 2011.
- [36] G. Weikum, A. Moenkeberg, C. Hasse, and P. Zabback, "Self-tuning database technology and information services: from wishful thinking to viable engineering," in *Proceedings of VLDB*, pp. 20–31, 2002.
- [37] S. Agrawal, S. Chaudhuri, A. Das, and V. Narasayya, "Automating layout of relational databases," in *Proceedings of the 19th International Conference on Data Engineering*, pp. 607–618, March 2003.
- [38] A. J. Storm, C. Garcia-Arellano, S. S. Lightstone, Y. Diao, and M. Surendra, "Adaptive self-tuning memory in DB2," in *Proceedings of VLDB*, pp. 1081–1092, 2006.
- [39] R. S. G. Lanzelotte, "Industrial-strength parallel query optimization: issues and lessons," *Information Systems*, vol. 19, no. 4, pp. 311–330, 1994.
- [40] L. Golshanara, S. M. T. R. Rankoohi, and H. Shah-Hosseini, "A multi-colony ant algorithm for optimizing join queries

- in distributed database systems,” *Knowledge and Information Systems*, vol. 39, no. 1, pp. 175–206, 2014.
- [41] F. Sourd and O. Spanjaard, “A multiobjective branch-and-bound framework: application to the biobjective spanning tree problem,” *INFORMS Journal on Computing*, vol. 20, no. 3, pp. 472–484, 2008.
- [42] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.
- [43] E. R. Hruschka and N. F. Ebecken, “A genetic algorithm for cluster analysis,” *Intelligent Data Analysis*, vol. 7, no. 1, pp. 15–25, 2003.
- [44] U. Tosun, T. Dokeroglu, and A. Cosar, “A robust island parallel genetic algorithm for the quadratic assignment problem,” *International Journal of Production Research*, vol. 51, no. 14, 2013.
- [45] E. Sevinç and A. Coşar, “An evolutionary genetic algorithm for optimization of distributed database queries,” *Computer Journal*, vol. 54, no. 5, pp. 717–725, 2011.
- [46] M. A. Bayir, I. H. Toroslu, and A. Cosar, “Genetic algorithm for the multiple-query optimization problem,” *IEEE Transactions on Systems, Man and Cybernetics C: Applications and Reviews*, vol. 37, no. 1, pp. 147–153, 2007.
- [47] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

Research Article

An Investigation of Generalized Differential Evolution Metaheuristic for Multiobjective Optimal Crop-Mix Planning Decision

Oluwole Adekanmbi,¹ Oludayo Olugbara,¹ and Josiah Adeyemo²

¹ Department of Information Technology, Durban University of Technology, P.O. Box 1334, Durban 4000, South Africa

² Department of Civil Engineering and Surveying, Durban University of Technology, P.O. Box 1334, Durban 4000, South Africa

Correspondence should be addressed to Oludayo Olugbara; oludayoo@dut.ac.za

Received 12 February 2014; Accepted 24 March 2014; Published 23 April 2014

Academic Editors: T. O. Ting and X.-S. Yang

Copyright © 2014 Oluwole Adekanmbi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper presents an annual multiobjective crop-mix planning as a problem of concurrent maximization of net profit and maximization of crop production to determine an optimal cropping pattern. The optimal crop production in a particular planting season is a crucial decision making task from the perspectives of economic management and sustainable agriculture. A multiobjective optimal crop-mix problem is formulated and solved using the generalized differential evolution 3 (GDE3) metaheuristic to generate a globally optimal solution. The performance of the GDE3 metaheuristic is investigated by comparing its results with the results obtained using epsilon constrained and nondominated sorting genetic algorithms—being two representatives of state-of-the-art in evolutionary optimization. The performance metrics of additive epsilon, generational distance, inverted generational distance, and spacing are considered to establish the comparability. In addition, a graphical comparison with respect to the true Pareto front for the multiobjective optimal crop-mix planning problem is presented. Empirical results generally show GDE3 to be a viable alternative tool for solving a multiobjective optimal crop-mix planning problem.

1. Introduction

The overarching objective of this work is to investigate the performance of the generalized differential evolution metaheuristic for multiobjective optimal crop-mix planning decision in the agricultural domain. The purpose of agricultural crop planning decision is generally to guarantee sufficient food resources for the human population, which is increasingly growing at a fast rate. In addition, the global demand for food items is growing at an accelerated rate. However, most of the available techniques for expanding agricultural systems have a serious long-term implication for the human environment [1]. The impact of increasing crop demand definitely depends heavily on the development of global agriculture. The needed development of the agricultural farming systems is directed toward achieving a great technology improvement. This should meet the year 2050 crop demand vision with much lower environmental impact. The impact of doubling

the global crop production will depend on how increased production is achieved [2].

The intensification of agricultural practices such as clearing land for massive crop production, achieving higher yields through increased agricultural inputs, and promoting innovations through the application of information communication technology could improve crop production and agricultural value chain [3]. The increasing growth of human population across the world has called for sustainable growth in agricultural products—so as to meet the primary needs of the human population [4]. One copious strategy of sustainable agriculture is crop-mix—sometimes called mixed cropping [5]. The formulation of multiobjective optimal crop-mix planning problem as presented in this paper simplifies the task of crop planning in agriculture setting that is generally aimed at maximizing returns from the meager resources available to farmers [6].

The current work explores an approach based on evolutionary metaheuristics to solve a multiobjective optimal crop-mix optimization problem. This could suggest an effective tool to support farmers in optimal crop planning decision making. There are numerous reasons for using evolutionary metaheuristics to solve optimization problems. One of such reasons is that evolutionary metaheuristics need little problem specific knowledge and can be applied to a broad range of problem types [7]. The evolutionary metaheuristics required the target objective function for a given problem to be optimized, but additional problem specific knowledge can be easily brought into metaheuristics to improve their performances [8]. In addition, metaheuristics require no derivative information; they are robust, flexible, and relatively simple to implement [9].

Previous studies on crop planning have used single and multiobjective optimization models, including linear programming [10–13], dynamic programming [14], and evolutionary metaheuristics [15–18] to solve diverse formulations of crop planning problem. The variety of optimization techniques previously considered for crop planning ranges from single to multiobjective and from linear to nonlinear forms, where computational intelligence techniques have been explored [18]. However, multiobjective optimization problems are frequently converted to single objective function optimization by means of weighting functions for several objective functions and solved using optimization techniques that are well suited for single objective function optimization. The relative importance of each objective function is expressed by the weighting functions. However, optimizing different objective functions concurrently without emphasizing the importance of each objective function *a priori* is called Pareto optimization. This relatively new optimization approach is more alluring for solving many nonlinear, multidimensional, multiobjective, combinatorial, nondifferentiable, nonconvex, and constrained practical problems often encountered in the real world phenomena.

2. Materials and Methods

2.1. Optimal Crop-Mix Planning Model. This section presents the mathematical formulation of the optimal crop-mix planning problem investigated in this work. The optimal crop-mix planning model is designed to maximize total net profit that can be produced by maximizing total crop production. The objective is to make an optimum use of the available limited resources in order to determine the land allocation for several competing crops required to be planted in a year. The soil characteristics, cropping patterns, crop produced, planting region, and cropping method are factors that contribute to the production cost, yield rate, and earning realized by a decision farmer. The crop-mix planning model is considered for a large scale planning incorporated with dataset collected from the South African abstract of agricultural statistics [19]. The model is specified as biobjective functions—profit function and crop production function with a set of constraints.

2.1.1. Objective Function 1: Profit Maximization. Maximize

$$F_1 = \sum_j^m \sum_{i \in M_j} (P_i \times U_{i,j,k=1} - B_1) \times X_{i,j,k=1} + \sum_j^m \sum_{i \in M_j} (P_i \times U_{i,j,k=2} - B_2) \times X_{i,j,k=2} + \sum_j^q \sum_{i \in Q_j} (P_i \times U_{i,j,k=3} - B_3) \times X_{i,j,k=3}, \quad (1)$$

where

$$B_s = (V_{i,j,k=s} \times R_{i,j,k=s}) + F_{i,j,k=s} \quad \forall s = 1, 2, 3. \quad (2)$$

Objective Function 2: Crop Production Maximization. Maximize

$$F_2 = \sum_j^m \sum_{i \in M_j} G_{i,j,k=1} \times X_{i,j,k=1} + \sum_j^n \sum_{i \in N_j} G_{i,j,k=2} \times X_{i,j,k=2} + \sum_j^q \sum_{i \in Q_j} G_{i,j,k=3} \times X_{i,j,k=3}. \quad (3)$$

The two objectives functions are to be concurrently solved, subject to the following constraints: food delivery, land allocation, labor cost, capital cost, and nonnegativity of decision variables.

Food Delivery Constraint. Consider

$$\sum_j^m \sum_{i \in M_j} G_{i,j,k=1} \times X_{i,j,k=1} + \sum_j^m \sum_{i \in M_j} G_{i,j,k=2} \times X_{i,j,k=2} + \sum_j^m \sum_{i \in M_j} G_{i,j,k=3} \times X_{i,j,k=3} \geq D_i \quad \forall i. \quad (4)$$

Land Allocation Constraint. Consider

$$\sum_i \sum_j W_k \times X_{i,j,k} \leq L_k, \quad \forall k. \quad (5)$$

Labor Cost Constraint. Consider

$$\begin{aligned} & \sum_j^m \sum_{i \in M_j} T_{i,j,k=1} \times X_{i,j,k=1} \\ & + \sum_j^m \sum_{i \in M_j} T_{i,j,k=2} \times X_{i,j,k=2} \\ & + \sum_j^m \sum_{i \in M_j} T_{i,j,k=3} \times X_{i,j,k=3} \geq H_k \quad \forall k. \end{aligned} \tag{6}$$

Capital Cost Constraint. Consider

$$\begin{aligned} & \sum_j^m \sum_{i \in M_j} B_1 \times X_{i,j,k=1} \\ & + \sum_j^m \sum_{i \in M_j} B_2 \times X_{i,j,k=2} \\ & + \sum_j^m \sum_{i \in M_j} B_3 \times X_{i,j,k=3} \leq C_a. \end{aligned} \tag{7}$$

Nonnegativity Constraint. Consider

$$X_{i,j,k} \geq 0 \quad \forall i, j, k, \tag{8}$$

where

- (i) i is a crop that can be considered for production,
- (ii) j is a crop combination made up of i ,
- (iii) k is land type, $k = 1$ for a single-cropped land, $k = 2$ for a double-cropped land, and $k = 3$ for a triple-cropped land.
- (iv) P_i is price in South Africa Rand (ZAR) of crop i per metric ton,
- (v) $V_{i,j,k}$ is the variable cost required per unit area for crop i of crop combination j in land type k ,
- (vi) $F_{i,j,k}$ is the fixed cost required per unit area for crop i of crop combination j in land type k ,
- (vii) $U_{i,j,k}$ is number of farming units of crop i of crop combination j in land type k ,
- (viii) $R_{i,j,k}$ is cultivating land area ratio of crop i of crop combination j in land type k ,
- (ix) $G_{i,j,k}$ is yield rate, which is the amount of production in metric tons per hectare of crop i of crop combination j in land type k ,
- (x) W_k is land type coefficient for land type k ,
- (xi) D_i is expected delivery (metric tons) of crop,
- (xii) L_k is available domain of land type k ,
- (xiii) C_a is working capital (ZAR), which indicates the total amount of money that can be invested for cropping,

- (xiv) M is number of alternative crops for single-cropped land,
- (xv) n is number of crop combinations for double-cropped land,
- (xvi) q is number of crop combinations for triple-cropped land,
- (xvii) M_j is a crop in each j for single-cropped land, $j = 1, \dots, m$,
- (xviii) N_j is the j th crop pair of the possible crop combinations of double-cropped land, $j = 1, \dots, n$,
- (xix) Q_j is the j th crop pair of the possible crop combinations of triple-cropped land, $j = 1, \dots, q$,
- (xx) $X_{i,j,k}$ is the area of land in hectare to be cultivated for a crop i of crop combination j in land type k .

2.2. Multiobjective Metaheuristics. The multiobjective evolutionary metaheuristics are population based techniques for solving complex multiobjective optimization problems. A metaheuristic is an iterative master process that guides and modifies the operation of a subordinate heuristic to efficiently produce high quality solutions by exploring and exploiting a solution search space [20]. The synonymous underlying principle for all evolutionary metaheuristics is that, given an initial population of individuals, an environmental pressure causes the natural selection of the surviving individuals—leading to a rise in the population fitness. The most surviving individuals, according to their measures of fitness, would steadily progress to the next generation by the application of recombination and mutation operators that create diversity. The recombination operator is applicable to two or more selected parents to produce a set of offsprings. The mutation operator is applicable to a single parent to reproduce an offspring and selection operator guarantees the quality of individuals in the given population. The execution of recombination and mutation operators yields a set of offsprings that compete with the existing population members for a place in the next generation. This process is iterated to refine the individuals produced and the iteration comes to a stop when a quality individual is found or a given termination condition is satisfied.

In this study, we investigated a set of metaheuristics to test the performance of generalized differential evolution for multiobjective optimal crop-mix planning problem. These metaheuristics are ϵ -constrained, widely used in practice to solve multiobjective optimization; the nondominated sorting genetic algorithm (NSGA), one of the most popular elitist multiobjective evolutionary algorithms; and the generalized differential evolution 3 (GDE3), a more recent metaheuristic that finds a global optimum solution for a multiobjective optimization problem.

2.2.1. Generalized Differential Evolution. The generalized differential evolution 3 (GDE3) [21] modifies the selection rule of the basic differential evolution (DE) [22] and extends DE/rand/1/bin strategy [23] to problems with M objective functions and K constraint functions. In DE/rand/1/bin notation, “rand” indicates how the vector for mutation is selected.

The number of vector differences used in the mutation is indicated next, and “bin” indicates the way the old vector and the trial vector are recombined. The basic principle behind the selection rule is that the trial vector $u_{i,G}$ is compared with the old vector $x_{i,G}$. If the trial vector has an equal or lower objective value, then it replaces the old vector in the next generation [24]. This can be presented as follows:

$$x_{i,G+1} = \begin{cases} u_{i,G} & \text{if } f(u_{i,G}) \leq f(x_{i,G}) \\ x_{i,G} & \text{otherwise.} \end{cases} \quad (9)$$

In the case of comparing feasible, incomparable, and nondominating solutions, both offspring and parent vectors are saved for the population of the next generation. This mechanism reduces the computational costs of the metaheuristic and improves its efficiency. The population size may increase at the end of a generation based on a similar selection method as used in NSGA-II; the population is reduced back to the original size. The sorting of the population members is based on the goal for a *posteriori* optimization. The worst population members are eliminated according to the principle of nondominance and crowding in order to reduce the population size to the original size. The GDE3 is similar to the differential evolution for multiobjective optimization (DEMO) [25], except that DEMO does not provide a mechanism for constraint handling nor recedes to basic DE in the case of a single objective. This is because DEMO modifies the basic DE and does not consider weak dominance in the selection. The GDE3 improves the ability to handle multiobjective optimization problems by giving a better distributed set of solutions and is less sensitive to the selection of control parameter values when compared to the earlier versions of GDE [26].

2.2.2. The ε -Constrained. The ε -constrained optimization technique is capable of generating widespread alternative solutions to constrained multiobjective version of the crop-mix planning model. The ε -constrained technique is based on the principle of selecting one objective function—usually the most preferred one to be optimized—whilst the other objective functions are treated as constraints that are bounded by some allowable levels of epsilon, ε_i [27]. This implies that a single objective function optimization problem is defined for the most relevant objective functions subject to additional constraints. The levels of ε_i required to generate the entire Pareto optimal set for an optimization problem are then altered. The apparent difficulty encountered when using this technique is how to choose the ε_i values. It is practically hard to know beforehand what the best values will be. If the value of ε is slightly increased, it could lead to a lot of redundant runs and if the steps between different runs are too large, it is possible to miss Pareto optimal solutions.

2.2.3. Nondominated Sorting Genetic Algorithm. The nondominated sorting genetic algorithm (NSGA) [28] has three essential properties: (i) it uses an elitist principle; (ii) it uses an explicit diversity preserving mechanism; and (iii) it emphasizes the nondominated solutions. In a typical scenario of the fast nondominated sorting, two entities are required to

be computed: (i) domination count n_p , which is the number of solutions that dominate the solution p , and (ii) S_p , which is a set of solutions that p dominates. The solutions with $n_p = 0$ represent the first nondominated Pareto front. Thereafter, for each solution with $n_p = 0$, each member (q) of its set S_p is visited and its domination count is reduced by one to remove solution p from n_q . The member for which the domination count becomes zero ($n_q = 0$) is put in a separate list Q, which represents the second domination front. These procedures are repeated for each member of Q to identify the third front and the procedure is continued until all fronts are identified. In order to obtain a density estimation of possible solutions surrounding a particular solution, the average distance of two points is computed on either side of the point along each of the objectives [29].

3. Results and Discussion

The multiobjective optimal crop-mix planning problem was solved using GDE3, NSGA-II, and ε -constrained. The population size was 100 and the number of generations was 50. The GDE3, NSGA-II, and ε -constrained metaheuristics were implemented using C-Sharp programming language in VISUAL-STUDIO version 2010 on an HP PC with Pentium dual core processor having 2.30 GHz clock speed and 4 GB of RAM. Simulation experiments were performed to determine the best values of step length “ F ” and crossover rate “ CR ” for better performance in GDE3 metaheuristic. The values of CR and F were varied from 0.1 to 1 with an increment of 0.1. The simulation experiments were conducted for each value of F with respect to all values of CR . Consequently, 100 such simulation experiments were performed.

The GDE3 was compared to NSGA-II and ε -constrained to investigate its performance when used to solve the multiobjective optimal crop-mix planning model considered in this work. It is interesting to discover that NSGA-II is very sensitive to the initial population. Due to the inability of NSGA-II to find a single feasible solution using many different seeds as experimented, the GDE3 was introduced to compare the outputs of the three metaheuristics. The improved selection based on crowding distance was demonstrated in the optimal crop-mix planning problem. The GDE3 found a solution that converged to the Pareto front in about 50 generations. In addition, the results of the comparative study show that better Pareto front is obtained by GDE3 with $F = 0.5$ and $CR = 0.9$. The control parameters for NSGA-II are the crossover probability $P_c = 0.9$ and mutation probability $P_m = 1/D$ (D is the number of decision variables). The distribution index of crossover operator $\eta_c = 20$ and the distribution index of mutation operator $\eta_m = 20$. The number of the needed function evaluations for GDE3, NSGA-II, and ε -constrained was set at 10000.

In order to compare the performance of GDE3 with performances of NSGA-II and ε -constrained, 100 trial runs were conducted for solving the optimal crop-mix planning problem. The best front was obtained from GDE3 based on the statistical performance metrics of *additive epsilon indicator (AEI)*, *generational distance (GD)*, *inverted generational distance (IGD)*, and *spacing (S)* [30, 31].

TABLE 1: Additive epsilon indicator metric (10^{-3}).

	GDE3	NSGA-II	ϵ -Constrained
Best	7.59	8.29	9.50
Average	8.24	9.16	10.90
Worst	9.65	10.5	13.10
Std. dev.	0.431	0.726	9.83

TABLE 2: Generational distance metric (10^{-3}).

	GDE3	NSGA-II	ϵ -Constrained
Best	2.15	2.73	2.94
Average	3.45	4.06	4.12
Worst	6.87	7.10	7.92
Std. dev.	1.48	1.74	1.91

TABLE 3: Inverted generational distance metric (10^{-3}).

	GDE3	NSGA-II	ϵ -Constrained
Best	2.40	2.62	2.73
Average	2.81	3.52	3.88
Worst	3.15	4.39	5.01
Std. dev.	0.171	0.398	0.512

TABLE 4: Spacing metric (10^{-3}).

	GDE3	NSGA-II	ϵ -Constrained
Best	1.07	1.23	1.62
Average	1.47	1.53	1.92
Worst	1.69	2.03	2.84
Std. dev.	0.415	0.653	0.976

Table 1 shows the results of AEI, wherein it can be seen that GDE3 recorded the best average value of 0.00824, closely followed by NSGA-II with a value of 0.00916, and ϵ -constrained ranked third with a value of 0.0109. The overall worst performing metaheuristic according to this metric is the ϵ -constrained.

Table 2 shows the results of GD, wherein it can be seen that GDE3 had the best performance, both in terms of the average value and the standard deviation, followed by NSGA-II. The GDE3 had the best average value of 0.00345, closely followed by NSGA-II with a value of 0.00406, and ϵ -constrained is ranked third with a value of 0.00412. The overall worst performing metaheuristic according to this metric is the ϵ -constrained.

Table 3 shows the results of IGD, wherein it can be seen that GDE3 gave the best result with an average value of 0.00281 followed by the NSGA-II with an average value of 0.00352. The NSGA-II and the ϵ -constrained were ranked second and third, respectively. The overall worst performing metaheuristic with respect to this performance metric is the ϵ -constrained.

Table 4 shows the result of spacing, wherein it can be seen that GDE3 gave the best result with an average value of 0.001469, closely followed by the NSGA-II with a value of

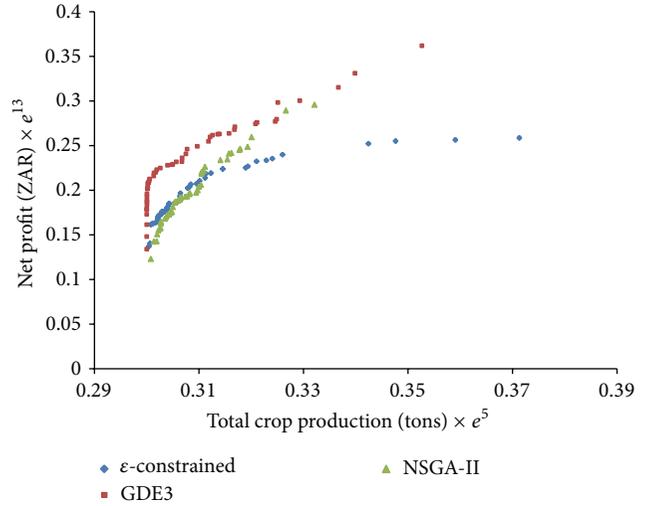


FIGURE 1: Pareto optimal set for GDE3, NSGA-II, and ϵ -constrained metaheuristics.

0.001534. The worst performing metaheuristic according to this metric is the ϵ -constrained.

In general, because performance metrics can sometimes be misleading in multiobjective optimization, it is always desirable to consider a graphical comparison whenever possible [32]. It is particularly germane when analyzing the Pareto front produced by a metaheuristic to consider two main evaluation criteria: (i) the placement of the solutions on the true Pareto front and (ii) uniform distribution of solutions along the Pareto front. These criteria were applied to establish graphical comparisons of the metaheuristics investigated in this work.

The first evaluation criterion requires us to determine the extent to which the points on the Pareto front are linearly correlated. Consequently, the metaheuristic that gives a Pareto front with points more linearly correlated is judged to be the best performing in solving the multiobjective optimal crop-mix planning problem. This approach is effective because it could indicate a natural association between crop production and net profit. The strength of the linearity of an association between two variables such as crop production and net profit can be determined by calculating the Pearson correlation coefficient. The correlation coefficient is a number between -1 and 1 that indicates the strength of the linear association between two variables, for instance, crop production and net profit. The higher positive value indicates a strong linear association in the same direction; that is, increase/decrease in one variable leads to increase/decrease in the other variable. If there is evidence of strong linearity, we would likely expect higher values of crop production to yield higher values of net profit. The second evaluation criterion suggests that solutions on the Pareto front should be uniformly distributed.

Figure 1 shows the Pareto fronts for the metaheuristics investigated in this work. In Figure 1, it can be seen that NSGA-II had a good distribution of solutions that it found, but it missed some portion of Pareto front. Hence, NSGA-II had an average performance of the multiobjective optimal

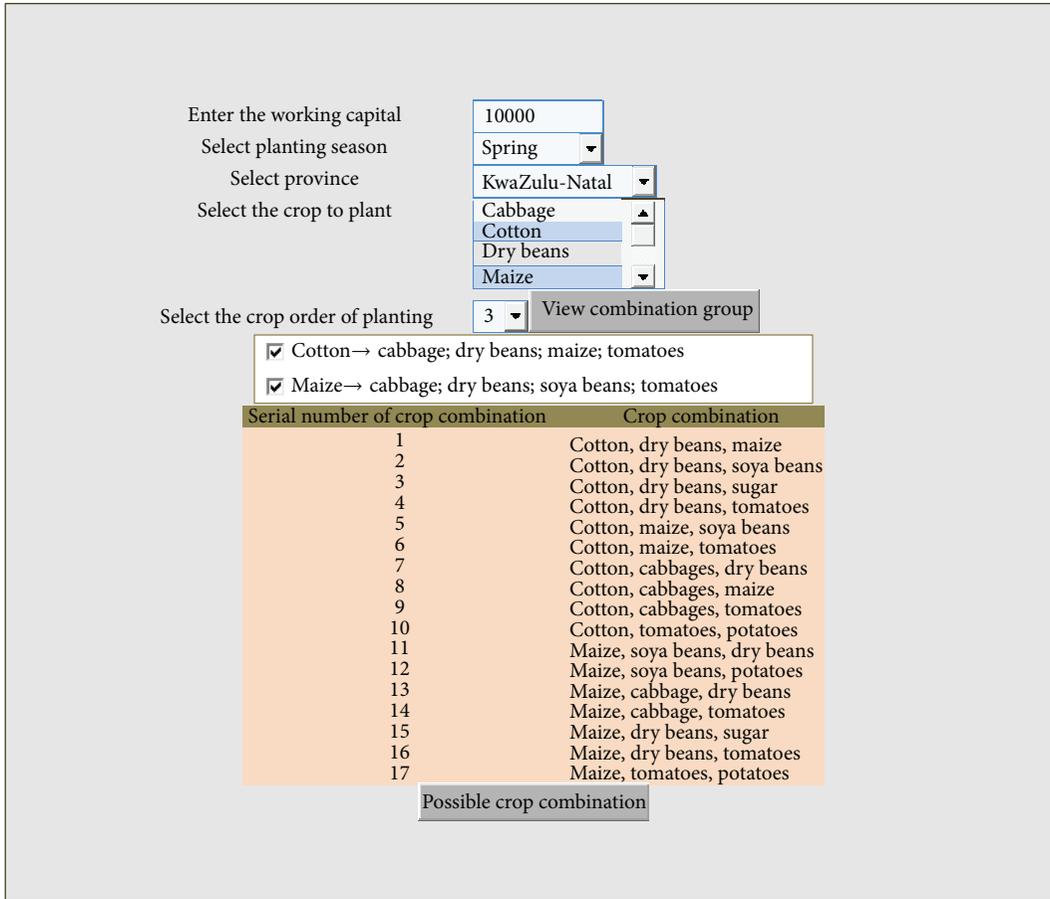


FIGURE 2: The screenshot of the process page of the decision support system.

crop-mix planning problem, despite its good spacing values. The correlation coefficient computed for the Pareto front of NSGA-II is 0.9711, which is slightly less than that of GDE3, which is 0.9736. The distribution of ϵ -constrained has better spacing but missed a considerable portion of the true Pareto front as in Figure 1. In addition, the metaheuristic has the least coefficient of correlation of 0.9708. It has been pointed out that having a good distribution of solutions becomes irrelevant when the metaheuristic does not converge to the true Pareto front of the optimal crop-mix planning problem [16]. The GDE3 as shown in Figure 1 clearly had the best overall performance, both in terms of distribution of solutions and the placement of solutions on Pareto front.

4. Implementation of a Crop Planning System

The crop planning system based on the generalized differential evolution 3 (GDE3) was implemented using the C-Sharp programming language in VISUAL-STUDIO. The purpose of the implementation was to provide a software tool to assist farmers in optimal crop planning decision making. The testing of the crop planning software was done with 10000 fitness function evaluations. The combination of parameters chosen for the testing is appropriate to have a reasonably good performance. This can be corroborated by checking the

original sources of the GDE3. The varied operations such as capturing crop information and managing the information on crop combination as provided by the crop planning system could be used for crop planning related activities such as land allocation and crop selection. The recorded data are stored in a database for easy accessibility and could be used in various planning and decision making processes. The prototype system is relatively easy to use and simple to accommodate basic users with very little literacy levels.

The system was tested with a scenario where a household farmer has a working capital of R10,000 (unit in South Africa rand) with the land mass of 1 hectare. The farmer chooses to plant crops that could be planted along with cotton and maize such that the crop combination should be of order 3; that is the farmer decided to plant on a tricropped land. The farmer supplied all the necessary inputs and clicked on the button (view combination group) to view the crop combination group, consisting of crops that could be planted with the selected crops (cotton, maize). In order to view the number of possible crop combinations that could be obtained, the farmer selected any of the crop combination group of his/her choice and clicked the button (possible crop combination). Figure 2 shows the screenshot of the process.

The system allocated a land portion to each crop combination; working with the scenario where the farmer decided

TABLE 5: The land allocation result.

Serial number of crop combination	Optimization Crop combination	Allocated land portion
1	Cotton, dry beans, maize	0.0538847660170879
2	Cotton, dry beans, soya beans	0.0552823360239594
3	Cotton, dry beans, sugar	0.0554001961353126
4	Cotton, dry beans, tomatoes	0.0537548193766635
5	Cotton, maize, soya beans	0.054378675341399
6	Cotton, maize, tomatoes	0.0537169683262593
7	Cotton, cabbages, dry beans	0.0560629255563752
8	Cotton, cabbages, maize	0.0529411764705882
9	Cotton, cabbages, tomatoes	0.0529411764705882
10	Cotton, tomatoes, potatoes	0.0530232065275474
11	Maize, soya beans, dry beans	0.053340093038395
12	Maize, soya beans, potatoes	0.054408535208579
13	Maize, cabbage, dry beans	0.0577712841409234
14	Maize, cabbage, tomatoes	0.0529411764705882
15	Maize, dry beans, sugar	0.0535396709191825
16	Maize, dry beans, tomatoes	0.0529411764705882
17	Maize, tomatoes, potatoes	0.0532893345994195

TABLE 6: Output of the optimization process.

Net profit (ZAR)	Total crop production (tons)	Total land utilization (ha)
995.31296475439	31.5857454386647	0.919617517093457

to choose both combination groups, the system produced the result in Table 5. Finally, Table 6 shows the best result of the optimization process while maximizing total crop production and minimizing total planting area concurrently.

5. Conclusions

This work suggests that generalized differential evolution 3 (GDE3) is a useful multiobjective optimization tool for optimal crop-mix planning decision support. The metaheuristic is able to produce improved results when compared to those generated by other two metaheuristics that are representatives of the state-of-the-art in evolutionary multiobjective optimization. The GDE3 uses a simple mechanism to deal with constraints and the results computed by the metaheuristic generally indicate that such mechanism, despite its simplicity, is effective in practice.

The following conclusions can be made about the performance of GDE3: (i) GDE3 is able to produce most of the true Pareto fronts of the optimal crop-mix planning problem considered and it has the best performance; (ii) the GDE3 is able to produce a good distribution of solutions of the multiobjective optimal crop-mix planning problem; and (iii) GDE3 is ranked first with respect to the selected performance metrics. It can be concluded that GDE3 is practically effective for supporting optimal crop planning decision making process. Given the features of GDE3, an extension of the paradigm for multiobjective optimization can be particularly useful to deal with dynamic functions. The

performance comparison of these metaheuristics is valuable for a decision maker to consider tradeoffs in accuracy versus complexity of solution techniques.

Future work will extend GDE3 for crop planning decision under uncertainty. This will produce a novel approach to deal with practical situations for which profit coefficients of agriculture are uncertain. The optimization approach can help farmers to efficiently utilize the available meager resources, including planting area, time, and money. The approach combines indigenous farming with information communication technology to optimize crop production, support efficient planning, and help farmers determine the possible combination of crops to plant on the same planting land year by year. As part of the future work, other optimization techniques can be compared to GDE3 to establish its superiority over many other techniques for crop planning decision making.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

[1] D. Tilman, C. Balzer, J. Hill, and B. L. Befort, "Global food demand and the sustainable intensification of agriculture," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 108, no. 50, pp. 20260–20264, 2011.

- [2] D. Gray, W. Parker, and E. Kemp, "Farm management research: a discussion of some of the important issues," *Journal of International Farm Management*, vol. 5, no. 1, pp. 1–24, 2009.
- [3] D. Tilman, K. G. Cassman, P. A. Matson, R. Naylor, and S. Polasky, "Agricultural sustainability and intensive production practices," *Nature*, vol. 418, no. 6898, pp. 671–677, 2002.
- [4] I. Lewandowski, M. Härdtlein, and M. Kaltschmitt, "Sustainable crop production: definition and methodological approach for assessing and implementing sustainability," *Crop Science*, vol. 39, no. 1, pp. 184–193, 1999.
- [5] W. Steve and L. Henri, "Raising agricultural productivity in Africa: options for action, and the role of subsidies," Africa Progress Panel Policy Brief, 2010.
- [6] W. Sadok, F. Angevin, J.-É. Bergez et al., "Ex ante assessment of the sustainability of alternative cropping systems: implications for using multi-criteria decision-aid methods—a review," in *Sustainable Agriculture*, pp. 753–767, Springer, Amsterdam, The Netherlands, 2009.
- [7] K. Deb, "Multi-objective optimization," in *Multi-Objective Optimization Using Evolutionary Algorithms*, pp. 13–46, John Wiley & Sons, New York, NY, USA, 2001.
- [8] R. Brunelli and C. von Lüken, "Optimal crop selection using multiobjective evolutionary algorithms," *AI Magazine*, vol. 30, no. 2, pp. 96–105, 2009.
- [9] V. Guliashki, H. Toshev, and C. Korsemov, "Survey of evolutionary algorithms used in multiobjective optimization," *Problems of Engineering Cybernetics and Robotics*, vol. 60, pp. 42–54, 2009.
- [10] D. Bai and W. Liang, "Optimal planning model of the regional water saving irrigation and its application," in *Proceedings of the International Symposium on Geomatics for Integrated Water Resources Management (GIWRM '12)*, pp. 1–4, Lanzhou, China, October 2012.
- [11] S. Chetty and A. Adewumi, "Comparison study of swarm intelligence techniques for the annual crop planning problem," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 2, pp. 258–268, 2014.
- [12] S. A. Mohaddes and M. G. Mohayidin, "Application of the fuzzy approach for agricultural production planning in a watershed, a case study of the Atrak watershed, Iran," *American-Eurasian Journal of Agricultural & Environmental Science*, vol. 3, no. 4, pp. 636–645, 2008.
- [13] R. A. Sarker, S. Talukdar, and A. Haque, "Determination of optimum crop mix for crop cultivation in Bangladesh," *Applied Mathematical Modelling*, vol. 21, no. 10, pp. 621–632, 1997.
- [14] M. Haouari and M. N. Azaiez, "Optimal cropping patterns under water deficits," *European Journal of Operational Research*, vol. 130, no. 1, pp. 133–146, 2001.
- [15] J. Adeyemo and F. O. Otieno, "Maximum irrigation benefit using multiobjective differential evolution algorithm (MDEA)," *OIDA International Journal of Sustainable Development*, vol. 1, no. 2, pp. 39–44, 2010.
- [16] K. S. Raju, A. Vasan, P. Gupta, K. Ganesan, and H. Mathur, "Multi-objective differential evolution application to irrigation planning," *ISH Journal of Hydraulic Engineering*, vol. 18, no. 1, pp. 54–64, 2012.
- [17] R. A. Sarker and M. A. Quaddus, "Modelling a nationwide crop planning problem using a multiple criteria decision making tool," *Computers & Industrial Engineering*, vol. 42, no. 2–4, pp. 541–553, 2002.
- [18] R. Sarker and T. Ray, "An improved evolutionary algorithm for solving multi-objective crop planning models," *Computers and Electronics in Agriculture*, vol. 68, no. 2, pp. 191–199, 2009.
- [19] AAS (Abstract of Agricultural Statistics), *Directorate Agricultural Information*, National Department of Agriculture, Pretoria, South Africa, 2012.
- [20] S. Voss, I. H. Osman, and C. Roucairol, *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer Academic, Boston, Mass, USA, 1999.
- [21] S. Kukkonen and J. Lampinen, "Performance assessment of generalized differential evolution 3 with a given set of constrained multi-objective test problems," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 1943–1950, Trondheim, Norway, May 2009.
- [22] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*, Natural Computing Series, Springer, Berlin, Germany, 2005.
- [23] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.
- [24] S. Kukkonen and J. Lampinen, "GDE3: the third evolution step of generalized differential evolution," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '05)*, pp. 443–450, Edinburgh, Scotland, September 2005.
- [25] T. Robič and B. Filipič, "DEMO: differential evolution for multiobjective optimization," in *Proceedings of the 3rd International Conference on Evolutionary Multi-Criterion Optimization (EMO '05)*, pp. 520–533, Guanajuato, Mexico, March 2005.
- [26] B. Luo, J. Zheng, J. Xie, and J. Wu, "Dynamic crowding distance? A new diversity maintenance strategy for MOEAs," in *Proceedings of the 4th International Conference on Natural Computation (ICNC '08)*, vol. 1, pp. 580–585, Jinan, China, October 2008.
- [27] T. Takahama and S. Sakai, "Constrained optimization by the ϵ constrained differential evolution with gradient-based mutation and feasible elites," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '06)*, pp. 1–8, Barcelona, Spain, July 2006.
- [28] L. Wang, T.-G. Wang, and Y. Luo, "Improved non-dominated sorting genetic algorithm (NSGA)-II in multi-objective optimization studies of wind turbine blades," *Applied Mathematics and Mechanics*, vol. 32, no. 6, pp. 739–748, 2011.
- [29] S. Panda, "Multi-objective PID controller tuning for a FACTS-based damping stabilizer using Non-dominated Sorting Genetic Algorithm-II," *International Journal of Electrical Power & Energy Systems*, vol. 33, no. 7, pp. 1296–1308, 2011.
- [30] J. Knowles and D. Corne, "On metrics for comparing non-dominated sets," in *Proceedings of the Congress on Evolutionary Computation (CEC '02)*, pp. 711–716, Honolulu, Hawaii, USA, May 2002.
- [31] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: an analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.
- [32] C. A. C. Coello and N. C. Cortés, "Solving multiobjective optimization problems using an artificial immune system," *Genetic Programming and Evolvable Machines*, vol. 6, no. 2, pp. 163–190, 2005.

Research Article

Towards the Novel Reasoning among Particles in PSO by the Use of RDF and SPARQL

Iztok Fister Jr.,¹ Xin-She Yang,² Karin Ljubič,³ Dušan Fister,¹
Janez Brest,¹ and Iztok Fister¹

¹ University of Maribor, Faculty of Electrical Engineering and Computer Science, Smetanova 17, 2000 Maribor, Slovenia

² Middlesex University Hendon Campus, London NW4 4BT, UK

³ University of Maribor Faculty of Medicine, Taborska 8, 2000 Maribor, Slovenia

Correspondence should be addressed to Iztok Fister Jr.; iztok.fister2@uni-mb.si

Received 15 January 2014; Accepted 26 February 2014; Published 27 March 2014

Academic Editors: N. Chakraborti, P. Melin, and F. Neri

Copyright © 2014 Iztok Fister Jr. et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The significant development of the Internet has posed some new challenges and many new programming tools have been developed to address such challenges. Today, semantic web is a modern paradigm for representing and accessing knowledge data on the Internet. This paper tries to use the semantic tools such as resource definition framework (RDF) and RDF query language (SPARQL) for the optimization purpose. These tools are combined with particle swarm optimization (PSO) and the selection of the best solutions depends on its fitness. Instead of the local best solution, a neighborhood of solutions for each particle can be defined and used for the calculation of the new position, based on the key ideas from semantic web domain. The preliminary results by optimizing ten benchmark functions showed the promising results and thus this method should be investigated further.

1. Introduction

Searching for the optimal solutions of the hardest real-world problems is an active field especially in computer science. An eternal desire of computer scientists is to develop a general problem solver that will be able to cope with all classes of real-world problems. Unfortunately, the most of the so-called clever algorithms are subject of the No Free Lunch Theorem [1]. Regarding this theorem, if one algorithm is good on one class of problems, it does not mean that it will also be good on the other classes of problems. Especially, three domains of algorithms have recently been appeared in the role of general problem solver, as follows: Artificial Intelligence (AI) [2], evolutionary algorithms (EA) [3], and Swarm Intelligence (SI) [4]. While the former mimics operating a human brain, the latter domains are inspired by nature. Evolutionary algorithms are inspired by Darwinian principles of natural evolution [5] according to which the fittest individuals have the greater possibilities for survival and pass on their characteristics to their offspring during a process of reproduction.

Nowadays, evolutionary computation (AC) [6] captures the algorithms involved in evolutionary domain and it considers genetic algorithms (GA) [7], genetic programming [8], evolution strategies (ES) [9], evolutionary programming [10], and differential evolution (DE) [11–13]. The mentioned algorithms differ between each other according to representation of individual. As a result, these kinds of algorithms have been applied to various optimization, modeling, and simulation problems.

However, this paper concentrates on the SI domain that is concerned with the design of multiagent systems with applications, for example, in optimization and in robotics [4]. Inspiration for the design of these systems is taken from the collective behavior of social insects, like ants, termites, and bees, as well as from the behavior of other animal societies, like flocks of birds or schools of fish. Recently, there exist a lot of different algorithms from this domain that is still being developed. Let us mention only the most important members of the SI algorithms, as follows: the particle swarm optimization (PSO) [14], the firefly algorithm (FA) [15, 16], cuckoo search [17], the bat algorithm (BA) [18, 19], and so forth.

The PSO is population-based algorithm that mimics movement of the swarm of particles (e.g., birds) by flying across a landscape, thus searching for food. Each particle in PSO represents the candidate solution of the problem to be solved. Position of the particle consists of the problem parameters that are modified when the virtual particle is moved in the search space. The motion depends on the current particle position and the current position of local best and global best solutions, respectively. The local best solutions denote the best solutions that are whenever arisen on the definite location in the population, while the current best is the best solution whenever found in the whole population. This solution has the main impact on the direction of moving the swarm towards the optimal solution. When this solution is not improved anymore, the population gets stuck into a local optimum.

Mainly, we focused on the new definition of the neighborhood within the PSO algorithm. In place of the local best solutions, the neighborhood is defined using the predefined radius of fitness values around each candidate solution, thus capturing all candidate solutions with the fitness value inside the predefined virtual radius. The size of this neighborhood can be variable. Therefore, at least one but maximum three candidate solutions can be permitted to form this neighborhood. Although this is not the first try how to define the variable neighborhood within the PSO algorithm [20–22], in this paper, such neighborhood is defined using the Resource Description Framework (RDF), SPARQL Protocol, and RDF query language (SPARQL) tools taken from semantic web domain. As a result, the modified RDF-PSO algorithm was developed. Both web tools are appropriate for describing and manipulating decentralized and distributed data. On the other hand, the original PSO algorithm maintains a population of particles that are also decentralized in their nature. An aim of using these web tools was to simulate a distributed population of particles, where each particle is placed on the different location in the Internet.

The remainder of this paper is structured as follows. In Section 2, we outline a short description of the PSO algorithm. The section is finished by introducing the semantic web tools, that is, RDF and SPARQL. Section 3 concentrates on development of the modified RDF-PSO algorithm. Section 4 presents the conducted experiments and results obtained with the RDF-PSO algorithms. Finally, Section 5 summarizes our work and potential future directions for the future work are outlined.

2. Background

2.1. Particle Swarm Optimization. Particle swarm optimization (PSO) was one of the first SI algorithms to be presented at an International Conference on Neural Networks by Kennedy and Eberhart in 1995 [14]. PSO is inspired by the social foraging behavior of some animals such as flocking behavior of birds (Figure 1) and schooling behavior of fish [23]. In nature, there are some individuals with better developed instinct for finding food. According to these individuals, the

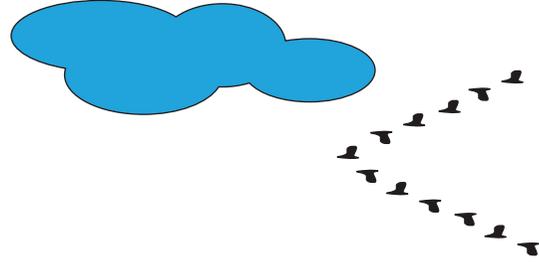


FIGURE 1: PSO.

whole swarm is directed into more promising regions in the landscape.

The PSO is a population-based algorithm that consists of N particles $\mathbf{x}_i^{(t)} = (x_{i1}, \dots, x_{iD})^T$ representing their position in a D -dimensional search space. These particles move across this space with velocity $\mathbf{v}_i^{(t)} = (v_{i1}, \dots, v_{iD})^T$ according to the position of the best particle $\mathbf{x}_{\text{best}}^{(t)}$ towards the more promising regions of the search space. However, this movement is also dependent on the local best position of each particle $\mathbf{p}_i^{(t)}$ and is mathematically expressed, as follows:

$$\begin{aligned} \mathbf{v}_i^{(t+1)} = & \mathbf{v}_i^{(t)} + c_1 r_1^{(t)} (\mathbf{p}_i^{(t)} - \mathbf{x}_i^{(t)}) \\ & + c_2 r_2^{(t)} (\mathbf{x}_{\text{best}}^{(t)} - \mathbf{x}_i^{(t)}), \quad \text{for } i = 1, \dots, N, \end{aligned} \quad (1)$$

where $r_1^{(t)}, r_2^{(t)}$ denote the random numbers drawn from the interval $[0, 1]$, and c_1, c_2 are constriction coefficients that determine the proportion, with which the local and global best solutions influence the current solution. Then, the new particle position is calculated according to the following expression:

$$\mathbf{x}_i^{(t+1)} = \mathbf{x}_i^{(t)} + \mathbf{v}_i^{(t)}, \quad \text{for } i = 1, \dots, N. \quad (2)$$

Pseudocode of the PSO algorithm is illustrated in Algorithm 1.

After finishing the initialization in function “init_particles” (Algorithm 1), the PSO algorithm optimizes a problem by iteratively improving the candidate solution [24, 25]. Thus, two functions are applied. The function “evaluate_the_new_solution” calculates the fitness value of particles obtained after initialization or movement. The movement according to (1) and (2) is implemented in the function “generate_new_solution.”

2.2. RDF. The RDF is an XML application devoted to encoding, exchanging, and reusing structural metadata [26]. It enables the knowledge to be represented in symbolical form. Fortunately, this knowledge is human readable. On the other hand, it is understandable to machines. The main characteristic of this framework is that RDF data can be manipulated on decentralized manner and distributed among various servers on the Internet. Resources identified by Uniform Resource Identifier (URI) are described in RDF graphs, where each resource representing the node has many properties that are associated with the resource using the

```

Input: PSO population of particles  $\mathbf{x}_i = (x_{i1}, \dots, x_{iD})^T$  for  $i = 1, \dots, N$ .
Output: The best solution  $\mathbf{x}_{\text{best}}$  and its corresponding value  $f_{\text{min}} = \min(f(\mathbf{x}))$ .
(1) init_particles;
(2) eval = 0;
(3) while termination_condition_not_meet do
(4)   for  $i = 1$  to  $N$  do
(5)      $f_i = \text{evaluate\_the\_new\_solution}(\mathbf{x}_i)$ ;
(6)     eval = eval + 1;
(7)     if  $f_i \leq p\text{Best}_i$  then
(8)        $\mathbf{p}_i = \mathbf{x}_i$ ;  $p\text{Best}_i = f_i$ ; // save the local best solution
(9)     end if
(10)    if  $f_i \leq f_{\text{min}}$  then
(11)       $\mathbf{x}_{\text{best}} = \mathbf{x}_i$ ;  $f_{\text{min}} = f_i$ ; // save the global best solution
(12)    end if
(13)     $\mathbf{x}_i = \text{generate\_new\_solution}(\mathbf{x}_i)$ ;
(14)  end for
(15) end while
    
```

ALGORITHM 1: Pseudocode of the classic PSO algorithm.

```

(1) <rdf:RDF>
(2) <rdf:Description rdf:about="http://www.example.org/Person">
(3) <ns1:Name>John</ns1:Name>
(4) <ns1:Surname>Smith</ns1:Surname>
(5) </rdf:Description>
(6) </rdf:RDF>
    
```

ALGORITHM 2: Pseudocode of the Person description in RDF.

property-type relationship. This relationship represents an edge in RDF graph. Thus, attributes may be atomic in nature (e.g., numbers, text strings, etc.) or represent other resources with their own properties [27].

The resource, property-type relation, and attribute present a triplet suitable for presentation in RDF graph. A sample of this graph is presented in Figure 2, from which two triples (also 2-triples) can be translated from the diagram. These 2-triples are written into a RDF database. In general, the format of this RDF data is serialization of N-triples obtained from the RDF graphs. For instance, the description of RDF database obtained from the RDF graph in Figure 2 is presented in Algorithm 2.

2.3. SPARQL. RDF enables data to be decentralized and distributed across the Internet. On the other hand, the SPARQL Protocol has been developed for accessing and discovering RDF data. SPARQL is an RDF query language that has its own syntax very similar to SQL queries. The SPARQL query consists of two parts [28]. The former SELECT clause identifies the variables that appear in the query results, while the latter WHERE clause provides the basic patterns that match against the RDF graph. Usually, these query patterns consist of three parts denoting the resource name, property-type relation, and attribute. As a result, the matched patterns are returned by the query. A sample of SPARQL query is illustrated in Algorithm 3.

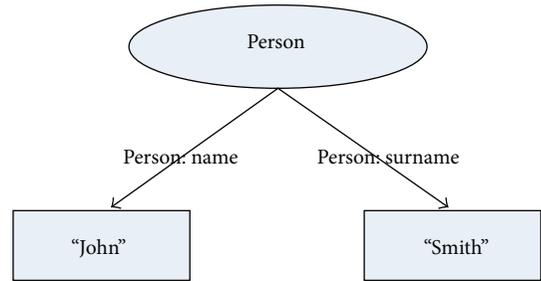


FIGURE 2: Diagram illustrates a resource Person in RDF graph. The resource consists of two atomic attributes “John” and “Smith” that are assigned to it with relations “PERSON: Name” and “PERSON: Surname”. In other words, the name of the person is “John” and the surname of the same person is “Smith.” Note that the word “PERSON:” denotes a location (URI), where a name space containing the description of semantics for this relation is located.

As a result of query presented in Algorithm 3, the name “John” and surname “Smith” are returned.

3. The Modified RDF-PSO Algorithm

The modified RDF-PSO algorithm implements two features:

- (i) using the variable neighborhood of candidate solutions in place of the local best solutions,

```

(1) SELECT ?name ?surname
(2) WHERE {
(3)   <http://www.example.org/Person> PERSON:Name ?name.
(4)   <http://www.example.org/Person> PERSON:Surname ?surname.
(5) }

```

ALGORITHM 3: Example of SPARQL query.

```

Input: PSO population of particles  $\mathbf{x}_i = (x_{i1}, \dots, x_{iD})^T$  for  $i = 1, \dots, N$ .
Output: The best solution  $\mathbf{x}_{\text{best}}$  and its corresponding value  $f_{\text{min}} = \min(f(\mathbf{x}))$ .
(1) init_particles;
(2) eval = 0;
(3) while termination_condition_not_meet do
(4)   for  $i = 1$  to  $N$  do
(5)      $f_i = \text{evaluate\_the\_new\_solution}(\mathbf{x}_i)$ ;
(6)     eval = eval + 1;
(7)     if  $f_i \leq f_{\text{min}}$  then
(8)        $\mathbf{x}_{\text{best}} = \mathbf{x}_i$ ;  $f_{\text{min}} = f_i$ ; // save the global best solution
(9)     end if
(10)     $\mathcal{N}(\mathbf{x}_i) = \{\mathbf{p}_j \mid \mathbf{p}_j \text{ is\_neighbor\_of } \mathbf{x}_i\}$ ;
(11)     $\mathbf{x}_i = \text{generate\_new\_solution}(\mathbf{x}_i, \mathcal{N}(\mathbf{x}_i))$ ;
(12)  end for
(13) end while

```

ALGORITHM 4: The proposed RDF-PSO algorithm.

- (ii) using the RDF for describing and SPARQL for manipulating this neighborhood.

The main reason for applying these well-known tools from the semantic web domain was to develop a distributed population model that could later be used in other SI algorithms. On the other hand, we try to use the semantic web tools for optimization purposes as well. Fortunately, RDF is suitable tool for describing the distributed population models, in general. In the PSO algorithm, it is applied for describing the relations between particles in population. For our purposes, a relation “is_neighbour_of” is important that each particle determines its neighborhood. Furthermore, SPARQL is used for determining the particles in its neighborhood. As a result, the RDF-PSO algorithm has been established, whose pseudocode is presented in Algorithm 4.

The three main differences distinguish the proposed RDF-PSO with the original PSO algorithm, as follows:

- (i) no local best solutions that are maintained by the RDF-PSO (lines 10–12 omitted in the Algorithm 1),
- (ii) defining the neighborhood of candidate solution (line 10 in Algorithm 4),
- (iii) generating the new solution according to the defined variable neighborhood relation (line 11 in Algorithm 4).

The relation $\mathcal{N}(\mathbf{x}_i) = \{\mathbf{x}_j \mid \mathbf{x}_j \text{ is_neighbor_of } \mathbf{x}_i\}$ (line 10 in Algorithm 4) is defined according to the following relation:

$$\mathbf{if} \quad \text{abs}(f(x_j) - f(x_i)) \leq R \quad \mathbf{then} \quad \mathbf{x}_j \in \mathcal{N}(\mathbf{x}_i), \quad (3)$$

where radius R defines the necessary maximum fitness distance of two candidate solutions that can be in neighborhood. In fact, this parameter regulates the number of candidate solutions in the neighborhood.

Here, the radius is expressed as $R = \sum_{i=1}^N |f(\mathbf{x}_i)| / \sqrt{N}$. Indeed, the neighborhood captures all solutions with the fitness differences less than the radius R . Typically, when the radius R is small, the size of neighborhood can also be small. However, this assertion holds if the population diversity is higher enough. When the particles are scattered across the search space, no particles are located in the vicinity of each other. Consequently, the size of neighborhood becomes zero. On the other hand, when the particles are crowded around some fitter individuals, the number of its neighbors can be increased enormously. In order to prevent this undersizing and oversizing, the neighborhood size is defined in such a manner that it cannot exceed the value of three and cannot be zero; in other words, $|\mathcal{N}(\mathbf{x}_i)| \in [1, 3]$.

For each observed particle \mathbf{x}_i , the new solution is generated according to the number of neighbors $|\mathcal{N}(\mathbf{x}_i)|$ in “generate_new_solution” function. The following modified equation is used in RDF-PSO for calculating the velocity:

$$\mathbf{v}_i^{(t+1)} = w \cdot \mathbf{v}_i^{(t)} + c_1 r_1^{(t)} (\mathbf{x}_{\text{best}}^{(t)} - \mathbf{x}_i^{(t)}) + \left[\frac{\sum_{j=1}^{|\mathcal{N}(\mathbf{x}_i)|} c_{j+1} r_{j+1}^{(t)} (\mathbf{p}_j^{(t)} - \mathbf{x}_i^{(t)})}{|\mathcal{N}(\mathbf{x}_i)|} \right], \quad (4)$$

where, r_1 , and r_2 are the real numbers randomly drawn from the interval $[0, 1]$, c_1 and c_2 denote the constriction

```

(1) <rdf:RDF>
(2)   <rdf:Description rdf:about="http://www.example.org/population">
(3)     <ns1:member_of rdf:resource="http://www.example.org/particle1"/>
(4)     <ns1:member_of rdf:resource="http://www.example.org/particle2"/>
(5)     ...
(6)     <ns1:member_of rdf:resource="http://www.example.org/particle $n$ "/>
(7)   </rdf:Description>
(8)   <rdf:Description rdf:about="http://www.example.org/particle1">
(9)     <ns1:is_neighbor_of rdf:Href="http://www.example.org/particle2"/>
(10)    <ns1:is_neighbor_of rdf:Href="http://www.example.org/particle5"/>
(11)    <ns1:id>1</ns1:id>
(12)  </rdf:Description>
(13)  ...
(14) </rdf:RDF>
    
```

ALGORITHM 5: Pseudocode of the PSO population in RDF.

coefficients, $\mathbf{p}_j^{(t)} = \{\mathbf{x}_k \mid \mathbf{x}_k \text{ is_neighbor_of } \mathbf{x}_i \wedge 1 \leq k \leq N\}$, $j \in [1, |\mathcal{N}(\mathbf{x}_i)|]$, and $\sum_{j=1}^{|\mathcal{N}(\mathbf{x}_i)|} c_{j+1} = 1$. Thus, it is expected that the movement of more crowded neighborhood depends on more neighbors. Furthermore, the term between square parenthesis ensures that the proportion of each neighbor as determined by constriction coefficients $\{c_2, c_3, c_4\}$ never exceeded the value of one.

3.1. *Representation of a Distributed Population.* The rapid growth of the Internet means that new kinds of application architectures have been emerged. The Internet applications are suitable to exploit enormous power of the computers connected to this huge network. Typically, these applications search for data distributed on many servers. These data need to be accessed easily, securely, and efficiently.

This paper proposes the first steps of developing the distributed population model within the PSO algorithm. In line with this, the RDF tool is applied that introduces a description of relations between particles in the population. These relations make us possible to manipulate population members on a higher abstraction level. At the moment, only the relation “is_neighbor_of” is implemented that determines the neighborhood of a specific particle in the population.

For this purpose, RDF is devoted for defining the various resources on different Internet servers. In our case, each particle in the population represents the resource that is defined with corresponding property-type relation (e.g., “is_neighbor_of”) and attributes. The RDF graph of the distributed population is illustrated in Figure 3.

The definition of a distributed population in RDF is presented in Algorithm 5, from which it can be seen that two kinds of attributes are encountered in this definition, that is, the references to neighbors of specific particle and its sequence number. Some details are omitted in this algorithm because of the space limitation of this paper. The missing parts of code are denoted by punctuation marks.

3.2. *Accessing the Distributed Population.* The distributed population in RDF can be accessed using the SPARQL query

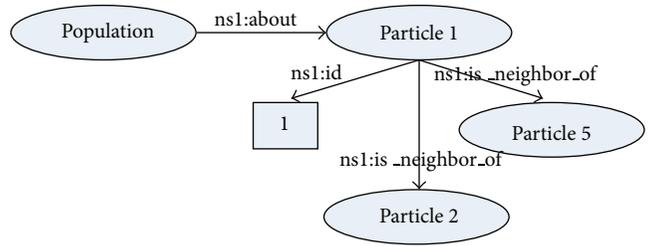


FIGURE 3: This PSO distributed population model contains the definitions of resources *population* and *particle*. Thus, each *particle* can relate to one or more neighbors and has an identification number. The former represents a reference to other *particles* in a swarm, while the latter is an atomic value.

```

(1) SELECT ?particle
(2) WHERE {
(3) <http://www.example.org/particle4>
(4) <http://www.example.org/is_neighbor_of>
(5) ?particle
(6) }
    
```

ALGORITHM 6: SPARQL query that returns particles in the neighborhood of particle4.

language, whose syntax is similar to the standard SQL syntax. An example of SPARQL query for returning the neighborhood of fourth particle is represented in Algorithm 6. Note that the SPARQL query from the mentioned algorithm will return all attributes that are related to the “resource4” with the relation “is_neighbor_of”

3.3. *Implementation Details.* The proposed RDF-PSO algorithm was implemented in Python programming language

TABLE 1: Definitions of benchmark functions.

f	Function name	Definition
f_1	Griewangk's function	$f_1(\mathbf{x}) = -\prod_{i=1}^n \cos\left(\frac{x_i}{\sqrt{i}}\right) + \sum_{i=1}^n \frac{x_i^2}{4000} + 1$
f_2	Rastrigin's function	$f_2(\mathbf{x}) = n * 10 + \sum_{i=1}^n (x_i^2 - 10 \cos(2\pi x_i))$
f_3	Rosenbrock's function	$f_3(\mathbf{x}) = \sum_{i=1}^{n-1} 100 (x_{i+1} - x_i^2)^2 + (x_i - 1)^2$
f_4	Ackley's function	$f_4(\mathbf{x}) = \sum_{i=1}^{n-1} \left(20 + e^{-0.2} e^{-0.2 \sqrt{0.5(x_{i+1}^2 + x_i^2)}} - e^{0.5(\cos(2\pi x_{i+1}) + \cos(2\pi x_i))} \right)$
f_5	Schwefel's function	$f_5(\mathbf{x}) = 418.9829 * D - \sum_{i=1}^D x_i \sin\left(\sqrt{ x_i }\right)$
f_6	De Jong's sphere function	$f_6(\mathbf{x}) = \sum_{i=1}^D x_i^2$
f_7	Easom's function	$f_7(\mathbf{x}) = -(-1)^D \left(\prod_{i=1}^D \cos^2(x_i) \right) \exp\left[-\sum_{i=1}^D (x_i - \pi)^2\right]$
f_8	Michalewicz's function	$f_8(\mathbf{x}) = -\sum_{i=1}^D \sin(x_i) \left[\sin\left(\frac{iA \cdot x_i^2}{\pi}\right) \right]^{2 \cdot 10}$
f_9	Xin-She Yang's function	$f_9(\mathbf{x}) = \left(\sum_{i=1}^D x_i \right) \exp\left[-\sum_{i=1}^D \sin(x_i^2)\right]$
f_{10}	Zakharov's function	$f_{10}(\mathbf{x}) = \sum_{i=1}^D x_i^2 + \left(\frac{1}{2} \sum_{i=1}^D i x_i \right)^2 + \left(\frac{1}{2} \sum_{i=1}^D i x_i \right)^4$

and executed on Linux operating system. Additionally, the following libraries were used:

- (i) *rdflib* which is a python library for working with RDF [29],
- (ii) *NumPy* that is the fundamental package for scientific computing with Python [30]
- (iii) *matplotlib* that is a python 2D plotting library [31].

The decision for using Python has been taken because there already existed a lot of the PSO implementation. Furthermore, the RDF and SPARQL semantic tools are also supported in this language and ultimately, programming in Python is easy.

4. Experiments and Results

The goal of our experimental work was to show that the semantic web tools, that is, RDF and SPARQL can be useful for the optimization purposes as well. Moreover, we want to show that using the variable neighborhood in RDF-PSO can also improve the results of the original PSO.

In line with this, the RDF-PSO algorithm was applied to the optimization of ten benchmark functions taken from literature. The function optimization belongs to a class of continuous optimization problems, where the objective function $f(\mathbf{x})$ is given and $\mathbf{x} = \{x_1, \dots, x_D\}$ is a vector of D design variables in a decision space S . Each design variable $x_i \in [Lb_i, Ub_i]$ is limited by its lower $Lb_i \in \mathbb{R}$ and upper

$Ub_i \in \mathbb{R}$ bounds. The task of optimization is to find the minimum of the objective functions.

In the remainder of this section, the benchmark suite is described; then, the experimental setup is presented and finally, the results of experiments are illustrated in detail.

4.1. Test Suite. The test suite consisted of ten functions, which were selected from the literature. However, the primary reference is the paper by Yang [32] that proposed a set of optimization functions suitable for testing the newly developed algorithms. The definitions of the benchmark functions are represented in Table 1, while their properties are illustrated in Table 2.

Table 2 consists of five columns that contain the function identifications (tag f), their global optimum (tag f^*), the values of optimal design variables (tag x^*), the lower and upper bounds of the design variables (tag *Bound*), and their characteristics (tag *Characteristics*). The lower and upper bounds of the design variables determine intervals that limit the size of the search space. The wider is the interval, the wider is the search space. Note that the intervals were selected, so that the search space was wider than those proposed in the standard literature. The functions within the benchmark suite can be divided into *unimodal* and *multimodal*. The multimodal functions have two or more local optima. Typically, the multimodal functions are more difficult to solve. The most complex functions are those that have an exponential number of local optima randomly distributed within the search space.

TABLE 2: Properties of benchmark functions.

f	f^*	x^*	Bounds	Characteristics
f_1	0.0000	(0, 0, ..., 0)	[-600, 600]	Highly multi-modal
f_2	0.0000	(0, 0, ..., 0)	[-15, 15]	Highly multi-modal
f_3	0.0000	(1, 1, ..., 1)	[-15, 15]	Multiple local optima
f_4	0.0000	(0, 0, ..., 0)	[-32.768, 32.768]	Highly multi-modal
f_5	0.0000	(0, 0, ..., 0)	[-500, 500]	Highly multi-modal
f_6	0.0000	(0, 0, ..., 0)	[-600, 600]	Uni-modal, convex
f_7	-1.0000	(π , π , ..., π)	$[-2\pi, 2\pi]$	Multiple local optima
f_8	-1.8013 ¹	(2.20319, 1.57049) ¹	[0, π]	Multiple local optima
f_9	0.0000	(0, 0, ..., 0)	$[-2\pi, 2\pi]$	Multiple local optima
f_{10}	0.0000	(0, 0, ..., 0)	[-5, 10]	Uni-modal

¹These values are valid for dimensions $D = 2$.

4.2. Experimental Setup. This experimental study compares the results of the RDF-PSO using different kind of distributed populations within the original PSO algorithm. All PSO algorithms used the following setup. The parameter w was randomly drawn from the interval [0.4, 0.9], while the constriction coefficients were set as $c_1 = c_2 = 1.0$. As a termination condition, the number of fitness function evaluations was considered. It was set to $FES = 1000 \cdot D$, where D denotes dimension of the problem. In this study, three different dimensions of functions were applied; that is, $D = 10$, $D = 30$, and $D = 50$. However, the population size is a crucial parameter for all population-based algorithms that have a great influence on their performance. In line with this, extensive experiments had been run in order to determine the most appropriate setting of this parameter by all algorithms in the test. As a result, the most appropriate setting of this parameter $N = 100$ was considered for the experiments. Parameters, like the termination condition, dimensions of the observed functions, and the population size were also used by the other algorithms in experiments.

The PSO algorithms are stochastic in nature. Therefore, statistical measures, like minimum, maximum, average, standard deviation, and median, were accumulated after 25 runs of the algorithms in order to fairly estimate the quality of solutions.

4.3. Results. The comparative study was conducted in which we would like to show, firstly, that the semantic web tools can be successfully applied to the optimization purposes as well and, secondly, that using the distributed population affects the results of the original PSO algorithm. In the remainder of this section, a detailed analysis of RDF-PSO algorithms is presented.

4.3.1. Analysis of the RDF-PSO Algorithms. In this experiment, the characteristics of the RDF-PSO algorithm were analyzed. In line with this, the RDF-PSO with neighborhood size of one (RDF1), the RDF-PSO with neighborhood size of two (RDF2), and the RDF-PSO with neighborhood size of tree (RDF3) were compared with the original PSO algorithm

(PSO) by optimizing ten benchmark functions with dimensions $D = 10$, $D = 30$, and $D = 50$. The obtained results by the optimization of functions with dimension $D = 30$ are aggregated in Table 3. Note that the best average values are for each function presented bold in the table.

From Table 3, it can be seen that the best average values were obtained by the RDF-1 algorithm eight times, that is, by $f_1 - f_4$, f_6 , f_8 , and f_{10} . The best results were two times observed also by the original PSO algorithm, that is, f_5 and f_9 . On average, the results of the other two RDF-PSO algorithms, that are, RDF-2 and RDF-3, were better than the results of the original PSO algorithm.

In order to statistically estimate the quality of solution, the Friedman nonparametric test was conducted. Each algorithm enters this test with five statistical measures for each of observed functions. As a result, each statistical classifier (i.e., various algorithms) consists of $5 \cdot 10 = 50$ different variables. The Friedman test [33, 34] compares the average ranks of the algorithms. The closer the rank to one, the better is the algorithm in this application. A null hypothesis states that two algorithms are equivalent and, therefore, their ranks should be equal. If the null hypothesis is rejected, that is, the performance of the algorithms is statistically different, the Bonferroni-Dunn test [35] is performed that calculates the critical difference between the average ranks of those two algorithms. When the statistical difference is higher than the critical difference, the algorithms are significantly different. The equation for the calculation of critical difference can be found in [35].

Friedman tests were performed using the significance level 0.05. The results of the Friedman nonparametric test are presented in Figure 4 where the three diagrams show the ranks and confidence intervals (critical differences) for the algorithms under consideration. The diagrams are organized according to the dimensions of functions. Two algorithms are significantly different if their intervals do not overlap.

The first diagram in Figure 4 shows that the RDF-1 algorithm significantly outperforms the RDF-3 algorithm. Interestingly, the results of the original PSO are also better than the results of the RDF-2 and RDF-3 algorithm. The situation is changed in the second (by $D = 30$) and third diagram (by $D = 50$), where RDF-3 improves the results

TABLE 3: Comparing the results of different PSO algorithms ($D = 30$).

Alg.	Meas.	f_1	f_2	f_3	f_4	f_5
PSO	Best	$7.40E - 001$	$5.83E + 002$	$7.08E + 004$	$2.00E + 001$	$1.55E + 003$
	Worst	$1.41E + 000$	$5.65E + 003$	$1.10E + 007$	$2.06E + 001$	$9.77E + 003$
	Mean	$1.06E + 000$	$1.44E + 003$	$3.22E + 006$	$2.03E + 001$	$5.16E + 003$
	StDev	$1.06E + 000$	$1.13E + 003$	$2.04E + 006$	$2.04E + 001$	$7.44E + 003$
	Mean	$1.37E - 001$	$1.02E + 003$	$3.14E + 006$	$1.90E - 001$	$5.97E + 003$
RDF-1	Best	$8.87E - 014$	$4.55E - 010$	$2.85E + 001$	$5.31E - 005$	$8.77E + 003$
	Worst	$3.33E - 010$	$1.95E - 006$	$2.88E + 001$	$9.64E - 003$	$1.05E + 004$
	Mean	$3.40E - 011$	$2.18E - 007$	$2.87E + 001$	$2.61E - 003$	$9.88E + 003$
	StDev	$1.27E - 011$	$4.81E - 008$	$2.87E + 001$	$1.53E - 003$	$9.95E + 003$
	Median	$6.91E - 011$	$4.90E - 007$	$5.85E - 002$	$2.68E - 003$	$3.99E + 002$
RDF-2	Best	$1.10E - 005$	$3.37E + 000$	$3.95E + 001$	$2.78E - 001$	$8.77E + 003$
	Worst	$2.42E - 001$	$7.03E + 001$	$3.42E + 002$	$3.64E + 000$	$1.04E + 004$
	Mean	$6.03E - 002$	$3.08E + 001$	$1.64E + 002$	$1.95E + 000$	$9.73E + 003$
	StDev	$4.12E - 002$	$2.52E + 001$	$1.36E + 002$	$1.72E + 000$	$9.65E + 003$
	Mean	$5.80E - 002$	$2.12E + 001$	$9.62E + 001$	$1.04E + 000$	$4.43E + 002$
RDF-3	Best	$3.07E - 005$	$1.85E + 001$	$5.49E + 001$	$9.11E - 001$	$8.81E + 003$
	Worst	$2.52E - 001$	$1.56E + 002$	$4.03E + 002$	$4.56E + 000$	$1.03E + 004$
	Mean	$8.94E - 002$	$7.75E + 001$	$1.77E + 002$	$2.63E + 000$	$9.70E + 003$
	StDev	$7.62E - 002$	$7.60E + 001$	$1.64E + 002$	$2.66E + 000$	$9.78E + 003$
	Mean	$5.51E - 002$	$3.46E + 001$	$8.38E + 001$	$1.11E + 000$	$4.19E + 002$
Evals	Meas.	f_6	f_7	f_8	f_9	f_{10}
PSO	Best	$1.10E - 003$	$0.00E + 000$	$-1.77E + 001$	$6.94E - 012$	$5.39E - 001$
	Worst	$3.75E - 001$	$0.00E + 000$	$-8.45E + 000$	$5.38E - 011$	$2.42E + 001$
	Mean	$1.13E - 001$	$0.00E + 000$	$-1.42E + 001$	$1.30E - 011$	$5.26E + 000$
	StDev	$6.89E - 002$	$0.00E + 000$	$-1.41E + 001$	$8.60E - 012$	$3.36E + 000$
	Mean	$1.24E - 001$	$0.00E + 000$	$1.87E + 000$	$1.05E - 011$	$5.35E + 000$
RDF-1	Best	$7.80E - 014$	$0.00E + 000$	$-6.45E + 000$	$1.83E - 007$	$2.47E - 012$
	Worst	$3.55E - 009$	$0.00E + 000$	$-3.85E + 000$	$1.39E - 005$	$5.70E - 007$
	Mean	$3.83E - 010$	$0.00E + 000$	$-4.81E + 000$	$3.68E - 006$	$2.59E - 008$
	StDev	$2.97E - 011$	$0.00E + 000$	$-4.74E + 000$	$3.35E - 006$	$1.34E - 009$
	Mean	$8.15E - 010$	$0.00E + 000$	$6.12E - 001$	$3.29E - 006$	$1.13E - 007$
RDF-2	Best	$6.63E - 004$	$0.00E + 000$	$-6.47E + 000$	$3.57E - 009$	$2.55E - 001$
	Worst	$2.03E + 000$	$0.00E + 000$	$-4.06E + 000$	$1.73E - 007$	$1.99E + 002$
	Mean	$4.36E - 001$	$0.00E + 000$	$-4.93E + 000$	$6.03E - 008$	$2.54E + 001$
	StDev	$1.48E - 001$	$0.00E + 000$	$-4.81E + 000$	$3.83E - 008$	$5.92E + 000$
	Mean	$5.58E - 001$	$0.00E + 000$	$5.92E - 001$	$4.64E - 008$	$4.90E + 001$
RDF-3	Best	$5.21E - 003$	$0.00E + 000$	$-6.27E + 000$	$2.42E - 009$	$5.94E + 000$
	Worst	$2.09E + 000$	$0.00E + 000$	$-4.09E + 000$	$1.31E - 007$	$4.53E + 002$
	Mean	$8.47E - 001$	$0.00E + 000$	$-5.03E + 000$	$3.82E - 008$	$1.09E + 002$
	StDev	$8.14E - 001$	$0.00E + 000$	$-5.05E + 000$	$3.35E - 008$	$6.65E + 001$
	Mean	$5.73E - 001$	$0.00E + 000$	$5.92E - 001$	$3.08E - 008$	$1.15E + 002$

of the RDF-3 and the original PSO, but not the RDF-2 algorithm. Additionally, the RDF-2 is significantly better than the original PSO also by $D = 50$.

In summary, the RDF-1 exposes the best results between all the other algorithms in tests by all observed dimensions of functions. On the other hand, the original PSO algorithm is only comparable with the modified PSO algorithms by optimizing the low dimensional functions ($D = 10$). The question why the RDF-PSO with neighborhood size of one

outperformed the other RDF-PSO algorithms remains open for the future work. At this moment, it seems that here the primary role plays the constriction coefficients that determine an influence of specific neighbors.

5. Conclusion

The aim of this paper was twofold. First is to prove that the semantic web tools, like RDF and SPARQL, can also

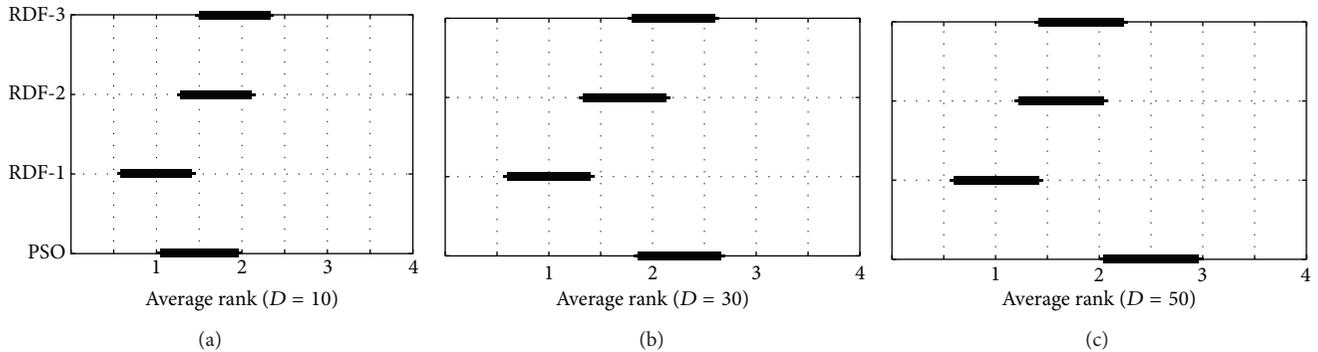


FIGURE 4: Results of the Friedman nonparametric test.

be used for the optimization purposes. Second is to show that the results of the modified RDF-PSO using the variable neighborhood are comparable with the results of the original PSO algorithm.

In line with the first hypothesis, a distributed population model was developed within the PSO algorithm that is suitable for describing the variable neighborhood of particles in the population. Furthermore, moving particles across the search space depends on all the particles in the neighborhood in place of the local best solutions as proposed in the original PSO algorithm.

In order to confirm the second hypothesis, the benchmark suite of ten well-known functions from the literature was defined. The results of extensive experiments by optimization of benchmark functions showed that the optimal neighborhood size within the RDF-PSO algorithm is one (RDF1). This variant of the RDF-PSO also outperformed the original PSO algorithm.

The distributed population model extends the concept of population in SI. This means that the population is no longer a passive data structure for storing particles. Not only can the particles now be distributed, but also some relations can be placed between the population members. In this proof of concept, only one relation was defined, that is, "is_neighbor_of." Additionally, not the whole definition of the distributed population was put onto Internet at this moment. Although we are at the beginning of the path of how to make an intelligent particle in swarm intelligence algorithms, the preliminary results are encouraging and future researches would investigate this idea of distributed population models in greater detail.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

References

- [1] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.
- [2] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, New York, NY, USA, 2009.
- [3] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, Springer, Berlin, Germany, 2003.
- [4] C. Blum and D. Merkle, *Swarm Intelligence: Introduction and Applications*, Springer, Berlin, Germany, 2008.
- [5] C. Darwin, *The Origin of Species*, John Murray, London, UK, 1859.
- [6] W. Paszkowicz, "Genetic algorithms, a nature-inspired tool: survey of applications in materials science and related fields," *Materials and Manufacturing Processes*, vol. 24, no. 2, pp. 174–197, 2009.
- [7] D. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Massachusetts, Mass, USA, 1996.
- [8] J. Koza, *Genetic Programming 2—Automatic Discovery of Reusable Programs*, The MIT Press, Cambridge, Mass, USA, 1994.
- [9] T. Bäck, *Evolutionary Algorithms in Theory and Practice—Evolution Strategies, Evolutionary Programming, Genetic Algorithms*, University Press, Oxford, UK, 1996.
- [10] L. Fogel, A. Owens, and M. Walsh, *Artificial Intelligence through Simulated Evolution*, John Wiley & Sons, New York, NY, USA, 1996.
- [11] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
- [12] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Žumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.
- [13] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.
- [14] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, vol. 4, pp. 1942–1948, IEEE, Perth, Australia, December 1995.
- [15] I. Fister, I. Fister Jr., X. -S. Yang, and J. Brest, "A comprehensive review of firefly algorithms," *Swarm and Evolutionary Computation*, vol. 13, pp. 34–46, 2013.

- [16] I. Fister, X. -S. Yang, J. Brest, and I. Fister Jr., "Modified firefly algorithm using quaternion representation," *Expert Systems with Applications*, vol. 40, no. 18, pp. 7220–7230, 2013.
- [17] I. Fister Jr., D. Fister, and I. Fister, "A comprehensive review of cuckoo search: variants and hybrids," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 4, no. 4, pp. 387–409, 2013.
- [18] X.-S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, pp. 65–74, Springer, New York, NY, USA, 2010.
- [19] I. Fister Jr., D. Fister, and I. Fister, "Differential evolution strategies with random forest regression in the bat algorithm," in *Proceeding of the 15th Annual Conference Companion on Genetic and Evolutionary Computation*, pp. 1703–1706, ACM, 2013.
- [20] P. N. Suganthan, "Particle swarm optimiser with neighbourhood operator," in *Proceedings of the Congress on Evolutionary Computation (CEC '99)*, vol. 3, IEEE, Washington, Wash, USA, July 1999.
- [21] H. Liu, A. Abraham, O. Choi, and S. H. Moon, "Variable neighborhood particle swarm optimization for multi-objective flexible job-shop scheduling problems," in *Simulated Evolution and Learning*, vol. 4247 of *Lecture Notes in Computer Science*, pp. 197–204, Springer, New York, NY, USA, 2006.
- [22] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of Machine Learning*, pp. 760–766, Springer, New York, NY, USA, 2010.
- [23] N. Chakraborti, R. Jayakanth, S. Das, E. D. Çalişir, and Ş. Erkoç, "Evolutionary and genetic algorithms applied to Li+-C system: calculations using differential evolution and particle swarm algorithm," *Journal of Phase Equilibria and Diffusion*, vol. 28, no. 2, pp. 140–149, 2007.
- [24] R. C. Eberhart and Y. Shi, "Particle swarm optimization: developments, applications and resources," in *Proceedings of the Congress on Evolutionary Computation*, vol. 1, pp. 81–86, IEEE, May 2001.
- [25] Y. Shi and R. Eberhart, "Modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, IEEE, May 1998.
- [26] E. Miller, "An introduction to the resource description framework," *D-Lib Magazine*, vol. 4, no. 5, pp. 14–25, 1998.
- [27] D. Allemang and J. Hendler, *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL*, Morgan Kaufmann, Amsterdam, The Netherlands, 2nd edition, 2011.
- [28] S. Harris and A. Seaborne, *Sparql 1.1 Query Language*, 2013.
- [29] rdflib: A python library for working with RDF, 2013, <http://code.google.com/p/rdflib/>.
- [30] Numpy, 2013, <http://www.numpy.org/>.
- [31] Matplotlib, 2013, <http://matplotlib.org/>.
- [32] X.-S. Yang, "Appendix A: test problems in optimization," in *Engineering Optimization*, X.-S. Yang, Ed., pp. 261–266, John Wiley & Sons, Hoboken, NJ, USA, 2010.
- [33] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, no. 200, pp. 675–701, 1937.
- [34] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *The Annals of Mathematical Statistics*, vol. 11, no. 1, pp. 86–92, 1940.
- [35] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *Journal of Machine Learning Research*, vol. 7, pp. 1–30, 2006.

Research Article

Support Vector Machine Based on Adaptive Acceleration Particle Swarm Optimization

Mohammed Hasan Abdulameer,^{1,2} Siti Norul Huda Sheikh Abdullah,¹
and Zulaiha Ali Othman³

¹ Pattern Recognition Research Group, Centre for Artificial Intelligence Technology, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 Bandar Baru Bangi, Malaysia

² Department of Computer Science, Faculty of Education for Women, University of Kufa, Iraq

³ Data Mining and Optimization Group, Centre for Artificial Intelligence Technology, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, 43600 Bandar Baru Bangi, Malaysia

Correspondence should be addressed to Mohammed Hasan Abdulameer; moh_ph135@yahoo.com

Received 1 December 2013; Accepted 20 February 2014; Published 25 March 2014

Academic Editors: S.-F. Chien, T. O. Ting, and X.-S. Yang

Copyright © 2014 Mohammed Hasan Abdulameer et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Existing face recognition methods utilize particle swarm optimizer (PSO) and opposition based particle swarm optimizer (OPSO) to optimize the parameters of SVM. However, the utilization of random values in the velocity calculation decreases the performance of these techniques; that is, during the velocity computation, we normally use random values for the acceleration coefficients and this creates randomness in the solution. To address this problem, an adaptive acceleration particle swarm optimization (AAPSO) technique is proposed. To evaluate our proposed method, we employ both face and iris recognition based on AAPSO with SVM (AAPSO-SVM). In the face and iris recognition systems, performance is evaluated using two human face databases, YALE and CASIA, and the UBiris dataset. In this method, we initially perform feature extraction and then recognition on the extracted features. In the recognition process, the extracted features are used for SVM training and testing. During the training and testing, the SVM parameters are optimized with the AAPSO technique, and in AAPSO, the acceleration coefficients are computed using the particle fitness values. The parameters in SVM, which are optimized by AAPSO, perform efficiently for both face and iris recognition. A comparative analysis between our proposed AAPSO-SVM and the PSO-SVM technique is presented.

1. Introduction

Support vector machine (SVM) is a machine-learning method based on the structure risk minimization principle. SVM can find global optimum solutions for problems with small training samples, high dimensions, and nonlinearity. SVM has attracted much attention during the past decade as a modern machine-learning approach in several domains, such as pattern recognition, bioinformatics, and other nonlinear problems with small sample sizes. SVM has strong theoretical foundations and a good generalization capability. From the implementation point of view, training an SVM in classification is equivalent to solving a linearly constrained quadratic programming (QP) problem, which consumes large amounts

of memory and computation time when the number of samples increases. Another issue in SVM is that the selection of the training parameters impacts its performance. Some of the SVM-based methods are utilized with face applications [1–15]. Li, Lijuan & Weiguo have proposed a multi-class SVM for face recognition [2]. They utilized the generalized two-dimensional Fisher's linear discriminant (G-2DFLD) method for feature extraction and used multiclass support vector machines as the classifier for face recognition. They proposed a multiobjective uniform design (MOUD) search method as an SVM model selection tool and then applied an optimized SVM classifier to face recognition. In their proposed method, LDA has been used for feature extraction. However, LDA is computationally high and suffers from the so-called small size

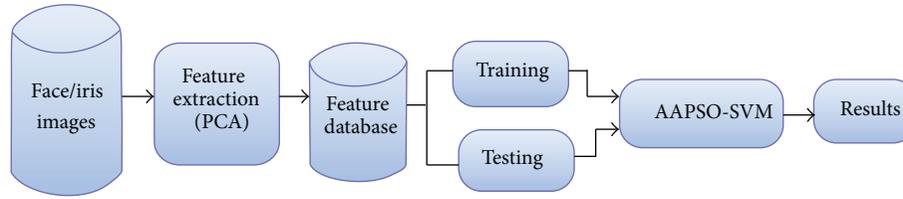


FIGURE 1: Structure of the proposed recognition technique based on AAPSO-SVM.

problem (SSS) problem [13]. Additionally, a new classification model based on SVM named as (SVM + NDA) has been proposed by Khan et al. [12]. In addition, they proposed a kernel extension of the model KSVM + KNDA to deal with nonlinear problems. However, it is obvious to notice that the SVM + NDA in linear case took more computational time than the LDA, NDA, HLDA, and SVM + LDA. On the other hand, the KSVM + KNDA required greater number of iterations than KNDA and KFD in nonlinear case.

Recently, existing face recognition methods utilize PSO and OPSO methods to optimize the parameters of SVM. Reference [3] presented a face recognition method based on support vector machine and particle swarm optimization (PSO-SVM). In PSO-SVM method, the parameters optimization problem in SVM is solved by particle swarm optimization. Nevertheless, this method lacks the initial phase of the PSO technique. In PSO, the populations are generated in a random manner. Due to this random process, the population results may also be in a random manner. Therefore, it is not certain that this method will produce a precise result when it is used with SVM. Later, and to avoid this drawback, a modified face recognition method based on opposition particle swarm optimization (OPSO) and SVM (OPSO-SVM) has been proposed by Hasan, Abdullah and Othman [13]. In OPSO-SVM, opposition particle swarm optimization (OPSO) [14] has been used instead of PSO to find the optimal parameters in SVM. In OPSO, the populations are generated in two ways: one is random population the same as the standard PSO technique and the other is opposition population, which is based on the random population values. The optimized parameters in SVM by OPSO efficiently perform the face recognition process. Accelerated PSO with SVM (APSO-SVM) has been introduced by Yang, Deb and Fong [15]. In APSO-SVM, APSO is used to find the best kernel parameters in SVM. Then, the kernel parameters are used to construct the support vector machines to solve the problem of interest. In APSO, the algorithm used the global best only and excluded the individual best and it does not use velocities or inertia parameter. Though, the PSO performance degrades by the utilization of random values in the velocity calculation and this will influence the parameter selection in SVM. In this paper, to address this problem, an adaptive acceleration particle swarm optimization (AAPSO) technique is proposed.

The rest of this paper is structured as follows. Section 2 describes the proposed model in relation to PSO and SVM. Section 3 explains the experimental results. Finally, we end our paper with conclusions and potential future studies in Section 4.

2. The Proposed Model

The standard PSO method has been utilized in many research works to obtain optimal problem solutions. To obtain a more accurate optimal result, the drawbacks which are present in the PSO method must be addressed by making modifications or enhancements to the PSO model. The major drawback of the PSO is the random value selection during new particle generation; that is, in the velocity computation, the acceleration coefficients are generated randomly. The random value selection in the velocity process means that the generated particles will also be random. Random populations do not produce more accurate results. Hence, to acquire a more accurate result and to reduce this PSO drawback, we propose an adaptive acceleration particle swarm optimization (AAPSO). To obtain more accurate classification results, the SVM parameters will be optimized by our AAPSO. The utilization of AAPSO in the SVM parameter optimization will reduce the PSO drawback and improve the classification result accuracy. In this research, our intent is to develop a face and iris recognition system for accurate recognition of face images from the databases. The proposed face recognition technique is performed in three phases, feature extraction by PCA, adaptive acceleration particle swarm optimization (AAPSO), and parameters selection for SVM with AAPSO. These three phases are performed repeatedly on the input database face images, and thus the face images are recognized more effectively. The three phases are discussed in Sections 2.1, 2.2, and 2.3. The basic structure of our proposed face recognition technique is shown in Figure 1.

2.1. Feature Extraction Using PCA. The purpose of the feature extraction is to extract the information that represents the face. Principal component analysis (PCA) is used for this purpose [9]. We apply PCA on the training and testing database face images and obtain the unique dimensional feature vectors.

2.2. The Proposed Adaptive Acceleration Particle Swarm Optimization (AAPSO). Particle swarm optimization (PSO) is a computational intelligence oriented, stochastic, population based global optimization technique proposed by Kennedy and Eberhart [4, 11]. It is inspired by the social behaviour of biological creatures, such as fishes and birds, which have the ability to group together to work as a whole to locate desirable positions in a certain area, for example, fish searching for a food source. This type of search behaviour is equivalent to

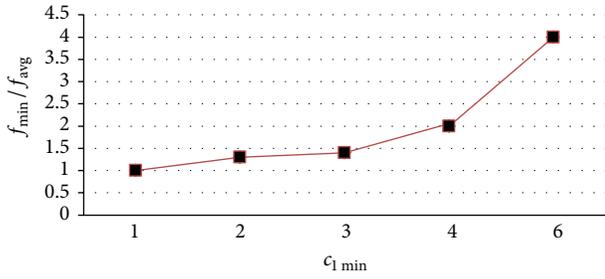


FIGURE 2: $c_{1 \min}$ behavior.

searching for solutions of equations in a real-valued search space [10]. PSO emulates the swarm behaviour of individuals who represent potential solutions in a D-dimensional search space. Particle i is often composed of four vectors: $X_i = (x_i^1, x_i^2, \dots, x_i^D)$, where x_i^D is its position in the d th dimension; $Pbest_i = (Pbest_i^1, Pbest_i^2, \dots, Pbest_i^D)$, where $Pbest_i^D$ is the best position in the d th dimension that particle i has found on its own; $V_i = (v_i^1, v_i^2, \dots, v_i^D)$, where v_i^D is the velocity in the d th dimension; and $gbest_i = (gbest_i^1, gbest_i^2, \dots, gbest_i^D)$, where $gbest_i^D$ is the global best position in the d th dimension that all particles have found. Particles in a swarm move through the search space as follows:

$$V_i^d = V_i^d + c_1 r_1 \cdot (Pbest_i^d - x_i^d) + c_2 r_2 \cdot (gbest_i^d - x_i^d), \quad (1)$$

$$x_i^d = x_i^d + \delta V_i^d, \quad (2)$$

where c_1 and c_2 are two constants, often with the value of 2.0, r_1 and r_2 are two independent random numbers uniformly generated in the range [0,1] at each updating iteration from $d = 1$ to D , V_i^d is the velocity of the i th particle, x_i^d is the current position of the particle i , $Pbest_i^d$ is the position of the best fitness value of the particle at the current iteration, and $gbest_i^d$ is the position of the particle with the best fitness value in the swarm. The random values of c_1 and c_2 in the velocity computation do not select the optimal SVM parameters so that the result of the recognition results will be random or inaccurate. Therefore, we have proposed an adaptive acceleration particle swarm optimization (AAPSO) method that selects the acceleration coefficients using particle fitness values. The AASPO method selects the optimal SVM parameters and formulates the SVM to provide a more accurate face recognition result. The AAPSO's acceleration coefficients are determined as follows:

$$nc_1 = \frac{2}{3} (c_{1 \max} - c_{1 \min}) \left(\frac{f_{\min}}{f_{\text{avg}}} + \frac{f_{\min}}{2f_{\max}} \right) + c_{1 \min}, \quad (3)$$

$$nc_2 = \frac{2}{3} (c_{2 \max} - c_{2 \min}) \left(\frac{f_{\min}}{f_{\text{avg}}} + \frac{f_{\min}}{2f_{\max}} \right) + c_{2 \min},$$

where $c_{1 \max}$ and $c_{1 \min}$ represent the minimum and maximum values of c_1 ; f_{\min} , f_{avg} , and f_{\max} are the particle minimum, average, and maximum fitness values of the entire population; and $c_{2 \max}$ and $c_{2 \min}$ represent the minimum and maximum

values of c_2 . By applying nc_1 and nc_2 in the velocity equation (1), the equation is updated as follows:

$$V_i^d = V_i^d + nc_1 r_1 \cdot (Pbest_i^d - x_i^d) + nc_2 r_2 \cdot (gbest_i^d - x_i^d). \quad (4)$$

Utilizing the above equations, the velocity function acceleration coefficients are computed in AAPSO. The evaluation of the coefficients using the AAPSO equations enables the SVM to provide more accurate results.

Example 1. Assume that the population pool has five particles/individuals, P_n (for simplicity), and their fitness values are as follows: $P_1 = 0.5$, $P_2 = 0.4$, $P_3 = 0.2$, $P_4 = 0.7$, $P_5 = 0.6$, and $f_{\min} = P_3$.

As a special case, f_{avg} is equal to f_{\min} (i.e., $f_{\min} = f_{\text{avg}}$) only if each member of the population has the same fitness value (i.e., $P_1 = P_2 = P_3 = P_4 = P_5$). Otherwise, $f_{\min} < f_{\text{avg}}$. Naturally, the probability of f_{\min}/f_{avg} will be between 0 and 1. Assume that the probability of $f_{\min}/f_{\text{avg}} = 1$ and the probability of $f_{\min}/f_{\max} = 1$; then the sum $((f_{\min}/f_{\text{avg}}) + (f_{\min}/2f_{\max}))$ is 3/2. Therefore, we only restrict the value within 1, and (3) and (4) are multiplied by 2/3. As a result, this calculation will produce a maximum value of 1 instead of 3/2. If f_{\min} , f_{avg} , and f_{\max} are equal, the acceleration will be constant at 1; that is, c_1 approaches $c_{1 \max}$. If c_1 increases, then the neighbourhood search space diverges using PSO. Otherwise, the neighbourhood search space converges. That is, c_1 is linearly proportional to its neighbourhood search space. The data in Figure 2 show that $c_{1 \min}$ varies linearly with f_{\min}/f_{avg} .

The acceleration constant, c_1 , accelerates the neighborhood search; that is, it determines the nearest or farthest one. If c_1 is small, the updated solutions will be near to the current solution, whereas if c_1 is high, the updated solution will be far from the current solution. In our adaptive method, instead of fixing a constant c_1 , we increase or decrease c_1 at every iteration. Therefore, the updated solutions may be far, near, or near-far.

Example 2. Assume a range of 10 and initial acceleration of 0; that is, the acceleration constant can vary within [0, 9]. For example, if a vehicle starts at 0 km/h and its maximum speed is 100 km/h, according to the traffic, it will accelerate between 0 and 100 km/h. Thus, the range of the speed is $100 - 0 = 100$ ($c_{1 \max} - c_{1 \min}$). In (3), we add $c_{1 \min}$ to calculate the speed from the initial condition. The conventional method fixes the acceleration constant at one value; thus, the velocity of the particles is updated without considering the relation to population fitness. By making this constant adaptive, we increase or decrease the velocity based on the population fitness.

2.3. Parameter Selection for SVM with AAPSO. To obtain more precise recognition, the SVM parameters are optimized using the AASPO method. The process is shown in Figure 3. The process of optimal parameter selection by AAPSO in SVM is shown as follows.

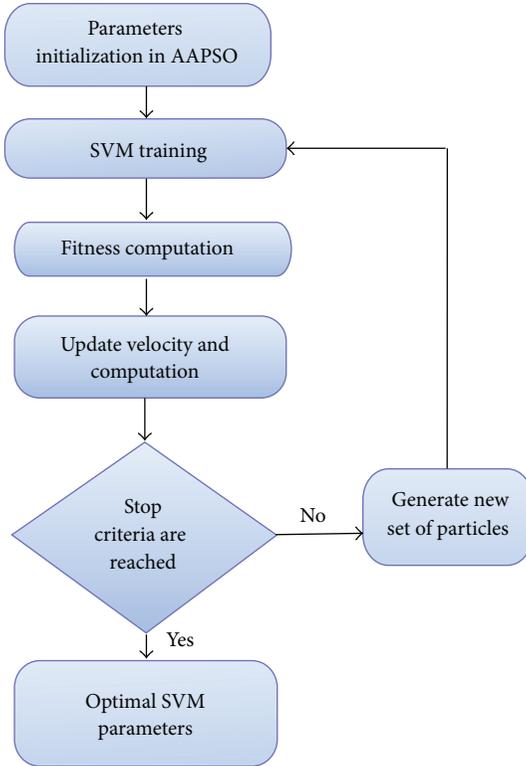


FIGURE 3: SVM parameter optimization using AAPSO.

Step 1. Initially, the particles are generated randomly within the interval $[x, y]$. The generated particles are composed of SVM parameters P_i . Then, the parameters of each particle are initiated, including position and velocity.

Step 2. The fitness value of every particle is calculated using (5). The particles that have the minimum fitness values are selected as the best particles as follows:

$$\min \frac{1}{2} \|P_i\|^2 + C \sum_{i=1}^N \xi_i \quad (5)$$

$$\text{Such that } \sum_{i=1}^N P_i x_i \geq \left(\frac{1 - \xi_i}{y_i} \right) - b, \quad (6)$$

$$i = 1, 2, \dots, N, \quad \xi_i \geq 0, \quad i = 1, 2, \dots, N,$$

where N is the size of the training dataset and C is a positive regularization constant or cost function, which defines the tradeoff between a large margin and a misclassification error.

Step 3. The \overline{Pbest}_i of each particle is updated and \overline{gbest}_i for the domain is updated. Based on these values, the velocity and position of every particle are updated using (4) and (2).

Step 4. Stop if the current optimization solution is good enough or if the stopping criterion is satisfied.

3. Experimental Results

We divide our experiment into two sections, face recognition evaluation and then iris recognition evaluation. The proposed face recognition technique is implemented using MATLAB (version 7.12) on an Intel core i5 processor that uses Windows 7 operating system and that has 3.20 GHz CPU speed and 4 GB of RAM. The performance of the proposed face recognition technique is evaluated using the face databases YALE [5] and CASIA [6]. The images are obtained from both databases and the feature extraction is computed using PCA, while recognition process is computed using the proposed AAPSO-SVM technique. Sample face images from the YALE and CASIA databases are shown in Figure 4.

The performance of our proposed method is analyzed in three evaluation steps: (i) evaluate the optimization using three standard functions, (ii) evaluate the classification results with the UBiris dataset [7], and (iii) evaluate the classification results with the face datasets. These three evaluation steps are explained below.

(i) Evaluate the Optimization Using Three Standard Functions. To accomplish the performance analysis, we performed 10 rounds of experiments using the AAPSO and PSO methods. Moreover, our proposed AAPSO method performance is evaluated using the standard functions [8]: sphere, Rosenbrock, and Rastrigin. These standard functions are computed using the following equations:

$$f_0(x) = \sum_{i=1}^n x_i^2,$$

$$f_1(x) = \sum_{i=1}^n \left(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right), \quad (7)$$

$$f_2(x) = \sum_{i=1}^n \left(x_i^2 - 10 \cos(2\pi x_i) + 10 \right).$$

The performance of our proposed AAPSO and the standard PSO methods on these standard functions, in terms of their fitness values for different numbers of iterations, is shown in Figure 5.

The results in Figure 5 show that our proposed AAPSO method yields more accurate particles that have lower fitness values than those generated by the PSO method. The results in Figures 5(a), 5(b), and 5(c) show that our proposed AAPSO method has obtained accurate fitness values for all of the three standard functions. The high performance result shows that our AAPSO method is able to determine the more accurate SVM parameters. Additionally, the performance of our proposed AAPSO method is compared to the performance of the PSO method using (5) in Figure 6.

In Figure 6, the proposed AAPSO technique obtained more accurate particles that have minimum fitness values smaller than those obtained with the PSO method. Therefore, our AAPSO technique has yielded more accurate SVM parameters. Figure 6(a) shows the fitness value performance for particles used on the YALE database face images. For all iterations, the fitness values of the particles of our proposed



FIGURE 4: Sample face images from (a) YALE [5] and (b) CASIA [6] databases.

TABLE 1: The accuracy of PSO and AAPSO based on SVM classification performance results for the UBiris dataset.

Experiment number	Accuracy (%) PSO	Accuracy (%) AAPSO
1	90	95
2	87	93
3	91	94
4	90	94
5	92	96
6	89	96
7	91	94
8	92	95
9	88	94
10	92	95
Average	90	95

AAPSO method are lower than those of the PSO method. However, when applied to the CASIA database, the PSO method particles have lower fitness values than the AAPSO particles for iterations 5, 8, and 9. In the remaining iterations, the AAPSO particles have the same or lower fitness values in comparison to the PSO particles.

(ii) *Evaluate Classification Results with the UBiris Dataset.* In this section, the classification performance is evaluated with the UBiris dataset. The classification accuracy results that are obtained for the UBiris dataset are given in Table 1. To analyze the classification performance, 10 experiments are conducted on the iris dataset. Sample iris dataset images are shown in Figure 7.

In 10 experiments, our proposed AAPSO method attained higher iris image classification accuracy than the standard PSO-SVM. The average classification accuracy is 95%.

(iii) *Evaluate Classification Results with Face Datasets.* In this section, the classification results are evaluated with two databases, Yale and CASIA. Moreover, the performance of our proposed technique is compared with the PSO-SVM method using based on accuracy rate. In the experiment, the face images are evaluated for four conditions: (i) same pose, same illumination, and different expression; (ii) same pose, same expression, and different illumination; (iii) same expression, same illumination, and different pose; and (iv) different expression, pose, or illumination. The accuracy results for the proposed AAPSO-SVM and the existing PSO-SVM face recognition techniques, applied to the YALE and CASIA databases, with the different conditions, are shown in Table 2 and Figure 8. The computational times for our proposed AAPSO and for the PSO methods are shown in Figure 9.

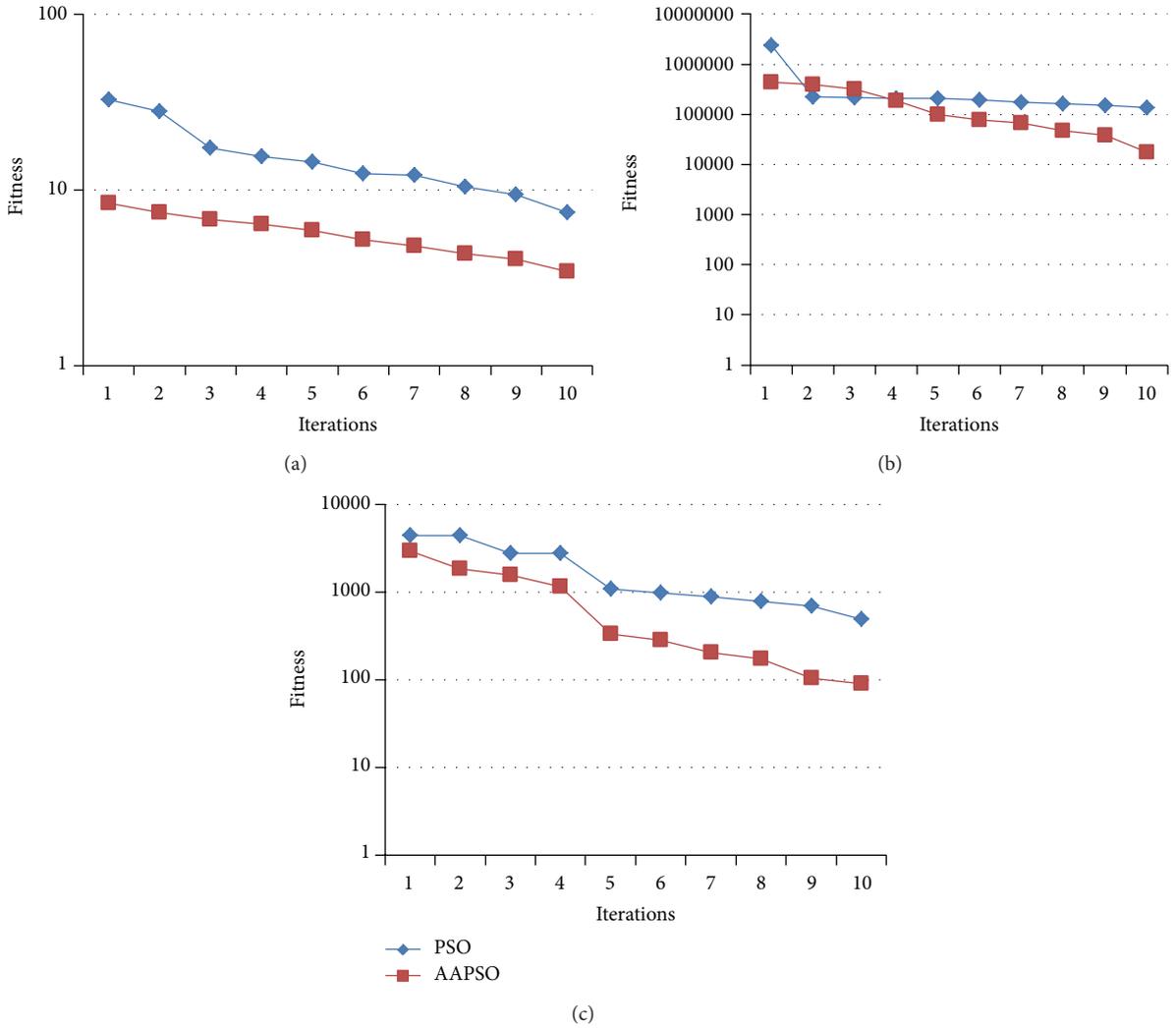


FIGURE 5: Performance of AAPSO and PSO methods with (a) sphere, (b) Rosenbrock, and (c) Rastrigin.

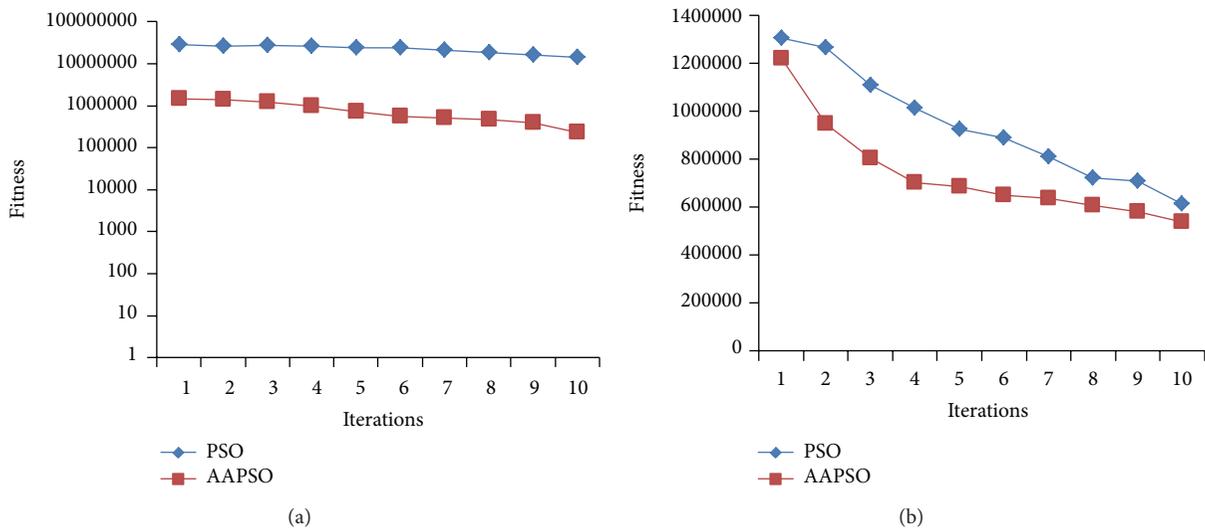


FIGURE 6: Performance of AAPSO and PSO methods from (a) YALE [5] database and (b) CASIA [6] database.

TABLE 2: Accuracy values of the proposed AAPSO-SVM and the PSO-SVM techniques with face datasets.

Conditions	Condition description	Yale dataset		CASIA dataset	
		Method	Accuracy	Method	Accuracy
1	Same pose, same illumination, and different expression	PSO-SVM	73	PSO-SVM	76
		AAPSO-SVM	86	AAPSO-SVM	85
2	Same pose, same expression, and different illumination	PSO-SVM	80	PSO-SVM	82
		AAPSO-SVM	95	AAPSO-SVM	96
3	Same expression, same illumination, and different pose	PSO-SVM	83	PSO-SVM	81
		AAPSO-SVM	94	AAPSO-SVM	93
4	Different expression, pose, or illumination	PSO-SVM	70	PSO-SVM	75
		AAPSO-SVM	82	AAPSO-SVM	84

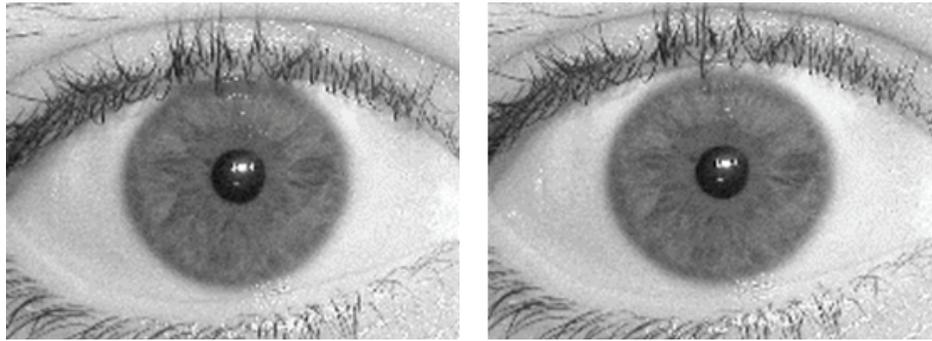


FIGURE 7: Sample images from the iris dataset.

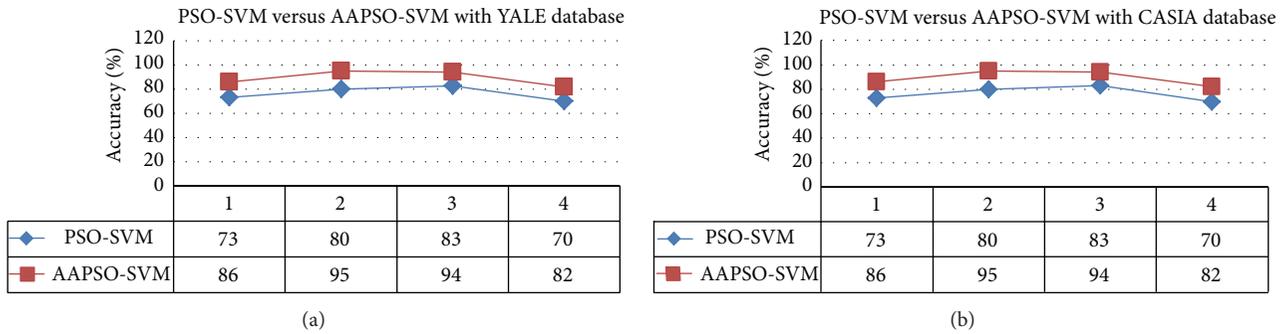


FIGURE 8: Performance accuracy of the AAPSO and PSO recognition methods from (a) YALE [5] database and (b) CASIA [6] database.

As shown in Figure 8, the recognition results of AAPSO-SVM have higher face recognition accuracy than the results of the PSO-SVM in all of the four conditions for both the Yale and CASIA databases. We measured the t -test values for the accuracy between AAPSO-SVM and PSO-SVM face recognition techniques. There was not much variation in the t -test results; however, the t -test result shows that the proposed AAPSO-SVM is statistically significant and that it outperforms the PSO-SVM with the t -test result $P < 0.05$; $P = 0.0265$ for the Yale database and $P = 0.0186$ for the CASIA database. Furthermore, Figure 9 shows the computational time used by our proposed AAPSO method and the conventional PSO to determine the optimal SVM parameters. It was shown that AAPSO optimized the SVM when compared with conventional PSO.

Figure 9 shows the computational time used by our proposed AAPSO method and the conventional PSO to determine the optimal SVM parameters. It was shown that AAPSO optimized the SVM, when compared with conventional PSO. The results in the table demonstrate the computational efficiency of AAPSO. That is, AAPSO uses less computational time to perform the optimization process compared with the conventional PSO. On average, AAPSO used 12% of the computational time to optimize the parameters and PSO used 21% of the computational time to optimize the parameters.

4. Conclusions

In this paper, we introduced AAPSO based on SVM to address the limitation of the standard PSO method that uses

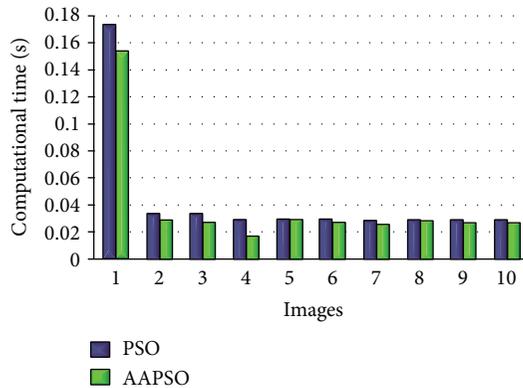


FIGURE 9: The computation time of AAPSO-SVM and PSO-SVM.

random selection of the coefficient factor for velocity. This may lead to performance instability. The optimized SVM, using the AAPSO technique, shows effective face recognition performance. Two human face databases, YALE and CASIA, were utilized to analyze the performance of our proposed AAPSO-SVM face recognition technique. The UBiris database was also used to illustrate the performance of our proposed technique in other domains. The performance and comparative analysis results show that our proposed AAPSO-SVM technique yields higher face recognition performance results than the PSO-SVM face recognition methods. In 10 experiments, our proposed AAPSO method attained a high iris image average classification accuracy of 95%, which is more than the standard PSO-SVM, which attained 90% in the same experiments. In addition, the SVM parameters for the YALE and CASIA databases are more optimal when obtained from AAPSO than from the conventional PSO. AAPSO also takes less computational time to perform the optimization process than the conventional PSO. On average, AAPSO used 12% of the computational time to optimize the parameters and PSO used 21% of the computational time to optimize the parameters. Hence, our proposed AAPSO with SVM technique is more robust and more precisely recognizes the face and iris images. Our proposed method can be made more robust if we test it with domains other than biometrics, such as bioinformatics and text categorization.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

This research is funded by the Ministry of Science, technology and Innovation (MOSTI) through ERGS/1/20 IISTG/UKM/2/48 (TK) under the title of 2D-3D Hybrid Face Matching via Fuzzy Bees Algorithm for Forensic Identification. The research also would like to thank CyberSecurity Malaysia and Royal Police of Malaysia's Forensics Lab for their support of the research.

References

- [1] C. Shiladitya, K. S. Jamuna, K. B. Dipak, and N. Mita, "Face recognition by generalized two-dimensional FLD method and multi-class support vector machines," *Applied Soft Computing*, vol. 11, no. 7, pp. 4282–4292, 2011.
- [2] W. Li, L. Lijuan, and G. Weiguo, "Multi-objective uniform design as a SVM model selection tool for face recognition," *Expert Systems with Applications*, vol. 38, no. 6, pp. 6689–6695, 2011.
- [3] W. Jin, J. Zhang, and X. Zhang, "Face recognition method based on support vector machine and particle swarm optimization," *Expert Systems with Applications*, vol. 38, no. 4, pp. 4390–4393, 2011.
- [4] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Piscataway, NJ, USA, December 1995.
- [5] P. N. Belhumeur, J. Hespanha, and D. Kriegman, "Eigenfaces vs. Fisherfaces: recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997.
- [6] Chinese Academy of Sciences, "CASIA face database," 2007, <http://biometrics.idealtest.org/>.
- [7] H. Proenc and L. Alexandre, "Ubiris iris image database," 2004, <http://iris.di.ubi.pt>.
- [8] K. James, R. C. Eberhart, and Y. Shi, *Swarm Intelligence*, Morgan Kaufmann, Burlington, Mass, USA, 1st edition, 2001.
- [9] S. Vaseghi and H. Jetelova, "Principal and independent component analysis in image processing," in *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, pp. 1–5, San Francisco, Calif, USA, 2006.
- [10] W. Chen, J. Zhang, S. H. Henry, Z. Wen-Liang, W. We-Gang, and S. Yu-Hui, "A novel set-based particle swarm optimization method for discrete optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 2, pp. 278–300, 2010.
- [11] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, pp. 39–43, IEEE Service Center Piscataway, Nagoya, Japan, October 1995.
- [12] N. M. Khan, R. Ksantini, I. S. Ahmad, and B. Boufama, "A novel SVM+NDA model for classification with an application to face recognition," *Pattern Recognition*, vol. 45, no. 1, pp. 66–79, 2012.
- [13] M. Hasan, S. H. S. Abdullah, and Z. A. Othman, "Face recognition based on opposition particle swarm optimization and support vector machine," in *Proceedings of the IEEE International Conference on Signal and Image Processing Applications (ICSIPA '13)*, pp. 417–424, Malaka, Malaysia, 2013.
- [14] H. Jabeen, Z. Jalil, and A. R. Baig, "Opposition based initialization in particle swarm optimization (O-PSO)," in *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers, Companion (GECCO '09)*, pp. 2047–2052, Material, Canada, July 2009.
- [15] X. S. Yang, S. Deb, and S. Fong, "Accelerated particle swarm optimization and support vector machine for business optimization and applications," in *Networked Digital Technologies*, vol. 136 of *Communications in Computer and Information Science*, pp. 53–66, Springer, Berlin, Germany, 2011.

Research Article

Novel Back Propagation Optimization by Cuckoo Search Algorithm

Jiao-hong Yi, Wei-hong Xu, and Yuan-tao Chen

School of Computer and Communication Engineering, Changsha University of Science and Technology, Changsha, Hunan 410014, China

Correspondence should be addressed to Jiao-hong Yi; yijiaohong@163.com

Received 1 January 2014; Accepted 16 February 2014; Published 20 March 2014

Academic Editors: S.-F. Chien, T. O. Ting, and X.-S. Yang

Copyright © 2014 Jiao-hong Yi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The traditional Back Propagation (BP) has some significant disadvantages, such as training too slowly, easiness to fall into local minima, and sensitivity of the initial weights and bias. In order to overcome these shortcomings, an improved BP network that is optimized by Cuckoo Search (CS), called CSBP, is proposed in this paper. In CSBP, CS is used to simultaneously optimize the initial weights and bias of BP network. Wine data is adopted to study the prediction performance of CSBP, and the proposed method is compared with the basic BP and the General Regression Neural Network (GRNN). Moreover, the parameter study of CSBP is conducted in order to make the CSBP implement in the best way.

1. Introduction

Though the traditional neural networks (such as BP) have been widely used in many areas, they have some inherent shortcomings. These disadvantages have become a major bottleneck that restricts their further development. In most cases, the gradient descent method is used in feedforward neural networks (FNN), which has the following main disadvantages.

- (1) Training slowly: many iterations are required in the gradient descent method in order to adjust weights and bias. Therefore, the training process takes long time.
- (2) It is easy to fall into local minimum, so that it cannot achieve the global minimum.
- (3) It is very sensitive to the choice of the initial weights and bias. Due to high influence on the performance for neural networks (NN), proper weights and bias must be carefully selected in order to obtain a more ideal network. If the selection of the weights and bias is improper, convergent speed of the algorithm will be very slow and the training process would take a long time.

Therefore, in order to enhance the performance of BP, many scholars are always striving for exploring a training algorithm that has a fast training speed, a global optimal solution, and a good generalization performance. Finding this training algorithm is also the main objective of the research in recent years.

Many metaheuristic methods have been proposed to solve optimization problems, such as the charged system search (CSS) [1], big bang-big crunch algorithm [2–5], harmony search (HS) [6–8], particle swarm optimization (PSO) [9–13], biogeography-based optimization (BBO) [14–18], firefly algorithm (FA) [19–23], differential evolution (DE) [24–27], krill herd (KH) [28–31], and bat algorithm (BA) [32–34].

In this paper, CS algorithm [35–37] that is a newly-developed metaheuristic method is used to optimize the weights and bias of BP. That is to say, CS is well capable of selecting the best initial weights and bias so as to construct the BP network instead of the randomly-generated weights and bias used in the basic BP. In order to prove the superiority of CSBP, it is used to solve the Italian wine classification problem. By comparing with the traditional BP and GRNN, this method has higher prediction accuracy and better generalization performance.

The remainder of this paper is organized as follows. The preliminaries including CS and BP are provided in Section 2. Section 3 represents the detailed BP optimized by CS. Then, in Section 4, a series of comparison experiments on Italian wine classification problem are conducted. The final section provides our concluding remarks and points out our future work orientation.

2. Preliminary

2.1. CS Algorithm. CS method [35, 36] is a novel metaheuristic swarm intelligence [38] optimization method for solving optimization problems. It is based on the behavior of some cuckoo species in combination with the Lévy flights. In the case of CS, how far a cuckoo can move forwards in a step can be determined by the Lévy flights.

In order to describe CS algorithm more easily, Yang and Deb [35] idealized the behavior of the cuckoo species into the following three rules:

- (1) for all the cuckoos in the population, every one lays only one egg at a time and randomly selects a nest in order to place this egg;
- (2) the population cannot change the eggs with the best fitness in order to make the whole population evolve forward all the time;
- (3) the host bird discovers the cuckoo eggs with a probability $p_a \in [0, 1]$. In this case, the cuckoo has no other choice and it has to build a fully new nest.

Based on the above hypothesis, the CS can be summarized as shown in Algorithm 1. We must point out that, for single objective problems, the cuckoos, eggs, and nests are equal to each other. So, we do not differentiate them in our works.

In order to make the balance of exploitation and exploration, CS uses a balanced combination of a local random walk and the global explorative random walk, controlled by a switching parameter p_a . The exploitation step can be represented as

$$x_i^{t+1} = x_i^t + \beta s \otimes H(p_a - \varepsilon) \otimes (x_j^t - x_k^t), \quad (1)$$

where x_j^t and x_k^t are two different randomly selected cuckoos, $H(u)$ is a Heaviside function, ε is a random number, and s is the step size. On the other hand, the exploration step is implemented by using Lévy flights as follows:

$$x_i^{t+1} = x_i^t + \beta L(s, \lambda), \quad (2)$$

where $L(s, \lambda) = (\lambda \Gamma(\lambda) \sin(\pi\lambda/2)/\pi)(1/s^{1+\lambda})$, ($s, s_0 > 0$), $\beta > 0$ is the scaling factor, and its value can be determined by the problem of interest. More information of CS can be referred to in [39–41].

2.2. BP Network. BP network was proposed by a team of scientists led by Rumelhart and McClelland in 1986 which is an error back propagation algorithm according to the former train multilayer feedforward network. It is one of the most widely used neural network models. BP network can learn

and remember a lot of input-output mapping model without prior mathematical equations that describe this mapping. The steepest descent method is used as the learning rules in order to adjust the weights and bias that can finally minimize the network error. In general, the topology of the BP network model includes input layer, hidden layer, and output layer. The number of layers and neurons in each hidden layer can be determined by the dimension of the input vector, and the output vector. In most cases, a single hidden layer is used in BP network.

BP network is a kind of supervised learning algorithm. Its main idea can be represented as follows. Firstly, training samples are input into the BP network, and then weights and bias are adjusted by using the error back propagation algorithm. This training process would minimize the error between the desired vector and the output vector. When the error is satisfied, weights and bias are remembered, which can be used to predict test samples. More information about BP can be referred to in [42].

3. CSBP

In the present work, the CS algorithm is used to optimize BP network. More specifically, the BP network is considered to be objective function (fitness function), and the weights and bias are optimized by the CS method in order to obtain the optimal weights and bias. The best weights and bias are well-suited to construct the BP that is significantly superior to the basic BP network.

The process of the BP network optimized by the CS is divided into three parts: determining BP network structure, obtaining the best weights and bias through CS, and predicting through neural network. The structure of BP network in the first part is determined based on the number of input and output parameters, and then the length of each cuckoo individual in CS is determined accordingly. In the second part, CS method is applied to optimize the weights and bias of the BP network. Each individual in the cuckoo population includes all the weights and bias in BP, and it is evaluated by the fitness function. The CS method implements initializing CS, determining fitness function, updating position operator, selecting operator, replacing operator, and eliminating operator in order to find the cuckoo individual with the best fitness. This optimization process is repeated until the satisfactory weights and bias are found. In the last part, the BP network with the optimal weights and bias is constructed and is trained to predict the output. Based on the above analyses, the flowchart of the CSBP algorithm can be shown in Figure 1.

In CSBP, CS is applied to optimize the initial weights and bias of BP network, so that the optimized BP network has better predicted output. The elements in CSBP include initializing CS, determining fitness function, updating position operator, selecting operator, replacing operator, and eliminating operator in order to find the cuckoo individual with the best fitness. The detailed steps of the CS algorithm (see Figure 1) are as follows.

```

Begin
  Step 1. Initialization. Set the generation counter  $G = 1$  and the discovery rate  $p_a$ ; initialize the population  $P$  of  $n$  host nests randomly.
  Step 2. While  $G < \text{MaxGeneration}$  do
    Sort all the cuckoos.
    Randomly select a cuckoo  $i$  and implement Lévy flights to replace its solution.
    Evaluate its fitness  $F_i$ .
    Randomly choose another nest  $j$ .
    if ( $F_i < F_j$ )
      Replace  $j$  by the new solution.
    end if
    Randomly generate a fraction ( $p_a$ ) of new nests and replace the worse nests.
    Keep the best nests.
    Sort the population and find the best cuckoo for this generation.
    Pass the current best to the next generation.
     $G = G + 1$ .
  Step 3. end while
End.
  
```

ALGORITHM 1: Cuckoo search.

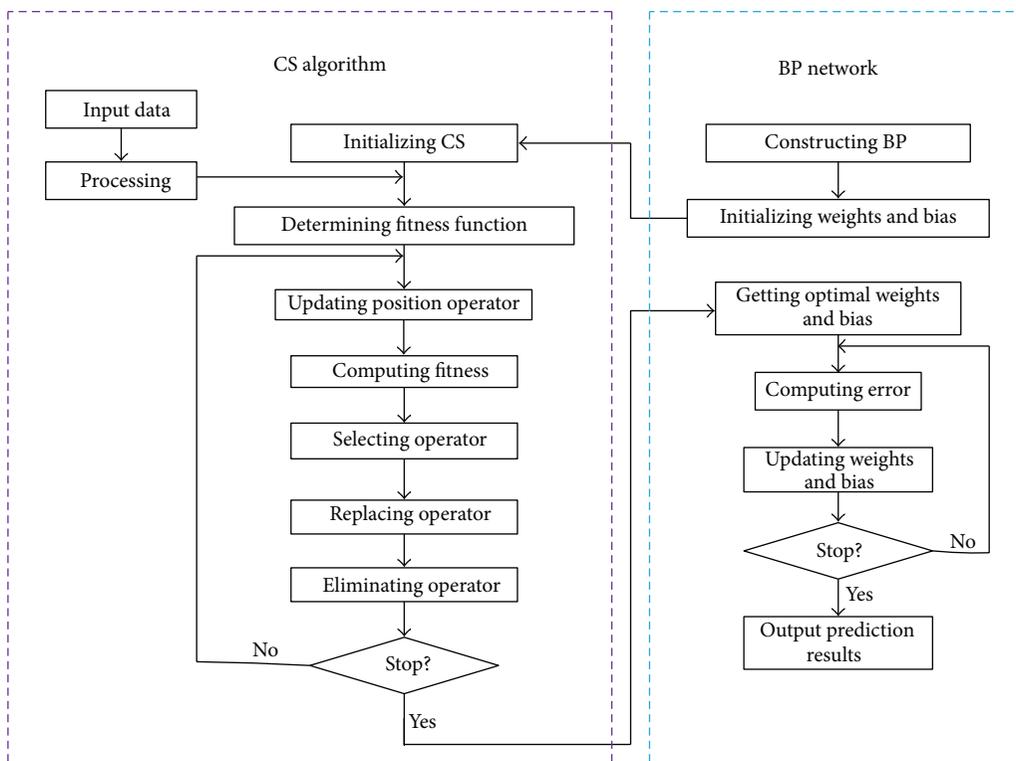


FIGURE 1: Flowchart of CSBP.

(1) *Initializing CS.* Cuckoo individual is encoded in the real-coded form, and each individual is composed of real-number string that consists of the following four parts: connection weights between the hidden layer and output layer, connection weights between the hidden layer and the input layer, the bias in the output layer, and the hidden layer. Each cuckoo individual contains all the weights and bias in

BP network. According to the weights and bias in BP network, a certain BP network can be constructed.

(2) *Determining Fitness Function.* The initial weights and bias of BP network can be determined according to the best individual. After training the BP network, it is used to predict the output. The fitness value of cuckoo individual F is the

sum of the absolute error between the desired output and the predicted output E as follows:

$$F = k \left(\sum_{i=1}^n |y_i - o_i| \right), \quad (3)$$

where n is the node number of the output layer in BP network and k is a coefficient. y_i and o_i are the desired output and the predicted output for the node i in BP network.

(3) *Updating Position Operator.* A cuckoo (say i) is randomly chosen in the cuckoo population and its position is updated according to (1). The fitness (F_i) of the i th cuckoo at generation t and position $x_i(t)$ is evaluated by (3).

(4) *Selecting Operator.* Similarly, another cuckoo (say j , $i \neq j$) is randomly chosen in the cuckoo population and its position fitness (F_j) of the i th cuckoo at generation t and position $x_j(t)$ is evaluated by (3).

(5) *Replacing Operator.* If the fitness value of the cuckoo i is bigger than the cuckoo j , that is, $F_i > F_j$, x_j is replaced by the new solution.

(6) *Eliminating Operator.* In order to make the population in an optimum state all the time, $\text{ceil}(n * p_a)$ worst cuckoos are removed in each generation. At the same time, in order to make the population size unchanged, $\text{ceil}(n * p_a)$ cuckoos would randomly be generated. The cuckoos with the best fitness will be passed directly to the next generation. Here, $\text{ceil}(x)$ rounds the elements of x to the nearest integers towards infinity.

BP network in CSBP (see Figure 1) is similar to an ordinary BP network, and the detailed steps can be represented as follows.

(1) *Determining BP Network Structure.* The weights and bias are randomly initialized, and then they are encoded according to the CS algorithm. The encoded weights and bias are input into the CS in order to optimize the BP network, followed by the CS algorithm (see Figure 1).

(2) *Construct CSBP Network.* The optimal weights and bias obtained from the CS algorithm are used to construct CSBP network. The training set is used to train the network and the training error is calculated. When the training error meets the requirements, training of the CSBP network stops.

(3) *Predicted Output.* The test set is input into the trained CSBP network to predict output.

4. Simulation

A classical wine classification problem (<http://archive.ics.uci.edu/ml/datasets/wine>) is used to test the prediction effectiveness of the CSBP network. Wine data that originated from UCI wine database records three different varieties of wine on the chemical composition analysis grown in the same region in Italy. Different kinds of wine are identified with 1, 2, and 3. Each sample contains 14 dimensions. The first dimension represents a class identifier, and the others represent the characteristics of wine. In these 178 samples, 1–59, 60–130, and 131–178 belong to the first, second, and third

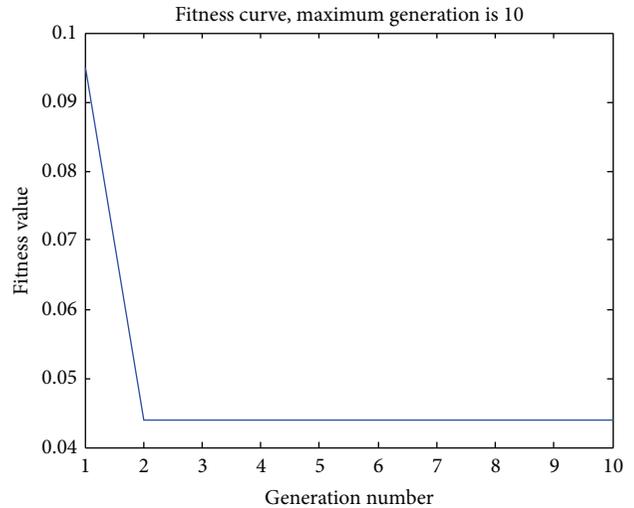


FIGURE 2: Convergent curve of CS with Max gen = 10, Popsiz = 10, and $p_a = 0.1$.

category, respectively. Each category is divided into two parts: training set and test set.

4.1. *Comparisons of CSBP with BP and GRNN.* In this section, CSBP is applied to solve wine classification problem, and the results are compared with the traditional feedforward neural networks (BP and GRNN).

For CSBP and BP, the neurons in input layer, hidden layer, and output layer are 13, 11, and 1, respectively. The length of encoded string number for each cuckoo individual is 166 that can be computed by the following equation: $13 * 11 + 11 + 11 * 1 + 1 = 166$. That is, CS would find the minimum of a 166-dimension function.

Firstly, the performance of CS when optimizing the weights and bias is tested with discovery rate $p_a = 0.1$ and few population sizes (10) and maximum generations (10). The fitness curve can be shown in Figure 2. From Figure 2, it can be seen that fitness value sharply decreases from 0.095 to 0.045 within two generations. This means that CS can significantly minimize the training error, and it does succeed in optimizing the basic BP network.

In the next experiments, all the parameters are set as follows. For BP network, epochs = 50, learning rate = 0.1, and objective = 0.00004. For GRNN, cyclic training method is used in order to select the best SPREAD value, making GRNN achieve the best prediction. For the CSBP, the BP network part has the same parameters with the basic BP; for CS algorithm part, we set discovery rate $p_a = 0.1$, population size NP = 50, and maximum generation Max gen = 50.

As intelligent algorithms always have some randomness, each run will generate different results. In order to get a typical statistical performance, 600 implementations are conducted for each method. The results are recorded in Figures 3 and 4 and Table 1.

From Table 1, for training set, the best performance and the average performance of BP, CSBP, and GRNN have little difference though CSBP performs slightly better than BP

TABLE I: The accuracy of BP, CSBP, and GRNN.

	Train set				Test set			
	Mean	Best	Worst	Std	Mean	Best	Worst	Std
BP	87.08	89	52	2.67	84.12	89	43	3.80
CSBP	88.78	89	88	0.44	87.33	89	82	0.97
GRNN	88.50	89	84	0.74	85.58	89	79	1.34

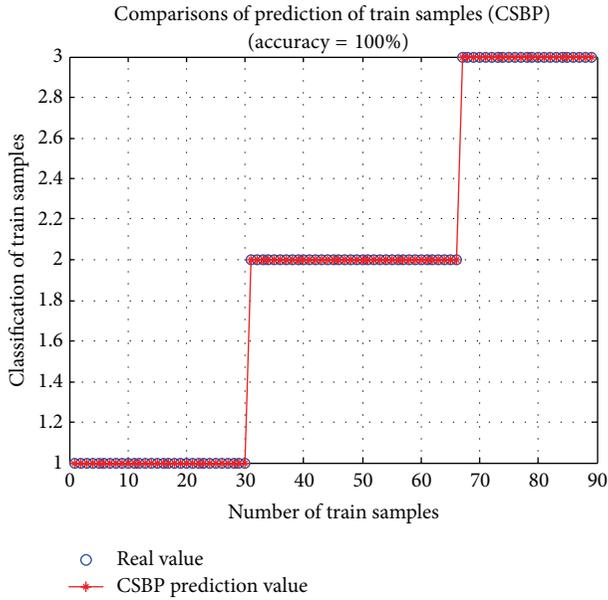


FIGURE 3: Prediction of train samples (CSBP) with Max gen = 50, Popsiz e = 50, and $p_a = 0.1$.

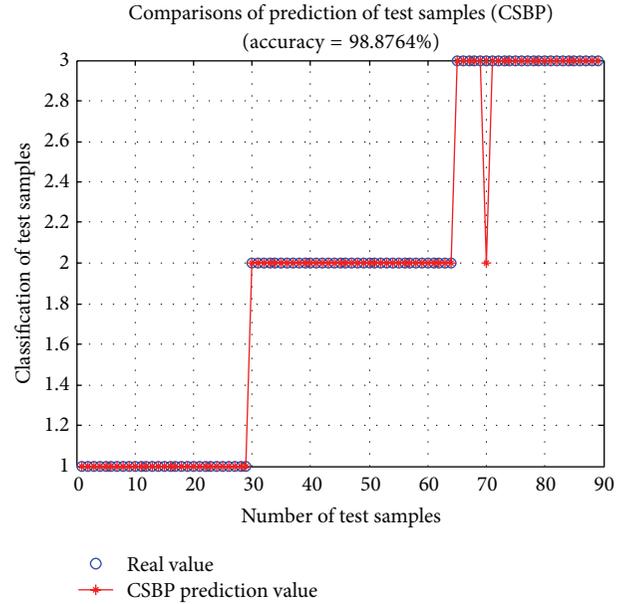


FIGURE 4: Prediction of test samples (CSBP) with Max gen = 50, Popsiz e = 50, and $p_a = 0.1$.

and GRNN. For the worst performance, CSBP is better than GRNN and is significantly superior to BP network. For test set, the overall prediction accuracy of CSBP is much better than BP and GRNN. In addition, the Std (standard deviation) of CSBP is clearly less than BP and GRNN. That is to say, CSBP would generate a more stable prediction output with little fluctuation. Moreover, from Figures 3 and 4, CSBP has a strong ability of solving the wine classification problem.

4.2. Parameter Study. As we are aware, parameter settings are of paramount importance to the performance of the metaheuristics. Here, the effectiveness of maximum generation, population size, and discovery rate will be analyzed and studied for CS algorithm.

4.2.1. Influence of the Maximum Generation for CSBP. Firstly, the number of maximum generations (Max gen) is studied, and the results are shown in Table 2. Table 2 shows that, when Max gen is equal to 40, 50, or 100, CSBP can approach all training samples without error. However, prediction accuracy is not always getting better with the increment of maximum generation. From prediction accuracy of test set, it can be seen that, when the number of maximum generations increases from 10 to 100, the prediction accuracy of test set is gradually

increased, decreased, and finally increased. Especially, when Max gen = 100, the prediction accuracy reaches maximum ($89/89 = 100\%$). Look carefully at Table 2; it is observed that the prediction accuracy changes in a very small range. That is, CSBP is insensitive to the parameter Max gen. Meanwhile, though more generations (such as 100) have a perfect prediction accuracy, it would take a longer time in order to optimize the weights and bias. Taking into consideration all the factors we analyzed earlier, the maximum generation is set to 50 in our present work.

4.2.2. Influence of the Population Size for CSBP. Subsequently, the influence of population size (NP) is studied (see Table 3). From Table 3, when NP is in the range [10, 100], especially equal to 100, CSBP can approach all training samples with little error. From prediction accuracy of test set, when the number of population size is equal to 100, the prediction accuracy of test set reaches maximum. Similar to the trend about Max gen, when the NP increases from 10 to 100, though prediction accuracy gradually increased, decreased, and finally increased, its fluctuation is little. This means that population size has little effect on the prediction accuracy of CSBP. In addition, when NP = 100, the prediction accuracy reaches maximum ($89/89 = 100\%$). The prediction accuracy

TABLE 2: The accuracy of CSBP with different maximum generations.

Max gen	Train set				Test set			
	Mean	Best	Worst	Std	Mean	Best	Worst	Std
10	88.60	89	88	0.55	86.4	88	83	2.30
20	87.8	89	85	1.64	87.4	89	85	1.52
30	88	89	87	1.00	86.6	89	84	1.95
40	89	89	89	0	88.17	89	85	1.60
50	89	89	89	0	88.2	89	87	1.10
60	87.6	89	87	1.14	84.4	89	77	4.93
70	88	89	86	1.41	85	87	84	1.41
80	89	89	89	0	86.8	89	82	3.19
90	87.2	89	85	1.79	86.2	89	82	2.77
100	88.8	89	88	0.45	89	89	89	0

TABLE 3: The accuracy of CSBP with different population sizes.

NP	Train set				Test set			
	Mean	Best	Worst	Std	Mean	Best	Worst	Std
10	88.6	88	88	0.55	86.4	88	83	2.30
20	88.4	89	86	1.34	87.6	89	86	1.34
30	88.6	89	88	0.55	86.4	89	84	2.51
40	88.8	89	88	0.45	87.8	89	87	0.84
50	88.8	89	88	0.45	88.8	89	88	0.45
60	88.4	89	87	0.89	88.4	89	87	0.89
70	87.5	89	82	2.01	84.6	89	67	6.72
80	88.4	89	82	1.78	86.9	89	67	5.42
90	88.4	89	87	0.89	87.4	89	86	1.14
100	89	89	89	0	89	89	89	0

TABLE 4: The accuracy of CSBP with different discovery rates.

P_a	Train set				Test set			
	Mean	Best	Worst	Std	Mean	Best	Worst	Std
0	88.4	89	88	0.52	84.7	89	77	3.30
0.1	88.6	89	87	0.70	87.1	89	83	2.23
0.2	88.2	89	87	1.03	86	89	77	3.56
0.3	88.4	89	86	1.08	84.9	89	75	5.17
0.4	88.9	89	88	0.32	85.9	89	76	4.45
0.5	88.1	89	83	1.91	84.6	89	76	4.92
0.6	88.4	89	86	1.08	86	89	81	3.01
0.7	88.5	89	87	0.71	86.4	89	82	2.50
0.8	88.2	89	86	1.14	85.7	89	77	3.56
0.9	88.1	89	83	1.91	86.9	89	82	2.51
1.0	88.1	89	85	1.29	85.8	89	76	4.15

of NP = 50 is only inferior to NP = 100. However, NP = 50 would take shorter time and obtain relatively satisfactory results. Comprehensively considering, the population size is set to 50 in other experiments.

4.2.3. *Influence of the Discovery Rate for CSBP.* Finally, in this section, the influence of the last important parameter discovery rate (p_a) is studied through a series of experiments (see Table 4). From Table 4, it is clear that, when p_a is in

the range [0, 1], the prediction accuracy of the CSBP is always bigger than 98.88% (88/89) for training samples. Due to CS, CSBP has a fast train speed. From the prediction accuracy of test set, when the discovery rate is equal to 0.1, the prediction accuracy of test set approaches the maximum 97.87% (87.1/89). Generally speaking, the prediction accuracy of CSBP for test samples varies a little with the incensement of the discovery rate. Comprehensively considering, the discovery rate p_a is set to 0.1 in our paper.

5. Conclusion

If BP network has bad initial weights and bias, it would fail to find the best solution. In order to overcome the disadvantages of BP, this paper uses the CS algorithm to optimize the weights and bias in the basic BP to solve the prediction problem. This method trains fast, can obtain the global optimal solution, and has good generalization performance. Most importantly, CSBP is insensitive to the initial weights and bias and the parameter settings of CS algorithm. We only need to input the training samples into the CSBP network, and then CSBP can obtain a unique optimal solution. By comparing to other traditional methods (such as BP and GRNN), this method has a faster and better generalization performance.

In future, our research highlights would be focused on in the following points. On the one hand, CSBP will be used to solve other regression and classification problems, and their results can be further compared to other methods, such as feedforward neural network [43], Wavelet Neural Network (WNN) [44, 45], and Extreme Learning Machine (ELM) [46, 47]. On the other hand, we will hybridize BP with some other metaheuristic algorithms, such as artificial plant optimization algorithm (APOA) [48], artificial physics optimization [49], flower pollination algorithm (FOA) [50], grey wolf optimizer (GSO) [51], and animal migration optimization (AMO) [52], so as to further improve the performance of BP.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors thank Dr. Gai-Ge Wang for his implementation of CSBP. The project is supported by the Ministry of Education Key Research Project under Grant no. 208098, the Hunan Science and Technology Planning Project under Grant no. 2012FJ3005, the Scientific Research Fund of Hunan Provincial Education Department (no. 12B005), and the Hunan Province Undergraduates Innovating Experimentation Project (no. (2013) 191-501).

References

- [1] A. Kaveh and S. Talatahari, "A novel heuristic optimization method: charged system search," *Acta Mechanica*, vol. 213, no. 3-4, pp. 267-289, 2010.
- [2] O. K. Erol and I. Eksin, "A new optimization method: Big Bang-Big Crunch," *Advances in Engineering Software*, vol. 37, no. 2, pp. 106-111, 2006.
- [3] A. Kaveh and S. Talatahari, "Size optimization of space trusses using Big Bang-Big Crunch algorithm," *Computers & Structures*, vol. 87, no. 17-18, pp. 1129-1140, 2009.
- [4] A. Kaveh and S. Talatahari, "Optimal design of Schwedler and ribbed domes via hybrid Big Bang-Big Crunch algorithm," *Journal of Constructional Steel Research*, vol. 66, no. 3, pp. 412-419, 2010.
- [5] A. Kaveh and S. Talatahari, "A discrete Big Bang-Big Crunch algorithm for optimal design of skeletal structures," *Asian Journal of Civil Engineering*, vol. 11, no. 1, pp. 103-122, 2010.
- [6] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, pp. 60-68, 2001.
- [7] P. Yadav, R. Kumar, S. K. Panda, and C. S. Chang, "An intelligent tuned harmony search algorithm for optimisation," *Information Sciences*, vol. 196, pp. 47-72, 2012.
- [8] S. Gholizadeh and A. Barzegar, "Shape optimization of structures for frequency constraints by sequential harmony search algorithm," *Engineering Optimization*, vol. 45, no. 6, pp. 627-646, 2012.
- [9] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942-1948, Perth, Australia, December 1995.
- [10] R. J. Kuo, Y. J. Syu, Z.-Y. Chen, and F. C. Tien, "Integration of particle swarm optimization and genetic algorithm for dynamic clustering," *Information Sciences*, vol. 195, pp. 124-140, 2012.
- [11] S. Talatahari, M. Kheirollahi, C. Farahmandpour, and A. H. Gandomi, "A multi-stage particle swarm for optimum design of truss structures," *Neural Computing & Applications*, vol. 23, no. 5, pp. 1297-1309, 2013.
- [12] Y. Zhang, D. Huang, M. Ji, and F. Xie, "Image segmentation using PSO and PCM with Mahalanobis distance," *Expert Systems with Applications*, vol. 38, no. 7, pp. 9036-9040, 2011.
- [13] S. Mirjalili and A. Lewis, "S-shaped versus V-shaped transfer functions for binary particle swarm optimization," *Swarm and Evolutionary Computation*, vol. 9, pp. 1-14, 2013.
- [14] G.-G. Wang, A. H. Gandomi, and A. H. Alavi, "An effective krill herd algorithm with migration operator in biogeography-based optimization," *Applied Mathematical Modelling*, 2013.
- [15] G. Wang, L. Guo, H. Duan, H. Wang, L. Liu, and M. Shao, "Hybridizing harmony search with biogeography based optimization for global numerical optimization," *Journal of Computational and Theoretical Nanoscience*, vol. 10, no. 10, pp. 2318-2328, 2013.
- [16] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702-713, 2008.
- [17] X. Li and M. Yin, "Multiobjective binary biogeography based optimization for feature selection using gene expression data," *IEEE Transactions on NanoBioscience*, vol. 12, no. 4, pp. 343-353, 2013.
- [18] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Let a biogeography-based optimizer train your Multi-Layer Perceptron," *Information Sciences*, 2014.
- [19] G.-G. Wang, L. Guo, H. Duan, and H. Wang, "A new improved firefly algorithm for global numerical optimization," *Journal of Computational and Theoretical Nanoscience*, vol. 11, no. 2, pp. 477-485, 2014.
- [20] X.-S. Yang, "Firefly algorithm, stochastic test functions and design optimisation," *International Journal of Bio-Inspired Computation*, vol. 2, no. 2, pp. 78-84, 2010.
- [21] L. Guo, G.-G. Wang, H. Wang, and D. Wang, "An effective hybrid firefly algorithm with harmony search for global numerical optimization," *The Scientific World Journal*, vol. 2013, Article ID 125625, 9 pages, 2013.
- [22] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Mixed variable structural optimization using Firefly Algorithm," *Computers & Structures*, vol. 89, no. 23-24, pp. 2325-2336, 2011.

- [23] X.-S. Yang, S. S. S. Hosseini, and A. H. Gandomi, "Firefly Algorithm for solving non-convex economic dispatch problems with valve loading effect," *Applied Soft Computing Journal*, vol. 12, no. 3, pp. 1180–1186, 2012.
- [24] X. Li and M. Yin, "Application of differential evolution algorithm on self-potential data," *PLoS ONE*, vol. 7, no. 12, Article ID e51199, 2012.
- [25] G.-G. Wang, A. H. Gandomi, A. H. Alavi, and G. S. Hao, "Hybrid krill herd algorithm with differential evolution for global numerical optimization," *Neural Computing & Applications*, 2013.
- [26] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
- [27] X. Li and M. Yin, "An opposition-based differential evolution algorithm for permutation flow shop scheduling based on diversity measure," *Advances in Engineering Software*, vol. 55, pp. 10–31, 2013.
- [28] A. H. Gandomi and A. H. Alavi, "Krill herd: a new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4831–4845, 2012.
- [29] G. Wang, L. Guo, H. Wang, H. Duan, L. Liu, and J. Li, "Incorporating mutation scheme into krill herd algorithm for global numerical optimization," *Neural Computing & Applications*, vol. 24, no. 3-4, pp. 853–871, 2014.
- [30] G.-G. Wang, A. H. Gandomi, and A. H. Alavi, "Stud krill herd algorithm," *Neurocomputing*, vol. 128, pp. 363–370, 2014.
- [31] G.-G. Wang, A. H. Gandomi, and A. H. Alavi, "A chaotic particle-swarm krill herd algorithm for global numerical optimization," *Kybernetes*, vol. 42, no. 6, pp. 962–978, 2013.
- [32] A. H. Gandomi, X.-S. Yang, A. H. Alavi, and S. Talatahari, "Bat algorithm for constrained optimization tasks," *Neural Computing & Applications*, vol. 22, no. 6, pp. 1239–1255, 2013.
- [33] X.-S. Yang and A. H. Gandomi, "Bat algorithm: a novel approach for global engineering optimization," *Engineering Computations*, vol. 29, no. 5, pp. 464–483, 2012.
- [34] S. Mirjalili, S. M. Mirjalili, and X.-S. Yang, "Binary bat algorithm," *Neural Computing & Applications*, 2013.
- [35] X.-S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, Coimbatore, India, December 2009.
- [36] X.-S. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, Frome, UK, 2nd edition, 2010.
- [37] X.-S. Yang and S. Deb, "Engineering optimisation by Cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343, 2010.
- [38] X.-S. Yang, Z. Cui, R. Xiao, A. H. Gandomi, and M. Karamanoglu, *Swarm Intelligence and Bio-Inspired Computation*, Elsevier, Waltham, Mass, USA, 2013.
- [39] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," *Engineering with Computers*, vol. 29, no. 1, pp. 17–35, 2013.
- [40] G.-G. Wang, A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "A new hybrid method based on Krill herd and Cuckoo search for global optimization tasks," *International Journal of Bio-Inspired Computation*, 2012.
- [41] X.-S. Yang and S. Deb, "Cuckoo search: recent advances and applications," *Neural Computing & Applications*, vol. 24, no. 1, pp. 169–174, 2014.
- [42] F. Trujillo-Romero, "Generation of neural networks using a genetic algorithm approach," *International Journal of Bio-Inspired Computation*, vol. 5, no. 5, pp. 289–302, 2013.
- [43] S. Mirjalili, S. Z. M. Hashim, and H. M. Sardroudi, "Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm," *Applied Mathematics and Computation*, vol. 218, no. 22, pp. 11125–11137, 2012.
- [44] G. Wang, L. Guo, and H. Duan, "Wavelet neural network using multiple wavelet functions in target threat assessment," *The Scientific World Journal*, vol. 2013, Article ID 632437, 7 pages, 2013.
- [45] Q. Zhang and A. Benveniste, "Wavelet networks," *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 889–898, 1992.
- [46] G.-B. Huang and L. Chen, "Convex incremental extreme learning machine," *Neurocomputing*, vol. 70, no. 16–18, pp. 3056–3062, 2007.
- [47] G.-B. Huang, X. Ding, and H. Zhou, "Optimization method based extreme learning machine for classification," *Neurocomputing*, vol. 74, no. 1–3, pp. 155–163, 2010.
- [48] X. Cai, S. Fan, and Y. Tan, "Light responsive curve selection for photosynthesis operator of APOA," *International Journal of Bio-Inspired Computation*, vol. 4, no. 6, pp. 373–379, 2012.
- [49] L. Xie, J. Zeng, and R. A. Formato, "Selection strategies for gravitational constant G in artificial physics optimisation based on analysis of convergence properties," *International Journal of Bio-Inspired Computation*, vol. 4, no. 6, pp. 380–391, 2012.
- [50] X.-S. Yang, M. Karamanoglu, and X. He, "Flower pollination algorithm: a novel approach for multiobjective optimization," *Engineering Optimization*, 2013.
- [51] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.
- [52] X. Li, J. Zhang, and M. Yin, "Animal migration optimization: an optimization algorithm inspired by animal migration behavior," *Neural Computing & Applications*, 2013.