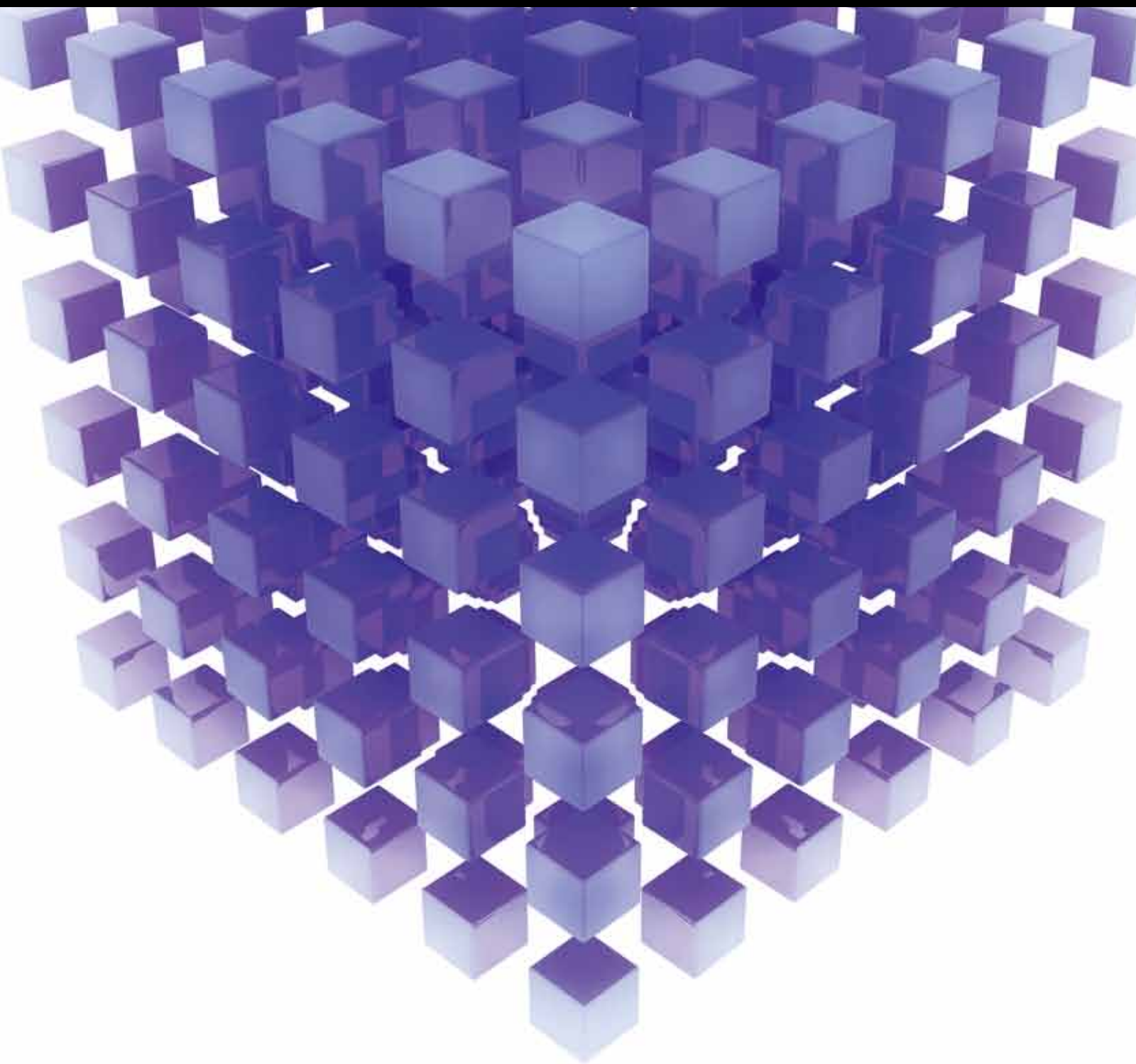


MATHEMATICAL PROBLEMS IN ENGINEERING

SWARM INTELLIGENCE in ENGINEERING

GUEST EDITORS: BIN YU, BAOZHEN YAO, AND RUI MU





Swarm Intelligence in Engineering

Mathematical Problems in Engineering

Swarm Intelligence in Engineering

Guest Editors: Baozhen Yao, Rui Mu, and Bin Yu



Copyright © 2013 Hindawi Publishing Corporation. All rights reserved.

This is a special issue published in "Mathematical Problems in Engineering." 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.

Editorial Board

- M. Abd El Aziz, Egypt
E. Abdel-Rahman, Canada
R. Abu Al-Rub, USA
Sarp Adali, South Africa
S. Alfonzetti, Italy
Igor Andrianov, Germany
S. Anita, Romania
W. Assawinchaichote, Thailand
Erwei Bai, USA
Ezzat Bakhoun, USA
José M. Balthazar, Brazil
Rasajit K. Bera, India
C. Bérenguer, France
J. Blakely, USA
S. Boccaletti, Spain
S. Bordas, UK
D. Boso, Italy
M. Boutayeb, France
Michael Brennan, UK
John Burns, USA
Salvatore Caddemi, Italy
Piermarco Cannarsa, Italy
Jose E. Capilla, Spain
Carlo Cattani, Italy
M. M. Cavalcanti, Brazil
Diego J. Celentano, Chile
Mohammed Chadli, France
Arindam Chakraborty, USA
Yong-Kui Chang, China
Michael J. Chappell, UK
Kue-Hong Chen, Taiwan
Kui Fu Chen, China
Xinkai Chen, Japan
Jyh H. Chou, Taiwan
Slim Choura, Tunisia
C. Cruz-Hernandez, Mexico
Swagatam Das, India
Filippo de Monte, Italy
M. de Pinho, Portugal
A. Desimone, Italy
Y. Dimakopoulos, Greece
Baocang Ding, China
Joao B. R. Do Val, Brazil
Daoyi Dong, Australia
Balram Dubey, India
Horst Ecker, Austria
M. Onder Efe, Turkey
E. Elabbasy, Egypt
Alex Elias-Zuniga, Mexico
Anders Eriksson, Sweden
V. S. Erturk, Turkey
Qi Fan, USA
Moez Feki, Tunisia
Ricardo Femat, Mexico
R. A. Valente, Portugal
C. Fuerte-Esquivel, Mexico
Zoran Gajic, USA
Ugo Galvanetto, Italy
Furong Gao, Hong Kong
Xin-Lin Gao, USA
Behrouz Gatmiri, Iran
Oleg V. Gendelman, Israel
Didier Georges, France
P. B. Goncalves, Brazil
Oded Gottlieb, Israel
Fabrizio Greco, Italy
Quang Phuc Ha, Australia
M. R. Hajj, USA
Thomas Hanne, Switzerland
Tasawar Hayat, Pakistan
K. (Stevanovic) Hedrih, Serbia
M.I. Herreros, Spain
Wei-Chiang Hong, Taiwan
J. Horacek, Czech Republic
C. Huang, China
Gordon Huang, Canada
Yi Feng Hung, Taiwan
Hai-Feng Huo, China
Asier Ibeas, Spain
Anuar Ishak, Malaysia
Reza Jazar, Australia
Zhijian Ji, China
J. Jiang, China
J. J. Judice, Portugal
Tadeusz Kaczorek, Poland
Tamas Kalmar-Nagy, USA
Tomasz Kapitaniak, Poland
H. R. Karimi, Norway
Metin Kaya, Turkey
Nikolaos Kazantzis, USA
Farzad Khani, Iran
K. Krabbenhoft, Australia
J. Kurths, Germany
Claude Lamarque, France
Marek Lefik, Poland
Stefano Lenci, Italy
Roman Lewandowski, Poland
Shanling Li, Canada
Jian Li, China
Ming Li, China
Tao Li, China
Shihua Li, China
Teh-Lu Liao, Taiwan
P. Liatsis, UK
Shueei Lin, Taiwan
Yi-Kuei Lin, Taiwan
Jui-Sheng Lin, Taiwan
Bin Liu, Australia
Wanquan Liu, Australia
Yuji Liu, China
Paolo Lonetti, Italy
Vassilios C. Loukopoulos, Greece
Junguo Lu, China
Chien-Yu Lu, Taiwan
Alexei Mailybaev, Brazil
Manoranjan Maiti, India
O. D. Makinde, South Africa
Rafael Martinez-Guerra, Mexico
Driss Mehdi, France
R. Melnik, Canada
Xinzhu Meng, China
Yuri V. Mikhlin, Ukraine
G. Milovanovic, Serbia
E. Momoniat, South Africa
Trung N. Thoi, Vietnam
Hung Nguyen-Xuan, Vietnam

Ben T. Nohara, Japan
Anthony Nouy, France
Sotiris Ntouyas, Greece
G. Olivar, Colombia
Claudio Padra, Argentina
F. Pellicano, Italy
Matjaz Perc, Slovenia
Vu Ngoc Phat, Vietnam
A. Pogromsky, The Netherlands
S. Pohjolainen, Finland
S. Potapenko, Canada
Sergio Preidikman, USA
C. Proppe, Germany
Hector Puebla, Mexico
Justo Puerto, Spain
Dane Quinn, USA
K. Rajagopal, USA
G. Ranzi, Australia
S. Ravindran, USA
G. Rega, Italy
Pedro Ribeiro, Portugal
J. Rodellar, Spain
R. Rodriguez-Lopez, Spain
A. Rodriguez-Luis, Spain
Ignacio Romero, Spain
Hamid Ronagh, Australia
Carla Roque, Portugal
R. R. García, Spain
M. Salehi, Iran
M. Sanjuán, Spain
I. F. Santos, Denmark
N. S. Sapidis, Greece

B. Sarler, Slovenia
A. V. Savkin, Australia
M. Scalia, Italy
M. A. Seddeek, Egypt
Alexander P. Seyranian, Russia
Leonid Shaikhet, Ukraine
Cheng Shao, China
Bo Shen, Germany
Daichao Sheng, Australia
Tony Sheu, Taiwan
Zhan Shu, UK
JianJun Shu, Singapore
Dan Simon, USA
L. Simoni, Italy
G. M. Sisoiev, UK
C. H. Skiadas, Greece
D. Spinello, Canada
Sri Sridharan, USA
Rolf Stenberg, Finland
Xi-Ming Sun, China
Jitao Sun, China
Changyin Sun, China
A. Swierniak, Poland
Allen Tannenbaum, USA
C.n Toma, Romania
I. N. Trendafilova, UK
Alberto Trevisani, Italy
Jung-Fa Tsai, Taiwan
John Tsinias, Greece
K. Vajravelu, USA
V. Vampa, Argentina
Josep Vehi, Spain

Stefano Vidoli, Italy
Cheng C. Wang, Taiwan
Youqing Wang, China
Yijing Wang, China
Xiaojun Wang, China
Dan Wang, China
Yongqi Wang, Germany
Moran Wang, China
Gerhard-Wilhelm W., Turkey
J. A.S. Witteveen, USA
Kwok-Wo Wong, Hong Kong
Ligang Wu, China
Zheng-Guang Wu, China
W. Xing-yuan, China
Xuping Xu, USA
X. Frank XU, USA
Jun-Juh Yan, Taiwan
Xing-Gang Yan, UK
Suh-Yuh Yang, Taiwan
M. T. Yassen, Egypt
M. I. Younis, USA
Huang Yuan, Germany
S.P. Yung, Hong Kong
Ion Zaballa, Spain
A. M. Zenkour, Saudi Arabia
J. Zhan, China
Xu Zhang, China
Y. Zhang, USA
Lu Zhen, China
L. Zheng, China
Jian Zhou, UK
Zexuan Zhu, China

Contents

Swarm Intelligence in Engineering, Baozhen Yao, Rui Mu, and Bin Yu
Volume 2013, Article ID 835251, pages

An Improved Harmony Search Based on Teaching-Learning Strategy for Unconstrained Optimization Problems, Shouheng Tuo, Longquan Yong, and Tao Zhou
Volume 2013, Article ID 413565, 29 pages

A Swarm Optimization Algorithm for Multimodal Functions and Its Application in Multicircle Detection, Erik Cuevas, Daniel Zaldívar, and Marco Pérez-Cisneros
Volume 2013, Article ID 948303, 22 pages

Solving Unconstrained Global Optimization Problems via Hybrid Swarm Intelligence Approaches, Jui-Yu Wu
Volume 2013, Article ID 256180, 15 pages

Escape-Route Planning of Underground Coal Mine Based on Improved Ant Algorithm, Guangwei Yan and Dandan Feng
Volume 2013, Article ID 687969, 14 pages

A Constructive Data Classification Version of the Particle Swarm Optimization Algorithm, Alexandre Szabo and Leandro Nunes de Castro
Volume 2013, Article ID 459503, 13 pages

Particle Swarm Optimization Algorithm for Unrelated Parallel Machine Scheduling with Release Dates, Yang-Kuei Lin
Volume 2013, Article ID 409486, 9 pages

Dynamic Route Guidance Using Improved Genetic Algorithms, Zhanke Yu, Mingfang Ni, Zeyan Wang, and Yanhua Zhang
Volume 2013, Article ID 765135, 6 pages

A Hybrid Algorithm of Traffic Accident Data Mining on Cause Analysis, Jianfeng Xi, Zhenhai Gao, Shifeng Niu, Tongqiang Ding, and Guobao Ning
Volume 2013, Article ID 302627, 8 pages

Improved Barebones Particle Swarm Optimization with Neighborhood Search and Its Application on Ship Design, Jingzheng Yao and Duanfeng Han
Volume 2013, Article ID 175848, 12 pages

A Simple and Efficient Artificial Bee Colony Algorithm, Yunfeng Xu, Ping Fan, and Ling Yuan
Volume 2013, Article ID 526315, 9 pages

Articulated Human Motion Tracking Using Sequential Immune Genetic Algorithm, Yi Li and Zhengxing Sun
Volume 2013, Article ID 921510, 16 pages



The Optimized Transport Scheme of Empty and Heavy Containers with Novel Genetic Algorithm, Lianwei Qu, Kang Chen, Bao Hongli, Jingshu Ma, and Wan Tao

Volume 2013, Article ID 434125, 5 pages

Daily Commute Time Prediction Based on Genetic Algorithm, Fang Zong, Haiyun Lin, Bo Yu, and Xiang Pan

Volume 2012, Article ID 321574, 20 pages

A Hybrid Approach Using an Artificial Bee Algorithm with Mixed Integer Programming Applied to a Large-Scale Capacitated Facility Location Problem, Guillermo Cabrera G., Enrique Cabrera, Ricardo Soto, L. Jose Miguel Rubio, Broderick Crawford, and Fernando Paredes

Volume 2012, Article ID 954249, 14 pages

Economic Analysis on Value Chain of Taxi Fleet with Battery-Swapping Mode Using Multiobjective Genetic Algorithm, Guobao Ning, Zijian Zhen, Peng Wang, Yang Li, and Huaixian Yin

Volume 2012, Article ID 175912, 15 pages

Bus Dispatching Interval Optimization Based on Adaptive Bacteria Foraging Algorithm, Zhong-hua Wei, Xia Zhao, Ke-wen Wang, and Yan Xiong

Volume 2012, Article ID 389086, 10 pages

Direct Index Method of Beam Damage Location Detection Based on Difference Theory of Strain Modal Shapes and the Genetic Algorithms Application, Bao Zhenming, Zhao Deyou, Lin Zhe, Ma Jun, and Zhen Yaobao

Volume 2012, Article ID 302482, 12 pages

Genetic Algorithm for Multiuser Discrete Network Design Problem under Demand Uncertainty, Wu Juan, Lu Huapu, Yu Xinxin, and Bian Changzhi

Volume 2012, Article ID 686272, 17 pages

Simulated Annealing-Based Ant Colony Algorithm for Tugboat Scheduling Optimization, Qi Xu, Jun Mao, and Zhihong Jin

Volume 2012, Article ID 246978, 22 pages

Guidance Compliance Behavior on VMS Based on SOAR Cognitive Architecture, Shiquan Zhong, Hongwei Ma, Lizhen Zhou, Xuelian Wang, Shoufeng Ma, and Ning Jia

Volume 2012, Article ID 530561, 21 pages

Editorial

Swarm Intelligence in Engineering

Baozhen Yao,¹ Rui Mu,² and Bin Yu³

¹ Dalian University of Technology, Dalian, Liaoning 116024, China

² Delft University of Technology, Delft, The Netherlands

³ Dalian Maritime University, Dalian, Liaoning 116026, China

Correspondence should be addressed to Bin Yu; ybzhyb@163.com

Received 24 February 2013; Accepted 24 February 2013

Copyright © 2013 Baozhen Yao 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.

Swarm intelligence (SI) is an artificial intelligence technique based on the study of behavior of simple individuals (e.g., ant colonies, bird flocking, animal herding, and honey bees), which has attracted much attention of researchers and has also been applied successfully to solve optimization problems in engineering. However, for large and complex problems, SI algorithms consume often much computation time due to stochastic feature of the search approaches. Therefore, there is a potential requirement to develop efficient algorithm to find solutions under the limited resources, time, and money in real-world applications.

Within this context, this special issue serves as a forum to highlight the most significant recent developments on the topics of SI and to apply SI algorithms in real-life scenario. The works in this issue contain new insights and findings in this field. A broad range of topics has been discussed, especially in the following areas, benchmarking and evaluation of new SI algorithms, convergence proof for SI algorithms, comparative theoretical and empirical studies on SI algorithms, and SI algorithms for real-world application.

Some works focus on the application of genetic algorithm in different area, for example, G. Ning et al.'s "*Economic analysis on value chain of taxi fleet with battery-swapping mode using multiobjective genetic algorithm*" presents an economic analysis model on value chain of taxi fleet with battery-swapping mode in a pilot city. A multiobjective genetic algorithm is used to solve the problem. The real data collected from the pilot city proves that the multiobjective genetic algorithm is tested as an effective method to solve this problem.

B. Zhenming et al. "*Direct index method of beam damage location detection based on difference theory of strain modal shapes and the genetic algorithms application*" applies direct index method SMSD and the Genetic Algorithms into structural damage identification. Numerical simulation shows that the criteria of damage location detection can be obtained by strain mode difference curve through cubic spline interpolation.

F. Zong et al.'s "*Daily commute time prediction based on genetic algorithm*" presents a joint discrete-continuous model for activity-travel time allocation by employing the ordered probit model for departure time choice and the hazard model for travel time prediction. Genetic algorithm (GA) is employed for optimizing the parameter in the hazard model. The results also show that the genetic algorithm contributes to the optimization and thus the high accuracy of the hazard model.

Qu et al.'s "*The optimized transport scheme of empty and heavy containers with novel genetic algorithm*" proposed a model with objective maximizing the route benefits to design the transport scheme of empty and heavy containers reasonably. A novel GA is developed to solve the model. The case study about China-Europe route proves that this model can improve the liner company's benefits effectively.

W. Juan et al.'s "*Genetic algorithm for multiuser discrete network design problem under demand uncertainty*" presents a bilevel model for discrete network design. An iterative approach including an improved genetic algorithm and a Frank-Wolfe algorithm is used to solve the bilevel model. The numerical results on the Nguyen Dupuis network show

that the model and the related algorithms were effective for discrete network design.

Z. Yu et al.'s "*Dynamic route guidance using improved genetic algorithms*" presents an improved genetic algorithm (IGA) for dynamic route guidance algorithm. The proposed IGA designs a vicinity crossover technique and a greedy backward mutation technique to increase the population diversity and strengthen local search ability. The simulation results show the effectiveness of the proposed algorithm.

Y. Li and Z. Sun's "*Articulated human motion tracking using sequential immune genetic algorithm*" proposed a novel generative method for human motion tracking in the framework of evolutionary computation. The paper designed an IGA-based method to estimate human pose from static images. It also proposed a sequential IGA (S-IGA) algorithm by incorporating the temporal continuity information into the traditional IGA. Experimental results show that our IGA-based pose estimation method can achieve viewpoint invariant 3D pose reconstruction, and the S-IGA-based tracking method can achieve accurate and stable tracking of 3D human motion.

Some works present improved algorithm based on particle swarm optimization. J. Yao and D. Han "*Improved bare-bones particle swarm optimization with neighborhood search and its application on ship design*" proposed a new BPSO variant called BPSO with neighborhood search (NSBPSO) to achieve a tradeoff between exploitation during the search process. In the paper, experiments are conducted on twelve benchmark functions and a real-world problem of ship design. Simulation results show that NSBPSO outperforms the standard PSO, BPSO, and six other improved PSO algorithms.

J. Xi et al.'s "*A hybrid algorithm of traffic accident data mining on cause analysis*" puts forward an improved association rule algorithm based on particle swarm optimization (PSO). The new method is used to analyze the correlation between traffic accident attributes and causes. *T*-test model and Delphi method were deployed to test and verify the accuracy of the improved algorithm, the result of which was ten times faster speed for random traffic accident data sampling analyses on average. And the final result proves that the improved algorithm was accurate and stable.

Y. Lin's "*Particle swarm optimization algorithm for unrelated parallel machine scheduling with release dates*" proposed a heuristic and a very effective particle swarm optimization (PSO) algorithm to tackle the problem of minimizing makespan for n jobs on m unrelated parallel machines with release dates. Computational results show that the proposed PSO is very accurate and that it outperforms the existing metaheuristic.

A. Szabo and L. de Castro's "*A constructive data classification version of the particle swarm optimization algorithm*" introduced new particle swarm optimization algorithm specially designed to solve continuous parameter optimization problems. The proposals were applied to wide range of databases from the literature, and the results show that they are competitive in relation to other approaches from the literature, with the advantage of having a dynamically constructed architecture.

Also, ant colony algorithm is discussed in some works. Q. Xu et al.'s "*Simulated annealing-based ant colony algorithm for tugboat scheduling optimization*" presents a hybrid simulated annealing-based ant colony algorithm to optimize the tugboat scheduling. In this paper, experiments are conducted to examine the effectiveness of the proposed algorithm for the tugboat scheduling problem.

G. Yan and D. Feng's "*Escape-route planning of underground coal mine based on improved ant algorithm*" proposed a new escape-route planning method of underground mines based on the improved ant algorithm. A tunnel network zoning method and max-min ant system method are used to improve the performance of the ant algorithm. Experiments show that the proposed method can find good escape routes correctly and efficiently and can be used in the escape-route planning of large and medium underground cone mines.

There are also some works discussing other algorithms in this field. J. Wu's "*Solving unconstrained global optimization problems via hybrid swarm intelligence approaches*" gives an overview of two efficient hybrid SGO approaches, namely, a real-coded genetic algorithm-based PSO (RGA-PSO) method and an artificial immune algorithm-based PSO (AIA-PSO) method. Numerical results indicate that the RGA-PSO and AIA-PSO approaches can be considered alternative SGO approaches for solving standard-dimensional UGO problems.

Z. Wei et al.'s "*Bus dispatching interval optimization based on adaptive bacteria foraging algorithm*" applied the improved bacterial algorithm to schedule the bus departing interval. Based on adaptive bacteria foraging algorithm (ABFA), a model on one bus line in Hohhot city in China was established and simulated. The final results showed that ABFA was most feasible in optimizing variables.

S. TUO et al.'s "*An improved harmony search based on teaching-learning strategy for unconstrained optimization problems*" presents an improved global harmony search algorithm named harmony search based on teaching-learning (HSTL) for high-dimensional complex optimization problems. The experimental results of 31 complex benchmark functions demonstrate that the HSTL method has strong convergence, robustness, and better balance capacity of space exploration and local exploitation on high-dimensional complex optimization problems.

Y. Xu et al.'s "*A simple and efficient artificial bee colony algorithm*" proposes a new artificial bee colony (NABC) algorithm, which modifies the search pattern of both employed and onlooker bees. Experiments are conducted on a set of twelve benchmark functions. Simulation results show that this approach is significantly better or at least comparable to the original ABC and seven other stochastic algorithms.

S. Zhong et al.'s "*Guidance compliance behavior on VMS based on SOAR cognitive architecture*" introduced SOAR to design the agent with the detailed description of the working memory, long-term memory, decision cycle, and learning mechanism based on the multiagent platform. Experiments are simulated many times under given simulation network and conditions. The results, including the comparison between guidance and no guidance, the state transition times, and average chunking times, are analyzed to

further study the laws of guidance compliance and learning mechanism.

Cuevas's "A Swarm Optimization Algorithm for Multimodal Functions and its Application in Multi-circle Detection" presents a new swarm multimodal optimization algorithm named as the Collective Animal Behavior (CAB). In the proposed algorithm, searcher agents emulate a group of animals which interact to each other based on simple biological laws that are modeled as evolutionary operators. Numerical experiments are conducted to compare the proposed method with the state-of-the-art methods on benchmark functions. The proposed algorithm has been also applied to the engineering problem of multi-circle detection, achieving satisfactory results.

G. Cabrera et al.'s "A hybrid approach using an artificial bee algorithm with mixed integer programming applied to a large-scale capacitated facility location problem" presents a hybridization of two different approaches applied to the well-known capacitated facility location problem (CFLP). The artificial bee algorithm (BA) is used to select a promising subset of locations (warehouses) which are solely included in the mixed integer programming (MIP) model. According to the results, combining the BA with a mathematical programming approach appears to be an interesting research area in the combinatorial optimization.

These articles demonstrate the advancement that swarm intelligence technologies have made for supporting problem solving in engineering. Developing the efficient algorithm to find solutions can provide solutions for large and complex problems under the limited resources, time, and money in real-world applications. We would like to express our gratitude to the many reviewers for their hard works. We would also like to thank the authors for their contributions to the special issue. This special issue could not have been completed without their dedication and support.

Baozhen Yao
Rui Mu
Bin Yu

Research Article

An Improved Harmony Search Based on Teaching-Learning Strategy for Unconstrained Optimization Problems

Shouheng Tuo,¹ Longquan Yong,¹ and Tao Zhou²

¹ School of Mathematics and Computer Science, Shaanxi University of Technology, Hanzhong, Shaanxi 723001, China

² School of Science, Ningxia Medical University, Yinchuan, Ningxia 750004, China

Correspondence should be addressed to Shouheng Tuo; tuo_sh@126.com

Received 22 August 2012; Accepted 12 November 2012

Academic Editor: Baozhen Yao

Copyright © 2013 Shouheng Tuo 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.

Harmony search (HS) algorithm is an emerging population-based metaheuristic algorithm, which is inspired by the music improvisation process. The HS method has been developed rapidly and applied widely during the past decade. In this paper, an improved global harmony search algorithm, named harmony search based on teaching-learning (HSTL), is presented for high dimension complex optimization problems. In HSTL algorithm, four strategies (harmony memory consideration, teaching-learning strategy, local pitch adjusting, and random mutation) are employed to maintain the proper balance between convergence and population diversity, and dynamic strategy is adopted to change the parameters. The proposed HSTL algorithm is investigated and compared with three other state-of-the-art HS optimization algorithms. Furthermore, to demonstrate the robustness and convergence, the success rate and convergence analysis is also studied. The experimental results of 31 complex benchmark functions demonstrate that the HSTL method has strong convergence and robustness and has better balance capacity of space exploration and local exploitation on high dimension complex optimization problems.

1. Introduction

With the development of scientific technology, many real-life optimization problems are becoming more and more complex and difficult. So how to resolve the complex problems in an exact manner within a reasonable time cost is very important. The traditional optimization algorithms are difficult to solve the nonlinear and nondifferential problems. In recent years, most popular swarm intelligence optimization algorithms, such as genetic algorithm (GA), particle swarm optimization (PSO), and differential evolution (DE) algorithm, have been successfully applied to large-scale complicated problems of scientific and engineering computing.

Inspired by the process of the musicians' improvisation of the harmony, the harmony search (HS) algorithm is proposed by Geem et al. [1, 2]. Similar to the GA and PSO, the HS algorithm is a meta-heuristic random optimization algorithm. In recent years, HS has been applied more broadly in the fields of engineering optimization. Such as pipe network design [3], structural optimization [4], clustering of text document [5], combined heat and power economic dispatch problem [6],

and scheduling of multiple dam system [7]. The aforementioned applications show that HS algorithm has significant in solving complex engineering application problems. It has strong ability of exploration and has a cheap running cost. However, the classical harmony search algorithm is not efficient enough for solving large-scale problems, which has a slow convergence speed and low-precision solution. So some improved HS algorithms were proposed. Mahdavi et al. proposed an improved HS algorithm (IHS) [8] that employed a novel method generating new solution vectors which enhanced accuracy and convergence speed. Recently, Omran and Mahdavi tried to improve the performance of HS by incorporating some techniques from swarm intelligence. Tuo and Yong presented an improved harmony search algorithm with chaos (HSCH) [9]. The new variant named GHS (Global Best Harmony Search) [10] reportedly outperformed the HS and IHS algorithm over the benchmark problems.

HS algorithm has strong ability on exploring the regions of the solution space in a reasonable time. However, it has lower exploitation ability in later period. Therefore, some improved HS algorithm is proposed to enhance the

local search ability and solution precision, such as local-best harmony search algorithm with dynamic subpopulations (DLHS) [11], harmony search algorithm with differential mutation operator (HSDE) [12], and novel global harmony search algorithm for unconstrained problems (NGHS) [13, 14]. In addition, Gao et al. proposed modified harmony search methods for unimodal and multimodal optimization [15]. Pan et al. proposed self-adaptive harmony search algorithm for optimization (SGHS) [16]. Yadav et al. presented an intelligent tuned HS algorithm [17].

Now some existing state-of-the-art harmony search algorithms, such as NGHS, HSDE, and SGHS, have shown exceptional problem-solving ability, but they have some disadvantages over the multimodal and high dimensional function. To enhance the performance of high dimensional multimodal problem by HS method, this paper proposed an improved large-scale HS algorithm with harmony search teaching-learning (HSTL). In the HSTL algorithm, the HS algorithm is improved by embedding teaching-learning-based-optimization (TLBO) [17, 18] method which has a strong searching capacity for high dimensional problem.

The rest of this paper is organized as follows. In Section 2, the basic framework of the classical harmony search method is summarized in a comprehensive style, and the two excellent variants of HS (SGHS and NGHS) are briefly presented. A novel teaching-learning-based-optimization algorithm is provided, and the details of HSTL algorithm are presented in Section 3. In Section 4, 31 different characteristic benchmark problems, which consist of separable problems, nonseparable problems, shifted problems, shifted rotated problems, and hybrid composite problems, are considered, and the numerical results of HSTL method are demonstrated. Furthermore, to investigate the robustness and convergent performance of the HSTL algorithm, the comparison results of convergence speed and success rate are presented, and convergence analysis is preliminary put out. Finally, the research findings and contributions of the proposed HSTL algorithm are discussed in Section 5.

2. HS Algorithm and Other Variants

In this section, we introduce the classical harmony search (HS) algorithm and two excellent variants of HS algorithms: self-adaptive global-best harmony search (SGHS) algorithm and novel global harmony search (NGHS) algorithm.

2.1. Classical Harmony Search Algorithm (HS). The steps in the procedure of classical harmony search algorithm are as follows.

Step 1 (initialize the problem and algorithm parameters). In this step, the optimization problem is specified as follows:

$$\begin{aligned} \min \quad & f(x), \\ \text{s.t.} \quad & x = (x_1, x_2, \dots, x_D), \\ & x_i \in [x_i^L, x_i^U], \quad i = 1, 2, \dots, D, \end{aligned} \quad (1)$$

where $f(x)$ is an objective function, D is the number of decision variables.

The HS algorithm parameters are also specified in this step.

HMS: the harmony memory size or the number of solution vectors in the population;

HMCR: the harmony considering rate;

PAR: pitch adjusting rate;

BW: bandwidth;

maxFEs: the maximum number of improvisations.

Step 2 (initialize the harmony memory). The harmony memory (HM) consists of HMS harmony vectors. Each harmony vector is generated from a uniform distribution in the feasible space, as

$$\begin{aligned} x_i^j &= x_i^L + (x_i^U - x_i^L) \times \text{rand}(), \\ i &= 1, 2, \dots, D; \quad j = 1, 2, \dots, \text{HMS}, \end{aligned} \quad (2)$$

where $\text{rand}()$ is a random between 0 and 1.

$$\text{HM} = \begin{bmatrix} x^1 \\ x^2 \\ \vdots \\ x^{\text{HMS}} \end{bmatrix} = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_D^1 \\ x_1^2 & x_2^2 & \cdots & x_D^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{\text{HMS}} & x_2^{\text{HMS}} & \cdots & x_D^{\text{HMS}} \end{bmatrix}. \quad (3)$$

Step 3 (improvise a new harmony). A new harmony vector $x^{\text{new}} = (x_1^{\text{new}}, x_2^{\text{new}}, \dots, x_D^{\text{new}})$ is generated based on three rules:

- (a) Memory consideration;
- (b) Pitch adjustment;
- (c) Random generation.

The improvisation procedure of new harmony vector works as Algorithm 1.

A new potential variation (or an offspring) is generated in Step 3, which is equivalent to mutation and crossover operator in standard evolution algorithms (EAs).

Step 4 (update harmony memory). Get the worst harmony memory x^{worst} from the HM. If x^{new} is better than x^{worst} then $x^{\text{worst}} := x^{\text{new}}$.

Step 5 (check stopping criterion). If the stopping criterion (maxFEs) is satisfied, computation is terminated. Otherwise, Steps 3 and 4 are repeated.

2.2. The SGHS Algorithm. In order to select the best parameters automatically, a self-adaptive global-best harmony search (SGHS) algorithm is proposed by Pan et al. [16]. The SGHS dynamically changes BW according to

$$\text{BW}(t) = \begin{cases} \text{BW}_{\max} - \frac{\text{BW}_{\max} - \text{BW}_{\min}}{T_{\max}} \times 2t, & \text{if } t < T_{\max}, \\ \text{BW}_{\min}, & \text{if } t \geq T_{\max}, \end{cases} \quad (4)$$

```

For  $i = 1$  to  $D$  do
  If  $\text{rand}() \leq \text{HMCR}$  then
     $x_i^{\text{new}} = x_i^j, j \in U\{1, 2, \dots, \text{HMS}\}$ 
    If  $\text{rand}() \leq \text{PAR}$  then
       $x_i^{\text{new}} = x_i^{\text{new}} \pm \text{rand}() \times \text{BW}(i)$ 
       $x_i^{\text{new}} = \min(\max(x_i^{\text{new}}, x_i^L), x_i^U)$ 
    End If
  Else
     $x_i^{\text{new}} = x_i^L + (x_i^U - x_i^L) \times \text{rand}()$ 
  End If
End For

```

ALGORITHM 1: The improvisation procedure of new harmony vector by HS.

```

For  $i = 1$  to  $D$  do
   $x_i^r = 2 \times x_i^{\text{best}} - x_i^{\text{worst}}$ 
   $x_i^r = \min(\max(x_i^r, x_i^L), x_i^U)$ 
   $x_i^{\text{new}} = x_i^{\text{worst}} + \text{rand}() \times (x_i^r - x_i^{\text{worst}})$ 
  If  $\text{rand}() \leq p_m$  //random mutation
     $x_i^{\text{new}} = x_i^L + (x_i^U - x_i^L) \times \text{rand}()$ 
  End If
End For.

```

ALGORITHM 2: The NGHS modifies improvisation step.

where BW_{\max} and BW_{\min} are the maximum and minimum distance bandwidths.

HMCR (PAR) is dynamically selected from a suitable normally distribution with mean HMCRm (PARm) and standard deviation 0.01 (0.05). Initially, HMCRm (PARm) is set at 0.98 (0.9). After a specified number of generations, HMCRm (PARm) is recalculated by averaging all the recorded HMCR (PAR) values during the period of evolution.

2.3. The NGHS Algorithm. In the novel global harmony search (NGHS) algorithm [13, 14], three significant parameters, harmony memory considering rate (HMCR), bandwidth (BW), and pitch adjusting rate (PAR), are excluded from NGHS, and a random select rate (p_m) is included in the NGHS. In Step 3, NGHS works as Algorithm 2.

Where x^{best} and x^{worst} , respectively, are the best harmony and the worst harmony in HM, $\text{rand}()$ is uniformly generated random number in $[0, 1]$.

3. HSTL Algorithm

In this section, we proposed a novel harmony search method with teaching-learning (HSTL) strategy which was derived from teaching-learning-based optimization (TLBO) algorithm. Above all, the TLBO algorithm is introduced and analyzed, and then we focus on the details of HSTL algorithm and the strategies of dynamically adjusting the parameters.

3.1. The TLBO Algorithm. Teaching-learning-based Optimization (TLBO) algorithm [18–20] is a new nature-inspired

```

For each learner  $x^j (j = 1, 2, \dots, \text{NP})$ 
  Randomly select another learner  $x^k (j \neq k)$ 
  If  $x^j$  is superior to  $x^k$  then
     $x^{j,\text{new}} = x^{j,\text{old}} + \text{rand}() \times (x^j - x^k)$ 
  Else
     $x^{j,\text{new}} = x^{j,\text{old}} + \text{rand}() \times (x^k - x^j)$ 
  End
End for
If  $x^{j,\text{new}}$  is superior to  $x^{j,\text{old}}$  then
   $x^j = x^{j,\text{new}}$ 
End If

```

ALGORITHM 3: The procedure of learner phase.

algorithm; it mimics the teaching process of teacher and learning process among learners in a class. TLBO shows a better performance with less computational effort for large scale problems [19], that is, problems of a high dimensionality. In addition, TLBO needs very few parameters.

In the TLBO method, it is the task of teacher to try to increase mean knowledge of all learners of the class in the subject taught by him or her depending on his or her capability. Learners make efforts to increase their knowledge by interaction among themselves. A learner is considered as a solution or a vector, and different design variables of vector will be analogous to different subjects offered to learners and the learners' result is analogous to the "fitness" as in other population-based optimization techniques. The teacher is considered as the best solution obtained so far. The process of working of TLBO is divided into two phases, teacher phase and learner phase.

(1) *Teacher Phase.* Assume there are D number of subjects (i.e., design variables), "NP" number of learners (i.e., population size), then x_i^{best} is the best learner (i.e., teacher) in subject $i (i = 1, 2, \dots, D)$. The works of teaching are as follows:

$$x_i^{j,\text{new}} = x_i^{j,\text{old}} + \text{rand}() \times (x_i^{\text{best}} - T_F \times \text{Mean}_i),$$

$$\text{Mean}_i = \frac{1}{\text{NP}} \sum_{j=1}^{\text{NP}} x_i^j, \quad j = 1, 2, \dots, \text{NP}, \quad i = 1, 2, \dots, D, \quad (5)$$

where $x_i^{j,\text{old}}$ denotes the result of the j th ($j = 1, 2, \dots, \text{NP}$) learner before learning the i th ($i = 1, 2, \dots, D$) subject, and $x_i^{j,\text{new}}$ is the result of the j th learner after learning the i th subject. T_F is the teaching factor which decides the value of Mean_i to be changed. The value of T_F is generated randomly with probability as $T_F = \text{round}[1 + \text{rand}()]$.

When the learner x^j finished his or her learning from the teacher, update the x^j by $x^{j,\text{new}}$ if $x^{j,\text{new}}$ is better than $x^{j,\text{old}}$.

(2) *Learner Phase.* Another important approach to increase knowledge for a learner is to interact with other learners. Learning method is expressed in Algorithm 3.

Even since the TLBO algorithm proposed in 2011 by Rao et al. [18], it has been applied in the fields of engineering optimization, such as mechanical design optimization [18, 21],

heat exchangers [22], thermoelectric cooler [23], and unconstrained and constrained real parameter optimization problems [24].

In the TLBO method, teacher phase relying on the best solution found so far usually has the fast convergence speed and the best ability of exploitation; it is more suitable for improving accuracy of the global optimal solution. Learner phase relying on other learners usually has the slow convergence speed; however, it bears stronger exploration capability for solving multimodal problems. Therefore, we use the TLBO method to improve the performance and efficiency of HS algorithm in the HSTL method.

3.2. The HSTL Algorithm. In order to achieve the most satisfactory optimization performance by applying the HS algorithm to a given problem, we develop a novel harmony search algorithm combined with teaching-learning strategy, in which both new harmony generation strategies and associated control parameter values can be dynamically changed according to the process of evolution.

It is of a high importance between the convergence and the diversity in order to improve the search efficiency. In the classical HS algorithm, a new harmony is generated in Step 3. After the selecting operation in Step 4, the population variance may increase or decrease. With a high population variance, the diversity and exploration power will increase, in the same time the convergence and the exploitation power will decrease accordingly. Conversely, with a low population variance, the convergence and the exploitation power will increase [25]; the diversity and the exploration power will decrease. So it is significant how to keep balance between the convergence and the diversity. Classical HS algorithm loses its ability easily at later evolution process [26], because of improvising new harmony from HM with a high HMCR and local adjusting with PAR. And HM diversity decreases gradually from the early iteration to the last. However, in HS algorithm, a low HMCR employed will increase the probability (1-HMCR) of random selection in search space; the exploration power will enhance, but the local search ability and the exploitation accuracy cannot be improved by single pitch adjusting strategy.

To overcome the inherent weaknesses of HS, in this section, we propose HSTL method. An improved teaching-learning strategy is employed to improve the search ability of optimal solution in the HSTL method. The HSTL algorithm works as follows.

- (1) Optimization target vector preparation: $x^{\text{new}} = x^r$, where x^r ($r = 1, 2, \dots, \text{HMS}$) is a harmony vector selected randomly in HM. Next, four strategies are employed to improve the target vector.

- (2) Improve the target vector x^{new} with the following 4 strategies.

(a) *Harmony Memory Consideration.* The i th design value of the target vector x_i^{new} , ($i = 1, 2, \dots, D$) is chosen randomly

from harmony memory (HM) with a probability of HMCR as

$$x_i^{\text{new}} = x_i^j, \quad j \in U\{1, 2, \dots, \text{HMS}\}, \quad i = 1, 2, \dots, D. \quad (6)$$

The HMCR is the rate of choosing one value from the historical values stored in the HM, which varies between 0 and 1.

(b) *Teaching-Learning Strategy.* If the i th ($i = 1, 2, \dots, D$) design variable of the target vector x_j^{new} has not been considered in HM, it will learn from the best harmony (i.e., teacher) with probability TLP in the teacher phase or from other harmony (i.e., learner) in the learner phase. The TLP is the probability of performing teaching-learning operator on design variables that have not been considered in (a). It works as follows.

Teacher Phase. In this phase, learner will learn from the best learner (i.e., teacher) in class. Learner modification is expressed as

$$x_i^{\text{new}} = x_i^{\text{new}} + \text{rand}() \times [x_i^{\text{best}} - T_F \times M_i], \quad (7)$$

$$M_i = \frac{(x_i^{\text{worst}} + x_i^{\text{new}})}{2}, \quad i = 1, 2, \dots, D,$$

where x^{best} is the best harmony in HM, x^{worst} is the worst harmony in HM, and $\text{rand}()$ is a uniformly distributed random number between 0 and 1.

The contribution in this section is that the mean value of population is replaced with $M_i = (x_i^{\text{worst}} + x_i^{\text{new}})/2$. There are two aspects to this apparent superiority. First, there are cheap to running cost because it do not need to compute the mean value of population in every iteration. Second, diversity of population will be enhanced more than the standard TLBO algorithm, that is because the new mean value M_i will be different for each individual, but the Mean_i in standard TLBO method is the same for every individual.

Learner Phase. In this phase, through comparing the advantages and disadvantages between the other two learners, the learner x^{new} will learn from their advantages, which draw on the idea of differential evolution algorithm. The process is as follows.

Randomly select two differential integer r_1 and r_2 from $1, 2, \dots, \text{HMS}$.

If x^{r_1} is better than x^{r_2}

$$x_i^{\text{new}} = x_i^{\text{new}} + \text{rand}() \times (x_i^{r_1} - x_i^{r_2})$$

Else

$$x_i^{\text{new}} = x_i^{\text{new}} + \text{rand}() \times (x_i^{r_2} - x_i^{r_1})$$

End If

(c) *Local Pitch Adjusting Strategy.* To achieve better solutions in the search space, it will carry out the local pitch adjusting strategy with probability PAR when the i th design variable has not performed Harmony memory consideration and Teaching-Learning Strategy as

$$x_i^{\text{new}} = x_i^{\text{new}} \pm \text{rand}() \times \text{BW}(i), \quad (8)$$

```

 $x^{new} = x^r$  ( $r = 1, 2, \dots, \text{HMS}$ ); // randomly select  $x^r$  as optimization target vector
For  $i = 1$  to  $D$  do
  If  $\text{rand}() \leq \text{HMCR}$  // (a) Harmony memory consideration
     $x_i^{new} = x_i^j$  ( $j = 1, 2, \dots, \text{HMS}$ )
  Else If  $\text{rand}() \leq \text{TLP}$  // (b) Teaching-Learning strategy
    If  $\text{rand}() \leq 0.5$ 
      // Teaching
       $x_i^{new} = x_i^{new} + \text{rand}() \times [x_i^{best} - 0.5T_F \times (x_i^{worst} + x_i^{new})]$ 
    Else
      // Learning
      Randomly select  $r_1$  and  $r_2$  from  $\{1, 2, \dots, \text{HMS}\}$ 
      If  $x^{r_1}$  is better than  $x^{r_2}$ 
         $x_i^{new} = x_i^{new} + \text{rand}() \times (x_i^{r_1} - x_i^{r_2})$ 
      Else
         $x_i^{new} = x_i^{new} + \text{rand}() \times (x_i^{r_2} - x_i^{r_1})$ 
      End
    End
     $x_i^{new} = \min(\max(x_i^{new}, x_i^L), x_i^U)$ 
  Else If  $\text{rand}(0, 1) \leq \text{PAR}$  // (c) Local pitch adjusting strategy
     $x_i^{new} = x_i^{new} \pm \text{rand}() \times \text{BW}(i)$ 
     $x_i^{new} = \min(\max(x_i^{new}, x_i^L), x_i^U)$ 
  Else If  $\text{rand}(0, 1) \leq P_m$  // (d) Random mutation operator
     $x_i^{new} = x_i^L + (x_i^U - x_i^L) \times \text{rand}()$ 
  End If
End For

```

ALGORITHM 4: The improvisation process of new harmony in HSTL algorithm.

where $\text{rand}()$ is a uniformly distributed random number between 0 and 1, and $\text{BW}(i)$ is an arbitrary distance bandwidth.

(d) *Random Mutation Operator*. As with the HS algorithm, if design variable did not perform previous actions (harmony memory consideration, teaching-learning strategy, and local pitch adjusting strategy), the HSTL method will carry out random mutation operator in feasible space with probability P_m on i th design variable of x^{new} as follows:

$$x_i^{new} = x_i^L + (x_i^U - x_i^L) \times \text{rand}(). \quad (9)$$

The improvisation of new target harmony in HSTL algorithm is shown in Algorithm 4.

The flow chart of HSTL algorithm is shown in Figure 1.

Parameters Dynamically Changed. To efficiently balance the exploration and exploitation power of the HSTL algorithm, HMCR, PAR, BW, and TLP parameters are dynamically adapted to a suitable range with the increase of generations. Equation (10) shows the dynamic change of HMCR, PAR, BW, and TLP, respectively.

Consider the following [8]:

$$\begin{aligned} \text{HMCR} &= \text{HMCR}_{\min} + (\text{HMCR}_{\max} - \text{HMCR}_{\min}) \\ &\quad \times \left(\frac{t}{T_{\max}} \right)^2, \\ \text{TLP} &= \text{TLP}_{\min} + (\text{TLP}_{\max} - \text{TLP}_{\min}) \times \left(\frac{t}{T_{\max}} \right)^k, \\ &\quad k = 5, \end{aligned}$$

$$\text{PAR} = \text{PAR}_{\max} - \frac{(\text{PAR}_{\max} - \text{PAR}_{\min}) \times t}{T_{\max}},$$

$$\text{BW} = \text{BW}_{\max} + \exp \left[\ln \left(\frac{\text{BW}_{\min}}{\text{BW}_{\max}} \right) \times \sqrt{\frac{t}{T_{\max}}} \right]. \quad (10)$$

4. Numerical Experiments and Results

4.1. Test Functions. The test functions F_1 – F_{23} [27–29] are shown in Table 1, and F_{24} – F_{31} are demonstrated. Table 2 in Functions F_{24} – F_{31} [30] are hybrid composition functions. The hybrid composition functions are built combining a non-separable (NS) function with other functions. The considered functions are as follows.

(i) Nonseparable functions:

- (a) F_{15} : shifted Rosenbrock's function,
- (b) F_{17} : shifted Griewank's function,
- (c) NS- F_{21} : nonshifted Extended F_{10} ,
- (d) NS- F_{22} : nonshifted Bohachevsky.

(ii) Other component functions:

- (a) F_{13} : shifted sphere function,
- (b) F_{16} : shifted Rastrigin function,
- (c) F_{20} : Schwefel2.22 function.

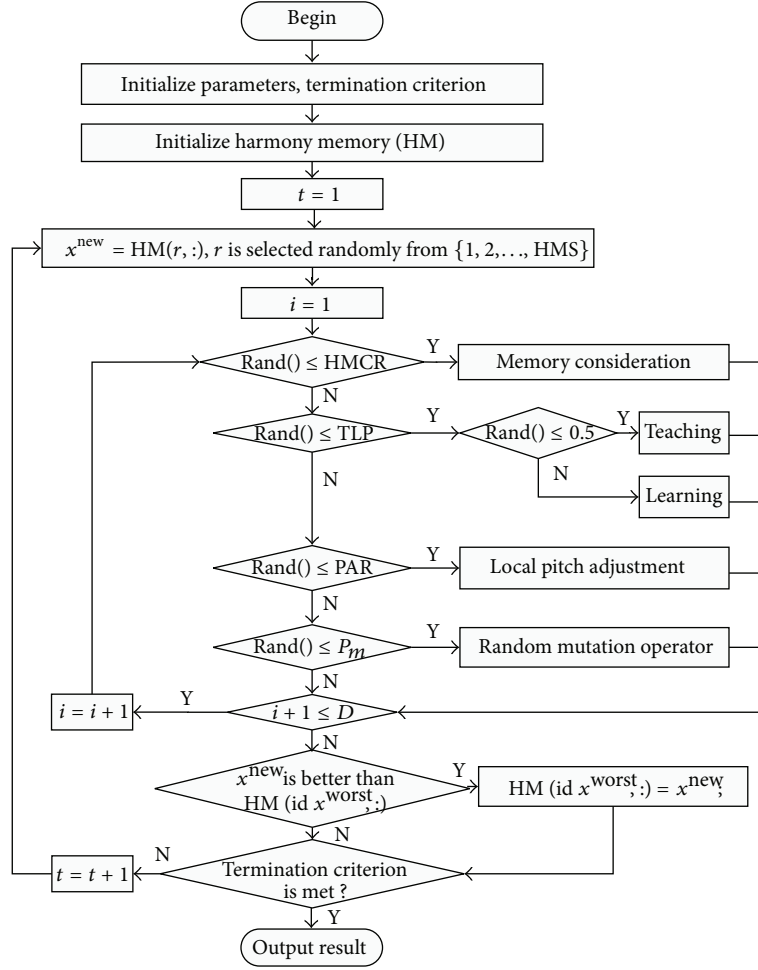


FIGURE 1: The flowchart of HSTL algorithm.

Hybrid function $F_{ns} \oplus F'(S)$ is composed of a nonseparable function F_{ns} and other function F' . The hybrid procedure is shown as follows. (1) S is divided into two parts ($part_1$ and $part_2$) as follows

If $m_{ns} \leq 0.5$ then

$part_1$ is composed by the first $D \cdot m_{ns}$ even variables. (length ($part_1$) = $D \cdot m_{ns}$)

$part_2$ is composed by the remaining variables. (length ($part_2$) = $D - \text{length} (part_1)$)

Else

$part_2$ is composed by the first $D \cdot (1 - m_{ns})$ odd variables. (length ($part_2$) = $D \cdot (1 - m_{ns})$)

$part_1$ is composed by the remaining variables. (length ($part_1$) = $D - \text{length} (part_2)$)

End If

(2) Return $F_{ns} (part_1) \oplus F'(part_2)$.

We have used 31 well-known unconstrained benchmark functions [27–30] as a testbed to evaluate the performance

of the HSTL algorithm presented in this paper. These test functions are considered as particularly challenging for many existing meta-heuristic optimization algorithms except for F_{10} . The composition of these test functions contains some characteristics, such as separable design variables, nonseparable design variables, strong unimodality, strong multimodality, and hybrid. Many of them blend different characteristics together. Table 3 shows the characteristics of all test functions.

4.2. Experimental Setup. Simulation experiments are carried out to compare the optimization (minimization) capabilities of the presented method (HSTL) with respect to (a) classical HS [1, 2], (b) SGHS [16], and (c) NGHS [13, 14]. All the experiments were performed on Windows XP 32 system with Intel(R) Core(TM) i3-2120 CPU@3.30 GHz and 2 GB RAM, and all the program codes were written in MATLAB R2009a.

In the experiments, the parameters settings for the compared HS algorithms are shown in Table 4; the dimensions of the benchmark problems are set as 50, 100, and 200, respectively. To make the comparison fair, the populations for all the competitor algorithms (for all problems tested) were initialized using the same random seeds. The variants

TABLE 1: Benchmark test functions.

Function name	Benchmark functions expression	Search range	Optimum value
F_1 : Ackley function	$F_1(x) = -20e^{-(1/5)\sqrt{(1/D)\sum_{i=1}^D x_i^2}} - e^{(1/D)\sum_{i=1}^D \cos(2\pi x_i)} + 20 - e$	$[-15, 30]^D$	$x^* = (0, 0, \dots, 0),$ $F_1(x^*) = 0$
F_2 : Dixon and Price function	$F_2(x) = (x_1 + 1)^2 + \sum_{i=2}^D i(2x_i^2 - x_{i-1})^2$	$[-10, 10]^D$	$F_2(x^*) = 0$
F_3 : Griewank function	$F_3 = (1/4000)\sum_{i=1}^D x_i^2 - \prod_{i=1}^D (\cos(x_i/\sqrt{i})) + 1$	$[-600, 600]^D$	$x^* = (0, 0, \dots, 0),$ $F_3(x^*) = 0$
F_4 : Levy function	$F_4(x) = \sin^2(\pi y_1) + \sum_{i=1}^{D-1} [(y_i - 1)^2 (1 + 10 \sin^2(\pi y_i + 1))] + (y_D - 1)^2 (1 + 10 \sin^2(2\pi y_D))$, where $y_i = 1 + ((x_i - 1)/4), i = 1, 2, \dots, D$	$[-10, 10]^D$	$x^* = (1, 1, \dots, 1),$ $F_4(x^*) = 0$
F_5 : Michalewics function	$F_5(x) = -\sum_{i=1}^D \sin(x_i) (\sin(ix_i^2/\pi))^{2m}$; $m = 10$	$[-10, \pi]^D$	$n = 5; F_5(x^*) = -4.687658$
F_6 : Powell function	$F_6(x) = \sum_{i=1}^{D/4} [(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - x_{4i-1})^2 + 10(x_{4i-3} - x_{4i})^4]$	$[-4, 5]^D$	$x^* = (3, -1, 0, 1, \dots, 3, -1, 0, 1), F_6(x^*) = 0$
F_7 : Rastrigin function	$F_7(x) = 10D + \sum_{i=1}^D (x_i^2 - 10 \cos(2\pi x_i))$	$[-5.12, 5.12]^D$	$x^* = (0, 0, \dots, 0),$ $F_7(x^*) = 0$
F_8 : Rosenbrock function	$F_8(x) = \sum_{i=1}^{D-1} [100(x_i^2 - x_{i-1})^2 + (x_i - 1)^2]$	$[-10, 10]^D$	$x^* = (1, 1, \dots, 1),$ $F_8(x^*) = 0$
F_9 : Schwefel2.26 function	$F_9(x) = 418.9829D + \sum_{i=1}^D x_i^2 \sin(\sqrt{ x_i })$	$[-512, 512]^D$	$x^* = (420.9687, 420.9687, \dots, 420.9687), F_9(x^*) = 0$
F_{10} : Sphere function	$F_{10}(x) = \sum_{i=1}^D x_i^2$	$[-100, 100]^D$	$x^* = (0, 0, \dots, 0),$ $F_{10}(x^*) = 0$
F_{11} : Trid function	$F_{11}(x) = \sum_{i=1}^D (x_i - 1)^2 - \sum_{i=2}^D x_i x_{i-1}$	$[-D^2, D^2]^D$	$D = 5; F_{11}(x^*) = -30$
F_{12} : Zakharov function	$F_{12}(x) = \sum_{i=1}^D x_i^2 + (\sum_{i=1}^D 0.5ix_i)^2 + (\sum_{i=1}^D 0.5ix_i)^4$	$[-5, 10]^D$	$x^* = (0, 0, \dots, 0),$ $F_{12}(x^*) = 0$
F_{13} : Sphere shift function	$F_{13}(x) = \sum_{i=1}^D z_i^2 + f_bias_{i3}$, $f_bias_{i3} = -450, z = x - o; o = [o_1, o_2, \dots, o_D]$	$[-100, 100]^D$	$x^* = o, F_{13}(x^*) = -450$
F_{14} : Schwefel shift function	$F_{14} = \max(z) + f_bias_{i4}$, $f_bias_{i4} = -450, z = x - o; o = [o_1, o_2, \dots, o_D]$	$[-100, 100]^D$	$x^* = o, F_{14}(x^*) = -450$
F_{15} : Rosenbrock shift function	$F_{15}(x) = \sum_{i=1}^{D-1} [100(z_i^2 - z_{i-1})^2 + (z_i - 1)^2] + f_bias_{i5}$, $f_bias_{i5} = 390, z = x - o; o = [o_1, o_2, \dots, o_D]$	$[-100, 100]^D$	$x^* = o, F_{15}(x^*) = 390$

TABLE 1: Continued.

Function name	Benchmark functions expression	Search range	Optimum value
F_{16} : Griewank shift function	$F_{16} = (1/4000) \sum_{i=1}^D z_i^2 - \prod_{i=1}^D (\cos(z_i/\sqrt{i})) + 1 + f_bias_{16}$ $f_bias_{16} = -180, z = (x - o) \times M, o = [o_1, o_2, \dots, o_D]$	$[-600, 600]^D$	$x^* = o, F_{16}(x^*) = -180$
F_{17} : Rastrigin shift function	$F_{17}(x) = 10D + \sum_{i=1}^D (z_i^2 - 10 \cos(2\pi z_i)) + f_bias_{17}$ $f_bias_{17} = -330, z = x - o; o = [o_1, o_2, \dots, o_D]$	$[-5.12, 5.12]^D$	$x^* = o, F_{17}(x^*) = -330$
F_{18} : Ackley shift function	$F_{18}(x) = -20e^{(1/5)\sqrt{(1/D)\sum_{i=1}^D z_i^2}} - e^{(1/D)\sum_{i=1}^D \cos(2\pi z_i)} + 20 - e$ $f_bias_{18} = -140, z = x - o; o = [o_1, o_2, \dots, o_D]$	$[-32, 32]^D$	$x^* = o, F_{18}(x^*) = -140$
F_{19} : Fast Fractal “Double Dip” function	$F_{19}(x) = \sum_{i=1}^D \text{fractal1D}(x_i + \text{twist}(x_{(i \bmod D)+1}))$	$[-1, 1]^D$	Unknown
	$\text{twist}(y) = 4(y^4 - 2y^3 + y^2)$		
	$\text{fractal1D}(x) \approx \sum_{k=1}^3 \sum_{i=1}^{2^{k-1}} \sum_1^{\text{ran}2(o)} \text{doubledip}(x, \text{ran } 1(o), \frac{1}{2^{k-1}(2 - \text{ran } 1(o))})$		
	$\text{doubledip}(x, c, s) = \begin{cases} c(-6144(x-c)^6 + 3088(x-c)^4 - 392(x-c)^2 + 1)s \\ 0, \text{ otherwise} \end{cases}$		
F_{20} : Schwefel2.22 function	ran 1 (o): double, pseudo randomly chosen, with seed o, with equal probability from the interval [0, 1]	$[-10, 10]^D$	$x^* = (0, 0, \dots, 0),$ $F_{20}(x^*) = 0$
	ran 2(o): integer, pseudo randomly chosen, with seed o, with equal probability from the set {0, 1, 2}		
	fractal1D(x) is an approximation to a recursive algorithm, it does not take account of wrapping at the boundaries, or local reseeding of the random generators.		
	$F_{20}(x) = \sum_{i=1}^D x_i + \prod_{i=1}^D x_i $		
F_{21} : Extended f10 shift function	$F_{21}(x) = \left(\sum_{i=1}^{D-1} f_{10}(z_i, z_{i+1}) \right) + f_{10}(z_D, z_1), \quad z = x - o$ $f_{10} = (x^2 + y^2)^{0.25} \cdot (\sin^2(50 \cdot (x^2 + y^2)^{0.1}) + 1)$	$[-100, 100]^D$	$x^* = o, F_{21}(x^*) = 0$
F_{22} : Bohachevsky function	$F_{22}(x) = \sum_{i=1}^D (z_i^2 + 2z_{i+1}^2 - 0.3 \cos(3\pi z_i) - 0.4 \cos(4\pi z_{i+1}) + 0.7), \quad z = x - o$	$[-15, 15]^D$	$x^* = o, F_{22}(x^*) = 0$
F_{23} : Schaffer shift function	$F_{23}(x) = \sum_{i=1}^{D-1} (z_i^2 + z_{i+1}^2)^{0.25} (\sin^2(50 \cdot (z_i^2 + z_{i+1}^2)^{0.1}) + 1), \quad z = x - o$	$[-100, 100]^D$	$x^* = o, F_{23}(x^*) = 0$

TABLE 2: The hybrid functions (F_{24} – F_{31}).

Function	F_{ns}	F'	m_{ns}	Range	Fitness optimum
F_{24}	NS- F_{21}	F_{13}	0.25	$[-100, 100]^D$	-450
F_{25}	NS- F_{22}	F_{15}	0.25	$[-100, 100]^D$	390
F_{26}	NS- F_{23}	F_{16}	0.25	$[-5, 5]^D$	-330
F_{27}	NS- F_{22}	NS- F_{20}	0.25	$[-10, 10]^D$	0
F_{28}	NS- F_{21}	F_{13}	0.5	$[-100, 100]^D$	-450
F_{29}	NS- F_{21}	F_{15}	0.75	$[-100, 100]^D$	390
F_{30}	NS- F_{21}	F_{16}	0.75	$[-5, 5]^D$	-330
F_{31}	NS- F_{22}	NS- F_{20}	0.75	$[-10, 10]^D$	0

TABLE 3: The characteristics of all test functions.

Function	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}	F_{16}
Unimodal (U)/multimodal (M)	M	M	M	M	M	M	M	M	M	U	U	U	U	U	M	M
Shifted (Y/N)	N	N	N	N	N	N	N	N	N	N	N	N	Y	Y	Y	Y
Separable (Y/N)	Y	N	N	Y	Y	N	Y	N	Y	Y	Y	Y	Y	N	N	Y
Hybrid (Y/N)	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N
Function	F_{17}	F_{18}	F_{19}	F_{20}	F_{21}	F_{22}	F_{23}	F_{24}	F_{25}	F_{26}	F_{27}	F_{28}	F_{29}	F_{30}	F_{31}	
Unimodal (U)/multimodal (M)	M	M	M	U	U	U	U	U	M	M	M	U	M	M	N	
Shifted (Y/N)	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	
Separable (Y/N)	N	N	N	N	N	N	N	N	N	N	N	N	N	N	N	
Hybrid (Y/N)	N	N	N	N	N	N	Y	Y	Y	Y	Y	Y	Y	Y	Y	

TABLE 4: Parameter settings for the compared HS algorithms (HS, SGHS, NGHS, HSTL).

Algorithm	HMS	HMCR	PAR	BW	Others
HS	35	0.98	0.33	$BW = (UB - LB)/1000$	
SGHS	35	$HMCR_m = 0.9$	$PAR_{max} = 0.99,$ $PAR_{min} = 0.01$	$BW_{max} = (UB - LB)/10$ $BW_{min} = 0.0005$	$LP = 100$
NGHS	35	/	/	/	$P_m = 0.01$
HSTL	35	$HMCR_{max} = 0.99$ $HMCR_{min} = 0.75$	$PAR_{max} = 0.8$ $PAR_{min} = 0.2$	$BW_{max} = (UB - LB)/20$ $BW_{min} = (UB - LB)/3500$	$TLP_{max} = 0.5,$ $TLP_{min} = 0.15$ $P_m = 0.15, k = 5$

of HS algorithm were set at the same termination criteria: the number of improvisations (function evaluation times: FEs) FEs = 500D, respectively.

The best and worst fitness values of each benchmark problem are recorded for 30 independent runs; the mean fitness, standard deviation (STD), and mean runtime of each function are calculated for 30 independent runs, and the Mean fitness, Standard Deviation (STD) and mean runtime of each function are calculated for 30 independent runs.

4.3. The Results Comparison between HSTL, HS, SGHS, and NGHS Algorithm. From Table 5, it can be found that the mean optimal fitness values of HSTL are always less than those of other 3 HS algorithms for all test functions except for F_{12} , F_{14} , F_{16} , and F_{29} when dimension are equal to 50 and

maxFEs = 25000, and all these mean optimal fitness values are slightly larger than the NGHS algorithm for F_{12} , F_{14} , F_{16} , and F_{29} . According to criteria (best, mean, worst, and STD), the over performance of HSTL method outperforms the other three HS algorithms for 31 benchmark problems except F_{12} , F_{14} , and F_{29} . Taken together, the best, mean, and STD obtained by the HSTL method are better than those of other three methods for most test functions. In addition, the runtime is less than SGHS for all problems.

In Tables 6 and 7, the results are shown for 100D and 200D problems, respectively. According to the results (best, mean, worst, and STD) of HSTL algorithm outperforms the other three harmony search algorithm. And it can be seen from Tables 6 and 7 that, compared with 50D problems, the HSTL method has more obvious advantages than other three HS algorithms (HS, SGHS, and NSGH).

TABLE 5: The optimization results of four harmony search algorithms for $F_1 - F_{31}$ ($D = 50$, FEs = 500D).

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_1	HS	2.56680E + 00	3.30204E + 00	3.88472E + 00	3.24846E - 01	5.916126E - 01
	SGHS	1.813356E + 00	2.557675E + 00	3.129460E + 00	3.60846E - 01	1.539694E + 00
	NGHS	3.447238E - 02	5.592492E - 02	8.840852E - 02	1.464912E - 02	6.187550E - 01
	HSTL	9.224783E - 04	1.775338E - 03	2.516676E - 03	3.736810E - 04	1.150359E + 00
F_2	HS	2.528891E + 01	1.303365E + 02	2.064702E + 02	4.764531E + 01	3.728292E - 01
	SGHS	2.291379E + 01	5.454353E + 01	1.405853E + 02	2.704011E + 01	1.335320E + 00
	NGHS	7.771989E - 01	5.744259E + 00	1.159369E + 01	3.130084E + 00	4.210442E - 01
	HSTL	6.666789E - 01	7.870821E - 01	1.185710E + 00	1.526289E - 01	8.835301E - 01
F_3	HS	1.498964E + 00	1.663430E + 00	1.988965E + 00	1.398770E - 01	6.367849E - 01
	SGHS	1.190177E + 00	1.475248E + 00	2.130067E + 00	2.263591E - 01	1.593317E + 00
	NGHS	4.418931E - 02	7.146802E - 02	1.559262E - 01	2.679798E - 02	6.544169E - 01
	HSTL	2.933239E - 05	8.315104E - 04	1.506785E - 02	3.351017E - 03	1.209138E + 00
F_4	HS	2.734048E - 01	3.341458E - 01	4.153356E - 01	3.844929E - 02	1.368443E + 00
	SGHS	1.417380E - 01	1.714592E + 00	3.739141E + 00	1.116863E + 00	2.390772E + 00
	NGHS	2.066964E - 04	3.193237E - 04	5.845768E - 04	9.202494E - 05	1.393585E + 00
	HSTL	4.120272E - 06	6.258354E - 06	9.738134E - 06	1.414984E - 06	2.027401E + 00
F_5	HS	-4.170235E + 01	-4.036729E + 01	-3.828157E + 01	9.770373E - 01	8.758810E - 01
	SGHS	-4.660214E + 01	-4.506838E + 01	-4.284438E + 01	9.381859E - 01	1.822759E + 00
	NGHS	-4.761973E + 01	-4.653808E + 01	-4.546623E + 01	5.188843E - 01	9.208168E - 01
	HSTL	-4.848365E + 01	-4.783102E + 01	-4.713305E + 01	3.899320E - 01	1.454742E + 00
F_6	HS	4.064420E + 00	1.023338E + 01	1.807872E + 01	4.174841E + 00	1.270949E + 00
	SGHS	3.787277E + 00	2.168953E + 01	5.037661E + 01	1.367783E + 01	2.307965E + 00
	NGHS	4.732459E - 02	9.566685E - 02	1.331749E - 01	2.187037E - 02	1.313182E + 00
	HSTL	3.072186E - 02	7.276520E - 02	1.260607E - 01	2.847471E - 02	1.955321E + 00
F_7	HS	2.373168E + 01	3.076467E + 01	3.740304E + 01	3.698242E + 00	4.206309E - 01
	SGHS	4.212332E + 01	5.047204E + 01	5.912495E + 01	5.715449E + 00	1.387887E + 00
	NGHS	4.155143E + 00	8.088647E + 00	1.126315E + 01	1.970674E + 00	4.668058E - 01
	HSTL	3.164383E - 05	3.450611E - 03	3.265254E - 02	8.899514E - 03	9.399841E - 01
F_8	HS	5.951376E + 01	4.482384E + 02	8.417699E + 02	2.411457E + 02	3.847198E - 01
	SGHS	2.024308E + 02	4.029735E + 02	1.163300E + 03	2.088718E + 02	1.346647E + 00
	NGHS	2.246655E + 01	1.232892E + 02	2.151188E + 02	5.247493E + 01	4.322725E - 01
	HSTL	4.179333E + 01	7.883687E + 01	1.109876E + 02	3.016135E + 01	9.006008E - 01
F_9	HS	1.891514E + 02	2.441957E + 02	3.147589E + 02	3.822653E + 01	4.938682E - 01
	SGHS	4.914947E + 02	1.155863E + 03	1.616206E + 03	3.125553E + 02	1.506911E + 00
	NGHS	1.177227E + 00	1.939242E + 02	5.928832E + 02	1.437114E + 02	5.429397E - 01
	HSTL	2.831025E - 03	4.800827E + 01	2.559129E + 02	8.324164E + 01	1.055470E + 00

TABLE 5: Continued.

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_{10}	HS	4.982674E + 01	7.634414E + 01	1.088877E + 02	1.384184E + 01	5.155853E - 01
	SGHS	1.941824E + 01	5.712271E + 01	1.173837E + 02	2.463712E + 01	1.483779E + 00
	NGHS	2.933452E - 02	5.495792E - 02	1.250118E - 01	2.413133E - 02	5.503947E - 01
	HSTL	5.196982E - 05	1.156867E - 04	1.822736E - 04	3.949670E - 05	1.105805E + 00
F_{11}	HS	-8.495269E + 03	2.112918E + 05	6.120867E + 05	1.507484E + 05	5.529535E - 01
	SGHS	1.495650E + 05	2.926573E + 05	4.921806E + 05	8.867408E + 04	1.522435E + 00
	NGHS	7.811770E + 02	6.724066E + 04	1.795737E + 05	4.051646E + 04	5.825563E - 01
	HSTL	3.586693E + 03	3.982094E + 04	1.129715E + 05	3.077074E + 04	1.154489E + 00
F_{12}	HS	1.808763E + 02	2.922832E + 02	3.779931E + 02	5.595106E + 01	5.278572E - 01
	SGHS	1.966065E + 02	2.939376E + 02	4.245442E + 02	6.167682E + 01	1.413886E + 00
	NGHS	2.263637E + 01	5.379553E + 01	9.021302E + 01	1.550666E + 01	5.732848E - 01
	HSTL	1.598764E + 02	2.741391E + 02	3.460420E + 02	5.654623E + 01	1.075394E + 00
F_{13}	HS	-3.982491E + 02	-3.703682E + 02	-3.359549E + 02	1.639155E + 01	1.281255E + 00
	SGHS	-4.342564E + 02	-4.020580E + 02	-3.757443E + 02	1.465006E + 01	2.355018E + 00
	NGHS	-4.499802E + 02	-4.499468E + 02	-4.498976E + 02	2.171095E - 02	1.325362E + 00
	HSTL	-4.499992E + 02	-4.499987E + 02	-4.499983E + 02	2.372395E - 04	2.063549E + 00
F_{14}	HS	-4.330907E + 02	-4.272501E + 02	-4.198265E + 02	3.339807E + 00	1.293516E + 00
	SGHS	-4.290403E + 02	-4.161920E + 02	-4.061673E + 02	6.869724E + 00	2.335600E + 00
	NGHS	-4.475734E + 02	-4.467607E + 02	-4.456620E + 02	5.729226E - 01	1.334976E + 00
	HSTL	-4.345355E + 02	-4.303615E + 02	-4.257539E + 02	2.813241E + 00	2.110056E + 00
F_{15}	HS	4.518000E + 04	1.178572E + 05	2.758509E + 05	5.318182E + 04	1.548986E + 00
	SGHS	1.546908E + 04	1.439934E + 05	1.140938E + 06	2.428257E + 05	2.663552E + 00
	NGHS	5.520145E + 02	1.096629E + 03	5.304441E + 03	1.408912E + 03	1.595152E + 00
	HSTL	4.364245E + 02	8.663389E + 02	6.714816E + 03	1.380399E + 03	2.384685E + 00
F_{16}	HS	-3.073606E + 02	-2.979182E + 02	-2.930230E + 02	3.428569E + 00	1.388468E + 00
	SGHS	-2.904807E + 02	-2.795281E + 02	-2.653651E + 02	7.917957E + 00	2.457957E + 00
	NGHS	-3.247746E + 02	-3.221057E + 02	-3.177697E + 02	2.003994E + 00	1.425334E + 00
	HSTL	-3.257734E + 02	-3.194510E + 02	-3.131576E + 02	3.257068E + 00	2.154368E + 00
F_{17}	HS	-1.784876E + 02	-1.782673E + 02	-1.779930E + 02	1.484480E - 01	1.920537E + 00
	SGHS	-1.788319E + 02	-1.785108E + 02	-1.778547E + 02	2.823212E - 01	3.026930E + 00
	NGHS	-1.799673E + 02	-1.799278E + 02	-1.798487E + 02	2.657279E - 02	1.936843E + 00
	HSTL	-1.799993E + 02	-1.799977E + 02	-1.799891E + 02	2.941948E - 03	2.661228E + 00
F_{18}	HS	-1.373170E + 02	-1.367418E + 02	-1.362105E + 02	3.320310E - 01	1.538989E + 00
	SGHS	-1.380207E + 02	-1.373137E + 02	-1.363707E + 02	3.842756E - 01	2.631186E + 00
	NGHS	-1.399544E + 02	-1.399264E + 02	-1.399026E + 02	1.478740E - 02	1.556766E + 00
	HSTL	-1.399944E + 02	-1.399932E + 02	-1.399923E + 02	4.802592E - 04	2.308382E + 00

TABLE 5: Continued.

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_{19}	HS	-7.783258E + 02	-7.574243E + 02	-7.440193E + 02	9.549819E + 00	3.364700E + 00
	SGHS	-7.977925E + 02	-7.830542E + 02	-7.648281E + 02	8.315008E + 00	4.449306E + 00
	NGHS	-8.126267E + 02	-7.983609E + 02	-7.807750E + 02	7.476016E + 00	3.453860E + 00
	HSTL	-8.13893E + 02	-8.054826E + 02	-7.933937E + 02	5.25994E + 00	4.110440E + 00
F_{20}	HS	3.235346E + 01	4.619696E + 01	6.603604E + 01	9.209638E + 00	6.108355E - 01
	SGHS	1.840610E + 01	2.931223E + 01	4.219485E + 01	6.705792E + 00	1.656616E + 00
	NGHS	1.023646E + 00	1.326115E + 00	1.638720E + 00	1.801950E - 01	6.740073E - 01
	HSTL	1.666479E - 02	2.377405E - 02	3.561417E - 02	3.989928E - 03	1.340302E + 00
F_{21}	HS	1.039417E + 02	1.371442E + 02	1.606956E + 02	1.457780E + 01	1.692515E + 01
	SGHS	6.337241E + 01	8.847938E + 01	1.032481E + 02	9.606610E + 00	1.883071E + 01
	NGHS	2.804776E + 01	3.706996E + 01	4.978956E + 01	6.433861E + 00	1.691865E + 01
	HSTL	4.874162E + 00	5.839427E + 00	7.688011E + 00	8.479931E - 01	1.766882E + 01
F_{22}	HS	2.466530E + 01	3.127048E + 01	3.564485E + 01	3.212730E + 00	1.153501E + 01
	SGHS	1.342097E + 01	2.030578E + 01	2.459678E + 01	3.326658E + 00	1.349578E + 01
	NGHS	4.017501E - 02	7.928514E - 02	1.477305E - 01	3.097912E - 02	1.160670E + 01
	HSTL	8.611143E - 04	2.201951E - 02	4.140857E - 01	9.228328E - 02	1.214935E + 01
F_{23}	HS	1.167773E + 02	1.315135E + 02	1.534918E + 02	1.135387E + 01	1.264720E + 01
	SGHS	7.270431E + 01	8.324425E + 01	1.008921E + 02	6.879689E + 00	1.442205E + 01
	NGHS	2.477052E + 01	3.758657E + 01	5.407804E + 01	7.976231E + 00	1.312881E + 01
	HSTL	4.038784E + 00	6.052038E + 00	1.261065E + 01	1.761817E + 00	1.385192E + 01
F_{24}	HS	-4.217519E + 02	-4.046735E + 02	-3.782667E + 02	1.257151E + 01	1.359617E + 01
	SGHS	-4.431356E + 02	-4.312111E + 02	-3.858681E + 02	1.369453E + 01	1.555383E + 01
	NGHS	-4.499996E + 02	-4.499993E + 02	-4.499982E + 02	3.724625E - 04	1.364959E + 01
	HSTL	-4.499996E + 02	-4.499994E + 02	-4.499991E + 02	1.730785E - 04	1.440449E + 01
F_{25}	HS	2.665579E + 04	6.003134E + 04	1.020993E + 05	2.295715E + 04	1.384424E + 01
	SGHS	6.783432E + 03	3.423766E + 04	1.447178E + 05	3.728901E + 04	1.574535E + 01
	NGHS	4.838503E + 02	5.491368E + 02	6.308092E + 02	4.002266E + 01	1.387828E + 01
	HSTL	4.283814E + 02	5.425790E + 02	1.213942E + 03	1.982288E + 02	1.433272E + 01
F_{26}	HS	-3.056734E + 02	-3.005925E + 02	-2.916669E + 02	3.782453E + 00	1.339199E + 01
	SGHS	-2.924154E + 02	-2.838442E + 02	-2.713927E + 02	6.793276E + 00	1.527401E + 01
	NGHS	-3.267093E + 02	-3.228070E + 02	-3.183256E + 02	2.244713E + 00	1.341534E + 01
	HSTL	-3.289412E + 02	-3.236531E + 02	-3.188403E + 02	2.872375E + 00	1.421595E + 01
F_{27}	HS	4.849375E + 00	6.498510E + 00	8.495851E + 00	1.017495E + 00	1.262153E + 01
	SGHS	1.619215E + 00	4.090404E + 00	6.379225E + 00	1.193628E + 00	1.455011E + 01
	NGHS	6.034321E - 02	8.531001E - 02	1.167611E - 01	1.717796E - 02	1.266599E + 01
	HSTL	8.329468E - 03	9.918090E - 03	1.162367E - 02	1.094430E - 03	1.344353E + 01

TABLE 5: Continued.

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_{28}	HS	-4.495086E + 02	-4.478733E + 02	-4.443561E + 02	1.460489E + 00	1.617566E + 01
	SGHS	-4.499731E + 02	-4.498848E + 02	-4.497073E + 02	8.308225E - 02	1.816665E + 01
	NGHS	-4.500000E + 02	-4.500000E + 02	-4.500000E + 02	5.050670E - 14	1.633242E + 01
	HSTL	-4.500000E + 02	-4.500000E + 02	-4.500000E + 02	3.894074E - 06	1.700242E + 01
F_{29}	HS	7.984189E + 02	5.231460E + 03	2.285396E + 04	5.423033E + 03	1.646778E + 01
	SGHS	6.857379E + 02	2.714991E + 03	1.007518E + 04	2.839634E + 03	1.841738E + 01
	NGHS	4.290961E + 02	5.562601E + 02	1.377085E + 03	2.419459E + 02	1.656795E + 01
	HSTL	4.073724E + 02	1.159148E + 03	9.324027E + 03	2.181373E + 03	1.728557E + 01
F_{30}	HS	-3.077579E + 02	-3.023452E + 02	-2.973287E + 02	3.007979E + 00	1.629385E + 01
	SGHS	-3.057815E + 02	-2.971863E + 02	-2.874010E + 02	4.692703E + 00	1.819898E + 01
	NGHS	-3.260430E + 02	-3.234723E + 02	-3.198728E + 02	1.579631E + 00	1.644573E + 01
	HSTL	-3.295820E + 02	-3.280941E + 02	-3.260903E + 02	1.159512E + 00	1.714851E + 01
F_{31}	HS	1.196768E + 01	1.679724E + 01	2.010432E + 01	2.439712E + 00	1.279175E + 01
	SGHS	4.187297E + 00	9.601911E + 00	1.575427E + 01	3.206880E + 00	1.473686E + 01
	NGHS	2.868378E - 02	4.841433E - 02	9.156237E - 02	1.566071E - 02	1.283577E + 01
	HSTL	2.103222E - 03	2.808338E - 03	3.640376E - 03	4.063424E - 04	1.358129E + 01

TABLE 6: The optimization results of four harmony search algorithms for $F_1 - F_{31}$ ($D = 100$, FEs = 500D).

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_1	HS	3.472069E + 00	3.820833E + 00	4.047963E + 00	1.596325E - 01	1.769219E + 00
	SGHS	3.132910E + 00	3.913786E + 00	5.083229E + 00	5.509827E - 01	4.010046E + 00
	NGHS	1.444919E + 00	1.767520E + 00	2.084939E + 00	1.790358E - 01	1.750870E + 00
	HSTL	2.287774E - 03	2.786403E - 03	3.224305E - 03	2.716016E - 04	3.008579E + 00
F_2	HS	3.167536E + 02	6.351839E + 02	1.255875E + 03	2.510796E + 02	1.193827E + 00
	SGHS	3.589095E + 02	1.471320E + 03	3.982050E + 03	1.068029E + 03	3.400621E + 00
	NGHS	3.974602E + 01	6.225108E + 01	8.429758E + 01	1.057944E + 01	1.243040E + 00
	HSTL	6.668194E - 01	1.286474E + 00	7.749294E + 00	1.726882E + 00	2.346918E + 00
F_3	HS	2.603003E + 00	3.327735E + 00	3.759840E + 00	2.519202E - 01	1.920741E + 00
	SGHS	2.174329E + 00	4.132405E + 00	6.281051E + 00	1.173173E + 00	4.125897E + 00
	NGHS	1.188965E + 00	1.242508E + 00	1.340110E + 00	3.865128E - 02	1.920139E + 00
	HSTL	1.376879E - 04	2.346120E - 04	5.029240E - 04	7.730570E - 05	3.209074E + 00
F_4	HS	8.948965E - 01	1.142211E + 00	1.519557E + 00	1.440158E - 01	5.019303E + 00
	SGHS	5.690498E + 00	1.420662E + 01	2.458487E + 01	5.224712E + 00	7.419859E + 00
	NGHS	9.494633E - 02	1.460202E - 01	2.463073E - 01	3.355741E - 02	5.010614E + 00
	HSTL	1.635147E - 05	1.380600E - 04	2.370741E - 03	5.255262E - 04	6.668101E + 00
F_5	HS	-6.672187E + 01	-6.350148E + 01	-6.043354E + 01	2.052348E + 00	2.969232E + 00
	SGHS	-8.968012E + 01	-8.648817E + 01	-8.264598E + 01	1.813756E + 00	5.086223E + 00
	NGHS	-8.313566E + 01	-8.029515E + 01	-7.690579E + 01	1.438337E + 00	2.968101E + 00
	HSTL	-9.585843E + 01	-9.464151E + 01	-9.352011E + 01	5.668992E - 01	4.289409E + 00
F_6	HS	2.275081E + 01	3.404998E + 01	4.701336E + 01	7.463806E + 00	4.601501E + 00
	SGHS	6.202169E + 01	1.021203E + 02	1.576931E + 02	2.604590E + 01	7.029177E + 00
	NGHS	2.559237E + 00	4.248504E + 00	7.194444E + 00	1.263185E + 00	4.571996E + 00
	HSTL	1.159057E - 01	2.164533E - 01	3.013113E - 01	4.862855E - 02	6.357763E + 00
F_7	HS	8.761768E + 01	9.968145E + 01	1.095680E + 02	5.125492E + 00	1.397587E + 00
	SGHS	1.204523E + 02	1.515871E + 02	1.806614E + 02	1.516934E + 01	3.643591E + 00
	NGHS	5.302534E + 01	7.188578E + 01	8.426157E + 01	7.828465E + 00	1.433040E + 00
	HSTL	1.577470E - 04	9.978969E - 02	9.954242E - 01	3.062646E - 01	2.611144E + 00
F_8	HS	5.880175E + 02	1.300531E + 03	2.082232E + 03	3.831144E + 02	1.223890E + 00
	SGHS	8.999245E + 02	1.790814E + 03	2.911281E + 03	5.768125E + 02	3.494810E + 00
	NGHS	3.079063E + 02	4.138027E + 02	5.638547E + 02	7.193641E + 01	1.255683E + 00
	HSTL	9.318870E + 01	1.469698E + 02	2.571664E + 02	4.541752E + 01	2.443950E + 00
F_9	HS	7.122610E + 02	8.396009E + 02	9.466086E + 02	6.566372E + 01	1.529895E + 00
	SGHS	3.806886E + 03	4.736018E + 03	6.005643E + 03	6.494470E + 02	3.877912E + 00
	NGHS	6.828367E + 02	1.783040E + 03	2.842955E + 03	4.795603E + 02	1.569362E + 00
	HSTL	7.968585E + 00	1.720210E + 02	3.973356E + 02	1.098869E + 02	2.854428E + 00

TABLE 6: Continued.

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_{10}	HS	2.129878E + 02	2.699577E + 02	3.362863E + 02	3.021370E + 01	1.482740E + 00
	SGHS	1.475702E + 02	3.215163E + 02	5.256382E + 02	1.308856E + 02	3.719739E + 00
	NGHS	1.599879E + 01	2.545197E + 01	3.908629E + 01	5.800758E + 00	1.488095E + 00
	HSTL	2.404241E - 04	3.881772E - 04	5.946775E - 04	9.559413E - 05	2.861329E + 00
F_{11}	HS	3.939318E + 06	7.748172E + 06	1.544812E + 07	2.861829E + 06	1.567541E + 00
	SGHS	1.156049E + 07	2.205330E + 07	3.666560E + 07	7.051009E + 06	3.787745E + 00
	NGHS	3.799257E + 06	6.908533E + 06	9.606363E + 06	1.677867E + 06	1.569956E + 00
	HSTL	3.921902E + 05	1.520828E + 06	3.598060E + 06	9.375356E + 05	2.978233E + 00
F_{12}	HS	7.150539E + 02	9.583770E + 02	1.225170E + 03	1.259891E + 02	1.530910E + 00
	SGHS	7.715070E + 02	9.708888E + 02	1.242580E + 03	1.201418E + 02	3.590301E + 00
	NGHS	2.108479E + 02	2.650550E + 02	3.204091E + 02	3.392485E + 01	1.553830E + 00
	HSTL	6.766070E + 02	8.735240E + 02	1.063928E + 03	1.027158E + 02	2.874153E + 00
F_{13}	HS	-2.375559E + 02	-1.738990E + 02	-1.079270E + 02	3.473333E + 01	3.058755E + 00
	SGHS	-3.131437E + 02	-2.097408E + 02	-8.276031E + 01	7.569102E + 01	5.492338E + 00
	NGHS	-4.332349E + 02	-4.262997E + 02	-4.192761E + 02	3.856018E + 00	3.064137E + 00
	HSTL	-4.499970E + 02	-4.499959E + 02	-4.499951E + 02	5.433935E - 04	4.894014E + 00
F_{14}	HS	-4.232215E + 02	-4.193171E + 02	-4.108272E + 02	2.989275E + 00	3.133809E + 00
	SGHS	-3.977762E + 02	-3.880149E + 02	-3.766792E + 02	5.980155E + 00	5.474201E + 00
	NGHS	-4.402313E + 02	-4.377563E + 02	-4.360122E + 02	1.032261E + 00	3.189262E + 00
	HSTL	-4.245886E + 02	-4.181521E + 02	-4.133119E + 02	3.501804E + 00	4.858567E + 00
F_{15}	HS	3.297507E + 05	5.050010E + 05	7.154538E + 05	1.134203E + 05	3.715503E + 00
	SGHS	2.141247E + 05	2.473414E + 06	9.215806E + 06	2.312600E + 06	6.170230E + 00
	NGHS	6.173097E + 03	1.336579E + 04	2.527368E + 04	6.553278E + 03	3.722490E + 00
	HSTL	6.679117E + 02	2.123307E + 03	6.663690E + 03	1.898390E + 03	5.447209E + 00
F_{16}	HS	-2.417986E + 02	-2.316766E + 02	-2.243841E + 02	4.720329E + 00	3.407872E + 00
	SGHS	-1.978875E + 02	-1.657988E + 02	-1.304433E + 02	1.954773E + 01	5.825132E + 00
	NGHS	-2.700495E + 02	-2.575340E + 02	-2.441387E + 02	7.491290E + 00	3.388501E + 00
	HSTL	-3.134902E + 02	-3.059890E + 02	-2.968384E + 02	4.108868E + 00	5.092343E + 00
F_{17}	HS	-1.769331E + 02	-1.764810E + 02	-1.760164E + 02	2.480479E - 01	5.401888E + 00
	SGHS	-1.779920E + 02	-1.770352E + 02	-1.750318E + 02	8.297157E - 01	7.875262E + 00
	NGHS	-1.788120E + 02	-1.787740E + 02	-1.787241E + 02	2.570359E - 02	5.350236E + 00
	HSTL	-1.799984E + 02	-1.799963E + 02	-1.799876E + 02	3.066164E - 03	7.034552E + 00
F_{18}	HS	-1.365537E + 02	-1.361589E + 02	-1.357947E + 02	2.159838E - 01	3.742110E + 00
	SGHS	-1.369490E + 02	-1.362044E + 02	-1.344587E + 02	6.516864E - 01	6.201829E + 00
	NGHS	-1.385604E + 02	-1.381231E + 02	-1.377560E + 02	2.656460E - 01	3.702051E + 00
	HSTL	-1.399928E + 02	-1.399894E + 02	-1.399455E + 02	1.034768E - 02	5.419278E + 00

TABLE 6: Continued.

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_{19}	HS	-1.397521E + 03	-1.360864E + 03	-1.328214E + 03	2.038863E + 01	1.091921E + 01
	SGHS	-1.452010E + 03	-1.426324E + 03	-1.397075E + 03	1.631505E + 01	1.335878E + 01
	NGHS	-1.451444E + 03	-1.428229E + 03	-1.393381E + 03	1.340884E + 01	1.124146E + 01
	HSTL	-1.513615E + 03	-1.499926E + 03	-1.485147E + 03	8.517545E + 00	1.261544E + 01
F_{20}	HS	1.193796E + 02	1.193796E + 02	1.193796E + 02	1.193796E + 02	1.193796E + 02
	SGHS	9.824839E + 01	9.824839E + 01	9.824839E + 01	9.824839E + 01	9.824839E + 01
	NGHS	3.473853E + 01	3.473853E + 01	3.473853E + 01	3.473853E + 01	3.473853E + 01
	HSTL	6.151721E - 02	6.151721E - 02	6.151721E - 02	6.151721E - 02	6.151721E - 02
F_{21}	HS	2.731607E + 02	3.018028E + 02	3.309987E + 02	1.465698E + 01	5.400367E + 01
	SGHS	2.061790E + 02	2.668907E + 02	3.219105E + 02	3.133660E + 01	5.782494E + 01
	NGHS	1.520321E + 02	1.848391E + 02	2.324962E + 02	1.945112E + 01	5.352526E + 01
	HSTL	1.216583E + 01	1.522370E + 01	1.868875E + 01	2.164395E + 00	5.544992E + 01
F_{22}	HS	7.035058E + 01	7.966205E + 01	8.675826E + 01	4.501205E + 00	3.134446E + 01
	SGHS	4.845036E + 01	7.002079E + 01	1.031254E + 02	1.472650E + 01	3.551839E + 01
	NGHS	1.890592E + 01	2.264589E + 01	2.708944E + 01	2.757052E + 00	3.129377E + 01
	HSTL	3.312806E - 03	4.788330E - 01	2.104085E + 00	7.191299E - 01	3.285056E + 01
F_{23}	HS	2.474923E + 02	2.926433E + 02	3.154565E + 02	1.633700E + 01	3.787547E + 01
	SGHS	2.077105E + 02	2.618490E + 02	3.486951E + 02	3.410635E + 01	4.192231E + 01
	NGHS	1.600018E + 02	1.868942E + 02	2.193507E + 02	1.802714E + 01	3.742503E + 01
	HSTL	1.226995E + 01	1.528464E + 01	2.013004E + 01	2.366879E + 00	3.842275E + 01
F_{24}	HS	-3.539791E + 02	-3.147048E + 02	-2.647908E + 02	1.949672E + 01	3.449387E + 01
	SGHS	-4.095286E + 02	-3.591410E + 02	-2.333805E + 02	5.324971E + 01	3.871345E + 01
	NGHS	-4.493291E + 02	-4.489227E + 02	-4.485923E + 02	2.120875E - 01	3.486910E + 01
	HSTL	-4.499990E + 02	-4.499985E + 02	-4.499980E + 02	2.414762E - 04	3.688151E + 01
F_{25}	HS	1.051141E + 05	1.874186E + 05	2.937699E + 05	4.432322E + 04	3.544980E + 01
	SGHS	5.004828E + 04	6.949195E + 05	3.958335E + 06	9.272106E + 05	3.953879E + 01
	NGHS	1.111344E + 03	1.579429E + 03	4.609411E + 03	7.698066E + 02	3.568322E + 01
	HSTL	4.768735E + 02	8.024700E + 02	2.168501E + 03	4.225175E + 02	3.736183E + 01
F_{26}	HS	-2.598286E + 02	-2.502187E + 02	-2.401834E + 02	5.848484E + 00	3.371420E + 01
	SGHS	-2.231800E + 02	-1.912732E + 02	-1.659675E + 02	1.303039E + 01	3.722651E + 01
	NGHS	-2.870567E + 02	-2.734730E + 02	-2.629992E + 02	5.510929E + 00	3.296890E + 01
	HSTL	-3.207702E + 02	-3.136777E + 02	-3.092977E + 02	2.615418E + 00	3.492102E + 01
F_{27}	HS	1.761478E + 01	1.978397E + 01	2.313474E + 01	1.461489E + 00	3.122492E + 01
	SGHS	1.215437E + 01	2.186805E + 01	3.511913E + 01	5.628417E + 00	3.560859E + 01
	NGHS	2.810761E + 00	3.983730E + 00	4.655769E + 00	4.301760E - 01	3.140541E + 01
	HSTL	2.089110E - 02	2.831916E - 02	5.808363E - 02	8.512059E - 03	3.310728E + 01

TABLE 6: Continued.

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_{28}	HS	-4.469079E + 02	-4.443488E + 02	-4.405620E + 02	1.967662E + 00	4.318177E + 01
	SGHS	-4.494424E + 02	-4.483361E + 02	-4.463611E + 02	1.058420E + 00	4.732766E + 01
	NGHS	-4.500000E + 02	-4.500000E + 02	-4.500000E + 02	3.611281E - 10	4.326785E + 01
	HSTL	-4.500000E + 02	-4.500000E + 02	-4.500000E + 02	9.511311E - 06	4.509948E + 01
F_{29}	HS	6.093107E + 03	1.397065E + 04	2.184820E + 04	1.114053E + 04	4.387041E + 01
	SGHS	8.168591E + 03	9.147798E + 03	1.012700E + 04	1.384807E + 03	4.815433E + 01
	NGHS	6.891292E + 02	7.565447E + 02	8.239602E + 02	9.533993E + 01	4.375258E + 01
	HSTL	4.954684E + 02	5.024741E + 02	5.094799E + 02	9.907604E + 00	4.518065E + 01
F_{30}	HS	-2.708213E + 02	-2.658142E + 02	-2.553643E + 02	4.447667E + 00	4.291836E + 01
	SGHS	-2.498452E + 02	-2.356727E + 02	-2.214623E + 02	6.894224E + 00	4.714809E + 01
	NGHS	-2.967689E + 02	-2.895710E + 02	-2.794822E + 02	4.509198E + 00	4.315103E + 01
	HSTL	-3.281517E + 02	-3.254159E + 02	-3.216770E + 02	1.727024E + 00	4.476932E + 01
F_{31}	HS	3.744918E + 01	4.335719E + 01	4.893459E + 01	3.361402E + 00	3.044300E + 01
	SGHS	2.842099E + 01	4.402938E + 01	6.465355E + 01	8.002441E + 00	3.374892E + 01
	NGHS	5.436447E + 00	8.517679E + 00	1.110289E + 01	1.334154E + 00	2.980321E + 01
	HSTL	7.377615E - 03	8.118152E - 02	4.421143E - 01	1.112750E - 01	3.135758E + 01

TABLE 7: The optimization results of four harmony search algorithms for $F_1 - F_{31}$ ($D = 200$, FEs = 500D).

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_1	HS	4.603726E + 00	4.929354E + 00	5.266226E + 00	1.613707E - 01	5.715312E + 00
	SGHS	4.421737E + 00	5.637606E + 00	7.262746E + 00	8.839935E - 01	1.138709E + 01
	NGHS	5.145771E + 00	5.521262E + 00	5.794585E + 00	1.715834E - 01	5.721544E + 00
	HSTL	3.575306E - 03	4.026710E - 03	4.538061E - 03	2.396758E - 04	8.701444E + 00
F_2	HS	4.078042E + 03	6.153632E + 03	8.488061E + 03	1.312734E + 03	4.147045E + 00
	SGHS	7.148166E + 03	3.459359E + 04	1.047654E + 05	2.687689E + 04	9.907156E + 00
	NGHS	2.696362E + 03	4.466500E + 03	5.439083E + 03	6.579299E + 02	4.042910E + 00
	HSTL	6.688204E - 01	9.705961E + 00	5.070641E + 01	1.456647E + 01	7.049820E + 00
F_3	HS	9.161714E + 00	1.244639E + 01	1.459380E + 01	1.320806E + 00	6.456437E + 00
	SGHS	8.285825E + 00	1.792049E + 01	5.102385E + 01	9.342384E + 00	1.205154E + 01
	NGHS	1.653665E + 01	1.945936E + 01	2.288375E + 01	1.916412E + 00	6.373141E + 00
	HSTL	3.399743E - 04	1.932473E - 03	2.008061E - 02	4.825732E - 03	9.435844E + 00
F_4	HS	3.848564E + 00	4.708132E + 00	5.294488E + 00	3.531660E - 01	1.828390E + 01
	SGHS	4.320375E + 01	6.698858E + 01	9.905690E + 01	1.803405E + 01	2.448338E + 01
	NGHS	7.186447E + 00	8.321538E + 00	9.657602E + 00	6.923441E - 01	1.812159E + 01
	HSTL	3.729585E - 02	1.541456E - 01	2.844556E - 01	7.373455E - 02	2.201131E + 01
F_5	HS	-9.228155E + 01	-8.912511E + 01	-8.545813E + 01	2.196104E + 00	1.075750E + 01
	SGHS	-1.691553E + 02	-1.632416E + 02	-1.554168E + 02	4.230945E + 00	1.607256E + 01
	NGHS	-1.266981E + 02	-1.212701E + 02	-1.165799E + 02	2.487909E + 00	1.057139E + 01
	HSTL	-1.879057E + 02	-1.859165E + 02	-1.841057E + 02	9.032725E - 01	1.379854E + 01
F_6	HS	1.059685E + 02	1.605423E + 02	2.171595E + 02	3.043869E + 01	1.913746E + 01
	SGHS	2.877414E + 02	5.597265E + 02	1.076033E + 03	1.825775E + 02	2.544270E + 01
	NGHS	1.476215E + 02	1.815020E + 02	2.722819E + 02	2.829572E + 01	1.891687E + 01
	HSTL	4.134323E - 01	8.303796E - 01	1.245398E + 00	1.952532E - 01	2.286965E + 01
F_7	HS	3.049237E + 02	3.272066E + 02	3.630817E + 02	1.403316E + 01	5.018255E + 00
	SGHS	3.796419E + 02	4.822636E + 02	5.650928E + 02	4.455035E + 01	1.075236E + 01
	NGHS	4.522705E + 02	5.209768E + 02	5.738802E + 02	3.656850E + 01	4.953818E + 00
	HSTL	1.658281E - 03	1.295575E + 00	3.005119E + 00	1.058554E + 00	7.836046E + 00
F_8	HS	3.896266E + 03	5.289774E + 03	7.844040E + 03	1.015647E + 03	4.297905E + 00
	SGHS	6.818663E + 03	1.570204E + 04	2.975647E + 04	8.010501E + 03	1.001014E + 01
	NGHS	3.307225E + 03	4.050635E + 03	5.045072E + 03	4.439356E + 02	4.207027E + 00
	HSTL	1.929455E + 02	2.304209E + 02	3.286657E + 02	4.180170E + 01	7.172235E + 00
F_9	HS	3.233837E + 03	3.714921E + 03	4.247033E + 03	2.604085E + 02	5.199977E + 00
	SGHS	1.175294E + 04	1.460607E + 04	1.810298E + 04	1.831608E + 03	1.119387E + 01
	NGHS	1.228090E + 04	1.400627E + 04	1.550115E + 04	8.852628E + 02	5.119688E + 00
	HSTL	1.550538E + 03	1.951142E + 03	2.716798E + 03	3.492572E + 02	8.289662E + 00

TABLE 7: Continued.

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_{10}	HS	1.035974E + 03	1.224840E + 03	1.408721E + 03	9.946126E + 01	4.635011E + 00
	SGHS	9.842122E + 02	1.758492E + 03	3.754911E + 03	7.377144E + 02	1.040168E + 01
	NGHS	1.861439E + 03	2.058453E + 03	2.504462E + 03	1.837061E + 02	4.467012E + 00
	HSTL	1.314733E - 03	1.760664E - 03	2.393748E - 03	2.837635E - 04	7.936856E + 00
F_{11}	HS	3.073770E + 08	5.529318E + 08	7.578658E + 08	1.318618E + 08	5.023786E + 00
	SGHS	8.707673E + 08	1.161818E + 09	1.767454E + 09	2.629050E + 08	1.075328E + 01
	NGHS	7.093037E + 08	9.105403E + 08	1.190901E + 09	1.208984E + 08	4.824641E + 00
	HSTL	2.965758E + 07	8.970849E + 07	1.931661E + 08	3.740898E + 07	8.290714E + 00
F_{12}	HS	2.178914E + 03	2.482882E + 03	2.815688E + 03	1.988376E + 02	4.845730E + 00
	SGHS	2.308245E + 03	2.584773E + 03	2.953531E + 03	2.051620E + 02	1.019302E + 01
	NGHS	9.284727E + 02	1.072448E + 03	1.250569E + 03	9.258004E + 01	4.699592E + 00
	HSTL	1.938323E + 03	2.308801E + 03	2.714633E + 03	1.905069E + 02	8.051031E + 00
F_{13}	HS	4.977364E + 02	8.339352E + 02	1.058941E + 03	1.248060E + 02	8.006508E + 00
	SGHS	2.137557E + 02	1.083748E + 03	2.437995E + 03	5.816088E + 02	1.404649E + 01
	NGHS	1.392416E + 03	1.708667E + 03	1.994529E + 03	1.697142E + 02	7.789651E + 00
	HSTL	-3.915048E + 02	-3.394840E + 02	-2.719662E + 02	2.748671E + 01	1.184497E + 01
F_{14}	HS	-4.084859E + 02	-4.048983E + 02	-3.994353E + 02	2.724485E + 00	8.170102E + 00
	SGHS	-3.756669E + 02	-3.637827E + 02	-3.512079E + 02	6.398155E + 00	1.406004E + 01
	NGHS	-4.248494E + 02	-4.217489E + 02	-4.171032E + 02	1.942003E + 00	7.962233E + 00
	HSTL	-4.078666E + 02	-4.041830E + 02	-4.006726E + 02	1.886381E + 00	1.207676E + 01
F_{15}	HS	2.261680E + 06	4.109099E + 06	5.954376E + 06	9.167839E + 05	9.762151E + 00
	SGHS	6.929055E + 06	5.330388E + 07	2.498381E + 08	5.898978E + 07	1.596695E + 01
	NGHS	5.823986E + 06	7.816998E + 06	1.131635E + 07	1.505165E + 06	9.593909E + 00
	HSTL	2.775018E + 05	5.853541E + 05	9.776046E + 05	2.019841E + 05	1.376301E + 01
F_{16}	HS	-2.948123E + 01	-5.791258E - 01	1.683795E + 01	1.195259E + 01	9.144801E + 00
	SGHS	7.067304E + 01	1.786652E + 02	2.626093E + 02	5.419678E + 01	1.508770E + 01
	NGHS	1.593170E + 02	1.986273E + 02	2.485464E + 02	2.374692E + 01	8.921733E + 00
	HSTL	-2.547415E + 02	-2.353550E + 02	-2.151499E + 02	9.688531E + 00	1.285946E + 01
F_{17}	HS	-1.685846E + 02	-1.671598E + 02	-1.654225E + 02	8.709843E - 01	1.678058E + 01
	SGHS	-1.751726E + 02	-1.626560E + 02	-1.461760E + 02	7.197573E + 00	2.278318E + 01
	NGHS	-1.634951E + 02	-1.604326E + 02	-1.574177E + 02	1.704898E + 00	1.650759E + 01
	HSTL	-1.783495E + 02	-1.781095E + 02	-1.778417E + 02	1.456349E - 01	2.036008E + 01
F_{18}	HS	-1.354532E + 02	-1.350248E + 02	-1.346804E + 02	1.775738E - 01	9.748667E + 00
	SGHS	-1.341637E + 02	-1.319425E + 02	-1.2799421E + 02	1.937096E + 00	1.586162E + 01
	NGHS	-1.348232E + 02	-1.344521E + 02	-1.342897E + 02	1.505267E - 01	9.520511E + 00
	HSTL	-1.385641E + 02	-1.382701E + 02	-1.380733E + 02	1.368319E - 01	1.358972E + 01

TABLE 7: Continued.

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_{19}	HS	-2.603380E + 03	-2.531804E + 03	-2.473751E + 03	3.179450E + 01	2.876496E + 01
	SGHS	-2.721608E + 03	-2.665352E + 03	-2.578190E + 03	3.671259E + 01	3.458217E + 01
	NGHS	-2.581654E + 03	-2.492258E + 03	-2.446278E + 03	3.258052E + 01	2.814525E + 01
	HSTL	-2.929585E + 03	-2.904859E + 03	-2.882272E + 03	1.460064E + 01	3.221904E + 01
F_{20}	HS	9.189454E + 02	5.077136E + 27	1.015427E + 29	2.270564E + 28	5.038898E + 00
	SGHS	5.059004E + 02	8.779452E + 02	1.348477E + 03	1.953273E + 02	1.104361E + 01
	NGHS	1.246908E + 03	2.028307E + 03	2.888645E + 03	4.719852E + 02	5.148672E + 00
	HSTL	1.837030E - 01	2.280353E - 01	2.673117E - 01	2.164823E - 02	8.886787E + 00
F_{21}	HS	6.377972E + 02	6.978627E + 02	7.582129E + 02	3.386564E + 01	1.646089E + 02
	SGHS	6.320206E + 02	7.707349E + 02	8.819430E + 02	5.316649E + 01	1.737214E + 02
	NGHS	7.097658E + 02	7.856465E + 02	8.525136E + 02	4.013482E + 01	1.646784E + 02
	HSTL	9.040644E + 01	9.986598E + 01	1.162748E + 02	6.666442E + 00	1.696227E + 02
F_{22}	HS	2.143132E + 02	2.280396E + 02	2.618505E + 02	1.143029E + 01	7.711841E + 01
	SGHS	1.576182E + 02	2.361854E + 02	4.033144E + 02	5.873130E + 01	8.653562E + 01
	NGHS	2.29408E + 02	2.658819E + 02	2.931895E + 02	1.367509E + 01	7.686817E + 01
	HSTL	9.638521E + 00	1.807740E + 01	2.585962E + 01	3.506528E + 00	8.063761E + 01
F_{23}	HS	6.234643E + 02	6.947096E + 02	7.419847E + 02	3.350395E + 01	1.019985E + 02
	SGHS	6.526572E + 02	7.693093E + 02	8.338596E + 02	4.956235E + 01	1.112867E + 02
	NGHS	6.898719E + 02	7.661332E + 02	8.566126E + 02	4.173909E + 01	1.018226E + 02
	HSTL	8.482421E + 01	9.938394E + 01	1.108417E + 02	8.042784E + 00	1.054352E + 02
F_{24}	HS	4.553252E + 01	2.076506E + 02	4.542336E + 02	1.063305E + 02	8.318836E + 01
	SGHS	-2.939057E + 02	7.490397E + 01	6.242859E + 02	3.186816E + 02	9.254373E + 01
	NGHS	-2.373270E + 02	-1.728535E + 02	-9.543890E + 01	4.073554E + 01	8.306743E + 01
	HSTL	-4.499950E + 02	-4.499933E + 02	-4.499913E + 02	9.985585E - 04	8.722270E + 01
F_{25}	HS	7.366636E + 05	1.041736E + 06	1.399154E + 06	1.825535E + 05	8.464179E + 01
	SGHS	8.054915E + 05	1.228918E + 07	4.659116E + 07	1.341661E + 07	9.395835E + 01
	NGHS	1.559065E + 05	2.605101E + 05	3.969066E + 05	6.581579E + 04	8.432677E + 01
	HSTL	1.448330E + 03	4.054597E + 03	9.585072E + 03	2.811645E + 03	8.853785E + 01
F_{26}	HS	-8.620111E + 01	-6.877671E + 01	-4.175610E + 01	1.198860E + 01	8.370164E + 01
	SGHS	3.837488E + 01	9.840901E + 01	2.221608E + 02	5.104657E + 01	9.300928E + 01
	NGHS	-1.214266E + 01	7.757135E + 00	3.280678E + 01	1.408601E + 01	8.382478E + 01
	HSTL	-2.834421E + 02	-2.731218E + 02	-2.591557E + 02	6.993937E + 00	8.785385E + 01
F_{27}	HS	6.017485E + 01	6.526806E + 01	7.041752E + 01	2.646571E + 00	8.031504E + 01
	SGHS	8.524694E + 01	1.156612E + 02	1.717014E + 02	2.490438E + 01	8.958076E + 01
	NGHS	5.648135E + 01	6.439351E + 01	7.123722E + 01	4.693589E + 00	7.798441E + 01
	HSTL	2.094078E + 00	2.833941E + 00	4.047095E + 00	5.585953E - 01	8.093007E + 01

TABLE 7: Continued.

Function	Algorithm	Best	Mean	Worst	STD	Runtime
F_{28}	HS	-4.289935E + 02	-4.211903E + 02	-4.163227E + 02	3.623053E + 00	1.249831E + 02
	SGHS	-4.488077E + 02	-4.432399E + 02	-4.317460E + 02	4.446639E + 00	1.353388E + 02
	NGHS	-4.499999E + 02	-4.499997E + 02	-4.499995E + 02	9.834457E - 05	1.265239E + 02
	HSTL	-4.500000E + 02	-4.499999E + 02	-4.499999E + 02	2.351848E - 05	1.325924E + 02
F_{29}	HS	2.233712E + 04	3.892963E + 04	5.633712E + 04	9.226538E + 03	1.300818E + 02
	SGHS	5.525834E + 03	2.751029E + 05	2.276536E + 06	5.361823E + 05	1.405805E + 02
	NGHS	1.510576E + 03	2.042038E + 03	6.840093E + 03	1.160782E + 03	1.305521E + 02
	HSTL	4.888426E + 02	1.232585E + 03	5.838617E + 03	1.452099E + 03	1.351203E + 02
F_{30}	HS	-1.772284E + 02	-1.610854E + 02	-1.510294E + 02	6.803769E + 00	1.289660E + 02
	SGHS	-9.536516E + 01	-6.623147E + 01	-3.230639E + 01	1.639332E + 01	1.391979E + 02
	NGHS	-1.574155E + 02	-1.463189E + 02	-1.301902E + 02	7.514495E + 00	1.291356E + 02
	HSTL	-3.201694E + 02	-3.156785E + 02	-3.090753E + 02	3.220219E + 00	1.337009E + 02
F_{31}	HS	1.212613E + 02	1.318822E + 02	1.410318E + 02	5.212928E + 00	8.330888E + 01
	SGHS	1.123687E + 02	1.636391E + 02	1.993816E + 02	2.540267E + 01	9.407166E + 01
	NGHS	1.189049E + 02	1.313797E + 02	1.457549E + 02	6.404969E + 00	8.467554E + 01
	HSTL	4.403576E + 00	7.020892E + 00	1.210083E + 01	1.807576E + 00	8.733987E + 01

The convergence of HSTL method is compared with three other HS algorithms: HS, SGHS, and NGHS on 50- D functions F_3 , F_{10} , and F_{27} , where the HSTL algorithm demonstrates an evident superiority on efficiency and stability, which can be observed from the convergence graphs (Figures 2(a) and 4(a)) and boxplots (Figures 2(b) and 4(b)).

For 50- D problems, the convergence graphs and the boxplots are shown in Figures 2, 3, and 4 on 50- D for sphere unimodal function, griewank multimodal inseparable function, and hybrid function, Hybrid5, which is composed of Bohachevsky and Schwefel2.22 function, where Figures 2(b) and 4(b) plot the boxplots of best results in 30 independent runs. Figures 2(a) and 4(a) portray the convergence curves. It is evident from the convergence graphs that strongly uniform convergence can be maintained throughout the procedure of evolution. From the boxplots we can see that the HSTL algorithm has better convergence, stability, and robustness in most cases than HS, SGHS, and NGHS algorithms.

Figures 5 and 6 show the boxplots and convergence graphs of 3 different problems for dimensions equal to 100 and 200, respectively, where HSTL algorithm demonstrates obvious superiority on efficiency and stability.

4.4. Comparison of the Convergence Speed and Success Rate. In order to give a fair chance to 4 HS algorithm (HS, SGHS, NGHS, and HSTL) compared. We run each algorithm on each benchmark test function and stop as soon as the minimum error value acquired by the algorithm falls below the predefined threshold or a maximum number of FEs is exceeded. For all test functions, the algorithms carry out 30 independent runs.

The respective thresholds values of the error for 30 benchmark problems are in Table 11.

Table 8 displays the statistics results about the success rate, average runtime, and average FEs of F_1 – F_{30} .

As Table 8 shows, for a high-dimensional problem ($D = 100$), the HS, SGHS, and NGHS have great difficulty in finding the global optima on all problems. The HSTL yields 100% success rate for F_1 , F_2 , F_4 – F_{11} , F_{14} , F_{15} , F_{18} , F_{20} , F_{21} , F_{24} , F_{26} , F_{29} , F_{30} . The HSTL algorithm performs much better with success rates of 100% on most problems, and over 50% on F_3 , F_{13} , F_{16} , F_{17} , F_{19} , F_{22} , and F_{28} where other algorithms fail in finding the global optima.

Table 9 illustrates the costing run of 4 HS algorithm for F_1 – F_{31} . The runtime is equal to the average value of all functions mean runtime; the FEs is equal to the average value of all functions mean FEs; the Success Rate is equal to the mean value of the success rate of all functions. In Table 9, we intend to show how well the presented HSTL algorithm performs when compared to HS, SGHS and NGHS algorithm. From the statistics shown in Table 9 can be seen that HSTL uses the least runtime and FEs, and acquires the best success rate among these algorithms on over performances.

4.5. Convergence Analysis. To investigate the convergence of proposed HSTL algorithm, we record the population variance of each algorithm. As Figure 7 shows, for each type function, the fluctuation of population variance in HSTL is smaller

than the fluctuation of population variance in SGHS and NGHS algorithm, and the population variance graphs fall steadily throughout the search process. As a consequence, the proposed HSTL algorithm has stronger robustness and convergence than other variants of HS.

4.6. Parameter HMS Study. In this section, the effect of HMS value on the performance of the HSTL method is investigated. The experimental results generated by using different HMS values (5, 10, 15, 20, 25, 30, 35, and 40) for dimensions equal to 50 are demonstrated in Table 10, respectively.

We can see from Table 10, in which there are some unimodal functions (i.e., F_{10} – F_{14} , F_{23}), a small value of HMS (i.e., 5 or 10) is superior to a large value. For other benchmark functions, there is no obvious indication that one setting value of HMS is superior to the other setting values. Therefore, we can think that, a small HMS is suitable for a simple problem. However, for some complex problems, we can set slightly more HMS that is not exceeding 50. It is reasonable and logical, for it is similar to the quickly memory of musician for a simple harmony improvisation and the depth memory of musician for an outstanding harmony improvisation.

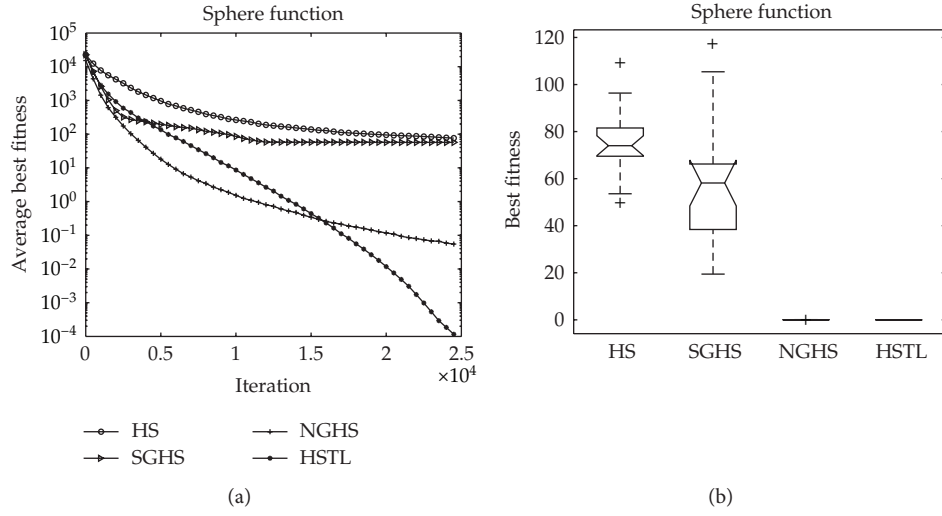
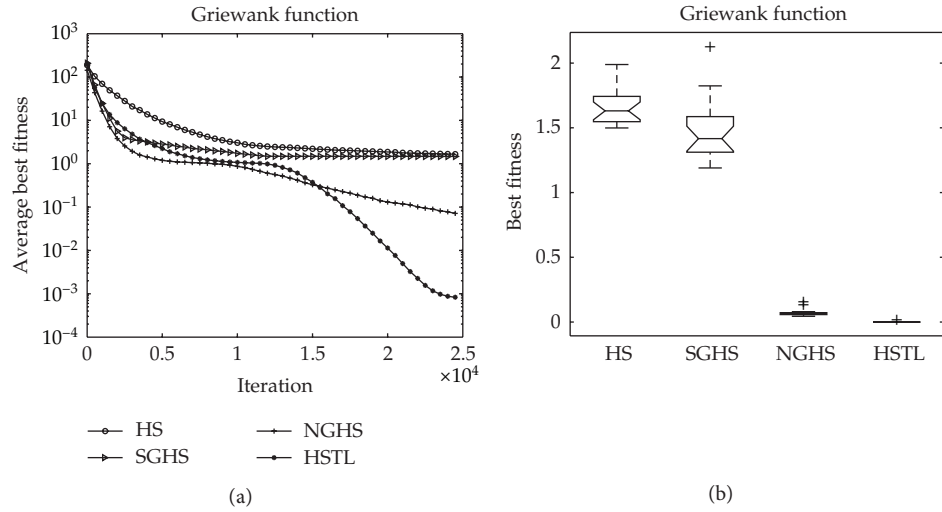
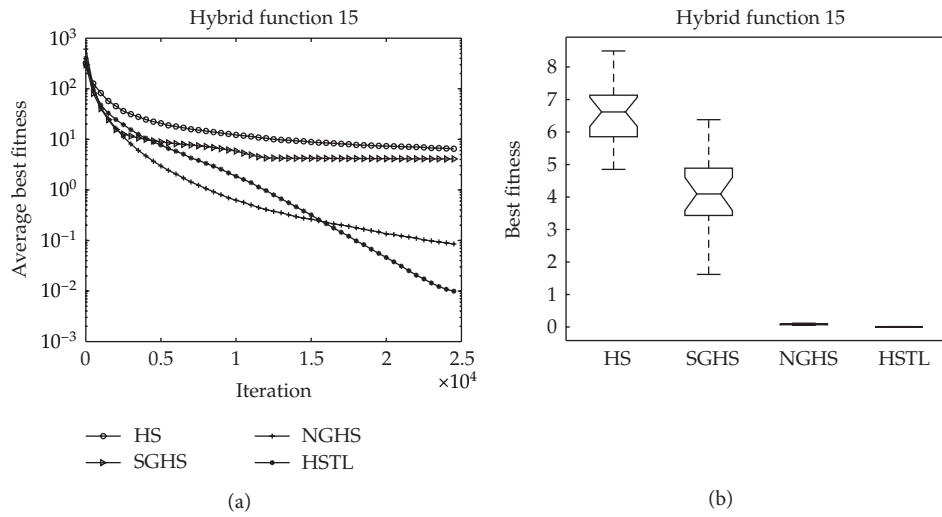
5. Conclusion

In this paper, a novel harmony search combined teaching-learning (HSTL) algorithm is presented to improve the performance and efficiency of the harmony search algorithm. The proposed HSTL algorithm employs the idea of teaching and learning. Four strategies (harmony memory consideration, teaching-learning strategy, local pitch adjusting, and random mutation) are employed to maintain the proper balance between convergence and population diversity. With the process of evolution, the dynamic strategy is adopted to change the parameters HMCR, TLP, BW, and PAR. Numerical experiments show that the dynamic changes of parameters are especially effective in balance between the exploration power and the exploitation power. The population variance analysis indicated that the HSTL algorithm has strong convergence throughout evolution progresses. The sensitivity analysis of HMS parameter showed that it does not have significant influence on complex multimodal problems.

We have compared the performance of proposed HSTL algorithm with the classical HS algorithm and two excellent variant algorithms over a suite of 31 unconstrained numerical optimization functions and evidently concluded that HSTL algorithm is more effective and stable in obtaining high quality solutions and has less FEs, less runtime, and higher success rates under the same conditions.

Acknowledgments

This work is supported by National Natural Science Foundation of China under Grant no. 60974082 and 81160183, Scientific Research Program funded by Shaanxi Provincial Education Department under Grant no. 12JK0863, Ministry of Education “Chunhui Plan” project (no. Z2011051), Natural

FIGURE 2: Convergence graph and boxplot for sphere function (F_{10}) on $D = 50$.FIGURE 3: Convergence graph and boxplot for Griewank function (F_4) on $D = 50$.FIGURE 4: Convergence graph and boxplot for hybrid 15 function (F_{27}) on $D = 50$.

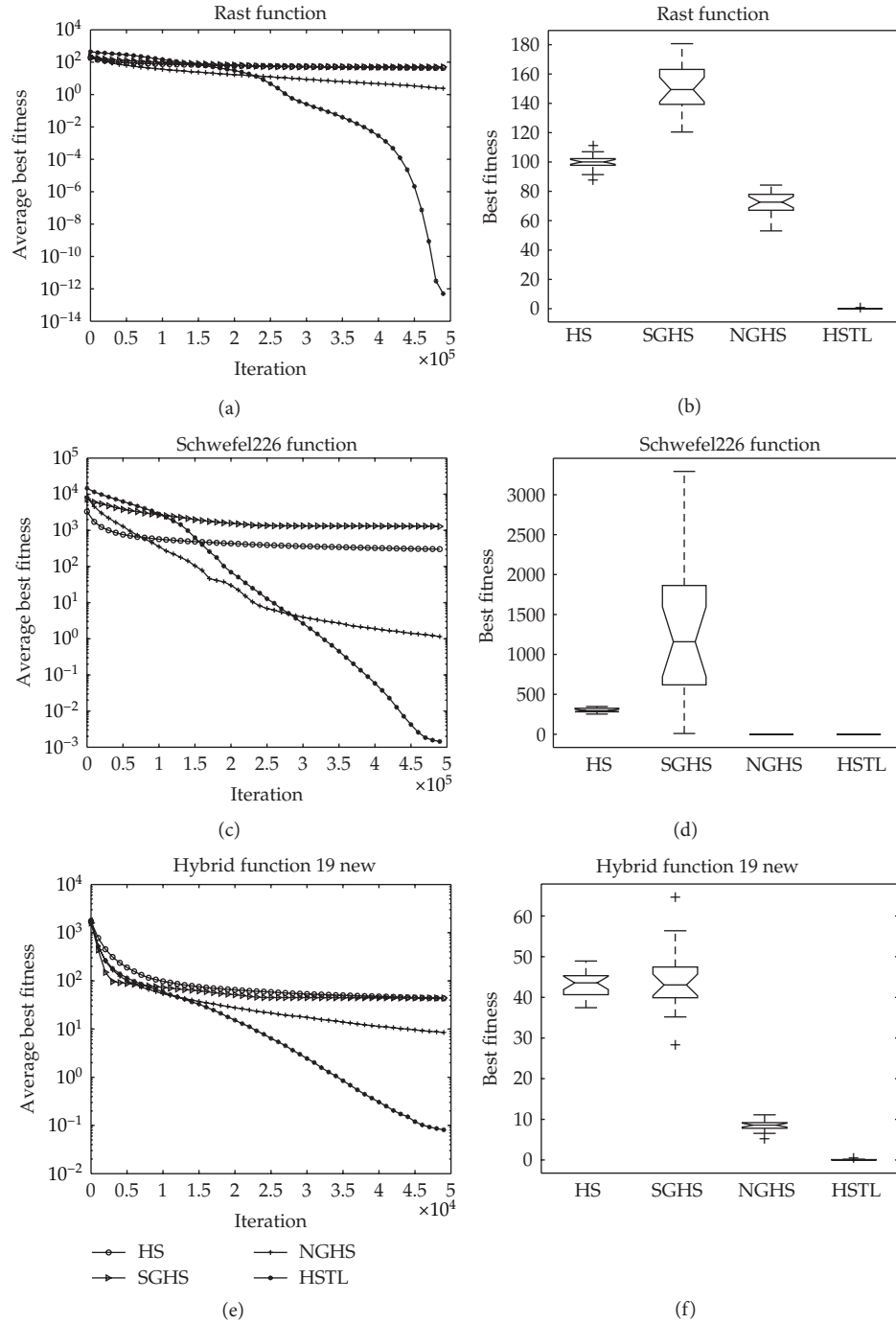
FIGURE 5: Convergence graphs and boxplots for functions Rastrigin, Schwefel, and hybrid 19 new on $D = 100$.

TABLE 8: The success rate, mean runtime, and FEs for function F_1-F_{19} , $F_{21}-F_{31}$.

Function	Algorithm	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
Success rate (SR)	HS	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
	SGHS	0%	0%	0%	0%	60%	0%	0%	0%	0%	0%
	NGHS	0%	0%	0%	0%	0%	0%	0%	45%	15%	0%
	HSTL	100%	100%	95%	100%	100%	100%	100%	100%	100%	100%
Mean runtime (s)	HS	19	14	21	50	31	48	15	14	17	16
	SGHS	39	34	41	73	40	72	36	35	39	37
	NGHS	18	13	20	49	30	48	14	11	16	16
	HSTL	20	10	21	48	26	46	18	4	14	19
Mean FEs	HS	500000	500000	500000	500000	500000	500000	500000	500000	500000	500000
	SGHS	500000	500000	500000	500000	397181	500000	500000	500000	500000	500000
	NGHS	500000	500000	500000	500000	500000	500000	500000	425302	497089	500000
	HSTL	475167	291492	452830	450158	386120	443818	482502	105551	332693	483873
Function	Algorithm	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}	F_{16}	F_{17}	F_{18}	F_{19}	F_{21}
Success rate (SR)	HS	70%	45%	0%	0%	0%	0%	0%	0%	0%	0%
	SGHS	0%	0%	0%	0%	65%	0%	0%	0%	0%	0%
	NGHS	75%	100%	0%	0%	90%	15%	0%	0%	0%	0%
	HSTL	100%	100%	90%	100%	100%	85%	90%	100%	80%	100%
Mean runtime (s)	HS	8	13	32	32	38	34	55	38	88	527
	SGHS	35	34	55	54	39	56	78	60	114	570
	NGHS	9	3	31	31	8	32	54	36	92	525
	HSTL	5	9	36	31	18	34	51	32	81	467
Mean Fes	HS	264713	471826	500000	500000	500000	500000	500000	500000	500000	500000
	SGHS	500000	500000	500000	500000	323631	500000	500000	500000	500000	500000
	NGHS	335021	107386	500000	500000	114380	485463	500000	500000	500000	500000
	HSTL	122555	240403	494888	418223	206975	437342	488064	434498	415872	445225
Function	Algorithm	F_{22}	F_{23}	F_{24}	F_{25}	F_{26}	F_{27}	F_{28}	F_{29}	F_{30}	F_{31}
Success rate (SR)	HS	0%	0%	0%	0%	100%	0%	0%	0%	0%	0%
	SGHS	0%	0%	0%	0%	95%	0%	100%	0%	0%	0%
	NGHS	0%	0%	0%	0%	100%	0%	100%	25%	0%	0%
	HSTL	100%	50%	0%	100%	100%	100%	100%	60%	100%	100%
Mean runtime (s)	HS	289	373	359	364	35	310	453	441	436	312
	SGHS	329	416	400	406	133	351	337	476	477	351
	NGHS	293	374	360	366	22	310	13	382	437	317
	HSTL	273	375	363	271	46	311	104	363	364	273
Mean Fes	HS	500000	500000	500000	500000	54677	500000	500000	500000	500000	500000
	SGHS	500000	500000	500000	500000	181162	500000	341216	500000	500000	500000
	NGHS	500000	500000	500000	500000	33331	500000	14644	451443	500000	500000
	HSTL	459671	497514	500000	367442	85659	500000	132414	432651	415004	429027

TABLE 9: The average runtime, average FEs, average success rate, and costing run of all functions.

Algorithm	Average runtime (ART)	Average FEs (AFE)	Average success rate (ASR)
HS	145.1399	477136	0.069355
SGHS	169.5528	475586.8	0.103226
NGHS	127.2487	434324.5	0.182258
HSTL	120.8951	384181.3	0.919355

Where $ART = \sum_{i=1}^{31} runtime_i$, $AFE = \sum_{i=1}^{31} FEs_i$, $ASR = \sum_{i=1}^{31} SR_i$.

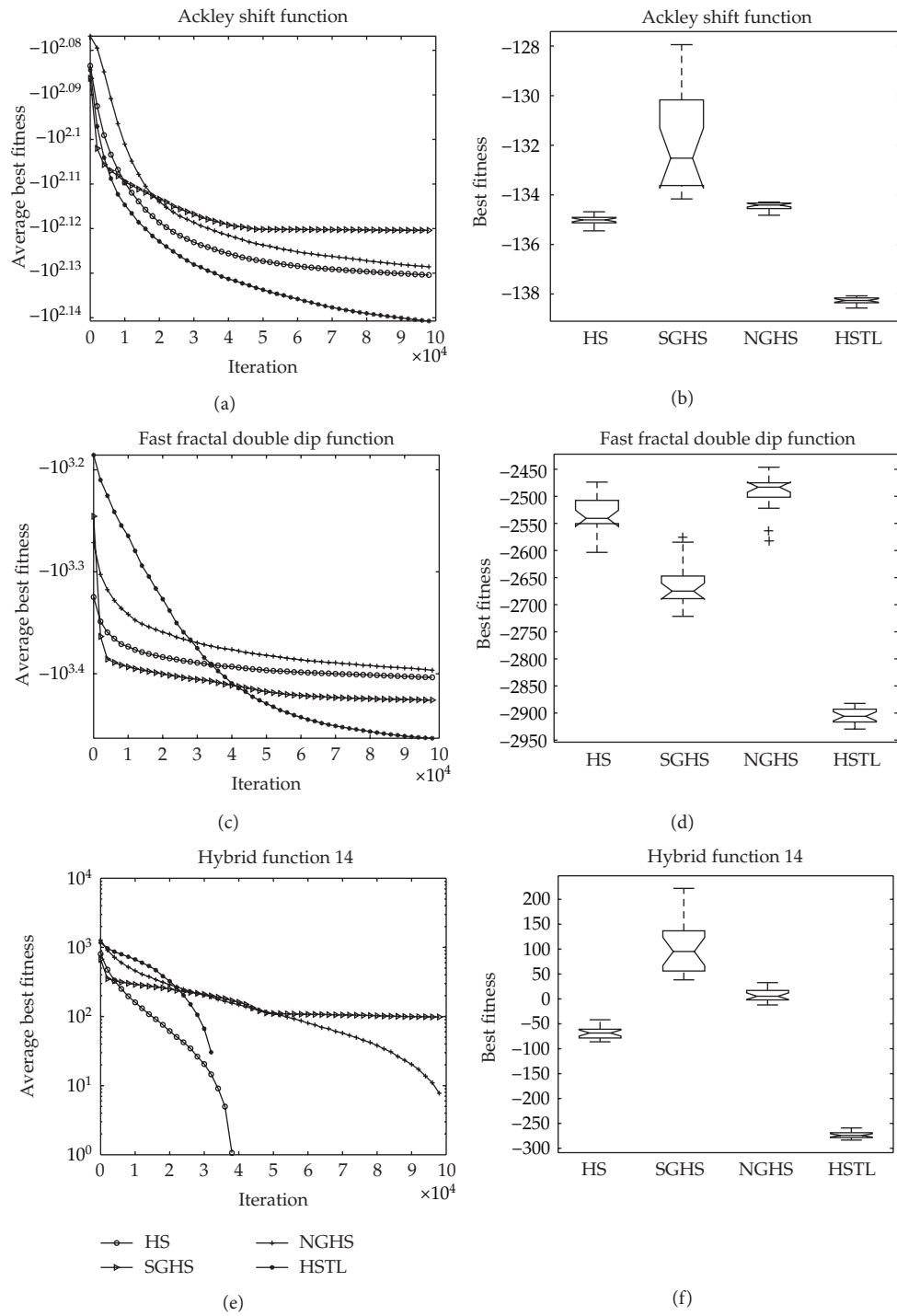


FIGURE 6: Convergence graphs and boxplots for shift functions Ackley, fast fractal double, and hybrid 14 on $D = 200$.

TABLE 10: The effect of HMS on the performance of the HSTL algorithm ($D = 50$, FEs = 25000).

HMS	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}
5	7.02E-06	1.28E+00	5.30E-03	1.12E-06	-4.82E+01	3.23E-03	4.97E-02	7.23E+01	1.36E-03	1.36E-03
10	2.10E-05	8.84E-01	2.71E-03	1.26E-06	-4.82E+01	4.83E-03	7.11E-10	4.39E+01	1.14E-03	4.92E-08
15	6.98E-05	7.11E-01	4.94E-04	1.29E-06	-4.83E+01	6.93E-03	7.66E-09	4.35E+01	1.06E-03	2.19E-07
20	1.03E-04	7.07E-01	8.64E-04	1.58E-06	-4.85E+01	9.06E-03	4.44E-08	4.16E+01	1.16E-03	3.70E-07
25	1.40E-04	7.23E-01	2.23E-03	1.47E-06	-4.80E+01	1.17E-02	7.97E-08	4.55E+01	1.27E-03	8.60E-07
30	2.05E-04	7.72E-01	1.82E-06	1.90E-06	-4.81E+01	1.46E-02	3.07E-07	4.39E+01	1.14E-03	1.78E-06
35	2.85E-04	7.03E-01	2.21E-06	1.69E-06	-4.82E+01	1.61E-02	4.45E-07	4.37E+01	1.09E-03	2.54E-06
40	3.81E-04	7.49E-01	3.34E-06	2.00E-06	-4.85E+01	1.97E-02	6.64E-07	4.46E+01	1.22E-03	3.88E-06
HMS	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}	F_{16}	F_{17}	F_{18}	F_{19}	F_{20}
05	7.97E+03	3.57E+00	-4.50E+02	-4.50E+02	6.45E+02	-3.24E+02	-1.80E+02	-1.39E+02	-8.04E+02	9.41E+00
10	2.13E+04	2.72E+01	-4.50E+02	-4.50E+02	5.21E+02	-3.24E+02	-1.80E+02	-1.40E+02	-8.03E+02	5.08E+00
15	1.80E+04	3.40E+01	-4.50E+02	-4.50E+02	6.78E+02	-3.24E+02	-1.80E+02	-1.40E+02	-8.08E+02	4.10E+00
20	1.82E+04	3.93E+01	-4.50E+02	-4.49E+02	5.96E+02	-3.25E+02	-1.80E+02	-1.40E+02	-8.10E+02	4.20E+00
25	9.80E+03	5.43E+01	-4.50E+02	-4.48E+02	5.19E+02	-3.24E+02	-1.80E+02	-1.40E+02	-8.10E+02	4.27E+00
30	2.18E+04	6.84E+01	-4.50E+02	-4.47E+02	5.33E+02	-3.25E+02	-1.80E+02	-1.40E+02	-8.11E+02	4.33E+00
35	1.46E+04	9.77E+01	-4.50E+02	-4.46E+02	8.04E+02	-3.26E+02	-1.80E+02	-1.40E+02	-8.11E+02	4.24E+00
40	1.24E+04	1.18E+02	-4.50E+02	-4.44E+02	8.18E+02	-3.26E+02	-1.80E+02	-1.40E+02	-8.10E+02	4.29E+00
HMS	F_{21}	F_{22}	F_{23}	F_{24}	F_{25}	F_{26}	F_{27}	F_{28}	F_{29}	F_{30}
05	2.02E-01	9.64E+00	-4.50E+02	4.74E+02	-3.26E+02	5.05E-03	-4.50E+02	6.01E+02	-3.29E+02	1.50E-03
10	6.50E-02	5.27E+00	-4.50E+02	4.71E+02	-3.26E+02	4.96E-03	-4.50E+02	7.29E+02	-3.28E+02	1.41E-03
15	7.34E-02	4.40E+00	-4.50E+02	4.70E+02	-3.27E+02	4.86E-03	-4.50E+02	4.19E+02	-3.29E+02	2.21E-02
20	5.28E-02	4.27E+00	-4.50E+02	4.96E+02	-3.26E+02	5.08E-03	-4.50E+02	4.26E+02	-3.29E+02	1.25E-03
25	3.54E-04	4.06E+00	-4.50E+02	4.93E+02	-3.26E+02	5.37E-03	-4.50E+02	9.77E+02	-3.29E+02	1.53E-03
30	3.34E-04	3.80E+00	-4.50E+02	4.91E+02	-3.27E+02	5.22E-03	-4.50E+02	6.57E+02	-3.29E+02	1.39E-03
35	4.41E-04	4.06E+00	-4.50E+02	4.93E+02	-3.27E+02	5.16E-03	-4.50E+02	5.06E+02	-3.29E+02	1.61E-03
40	3.65E-04	4.06E+00	-4.50E+02	4.86E+02	-3.27E+02	5.40E-03	-4.50E+02	8.34E+02	-3.29E+02	1.45E-03

TABLE 11

$F_1 \sim F_{10}$:	$1.0E-05$	$1.0E+00$	$1.0E-05$	$1.0E-05$	$5.0E+00$	$1.0E-02$	$1.0E-10$	$2D$	$1.0E+00$	$1.0E-10$
$F_{11} \sim F_{20}$:	$2DE+04$	D	$1.0E-04$	$1.0E-01$	$50D$	$1.0E+00$	$1.0E-04$	$1.0E-02$	$1.0E+00$	$1.0E+01$
$F_{21} \sim F_{30}$:	$1.0E-03$	$1.0E+01$	$1.0E-04$	$2D$	$2D$	$1.0E-02$	$1.0E-05$	D	$1.0E+00$	$1.0E-02$

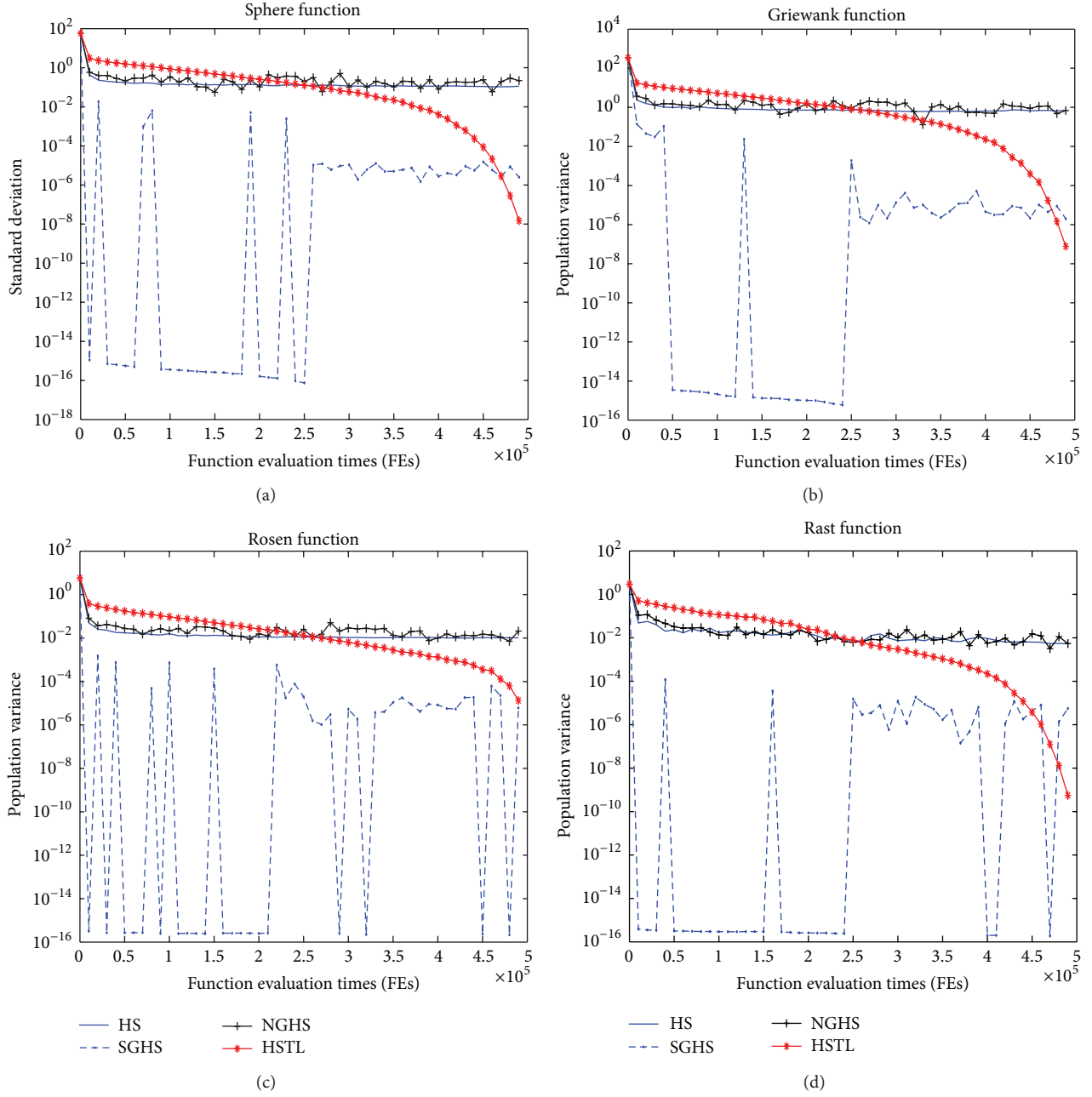


FIGURE 7: The population variance graphs.

Science Foundation of Ningxia Hui Autonomous Region (no. NZ12179), and Scientific Research Fund of Ningxia Education Department (no. NGY2011042).

References

- [1] 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.
- [2] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, no. 36–38, pp. 3902–3933, 2005.
- [3] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "Harmony search optimization: application to pipe network design," *International Journal of Modelling and Simulation*, vol. 22, no. 2, pp. 125–133, 2002.
- [4] K. S. Lee and Z. W. Geem, "A new structural optimization method based on the harmony search algorithm," *Computers and Structures*, vol. 82, no. 9–10, pp. 781–798, 2004.
- [5] J. Kang and W. Zhang, "Combination of fuzzy C-means and harmony search algorithms for clustering of text document," *Journal of Computational Information Systems*, vol. 16, no. 7, pp. 5980–5986, 2011.
- [6] A. Vasebi, M. Fesanghary, and S. M. T. Bathaee, "Combined heat and power economic dispatch by harmony search algorithm," *International Journal of Electrical Power and Energy Systems*, vol. 29, no. 10, pp. 713–719, 2007.
- [7] Z. W. Geem, "Optimal scheduling of multiple dam system using harmony search algorithm," in *Lecture Notes in Computer Science*, vol. 4507, pp. 316–323, 2007.
- [8] M. Mahdavi, M. Fesanghary, and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Applied Mathematics and Computation*, vol. 188, no. 2, pp. 1567–1579, 2007.
- [9] S. Tuo and L. Yong, "An improved harmony search algorithm with chaos," *Journal of Computational Information Systems*, vol. 8, no. 10, pp. 4269–4276, 2012.
- [10] M. G. H. Omran and M. Mahdavi, "Global-best harmony search," *Applied Mathematics and Computation*, vol. 198, no. 2, pp. 643–656, 2008.
- [11] Q. K. Pan, P. N. Suganthan, J. J. Liang, and M. F. Tasgetiren, "A local-best harmony search algorithm with dynamic subpopulations," *Engineering Optimization*, vol. 42, no. 2, pp. 101–117, 2010.
- [12] P. Chakraborty, G. G. Roy, S. Das, D. Jain, and A. Abraham, "An improved harmony search algorithm with differential mutation operator," *Fundamenta Informaticae*, vol. 95, no. 4, pp. 401–426, 2009.
- [13] D. X. Zou, L. Q. Gao, J. Wu, S. Li, and Y. Li, "A novel global harmony search algorithm for reliability problems," *Computers and Industrial Engineering*, vol. 58, no. 2, pp. 307–316, 2010.
- [14] D. Zou, L. Gao, J. Wu, and S. Li, "Novel global harmony search algorithm for unconstrained problems," *Neurocomputing*, vol. 73, no. 16–18, pp. 3308–3318, 2010.
- [15] X. Z. Gao, X. Wang, and S. J. Ovaska, "Uni-modal and multi-modal optimization using modified Harmony Search methods," *International Journal of Innovative Computing, Information and Control*, vol. 5, no. 10, pp. 2985–2996, 2009.
- [16] Q. K. Pan, P. N. Suganthan, M. F. Tasgetiren, and J. J. Liang, "A self-adaptive global best harmony search algorithm for continuous optimization problems," *Applied Mathematics and Computation*, vol. 216, no. 3, pp. 830–848, 2010.
- [17] P. Yadav, R. Kumar, S. K. Panda, and C. S. Chang, "An intelligent tuned harmony search algorithm for optimization," *Information Sciences*, vol. 196, pp. 47–72, 2012.
- [18] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems," *CAD Computer Aided Design*, vol. 43, no. 3, pp. 303–315, 2011.
- [19] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-learning-based optimization: an optimization method for continuous non-linear large scale problems," *Information Sciences*, vol. 183, pp. 1–15, 2012.
- [20] R. V. Rao and V. Patel, "An elitist teaching-learning-based optimization algorithm for solving complex constrained optimization problems," *International Journal of Industrial Engineering Computations*, vol. 3, pp. 535–560, 2012.
- [21] R. V. Rao and V. J. Savsani, *Mechanical Design Optimization Using Advanced Optimization Techniques*, Springer-Verlag, London, UK, 2012.
- [22] R. V. Rao and V. Patel, "Multi-objective optimization of heat exchangers using a modified teaching-learning based optimization algorithm," *Applied Mathematical Modelling*, vol. 37, no. 3, pp. 1147–1162, 2013.
- [23] V. R. Rao and V. Patel, "Multi-objective optimization of two stage thermoelectric cooler using a modified teaching-learning-based optimization algorithm," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 1, pp. 430–445, 2013.
- [24] R. V. Rao, V. J. Savsani, and J. Balic, "Teaching-learning-based optimization algorithm for unconstrained and constrained real parameter optimization problems," *Engineering Optimization*, vol. 44, no. 12, pp. 1447–1462, 2012.
- [25] S. Das, A. Mukhopadhyay, A. Roy, A. Abraham, and B. K. Panigrahi, "Exploratory power of the harmony search algorithm: analysis and improvements for global numerical optimization," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 41, no. 1, pp. 89–106, 2011.
- [26] H. Sarvari and K. Zamanifar, "Improvement of harmony search algorithm by using statistical analysis," *Artificial Intelligence Review*, vol. 37, no. 3, pp. 181–215, 2012.
- [27] M. Fukushima, Test Functions for Unconstrained Global Optimization, http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page364.htm.
- [28] K. Tang, X. Yao, P. N. Suganthan et al., Benchmark Functions for the CEC'2008 Special Session and Competition on Large Scale Global Optimization, <http://www.ntu.edu.sg/home/EPNSugan/>, 2008.
- [29] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC'2010 special session and competition on large scale global optimization," Tech. Rep., Nature Inspired Computation and Applications Laboratory, USTC, China & Nanyang Technological University, Nanyang Avenue, Singapore, 2009.
- [30] F. Herrera, M. Lozano, and D. Molina, "Test suite for the special issue of soft computing on scalability of evolutionary algorithms and other meta-heuristics for large scale continuous optimization problems," <http://sci2s.ugr.es/eamhco/CFP.php>.

Research Article

A Swarm Optimization Algorithm for Multimodal Functions and Its Application in Multicircle Detection

Erik Cuevas, Daniel Zaldívar, and Marco Pérez-Cisneros

Departamento de Ciencias Computacionales, Universidad de Guadalajara, CUCEI, Avenida Revolución 1500, C.P 44430, Guadalajara, Jal, Mexico

Correspondence should be addressed to Erik Cuevas; erik.cuevas@cucei.udg.mx

Received 3 September 2012; Accepted 25 December 2012

Academic Editor: Baozhen Yao

Copyright © 2013 Erik Cuevas 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 engineering problems due to physical and cost constraints, the best results, obtained by a global optimization algorithm, cannot be realized always. Under such conditions, if multiple solutions (local and global) are known, the implementation can be quickly switched to another solution without much interrupting the design process. This paper presents a new swarm multimodal optimization algorithm named as the collective animal behavior (CAB). Animal groups, such as schools of fish, flocks of birds, swarms of locusts, and herds of wildebeest, exhibit a variety of behaviors including swarming about a food source, milling around a central location, or migrating over large distances in aligned groups. These collective behaviors are often advantageous to groups, allowing them to increase their harvesting efficiency to follow better migration routes, to improve their aerodynamic, and to avoid predation. In the proposed algorithm, searcher agents emulate a group of animals which interact with each other based on simple biological laws that are modeled as evolutionary operators. Numerical experiments are conducted to compare the proposed method with the state-of-the-art methods on benchmark functions. The proposed algorithm has been also applied to the engineering problem of multi-circle detection, achieving satisfactory results.

1. Introduction

A large number of real-world problems can be considered as multimodal function optimization subjects. An objective function may have several global optima, that is, several points holding objective function values which are equal to the global optimum. Moreover, it may exhibit some other local optima points whose objective function values lay near-by a global optimum. Since the mathematical formulation of a real-world problem often produces a multimodal optimization issue, finding all global or even these local optima would provide to the decision makers multiple options to choose from [1].

Several methods have recently been proposed for solving the multimodal optimization problem. They can be divided into two main categories: deterministic and stochastic (metaheuristic) methods. When facing complex multimodal optimization problems, deterministic methods, such as gradient descent method, the quasi-Newton method, and the Nelder-Mead's simplex method, may get easily trapped into the local

optimum as a result of deficiently exploiting local information. They strongly depend on a priori information about the objective function, yielding few reliable results.

Metaheuristic algorithms have been developed combining rules and randomness mimicking several phenomena. These phenomena include evolutionary processes (e.g., the evolutionary algorithm proposed by Fogel et al. [2], de Jong [3], and Koza [4] and the genetic algorithms (GAs) proposed by Holland [5] and Goldberg [6]), immunological systems (e.g., the artificial immune systems proposed by de Castro et al. [7]), physical processes (e.g., simulated annealing proposed by Kirkpatrick et al. [8], electromagnetism-like proposed by Birbil et al. [9], and the gravitational search algorithm proposed by Rashedi et al. [10]), and the musical process of searching for a perfect state of harmony (proposed by Geem et al. [11], Lee and Geem [12], Geem [13], and Gao et al. [14]).

Traditional GAs perform well for locating a single optimum but fail to provide multiple solutions. Several methods have been introduced into the GA's scheme to achieve

multimodal function optimization, such as sequential fitness sharing [15, 16], deterministic crowding [17], probabilistic crowding [18], clustering-based niching [19], clearing procedure [20], species conserving genetic algorithm [21], and elitist-population strategies [22]. However, algorithms based on the GAs do not guarantee convergence to global optima because of their poor exploitation capability. GAs exhibit other drawbacks such as the premature convergence which results from the loss of diversity in the population and becomes a common problem when the search continues for several generations. Such drawbacks [23] prevent the GAs from practical interest for several applications.

Using a different metaphor, other researchers have employed artificial immune systems (AIS) to solve the multimodal optimization problems. Some examples are the clonal selection algorithm [24] and the artificial immune network (AiNet) [25, 26]. Both approaches use some operators and structures which attempt to algorithmically mimic the natural immune system's behavior of human beings and animals.

Several studies have been inspired by animal behavior phenomena in order to develop optimization techniques such as the particle swarm optimization (PSO) algorithm which models the social behavior of bird flocking or fish schooling [27]. In recent years, there have been several attempts to apply the PSO to multimodal function optimization problems [28, 29]. However, the performance of such approaches presents several flaws when it is compared to the other multi-modal metaheuristic counterparts [26].

Recently, the concept of individual organization [30, 31] has been widely used to understand collective behavior of animals. The central principle of individual organization is that simple repeated interactions between individuals can produce complex behavioral patterns at group level [30, 32, 33]. Such inspiration comes from behavioral patterns seen in several animal groups, such as ant pheromone trail networks, aggregation of cockroaches, and the migration of fish schools, which can be accurately described in terms of individuals following simple sets of rules [34]. Some examples of these rules [33, 35] include keeping current position (or location) for best individuals, local attraction or repulsion, random movements, and competition for the space inside of a determined distance. On the other hand, new studies have also shown the existence of collective memory in animal groups [36–38]. The presence of such memory establishes that the previous history, of group structure, influences the collective behavior exhibited in future stages. Therefore, according to these new developments, it is possible to model complex collective behaviors by using simple individual rules and configuring a general memory.

On the other hand, the problem of detecting circular features holds paramount importance in several engineering applications. The circle detection in digital images has been commonly solved through the circular Hough transform (CHT) [39]. Unfortunately, this approach requires a large storage space that augments the computational complexity and yields a low processing speed. In order to overcome this problem, several approaches which modify the original CHT have been proposed. One well-known example is the randomized Hough transform (RHT) [40]. As an alternative to

Hough-transform-based techniques, the problem of shape recognition has also been handled through optimization methods. In general, they have demonstrated to deliver better results than those based on the HT considering accuracy, speed, and robustness [41]. Such approaches have produced several robust circle detectors using different optimization algorithms such as genetic algorithms (GAs) [41], harmony search (HSA) [42], electromagnetism-like (EMO) [43], differential evolution (DE) [44], and bacterial foraging optimization (BFOA) [45]. Since such evolutionary algorithms are global optimizers, they detect only the global optimum (only one circle) of an objective function that is defined over a given search space. However, extracting multiple-circle primitives falls into the category of multi-modal optimization, where each circle represents an optimum which must be detected within a feasible solution space. The quality for such optima is characterized by the properties of their geometric primitives. Big and well-drawn circles normally represent points in the search space with high fitness values (possible global maximum) whereas small and dashed circles describe points with fitness values which account for local maxima. Likewise, circles holding similar geometric properties, such as radius and size, tend to represent locations with similar fitness values. Therefore, a multi-modal method must be applied in order to appropriately solve the problem of multi-shape detection. In this paper, a new multimodal optimization algorithm based on the collective animal behavior is proposed and also applied to multicircle detection.

This paper proposes a new optimization algorithm inspired by the collective animal behavior. In this algorithm, the searcher agents emulate a group of animals that interact with each other based on simple behavioral rules which are modeled as evolutionary operators. Such operations are applied to each agent considering that the complete group has a memory which stores its own best positions seen so far by applying a competition principle. Numerical experiments have been conducted to compare the proposed method with the state-of-the-art methods on multi-modal benchmark functions. Besides, the proposed algorithm is also applied to the engineering problem of multicircle detection, achieving satisfactory results.

This paper is organized as follows. Section 2 introduces the basic biological aspects of the algorithm. In Section 3, the proposed algorithm and its characteristics are described. A numerical study on different multi-modal benchmark functions is presented in Section 4. Section 5 presents the application of the proposed algorithm to multi-circle detection whereas Section 6 shows the obtained results. Finally, in Section 7 the conclusions are discussed.

2. Biological Fundamentals

The remarkable collective behavior of organisms such as swarming ants, schooling fish, and flocking birds has long captivated the attention of naturalists and scientists. Despite a long history of scientific investigation, just recently we are beginning to decipher the relationship between individuals and group-level properties [46]. Grouping individuals often have to make rapid decisions about where to move

or what behavior to perform, in uncertain and dangerous environments. However, each individual typically has only relatively local sensing ability [47]. Groups are, therefore, often composed of individuals that differ with respect to their informational status, and individuals are usually not aware of the informational state of others [48], such as whether they are knowledgeable about a pertinent resource or of a threat.

Animal groups are based on a hierarchic structure [49] which differentiates individuals according to a fitness principle known as dominance [50]. Such concept represents the domain of some individuals within a group and occurs when competition for resources leads to confrontation. Several studies [51, 52] have found that such animal behavior leads to stable groups with better cohesion properties among individuals.

Recent studies have illustrated how repeated interactions among grouping animals scale to collective behavior. They have also remarkably revealed that collective decision-making mechanisms across a wide range of animal group types, ranging from insects to birds (and even among humans in certain circumstances), seem to share similar functional characteristics [30, 34, 53]. Furthermore, at a certain level of description, collective decision-making in organisms shares essential common features such as general memory. Although some differences may arise, there are good reasons to increase communication between researchers working in collective animal behavior and those involved in cognitive science [33].

Despite the variety of behaviors and motions of animal groups, it is possible that many of the different collective behavioral patterns are generated by simple rules followed by individual group members. Some authors have developed different models, such as the self-propelled particle (SPP) model which attempts to capture the collective behavior of animal groups in terms of interactions between group members following a diffusion process [54–57].

On other hand, following a biological approach, Couzin et al. [33, 34] have proposed a model in which individual animals follow simple rules of thumb: (1) keep the position of best individuals, (2) move from or to nearby neighbors (local attraction or repulsion), (3) move randomly, and (4) compete for the space inside of a determined distance. Each individual thus admits three different movements: attraction, repulsion, or random, while holding two kinds of states: preserve the position or compete for a determined position. In the model, the movement experimented by each individual is decided randomly (according to an internal motivation); meanwhile the states are assumed according to fixed criteria.

The dynamical spatial structure of an animal group can be explained in terms of its history [54]. Despite this, the majority of the studies have failed in considering the existence of memory in behavioral models. However, recent researches [36, 58] have also shown the existence of collective memory in animal groups. The presence of such memory establishes that the previous history of the group structure influences the collective behavior exhibited in future stages. Such memory can contain the position of special group members (the dominant individuals) or the averaged movements produced by the group.

According to these new developments, it is possible to model complex collective behaviors by using simple individual rules and setting a general memory. In this work, the behavioral model of animal groups is employed for defining the evolutionary operators through the proposed meta-heuristic algorithm. A memory is incorporated to store best animal positions (best solutions) considering a competition-dominance mechanism.

3. Collective Animal Behaviour Algorithm (CAB)

The CAB algorithm assumes the existence of a set of operations that resembles the interaction rules that model the collective animal behavior. In the approach, each solution within the search space represents an animal position. The “fitness value” refers to the animal dominance with respect to the group. The complete process mimics the collective animal behavior.

The approach in this paper implements a memory for storing best solutions (animal positions) mimicking the aforementioned biologic process. Such memory is divided into two different elements, one for maintaining the best found positions in each generation (\mathbf{M}_g) and the other for storing best history positions during the complete evolutionary process (\mathbf{M}_h).

3.1. Description of the CAB Algorithm. Like other meta-heuristic approaches, the CAB algorithm is also an iterative process. It starts by initializing the population randomly, that is, generating random solutions or animal positions. The following four operations are thus applied until the termination criterion is met, that is, the iteration number NI is reached as follows.

- (1) Keep the position of the best individuals.
- (2) Move from or nearby neighbors (local attraction and repulsion).
- (3) Move randomly.
- (4) Compete for the space inside of a determined distance (updating the memory).

3.1.1. Initializing the Population. The algorithm begins by initializing a set \mathbf{A} of N_p animal positions ($\mathbf{A} = \{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_{N_p}\}$). Each animal position \mathbf{a}_i is a D -dimensional vector containing the parameter values to be optimized, which are randomly and uniformly distributed between the prespecified lower initial parameter bound a_j^{low} and the upper initial parameter bound a_j^{high} :

$$a_{j,i} = a_j^{\text{low}} + \text{rand}(0, 1) \cdot (a_j^{\text{high}} - a_j^{\text{low}}), \quad (1)$$

$$j = 1, 2, \dots, D; \quad i = 1, 2, \dots, N_p.$$

with j and i being the parameter and individual indexes, respectively. Hence, $a_{j,i}$ is the j th parameter of the i th individual.

All the initial positions \mathbf{A} are sorted according to the fitness function (dominance) to form a new individual set $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{N_p}\}$, so that we can choose the best B positions and store them in the memory \mathbf{M}_g and \mathbf{M}_h . The fact that both memories share the same information is only allowed at this initial stage.

3.1.2. Keep the Position of the Best Individuals. Analogously to the biological metaphor, this behavioral rule, typical in animal groups, is implemented as an evolutionary operation in our approach. In this operation, the first B elements of the new animal position set $\mathbf{A}(\{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_B\})$ are generated. Such positions are computed by the values contained in the historic memory \mathbf{M}_h considering a slight random perturbation around them. This operation can be modelled as follows:

$$\mathbf{a}_l = \mathbf{m}_h^l + \mathbf{v}, \quad (2)$$

where $l \in \{1, 2, \dots, B\}$ while \mathbf{m}_h^l represents the l -element of the historic memory \mathbf{M}_h and \mathbf{v} is a random vector holding an appropriate small length.

3.1.3. Move from or to Nearby Neighbours. From the biological inspiration, where animals experiment a random local attraction or repulsion according to an internal motivation, we implement the evolutionary operators that mimic them. For this operation, a uniform random number r_m is generated within the range $[0, 1]$. If r_m is less than a threshold H , a determined individual position is moved (attracted or repelled) considering the nearest best historical value of the group (the nearest position contained in \mathbf{M}_h); otherwise it is considered the nearest best value in the group of the current generation (the nearest position contained in \mathbf{M}_g). Therefore, such operation can be modeled as follows:

$$\mathbf{a}_i = \begin{cases} \mathbf{x}_i \pm r \cdot (\mathbf{m}_h^{\text{nearest}} - \mathbf{x}_i) & \text{with probability } H \\ \mathbf{x}_i \pm r \cdot (\mathbf{m}_g^{\text{nearest}} - \mathbf{x}_i) & \text{with probability } (1 - H), \end{cases} \quad (3)$$

where $i \in \{B+1, B+2, \dots, N_p\}$, $\mathbf{m}_h^{\text{nearest}}$ and $\mathbf{m}_g^{\text{nearest}}$ represent the nearest elements of \mathbf{M}_h and \mathbf{M}_g to \mathbf{x}_i , while r is a random number between $[-1, 1]$. Therefore, if $r > 0$, the individual position \mathbf{x}_i is attracted to the position $\mathbf{m}_h^{\text{nearest}}$ or $\mathbf{m}_g^{\text{nearest}}$; otherwise such movement is considered as a repulsion.

3.1.4. Move Randomly. Following the biological model, under some probability P an animal randomly changes its position. Such behavioral rule is implemented considering the next expression:

$$\mathbf{a}_i = \begin{cases} \mathbf{r} & \text{with probability } P \\ \mathbf{x}_i & \text{with probability } (1 - P), \end{cases} \quad (4)$$

where $i \in \{B+1, B+2, \dots, N_p\}$ and \mathbf{r} is a random vector defined within the search space. This operator is similar to reinitialize the particle in a random position as it is done by (1).

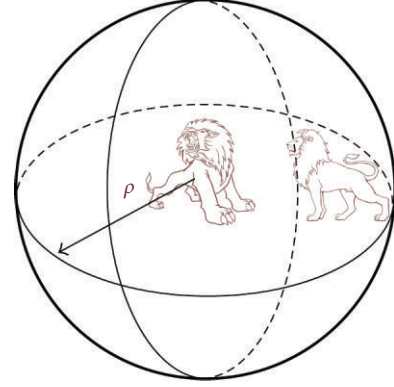


FIGURE 1: Dominance concept, presented when two animals confront each other inside of a ρ distance.

3.1.5. Compete for the Space Inside of a Determined Distance (Updating the Memory). Once the operations to preserve the position of the best individuals, to move from or to nearby neighbors and to move randomly, have all been applied to all the N_p animal positions, generating N_p new positions, it is necessary to update the memory \mathbf{M}_h .

In order to update the memory \mathbf{M}_h , the concept of dominance is used. Animals that interact in a group keep a minimum distance among them. Such distance ρ depends on how aggressive the animal behaves [50, 58]. Hence, when two animals confront each other inside of such distance, the most dominant individual prevails as the other withdraws. Figure 1 shows this process.

In the proposed algorithm, the historic memory \mathbf{M}_h is updated considering the following procedure.

- (1) The elements of \mathbf{M}_h and \mathbf{M}_g are merged into $\mathbf{M}_U (\mathbf{M}_U = \mathbf{M}_h \cup \mathbf{M}_g)$.
- (2) Each element \mathbf{m}_U^i of the memory \mathbf{M}_U is compared pairwise with the remainder memory elements ($\{\mathbf{m}_U^1, \mathbf{m}_U^2, \dots, \mathbf{m}_U^{2B-1}\}$). If the distance between both elements is less than ρ , the element holding a better performance in the fitness function will prevail; meanwhile the other will be removed.
- (3) From the resulting elements of \mathbf{M}_U (as they are obtained in Step 2), the B best value is selected to integrate the new \mathbf{M}_h .

Unsuitable values of ρ result in a lower convergence rate, longer computation time, larger function evaluation number, convergence to a local maximum, or unreliability of solutions. The ρ value is computed considering the following equation:

$$\rho = \frac{\prod_{j=1}^D (a_j^{\text{high}} - a_j^{\text{low}})}{10 \cdot D}, \quad (5)$$

where a_j^{low} and a_j^{high} represent the prespecified lower bound and the upper bound of the j -parameter, respectively, within a D -dimensional space.

3.1.6. Computational Procedure. The computational procedure for the proposed algorithm can be summarized as follows.

Step 1. Set the parameters N_p , B , H , P , and NI .

Step 2. Generate randomly the position set $\mathbf{A} = \{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_{N_p}\}$ using (1).

Step 3. Sort \mathbf{A} , according to the objective function (dominance), building $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{N_p}\}$.

Step 4. Choose the first B positions of \mathbf{X} and store them into the memory \mathbf{M}_g .

Step 5. Update \mathbf{M}_h according to Section 3.1.5 (for the first iteration $\mathbf{M}_h = \mathbf{M}_g$).

Step 6. Generate the first B positions of the new solution set $\mathbf{A} = \{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_B\}$. Such positions correspond to elements of \mathbf{M}_h making a slight random perturbation around them: $\mathbf{a}_i = \mathbf{m}_h^i + \mathbf{v}$, \mathbf{v} being a random vector holding an appropriate small length.

Step 7. Generate the rest of the \mathbf{A} elements using the attraction, repulsion, and random movements:

```

for  $i = B + 1 : N_p$ 
  if ( $r_1 < 1 - P$ ) then
    attraction and repulsion movement
    {if ( $r_2 < H$ ) then
       $\mathbf{a}_i = \mathbf{x}_i \pm r \cdot (\mathbf{m}_h^{\text{nearest}} - \mathbf{x}_i)$ 
    else if
       $\mathbf{a}_i = \mathbf{x}_i \pm r \cdot (\mathbf{m}_g^{\text{nearest}} - \mathbf{x}_i)$ 
    }
  else if
    random movement
    {
       $\mathbf{a}_i = \mathbf{r}$ 
    }
end for
where  $r_1, r_2, r \in \text{rand}(0, 1)$ .

```

Step 8. If NI is completed, the process is thus completed; otherwise go back to Step 3.

3.1.7. Optima Determination. Just after the optimization process has finished, an analysis of the final \mathbf{M}_h memory is executed in order to find the global and significant local minima. For it, a threshold value Th is defined to decide which elements will be considered as a significant local minimum. Such threshold is thus computed as

$$\text{Th} = \frac{\max_{\text{fitness}}(\mathbf{M}_h)}{6}, \quad (6)$$

where $\max_{\text{fitness}}(\mathbf{M}_h)$ represents the best fitness value among \mathbf{M}_h elements. Therefore, memory elements whose fitness values are greater than Th will be considered as global and local optima as other elements are discarded.

3.1.8. Capacities of CAB and Differences with PSO. Evolutionary algorithms (EAs) have been widely employed for solving complex optimization problems. These methods are found to be more powerful than conventional methods based on formal logics or mathematical programming [59]. Exploitation and exploration are two main features of the EA [60]. The exploitation phase searches around the current best solutions and selects the best candidates or solutions. The exploration phase ensures that the algorithm seeks the search space more efficiently in order to analyze potential unexplored areas.

The EAs do not have limitations in using different sources of inspiration (e.g., music-inspired [11] or physic-inspired charged system search [9]). However, nature is a principal inspiration for proposing new metaheuristic approaches, and the nature-inspired algorithms have been widely used in developing systems and solving problems [61]. Biologically inspired algorithms are one of the main categories of the nature-inspired metaheuristic algorithms. The efficiency of the bio inspired algorithms is due to their significant ability to imitate the best features in nature. More specifically, these algorithms are based on the selection of the most suitable elements in biological systems which have evolved by natural selection.

Particle swarm optimization (PSO) is undoubtedly one of the most employed EA methods that use biologically inspired concepts in the optimization procedure. Unfortunately, like other stochastic algorithms, PSO also suffers from the premature convergence [62], particularly in multi modal problems. Premature convergence, in PSO, is produced by the strong influence of the best particle in the evolution process. Such particle is used by the PSO movement equations as a main individual in order to attract other particles. Under such conditions, the exploitation phase is privileged by allowing the evaluation of new search position around the best individual. However, the exploration process is seriously damaged, avoiding searching in unexplored areas.

As an alternative to PSO, the proposed scheme modifies some evolution operators for allowing not only attracting but also repelling movements among particles. Likewise, instead of considering the best position as reference, our algorithm uses a set of neighboring elements that are contained in an incorporated memory. Such improvements allow increasing the algorithm's capacity to explore and to exploit the set of solutions which are operated during the evolving process.

In the proposed approach, in order to improve the balance between exploitation and exploration, we have introduced three new concepts. The first one is the "attracting and repelling movement", which outlines that one particle cannot be only attracted but also repelled. The application of this concept to the evolution operators (3) increases the capacity of the proposed algorithm to satisfactorily explore the search space. Since the process of attraction or repulsion of each particle is randomly determined, the possibility of

premature convergence is very low, even for cases that hold an exaggerated number of local minima (excessive number of multimodal functions).

The second concept is the use of the main individual. In the approach, the main individual, that is considered as pivot in the equations (in order to generate attracting and repulsive movements), is not the best (as in PSO) but one element ($\mathbf{m}_h^{\text{nearest}}$ or $\mathbf{m}_g^{\text{nearest}}$) of a set which is contained in memories that store the best individual seen so far. Such pivot is the nearest element in memory with regard to the individual whose position is necessary to evolve. Under such conditions, the points considered to prompt the movement of a new individual are multiple. Such fact allows to maintain a balance between exploring new positions and exploiting the best positions seen so far.

Finally, the third concept is the use of an incorporated memory which stores the best individuals seen so far. As it has been discussed in Section 3.1.5, each candidate individual to be stored in the memory must compete with elements already contained in the memory in order to demonstrate that such new point is relevant. For the competition, the distance between each individual and the elements in the memory is used to decide pair-wise which individuals are actually considered. Then, the individual with better fitness value prevails whereas its pair is discarded. The incorporation of such concept allows simultaneous registering and refining of the best-individual set seen so far. This fact guarantees a high precision for final solutions of the multi-modal landscape through an extensive exploitation of the solution set.

3.1.9. Numerical Example. In order to demonstrate the algorithm's step-by-step operation, a numerical example has been set by applying the proposed method to optimize a simple function which is defined as follows:

$$f(x_1, x_2) = e^{-((x_1-4)^2 - (x_2-4)^2)} + e^{-((x_1+4)^2 - (x_2-4)^2)} + 2 \cdot e^{-((x_1)^2 + (x_2)^2)} + 2 \cdot e^{-((x_1)^2 - (x_2+4)^2)}. \quad (7)$$

Considering the interval of $-5 \leq x_1, x_2 \leq 5$, the function possesses two global maxima of value 2 at $(x_1, x_2) = (0, 0)$ and $(0, -4)$. Likewise, it holds two local minima of value 1 at $(-4, 4)$ and $(4, 4)$. Figure 2(a) shows the 3D plot of this function. The parameters for the CAB algorithm are set as $N_p = 10$, $B = 4$, $H = 0.8$, $P = 0.1$, $\rho = 3$, and $NI = 30$.

Like all evolutionary approaches, CAB is a population-based optimizer that attacks the starting point problem by sampling the objective function at multiple, randomly chosen, initial points. Therefore, after setting parameter bounds that define the problem domain, 10 (N_p) individuals ($\mathbf{i}_1, \mathbf{i}_2, \dots, \mathbf{i}_{10}$) are generated using (1). Following an evaluation of each individual through the objective function (7), all are sorted decreasingly in order to build vector $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{10})$. Figure 2(b) depicts the initial individual distribution in the search space. Then, both memories $\mathbf{M}_g (\mathbf{m}_g^1, \dots, \mathbf{m}_g^4)$ and $\mathbf{M}_h (\mathbf{m}_h^1, \dots, \mathbf{m}_h^4)$ are filled with the first four (B) elements present in \mathbf{X} . Such memory elements are represented by solid points in Figure 2(c).

The new 10 individuals ($\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_{10}$) are evolved at each iteration following three different steps: (1) keep the position of best individuals, (2) move from or nearby neighbors, and (3) move randomly. The first new four elements ($\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4$) are generated considering the first step (keeping the position of best individuals). Following such step, new individual positions are calculated as perturbed versions of all the elements which are contained in the \mathbf{M}_h memory (that represent the best individuals known so far). Such perturbation is done by using $\mathbf{a}_l = \mathbf{m}_h^l + \mathbf{v}$ ($l \in 1, \dots, 4$). Figure 2(d) shows a comparative view between the memory element positions and the perturbed values of ($\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3, \mathbf{a}_4$).

The remaining 6 new positions ($\mathbf{a}_5, \dots, \mathbf{a}_{10}$) are individually computed according to Steps 2 and 3 of the numerical example. For such operation, a uniform random number r_1 is generated within the range $[0, 1]$. If r_1 is less than $1 - P$, the new position \mathbf{a}_j ($j \in 5, \dots, 10$) is generated through Step 2; otherwise, \mathbf{a}_j is obtained from a random reinitialization (Step 3) between search bounds.

In order to calculate a new position \mathbf{a}_j at Step 2, a decision must be made on whether it should be generated by using the elements of \mathbf{M}_h or \mathbf{M}_g . For such decision, a uniform random number r_2 is generated within the range $[0, 1]$. If r_2 is less than H , the new position \mathbf{a}_j is generated by using $\mathbf{x}_j \pm r \cdot (\mathbf{m}_h^{\text{nearest}} - \mathbf{x}_j)$; otherwise, \mathbf{a}_j is obtained by considering $\mathbf{x}_j \pm r \cdot (\mathbf{m}_g^{\text{nearest}} - \mathbf{x}_j)$, where $\mathbf{m}_h^{\text{nearest}}$ and $\mathbf{m}_g^{\text{nearest}}$ represent the closest elements to \mathbf{x}_j in memory \mathbf{M}_h and \mathbf{M}_g , respectively. In the first iteration, since there is not available information from previous steps, both memories \mathbf{M}_h and \mathbf{M}_g share the same information which is only allowed at this initial stage. Figure 2(e) shows graphically the whole procedure employed by Step 2 in order to calculate the new individual position \mathbf{a}_8 whereas Figure 2(f) presents the positions of all new individuals ($\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_{10}$).

Finally, after all new positions ($\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_{10}$) have been calculated, memories \mathbf{M}_h and \mathbf{M}_g must be updated. In order to update \mathbf{M}_h , new calculated positions ($\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_{10}$) are arranged according to their fitness values by building vector $\mathbf{X} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{10})$. Then, the elements of \mathbf{M}_h are replaced by the first four elements in \mathbf{X} (the best individuals of its generation). In order to calculate the new elements of \mathbf{M}_h , current elements of \mathbf{M}_h (the present values) and \mathbf{M}_g (the updated values) are merged into \mathbf{M}_U . Then, by using the dominance concept (explained in Section 3.1.5) over \mathbf{M}_U , the best four values are selected to replace the elements in \mathbf{M}_g . Figures 2(g) and 2(h) show the updating procedure for both memories. Applying the dominance (see Figure 2(g)), since the distances $a = \text{dist}(\mathbf{m}_h^3, \mathbf{m}_g^4)$, $b = \text{dist}(\mathbf{m}_h^2, \mathbf{m}_g^3)$, and $c = \text{dist}(\mathbf{m}_h^1, \mathbf{m}_g^1)$ are less than $\rho = 3$, elements with better fitness evaluation will build the new memory \mathbf{M}_h . Figure 2(h) depicts final memory configurations. The circles and solid circles points represent the elements of \mathbf{M}_g and \mathbf{M}_h , respectively, whereas the bold squares perform as elements shared by both memories. Therefore, if the complete procedure is repeated over 30 iterations, the memory \mathbf{M}_h will contain the 4 global and local maxima as elements. Figure 2(i) depicts the final configuration after 30 iterations.

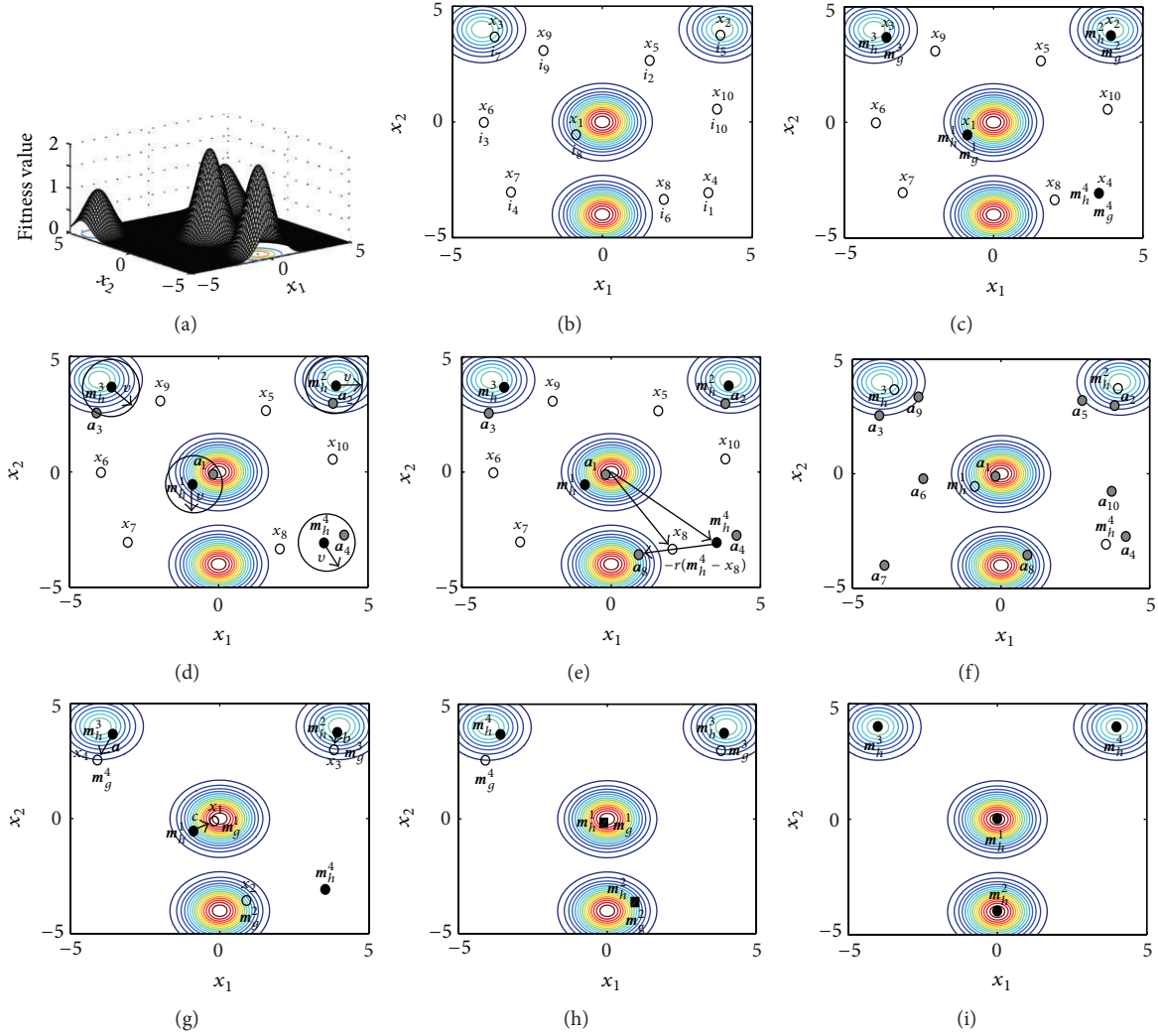


FIGURE 2: CAB numerical example. (a) 3D plot of the function used as example. (b) Initial individual distribution. (c) Initial configuration of memories M_g and M_h . (d) The computation of the first four individuals (a_1, a_2, a_3, a_4). (e) It shows the procedure employed by Step 2 in order to calculate the new individual position a_8 . (f) Positions of all new individuals (a_1, a_2, \dots, a_{10}). (g) Application of the dominance concept over elements of M_g and M_h . (h) Final memory configurations of M_g and M_h after the first iteration. (i) Final memory configuration of M_h after 30 iterations.

4. Results on Multimodal Benchmark Functions

In this section, the performance of the proposed algorithm is tested. Section 4.1 describes the experiment methodology. Sections 4.2 and 4.3 report on a comparison between the CAB experimental results and other multimodal metaheuristic algorithms for different kinds of optimization problems.

4.1. Experiment Methodology. In this section, we will examine the search performance of the proposed CAB by using a test suite of 8 benchmark functions with different complexities. They are listed in Tables 1 and 2. The suite mainly contains some representative, complicated, and multimodal functions with several local optima. These functions are normally regarded as difficult to be optimized as they are particularly challenging to the applicability and efficiency of multimodal

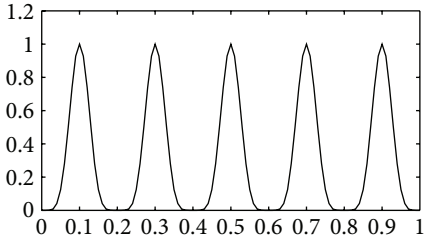
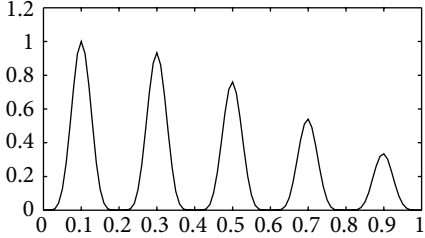
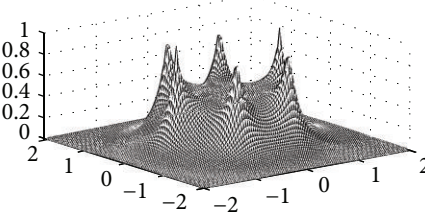
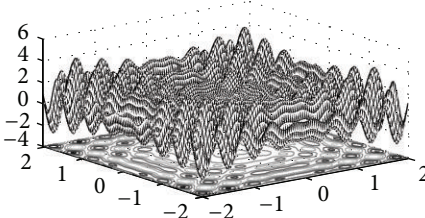
metaheuristic algorithms. The performance measurements considered at each experiment are the following:

- (i) the consistency of locating all known optima;
- (ii) the averaged number of objective function evaluations that are required to find such optima (or the running time under the same condition).

The experiments compare the performance of CAB against the deterministic crowding [17], the probabilistic crowding [18], the sequential fitness sharing [15], the clearing procedure [20], the clustering based niching (CBN) [19], the species conserving genetic algorithm (SCGA) [21], the elitist-population strategy (AEGA) [22], the clonal selection algorithm [24], and the artificial immune network (AiNet) [25].

Since the approach solves real-valued multimodal functions, we have used, in the GA approaches, consistent real coding variable representation, uniform crossover, and

TABLE 1: The test suite of multimodal functions for Experiment 4.2.

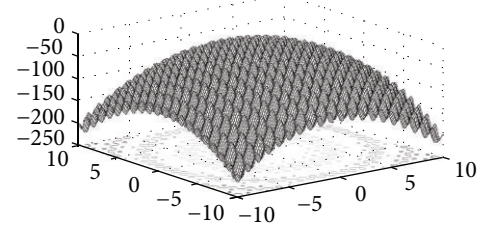
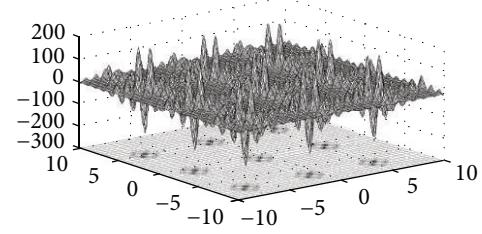
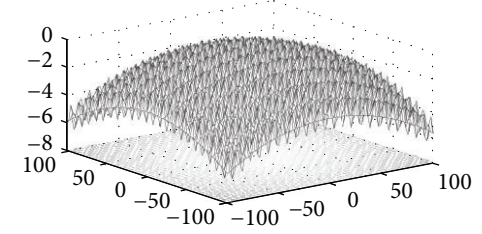
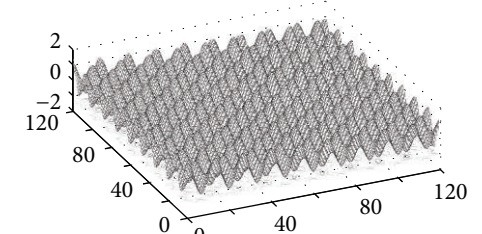
Function	Search space	Sketch
Deb's function 5 optima $f_1 = \sin^6(5\pi x)$	$x \in [0, 1]$	
Deb's decreasing function 5 optima $f_2(x) = 2^{-2((x-0.1)/0.9)^2} \cdot \sin(5\pi x)$	$x \in [0, 1]$	
Roots function 6 optima $f_3(z) = \frac{1}{1 + z^6 + 1 }$	$z \in C, z = x_1 + ix_2$ $x_1, x_2 \in [-2, 2]$	
Two dimensional multi-modal function 100 optima $f_4(x_1, x_2) = x_1 \sin(4\pi x_1) - x_2 \sin(4\pi x_2 + \pi) + 1$	$x_1, x_2 \in [-2, 2]$	

mutation operators for each algorithm seeking a fair comparison. The crossover probability $P_c = 0.8$ and the mutation probability $P_m = 0.1$ have been used. We use the standard tournament selection operator with a tournament size of 2 in our implementation of sequential fitness sharing, clearing procedure, CBN, clonal selection algorithm, and SCGA. On

the other hand, the parameter values for the AiNet algorithm have been defined as suggested in [25], with the mutation strength $\beta = 100$, the suppression threshold $\sigma_{s(aiNet)} = 0.2$, and the update rate $d = 40\%$.

In the case of the CAB algorithm, the parameters are set to $N_p = 200$, $B = 100$, $P = 0.8$, and $H = 0.6$. Once they have

TABLE 2: The test suite of multimodal functions used in the Experiment 4.3.

Function	Search space	Sketch
Rastrigin's function 100 optima		
$f_5(x_1, x_2) = -(20 + x_1^2 + x_2^2 - 10(\cos(2\pi x_1) + \cos(2\pi x_2)))$	$x_1, x_2 \in [-10, 10]$	
Shubert function 18 optima		
$f_6(x_1, x_2) = -\prod_{i=1}^2 \sum_{j=1}^5 \cos((j+1)x_i + j)$	$x_1, x_2 \in [-10, 10]$	
Griewank function 100 optima		
$f_7(x_1, x_2) = \frac{1}{4000} \sum_{i=1}^2 x_i^2 - \prod_{i=1}^2 \cos\left(\frac{x_i}{\sqrt{2}}\right) + 1$	$x_1, x_2 \in [-100, 100]$	
Modified Griewank function 100 optima		
$f_8(x_1, x_2) = \frac{\cos(0.5x_1) + \cos(0.5x_2)}{4000} + \cos(10x_1) \cos(10x_2)$	$x_1, x_2 \in [0, 120]$	

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 given in the literature. An optimum o_j is considered as found if $\exists x_i \in \text{Pop}(k = T) \mid d(x_i, o_j) < 0.005$, where $\text{Pop}(k = T)$ is the complete population at the end of the run T and x_i is an individual in $\text{Pop}(k = T)$.

All algorithms have been tested in MATLAB over the same Dell OptiPlex GX260 computer with a Pentium 4 2.66 G HZ processor, running Windows XP operating system over 1 Gb of memory. Next sections present experimental results for multimodal optimization problems which have been divided into two groups with different purposes. The first one consists of functions with smooth landscapes and well-defined optima (local and global values), while the second gathers functions holding rough landscapes and complex location optima.

4.2. Comparing CAB Performance for Smooth Landscapes Functions. This section presents a performance comparison for different algorithms solving multimodal problems f_1 – f_4 in Table 1. The aim is to determine whether CAB is more efficient and effective than other existing algorithms for finding all multiple optima of f_1 – f_4 . The stopping criterion analyzes if the number-identified optima cannot be further increased over 10 successive generations after the first 100 generations; then the execution will be stopped. Four measurements have been employed to evaluate the performance:

- (i) the average of optima found within the final population (NO);
- (ii) the average distance between multiple optima detected by the algorithm and their closest individuals in the final population (DO);
- (iii) the average of function evaluations (FE);
- (iv) the average of execution time in seconds (ET).

Table 3 provides a summarized performance comparison among several algorithms. Best results have been bold faced. From the NO measure, CAB always finds better or equally optimal solutions for the multimodal problems f_1 – f_4 . It is evident that each algorithm can find all optima of f_1 . For function f_2 , only AEGA, clonal selection algorithm, aiNet, and CAB can eventually find all optima each time. For function f_3 , clearing procedure, SCGA, AEGA, and CAB can get all optima at each run. For function f_4 , deterministic crowding leads to premature convergence and all other algorithms cannot get any better results, but CAB yet can find all multiple optima 48 times in 50 runs and its average successful rate for each run is higher than 99%. By analyzing the DO measure in Table 3, CAB has obtained the best score for all the multimodal problems except for f_3 . In the case of f_3 , the solution precision of CAB is only worse than that of clearing procedure. On the other hand, CAB has smaller standard deviations in the NO and DO measures than all other algorithms and hence its solution is more stable.

From the FE measure in Table 3, it is clear that CAB needs fewer function evaluations than other algorithms considering the same termination criterion. Recall that all algorithms use the same conventional crossover and mutation operators. It can be easily deduced from results that the CAB algorithm is able to produce better search positions (better compromise between exploration and exploitation), in a more efficient and effective way than other multimodal search strategies.

To validate that CAB improvement over other algorithms occurs as a result of CAB producing better search positions over iterations, Figure 3 shows the comparison of CAB and other multimodal algorithms for f_4 . The initial populations for all algorithms have 200 individuals. In the final population of CAB, the 100 individuals belonging to the M_h memory correspond to the 100 multiple optima, while, on the contrary, the final population of the other nine algorithms fail consistently in finding all optima, despite that they have superimposed several times over some previously found optima.

When comparing the execution time (ET) in Table 3, CAB uses significantly less time to finish than other algorithms. The situation can be registered by the reduction of the redundancy in the M_h memory due to competition (dominance) criterion. All these comparisons show that CAB generally outperforms all other multimodal algorithms regarding efficacy and efficiency.

4.3. Comparing CAB Performance in Rough Landscapes Functions. This section presents the performance comparison among different algorithms solving multimodal optimization problems which are listed in Table 2. Such problems hold lots of local optima and very rugged landscapes. The goal of multimodal optimizers is to find as many global optima as possible and possibly good local optima. Rastrigin's function f_5 and Griewank's function f_7 have 1 and 18 global optima, respectively, becoming practical as to test whether a multimodal algorithm can find a global optimum and at least 80 higher fitness local optima to validate the algorithms' performance.

Our main objective in these experiments is to determine whether CAB is more efficient and effective than other existing algorithms for finding the multiple high fitness optima of functions f_5 – f_8 . In the experiments, the initial population size for all algorithms has been set to 1000. For sequential fitness sharing, clearing procedure, CBN, clonal selection, SCGA, and AEGA, we have set the distance threshold σ_s to 5. The algorithms' stopping criterion checks whenever the number of optima found cannot be further increased in 50 successive generations after the first 500 generations. If such condition prevails, then the algorithm is halted. We still evaluate the performance of all algorithms using the aforementioned four measures NO, DO, FE, and ET.

Table 4 provides a summary of the performance comparison among different algorithms. From the NO measure, we observe that CAB could always find more optimal solutions for the multimodal problems f_5 – f_8 . For Rastrigin's function f_5 , only CAB can find all multiple high fitness optima 49 times out of 50 runs and its average successful rate for each run is higher than 97%. On the contrary, other algorithms cannot find all multiple higher fitness optima for any run. For f_6 , 5 algorithms (clearing procedure, SCGA, AEGA, clonal selection algorithm, AiNet, and CAB) can get all multiple higher fitness maxima for each run, respectively. For Griewank's function (f_7), only CAB can get all multiple higher fitness optima for each run. In case of the modified Griewank's function (f_8), it has numerous optima whose value is always

TABLE 3: Performance comparison among the multimodal optimization algorithms for the test functions f_1 – f_4 . The standard unit in the column ET is seconds. (For all the parameters, numbers in parentheses are the standard deviations.) Bold faced letters represent best obtained results.

Function	Algorithm	NO	DO	FE	ET
f_1	Deterministic crowding	5 (0)	1.52×10^{-4} (1.38×10^{-4})	7,153 (358)	0.091 (0.013)
	Probabilistic crowding	5 (0)	3.63×10^{-4} (6.45×10^{-5})	10,304 (487)	0.163 (0.011)
	Sequential fitness sharing	5 (0)	4.76×10^{-4} (6.82×10^{-5})	9,927 (691)	0.166 (0.028)
	Clearing procedure	5 (0)	1.27×10^{-4} (2.13×10^{-5})	5,860 (623)	0.128 (0.021)
	CBN	5 (0)	2.94×10^{-4} (4.21×10^{-5})	10,781 (527)	0.237 (0.019)
	SCGA	5 (0)	1.16×10^{-4} (3.11×10^{-5})	6,792 (352)	0.131 (0.009)
	AEGA	5 (0)	4.6×10^{-5} (1.35×10^{-5})	2,591 (278)	0.039 (0.007)
	Clonal selection algorithm	5 (0)	1.99×10^{-4} (8.25×10^{-5})	15,803 (381)	0.359 (0.015)
	AiNet	5 (0)	1.28×10^{-4} (3.88×10^{-5})	12,369 (429)	0.421 (0.021)
	CAB	5 (0)	1.69×10^{-5} (5.2×10^{-6})	1,776 (125)	0.020 (0.009)
f_2	Deterministic crowding	3.53 (0.73)	3.61×10^{-3} (6.88×10^{-4})	6,026 (832)	0.271 (0.06)
	Probabilistic crowding	4.73 (0.64)	2.82×10^{-3} (8.52×10^{-4})	10,940 (9517)	0.392 (0.07)
	Sequential fitness sharing	4.77 (0.57)	2.33×10^{-3} (4.36×10^{-4})	12,796 (1,430)	0.473 (0.11)
	Clearing procedure	4.73 (0.58)	4.21×10^{-3} (1.24×10^{-3})	8,465 (773)	0.326 (0.05)
	CBN	4.70 (0.53)	2.19×10^{-3} (4.53×10^{-4})	14,120 (2,187)	0.581 (0.14)
	SCGA	4.83 (0.38)	3.15×10^{-3} (4.71×10^{-4})	10,548 (1,382)	0.374 (0.09)
	AEGA	5 (0)	1.38×10^{-4} (2.32×10^{-5})	3,605 (426)	0.102 (0.04)
	Clonal selection algorithm	5 (0)	1.37×10^{-3} (6.87×10^{-4})	21,922 (746)	0.728 (0.06)
	AiNet	5 (0)	1.22×10^{-3} (5.12×10^{-4})	18,251 (829)	0.664 (0.08)
	CAB	5 (0)	4.5×10^{-5} (8.56×10^{-6})	2,065 (92)	0.08 (0.007)
f_3	Deterministic crowding	4.23 (1.17)	7.79×10^{-4} (4.76×10^{-4})	11,009 (1,137)	1.07 (0.13)
	Probabilistic crowding	4.97 (0.64)	2.35×10^{-3} (7.14×10^{-4})	16,391 (1,204)	1.72 (0.12)
	Sequential fitness sharing	4.87 (0.57)	2.56×10^{-3} (2.58×10^{-3})	14,424 (2,045)	1.84 (0.26)
	Clearing procedure	6 (0)	7.43×10^{-5} (4.07×10^{-5})	12,684 (1,729)	1.59 (0.19)
	CBN	4.73 (1.14)	1.85×10^{-3} (5.42×10^{-4})	18,755 (2,404)	2.03 (0.31)
	SCGA	6 (0)	3.27×10^{-4} (7.46×10^{-5})	13,814 (1,382)	1.75 (0.21)
	AEGA	6 (0)	1.21×10^{-4} (8.63×10^{-5})	6,218 (935)	0.53 (0.07)
	Clonal selection algorithm	5.50 (0.51)	4.95×10^{-3} (1.39×10^{-3})	25,953 (2,918)	2.55 (0.33)
	AiNet	4.8 (0.33)	3.89×10^{-3} (4.11×10^{-4})	20,335 (1,022)	2.15 (0.10)
	CAB	6 (0)	9.87×10^{-5} (1.69×10^{-5})	4,359 (75)	0.11 (0.023)
f_4	Deterministic crowding	76.3 (11.4)	4.52×10^{-3} (4.17×10^{-3})	1,861,707 (329,254)	21.63 (2.01)
	Probabilistic crowding	92.8 (3.46)	3.46×10^{-3} (9.75×10^{-4})	2,638,581 (597,658)	31.24 (5.32)
	Sequential fitness sharing	89.9 (5.19)	2.75×10^{-3} (6.89×10^{-4})	2,498,257 (374,804)	28.47 (3.51)
	Clearing procedure	89.5 (5.61)	3.83×10^{-3} (9.22×10^{-4})	2,257,964 (742,569)	25.31 (6.24)
	CBN	90.8 (6.50)	4.26×10^{-3} (1.14×10^{-3})	2,978,385 (872,050)	35.27 (8.41)
	SCGA	91.4 (3.04)	3.73×10^{-3} (2.29×10^{-3})	2,845,789 (432,117)	32.15 (4.85)
	AEGA	95.8 (1.64)	1.44×10^{-4} (2.82×10^{-5})	1,202,318 (784,114)	12.17 (2.29)
	Clonal selection algorithm	92.1 (4.63)	4.08×10^{-3} (8.25×10^{-3})	3,752,136 (191,849)	45.95 (1.56)
	AiNet	93.2 (7.12)	3.74×10^{-3} (5.41×10^{-4})	2,745,967 (328,176)	38.18 (3.77)
	CAB	100 (2)	2.31×10^{-5} (5.87×10^{-6})	697,578 (57,089)	5.78 (1.26)

the same. However, CAB still can find all global optima with an effectiveness rate of 95%.

From the FE and ET measures in Table 4, we can clearly observe that CAB uses significantly fewer function evaluations and a shorter running time than all other algorithms under the same termination criterion. Moreover, deterministic crowding leads to premature convergence as CAB is at least 2.5, 3.8, 4, 3.1, 4.1, 3.7, 1.4, 7.9, and 4.9 times faster than

all others, respectively, according to Table 4 for functions f_5 – f_8 .

5. Application of CAB in Multicircle Detection

5.1. Individual Representation. In order to detect circle shapes, candidate images must be preprocessed first by the well-known Canny algorithm which yields a single-pixel

TABLE 4: Performance comparison among multimodal optimization algorithms for the test functions $f_5 - f_8$. The standard unit of the column ET is seconds (numbers in parentheses are standard deviations). Bold faced letters represent best results.

Function	Algorithm	NO	DO	FE	ET
f_5	Deterministic crowding	62.4 (14.3)	4.72×10^{-3} (4.59×10^{-3})	1,760,199 (254,341)	14.62 (2.83)
	Probabilistic crowding	84.7 (5.48)	1.50×10^{-3} (9.38×10^{-4})	2,631,627 (443,522)	34.39 (5.20)
	Sequential fitness sharing	76.3 (7.08)	3.51×10^{-3} (1.66×10^{-3})	2,726,394 (562,723)	36.55 (7.13)
	Clearing procedure	93.6 (2.31)	2.78×10^{-3} (1.20×10^{-3})	2,107,962 (462,622)	28.61 (6.47)
	CBN	87.9 (7.78)	4.33×10^{-3} (2.82×10^{-3})	2,835,119 (638,195)	37.05 (8.23)
	SCGA	97.4 (4.80)	1.34×10^{-3} (8.72×10^{-4})	2,518,301 (643,129)	30.27 (7.04)
	AEGA	99.4 (1.39)	6.77×10^{-4} (3.18×10^{-4})	978,435 (71,135)	10.56 (4.81)
	Clonal selection algorithm	90.6 (9.95)	3.15×10^{-3} (1.47×10^{-3})	5,075,208 (194,376)	58.02 (2.19)
	AiNet	93.8 (7.8)	2.11×10^{-3} (3.2×10^{-3})	3,342,864 (549,452)	51.65 (6.91)
	CAB	100 (2)	2.22×10^{-4} (3.1×10^{-5})	680,211 (12,547)	7.33 (1.84)
f_6	Deterministic crowding	9.37 (1.91)	3.26×10^{-3} (5.34×10^{-4})	832,546 (75,413)	4.58 (0.57)
	Probabilistic crowding	15.17 (2.43)	2.87×10^{-3} (5.98×10^{-4})	1,823,774 (265,387)	12.92 (2.01)
	Sequential fitness sharing	15.29 (2.14)	1.42×10^{-3} (5.29×10^{-4})	1,767,562 (528,317)	14.12 (3.51)
	Clearing procedure	18 (0)	1.19×10^{-3} (6.05×10^{-4})	1,875,729 (265,173)	11.20 (2.69)
	CBN	14.84 (2.70)	4.39×10^{-3} (2.86×10^{-3})	2,049,225 (465,098)	18.26 (4.41)
	SCGA	4.83 (0.38)	1.58×10^{-3} (4.12×10^{-4})	2,261,469 (315,727)	13.71 (1.84)
	AEGA	18 (0)	3.34×10^{-4} (1.27×10^{-4})	656,639 (84,213)	3.12 (1.12)
	Clonal selection algorithm	18 (0)	3.42×10^{-3} (1.58×10^{-3})	4,989,856 (618,759)	33.85 (5.36)
	AiNet	18 (0)	2.11×10^{-3} (3.31×10^{-3})	3,012,435 (332,561)	26.32 (2.54)
	CAB	18 (0)	1.02×10^{-4} (4.27×10^{-5})	431,412 (21,034)	2.21 (0.51)
f_7	Deterministic crowding	52.6 (8.86)	3.71×10^{-3} (1.54×10^{-3})	2,386,960 (221,982)	19.10 (2.26)
	Probabilistic crowding	79.2 (4.94)	3.48×10^{-3} (3.79×10^{-3})	3,861,904 (457,862)	43.53 (4.38)
	Sequential fitness sharing	63.0 (5.49)	4.76×10^{-3} (3.55×10^{-3})	3,619,057 (565,392)	42.98 (6.35)
	Clearing procedure	79.4 (4.31)	2.95×10^{-3} (1.64×10^{-3})	3,746,325 (594,758)	45.42 (7.64)
	CBN	71.3 (9.26)	3.29×10^{-3} (4.11×10^{-3})	4,155,209 (465,613)	48.23 (5.42)
	SCGA	94.9 (8.18)	2.63×10^{-3} (1.81×10^{-3})	3,629,461 (373,382)	47.84 (0.21)
	AEGA	98 (2)	1.31×10^{-3} (8.76×10^{-4})	1,723,342 (121,043)	12.54 (1.31)
	Clonal selection algorithm	89.2 (5.44)	3.02×10^{-3} (1.63×10^{-3})	5,423,739 (231,004)	47.84 (6.09)
	AiNet	92.7 (3.21)	2.79×10^{-3} (3.19×10^{-4})	4,329,783 (167,932)	41.64 (2.65)
	CAB	100 (1)	3.32×10^{-4} (5.25×10^{-5})	953,832 (9,345)	8.82 (1.51)
f_8	Deterministic crowding	44.2 (7.93)	4.45×10^{-3} (3.63×10^{-3})	2,843,452 (353,529)	23.14 (3.85)
	Probabilistic crowding	70.1 (8.36)	2.52×10^{-3} (1.47×10^{-3})	4,325,469 (574,368)	49.51 (6.72)
	Sequential fitness sharing	58.2 (9.48)	4.14×10^{-3} (3.31×10^{-3})	4,416,150 (642,415)	54.43 (12.6)
	Clearing procedure	67.5 (10.11)	2.31×10^{-3} (1.43×10^{-3})	4,172,462 (413,537)	52.39 (7.21)
	CBN	53.1 (7.58)	4.36×10^{-3} (3.53×10^{-3})	4,711,925 (584,396)	61.07 (8.14)
	SCGA	87.3 (9.61)	3.15×10^{-3} (2.07×10^{-3})	3,964,491 (432,117)	53.87 (8.46)
	AEGA	90.6 (1.65)	2.55×10^{-3} (9.55×10^{-4})	2,213,754 (412,538)	16.21 (3.19)
	Clonal selection algorithm	74.4 (7.32)	3.52×10^{-3} (2.19×10^{-3})	5,835,452 (498,033)	74.26 (5.47)
	AiNet	83.2 (6.23)	3.11×10^{-3} (2.41×10^{-4})	4,123,342 (213,864)	60.38 (5.21)
	CAB	97 (2)	1.54×10^{-3} (4.51×10^{-4})	1,121,523 (51,732)	12.21 (2.66)

edge-only image. Then, the (x_i, y_i) coordinates for each edge pixel p_i are stored inside the edge vector $P = \{p_1, p_2, \dots, p_{N_p}\}$, with N_p being the total number of edge pixels. Each circle C uses three edge points as individuals in the optimization algorithm. In order to construct such individuals, three indexes p_i , p_j , and p_k are selected from vector P , considering the circle's contour that connects them. Therefore, the circle

$C = \{p_i, p_j, p_k\}$ that crosses over such points may be considered as a potential solution for the detection problem. Considering the configuration of the edge points shown by Figure 4, the circle center (x_0, y_0) and the radius r of C can be computed as follows:

$$(x - x_0)^2 + (y - y_0)^2 = r^2. \quad (8)$$

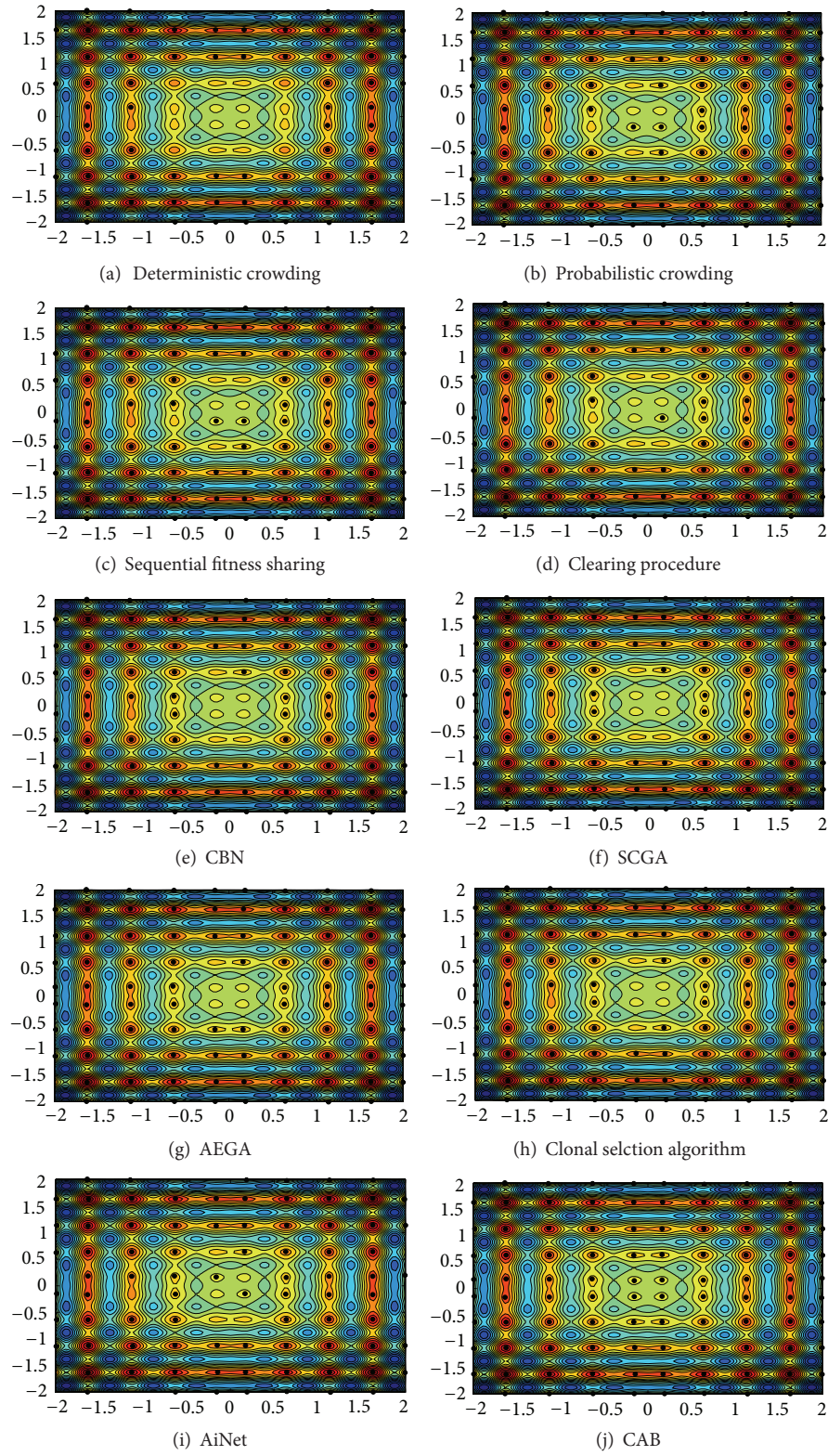


FIGURE 3: Typical results of the maximization of f_4 . (a)–(j) Local and global optima located by all ten algorithms in the performance comparison.

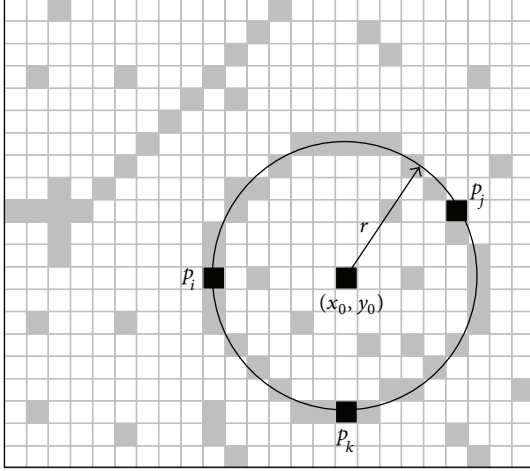


FIGURE 4: Circle candidate (individual) built from the combination of points p_i , p_j , and p_k .

Consider

$$\mathbf{A} = \begin{bmatrix} x_j^2 + y_j^2 - (x_i^2 + y_i^2) & 2 \cdot (y_j - y_i) \\ x_k^2 + y_k^2 - (x_i^2 + y_i^2) & 2 \cdot (y_k - y_i) \end{bmatrix}, \quad (9)$$

$$\mathbf{B} = \begin{bmatrix} 2 \cdot (x_j - x_i) & x_j^2 + y_j^2 - (x_i^2 + y_i^2) \\ 2 \cdot (x_k - x_i) & x_k^2 + y_k^2 - (x_i^2 + y_i^2) \end{bmatrix},$$

$$x_0 = \frac{\det(\mathbf{A})}{4((x_j - x_i)(y_k - y_i) - (x_k - x_i)(y_j - y_i))}, \quad (10)$$

$$y_0 = \frac{\det(\mathbf{B})}{4((x_j - x_i)(y_k - y_i) - (x_k - x_i)(y_j - y_i))},$$

$$r = \sqrt{(x_0 - x_d)^2 + (y_0 - y_d)^2}, \quad (11)$$

with $\det(\cdot)$ being the determinant and $d \in \{i, j, k\}$. Figure 2 illustrates the parameters defined by (8) to (11).

5.2. Objective Function. In order to calculate the error produced by a candidate solution C , a set of test points is calculated as a virtual shape which, in turn, must be validated, that is if it really exists in the edge image. The test set is represented by $S = \{s_1, s_2, \dots, s_{N_s}\}$, where N_s is the number of points over which the existence of an edge point, corresponding to C , should be validated. In our approach, the set S is generated by the midpoint circle algorithm (MCA) [63]. The MCA is a searching method which seeks the required points for drawing a circle digitally. Therefore MCA calculates the necessary number of test points N_s to totally draw the complete circle. Such a method is considered the fastest because MCA avoids computing square-root calculations by comparing the pixel separation distances among them.

The objective function $J(C)$ represents the matching error produced between the pixels S of the circle candidate C

(animal position) and the pixels that actually exist in the edge image, yielding

$$J(C) = 1 - \frac{\sum_{v=1}^{N_s} E(x_v, y_v)}{N_s}, \quad (12)$$

where $E(x_i, y_i)$ is a function that verifies the pixel existence in (x_v, y_v) , with $(x_v, y_v) \in S$, and N_s being the number of pixels lying on the perimeter corresponding to C currently under testing. Hence, function $E(x_v, y_v)$ is defined as

$$E(x_v, y_v) = \begin{cases} 1 & \text{if the pixel } (x_v, y_v) \text{ is an edge point} \\ 0 & \text{otherwise.} \end{cases} \quad (13)$$

A value near zero of $J(C)$ implies a better response from the “circularity” operator. Figure 5 shows the procedure to evaluate a candidate solution C with its representation as a virtual shape S . In Figure 5(b), the virtual shape is compared to the original image, point by point, in order to find coincidences between virtual and edge points. The virtual shape is built from points p_i , p_j , and p_k shown by Figure 5(a). The virtual shape S gathers 56 points ($N_s = 56$) with only 18 of such points existing in both images (shown as blue points plus red points in Figure 5(c)) yielding $\sum_{v=1}^{N_s} E(x_v, y_v) = 18$ and therefore $J(C) \approx 0.67$.

5.3. The Multiple-Circle Detection Procedure. In order to detect multiple circles, most detectors simply apply a one-minimum optimization algorithm, which is able to detect only one circle at a time, repeating the same process several times as previously detected primitives are removed from the image. Such algorithms iterate until there are no more candidates left in the image.

On the other hand, the method in this paper is able to detect single or multiples circles through only one optimization step. The multidetection procedure can be summarized as follows: guided by the values of a matching function, the whole group of encoded candidate circles is evolved through the set of evolutionary operators. The best circle candidate (global optimum) is considered to be the first detected circle over the edge-only image. An analysis of the historical memory \mathbf{M}_h is thus executed in order to identify other local optima (other circles).

In order to find other possible circles contained in the image, the historical memory \mathbf{M}_h is carefully examined. The approach aims to explore all elements, one at a time, assessing which of them represents an actual circle in the image. Since several elements can represent the same circle (i.e., circles slightly shifted or holding small deviations), a distinctiveness factor $D_{A,B}$ is required to measure the mismatch between two given circles (A and B). Such distinctiveness factor is defined as follows:

$$D_{A,B} = |x_A - x_B| + |y_A - y_B| + |r_A - r_B|, \quad (14)$$

with (x_A, y_A) and r_A being the central coordinates and radius of the circle C_A , respectively, while (x_B, y_B) and r_B represent

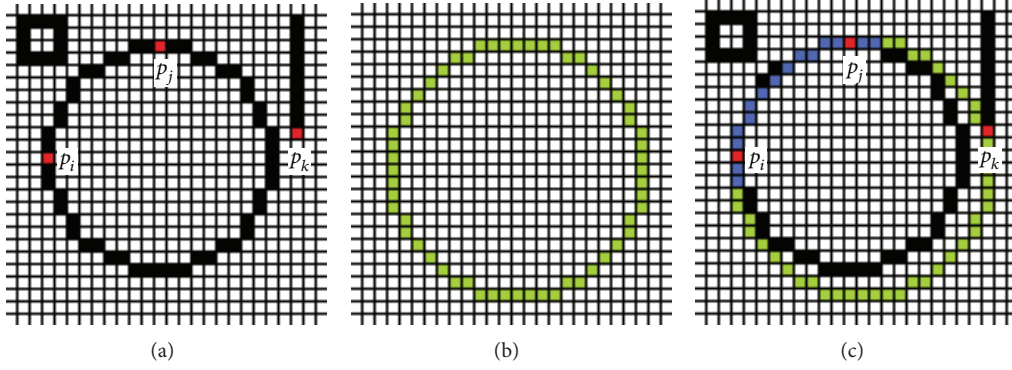


FIGURE 5: Evaluation of candidate solutions C : the image in (a) shows the original image while (b) presents the virtual shape generated including points p_i , p_j , and p_k . The image in (c) shows coincidences between both images marked by blue or red pixels while the virtual shape is also depicted in green.

the corresponding parameters of the circle C_B . One threshold value E_{sth} is also calculated to decide whether two circles must be considered different or not. Th is computed as:

$$Th = \frac{r_{\max} - r_{\min}}{d}, \quad (15)$$

where $[r_{\min}, r_{\max}]$ is the feasible radii's range and d is a sensitivity parameter. By using a high d value, two very similar circles would be considered different while a smaller value for d would consider them as similar shapes. In this work, after several experiments, the d value has been set to 2.

Thus, since the historical memory $M_h \{C_1^M, C_2^M, \dots, C_B^M\}$ groups the elements in descending order according to their fitness values, the first element C_1^M , whose fitness value represents the best value $J(C_1^M)$, is assigned to the first circle. Then, the distinctiveness factor ($D_{C_1^M, C_2^M}$) over the next element C_2^M is evaluated with respect to the prior C_1^M . If $D_{C_1^M, C_2^M} > Th$, then C_2^M is considered as a new circle; otherwise the next element C_3^M is selected. This process is repeated until the fitness value $J(C_i^M)$ reaches a minimum threshold J_{Th} . According to such threshold, other values above J_{Th} represent individuals (circles) that are considered significant while other values lying below such boundary are considered as false circles and hence they are not contained in the image. After several experiments the value of J_{Th} is set to $(J(C_1^M)/10)$.

The fitness value of each detected circle is characterized by its geometric properties. Big and well-drawn circles normally represent points in the search space with higher fitness values whereas small and dashed circles describe points with lower fitness values. Likewise, circles with similar geometric properties, such as radius and size tend to represent locations holding similar fitness values. Considering that the historical memory M_h groups the elements in descending order according to their fitness values, the proposed procedure allows the cancelling of those circles which belong to the same circle and hold a similar fitness value.

5.4. Implementation of CAB Strategy for Circle Detection. The implementation of the proposed algorithm can be summarized in the following steps.

Step 1. Adjust the algorithm parameters N_p , B , H , P , NI , and d .

Step 2. Randomly generate a set of N_p candidate circles (position of each animal) $C = \{C_1, C_2, \dots, C_{N_p}\}$ set using (1).

Step 3. Sort C according to the objective function (dominance) to build $X = \{x_1, x_2, \dots, x_{N_p}\}$.

Step 4. Choose the first B positions of X and store them into the memory M_g .

Step 5. Update M_h according to Section 3.1.5. (during the first iteration: $M_h = M_g$).

Step 6. Generate the first B positions of the new solution set $C(\{C_1, C_2, \dots, C_B\})$. Such positions correspond to the elements of M_h making a slight random perturbation around them:

$$C_i = m_h^i + v, \quad \text{with being } v \text{ a random vector of} \quad (16)$$

a small enough length.

Step 7. Generate the rest of the C elements using the attraction, repulsion, and random movements:

```

for  $i = B + 1 : N_p$ 
  if ( $r_1 < P$ ) then
    attraction and repulsion movement
    {if ( $r_2 < H$ ) then
       $C_i = x_i \pm r \cdot (m_h^{\text{nearest}} - x_i)$ 
    else if
       $C_i = x_i \pm r \cdot (m_g^{\text{nearest}} - x_i)$ 
    }
  else if
    random movement
    {
       $C_i = r$ 

```

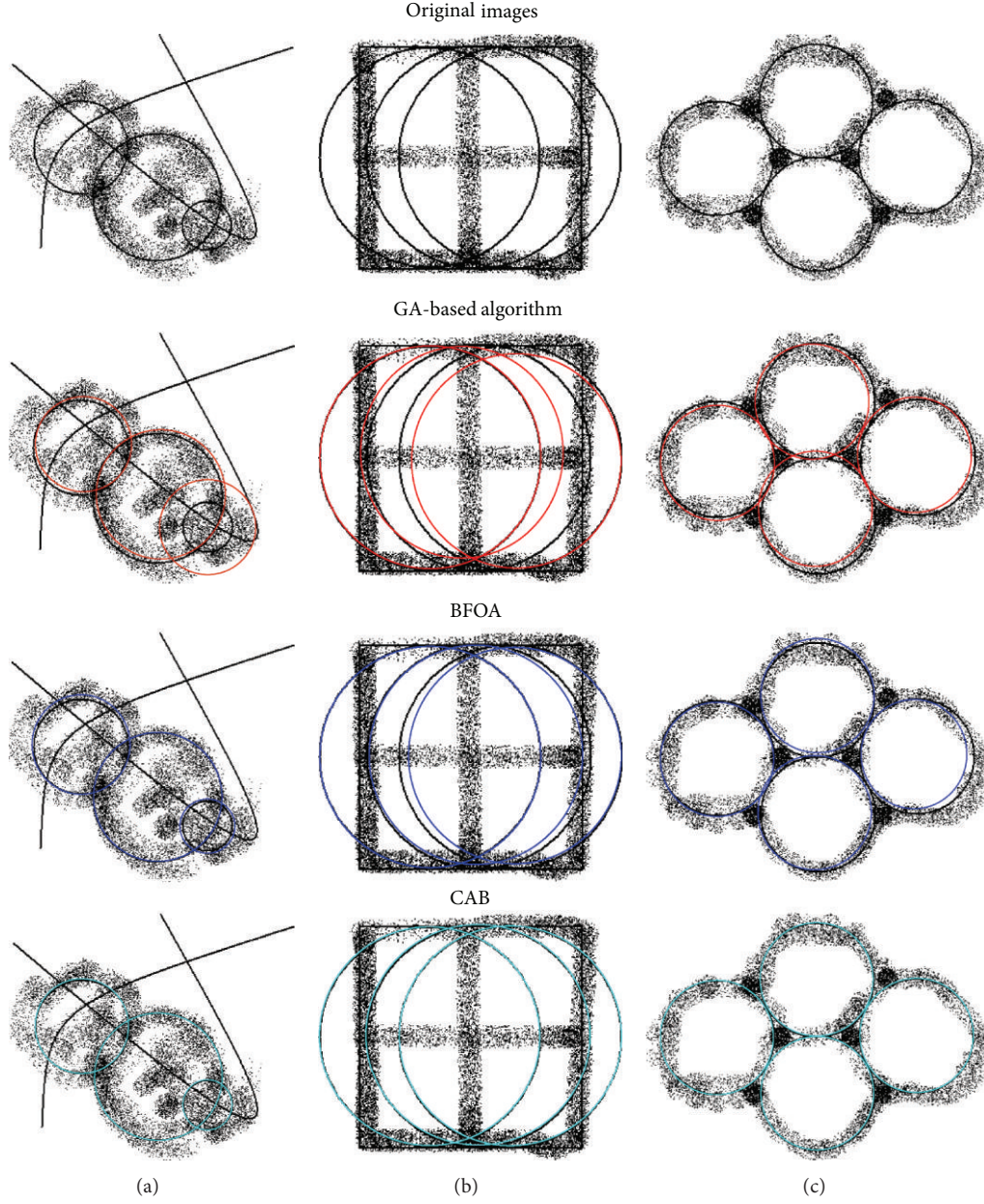


FIGURE 6: Synthetic images and their detected circles for GA-based algorithm, the BFOA method, and the proposed CAB algorithm.

}

end for where $r_1, r_2, r \in \text{rand}(0, 1)$.

Step 8. If NI is not completed, the process goes back to Step 3. Otherwise, the best values in M_h $\{C_1^M, C_2^M, \dots, C_B^M\}$ represent the best solutions (the best found circles).

Step 9. The element with the highest fitness value $J(C_1^M)$ is identified as the first circle C_1 .

Step 10. The distinctiveness factor $D_{C_m^M, C_{m-1}^M}$ of circle C_m^M (element m) with the next highest probability is evaluated with

respect to C_{m-1}^M . If $D_{C_m^M, C_{m-1}^M} > Th$, then C_m^M is considered as a new circle; otherwise the next action is evaluated.

Step 11. Step 10 is repeated until the element's fitness value reaches $(J(C_1^M)/10)$.

The number of candidate circles N_p is set considering a balance between the number of local minima to be detected and the computational complexity. In general terms, a large value of N_p suggests the detection of a great amount of circles at the cost of excessive computer time. After exhaustive experimentation, it has been found that a value of $N_p = 30$ represents the best tradeoff between computational overhead

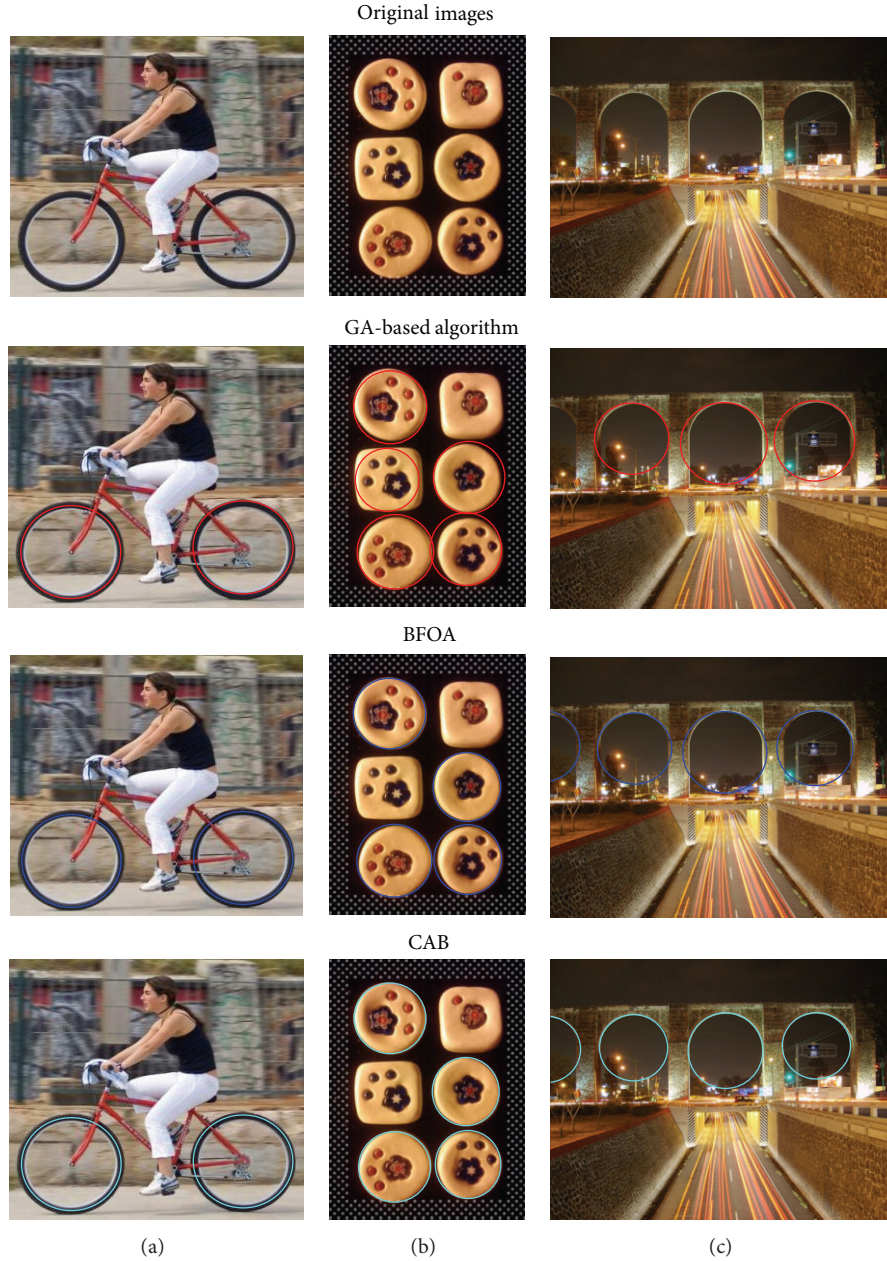


FIGURE 7: Real-life images and their detected circles for GA-based algorithm, the BFOA method, and the proposed CAB algorithm.

and accuracy and therefore such value is used throughout the study.

6. Results on Multicircle Detection

In order to achieve the performance analysis, the proposed approach is compared to the BFAO detector, the GA-based algorithm, and the RHT method over an image set.

The GA-based algorithm follows the proposal of Ayala-Ramirez et al. [41], which considers the population size as 70, the crossover probability as 0.55, the mutation probability as 0.10, and the number of elite individuals as 2. The roulette wheel selection and the 1-point crossover operator are both

applied. The parameter setup and the fitness function follow the configuration suggested in [41]. The BFAO algorithm follows the implementation from [45] considering the experimental parameters as $S = 50$, $N_c = 350$, $N_s = 4$, $N_{ed} = 1$, $P_{ed} = 0.25$, $d_{attract} = 0.1$, $w_{attract} = 0.2$, $w_{repellant} = 10$, $h_{repellant} = 0.1$, $\lambda = 400$, and $\psi = 6$. Such values are found to be the best configuration set according to [45]. Both, the GA-based algorithm and the BAFO method use the same objective function that is defined by (12). Likewise, the RHT method has been implemented as it is described in [40]. Finally, Table 5 presents the parameters for the CAB algorithm used in this work. They have been kept for all test images after being experimentally defined.

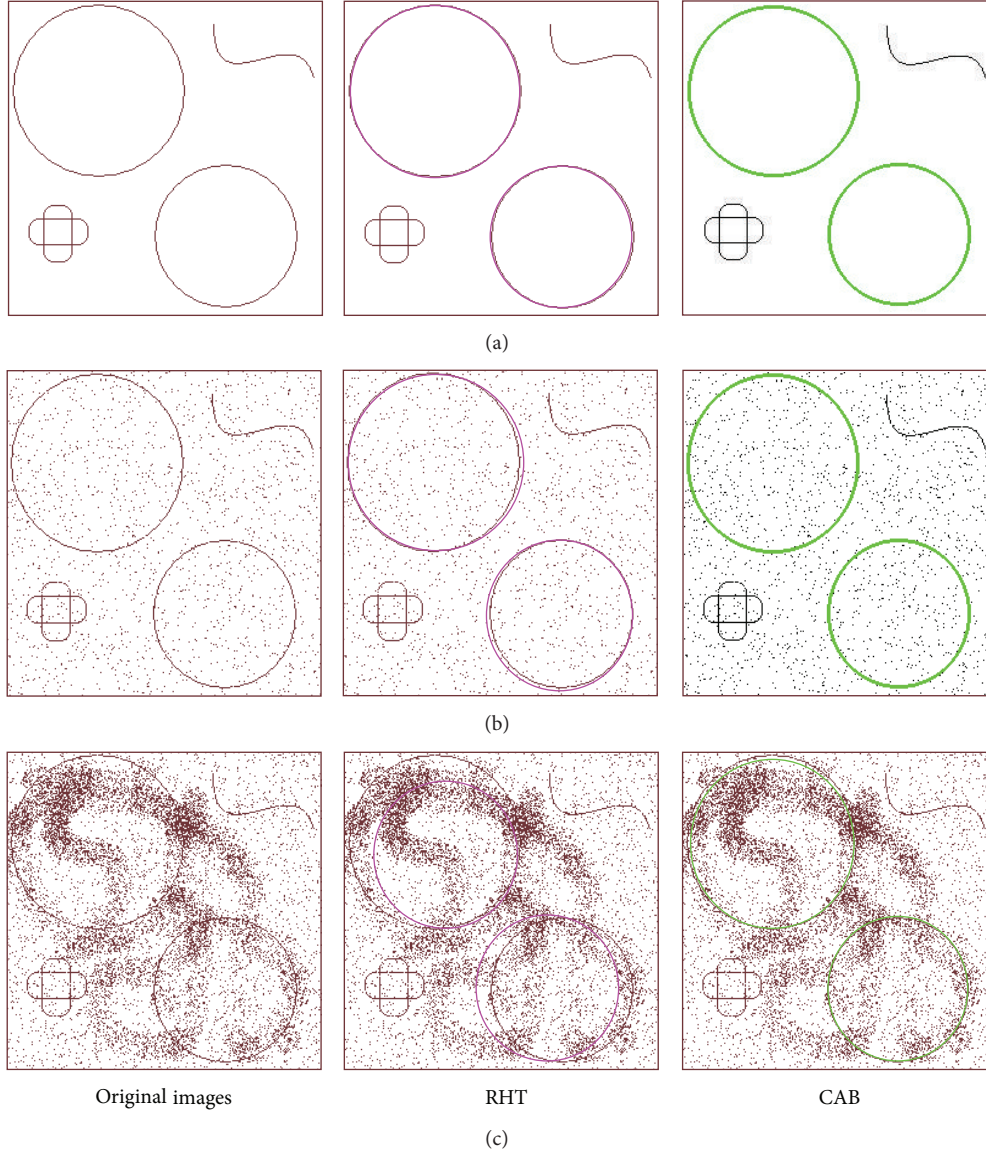


FIGURE 8: Relative performance of the RHT and the CAB.

TABLE 5: CAB detector parameters.

N_p	H	P	B	NI
30	0.5	0.1	12	200

Images rarely contain perfectly shaped circles. Therefore, with the purpose of testing accuracy for a single circle, the detection is challenged by a ground-truth circle which is determined from the original edge map. The parameters $(x_{\text{true}}, y_{\text{true}}, r_{\text{true}})$ representing the testing circle are computed using (6)–(9) for three circumference points over the manually drawn circle. Considering the center and the radius of the detected circle are defined as (x_D, y_D) and r_D , the error score (Es) can be accordingly calculated as

$$\text{Es} = \eta \cdot (|x_{\text{true}} - x_D| + |y_{\text{true}} - y_D|) + \mu \cdot |r_{\text{true}} - r_D|. \quad (17)$$

The central point difference $(|x_{\text{true}} - x_D| + |y_{\text{true}} - y_D|)$ represents the center shift for the detected circle as it is compared to a benchmark circle. The radio mismatch $(|r_{\text{true}} - r_D|)$ accounts for the difference between their radii. η and μ represent two weighting parameters which are to be applied separately to the central point difference and to the radio mismatch for the final error Es. At this work, they are chosen as $\eta = 0.05$ and $\mu = 0.1$. Such particular choice ensures that the radii difference would be strongly weighted in comparison to the difference of central circular positions between the manually detected and the machine-detected circles. Here we assume that if Es is found to be less than 1, then the algorithm gets a success; otherwise, we say that it has failed to detect the edge circle. Note that for $\eta = 0.05$ and $\mu = 0.1$, $\text{Es} < 1$ means that the maximum difference of radius tolerated is 10 while the maximum mismatch in the location of the center can be 20 (in number of pixels). In order to appropriately compare

TABLE 6: The averaged execution time, detection rate, and the averaged multiple error for the GA-based algorithm, the BFOA method, and the proposed CAB algorithm, considering six test images (shown by Figures 6 and 7).

Image	Averaged execution time \pm standard deviation (s)			Success rate (DR) (%)			Averaged ME \pm standard deviation		
	GA	BFOA	CAB	GA	BFOA	CAB	GA	BFOA	CAB
Synthetic images									
(a)	2.23 \pm (0.41)	1.71 \pm (0.51)	0.21 \pm (0.22)	88	99	100	0.41 \pm (0.044)	0.33 \pm (0.052)	0.22 \pm (0.033)
(b)	3.15 \pm (0.39)	2.80 \pm (0.65)	0.36 \pm (0.24)	79	92	99	0.51 \pm (0.038)	0.37 \pm (0.032)	0.26 \pm (0.041)
(c)	4.21 \pm (0.11)	3.18 \pm (0.36)	0.20 \pm (0.19)	74	88	100	0.48 \pm (0.029)	0.41 \pm (0.051)	0.15 \pm (0.036)
Natural images									
(a)	5.11 \pm (0.43)	3.45 \pm (0.52)	1.10 \pm (0.24)	90	96	100	0.45 \pm (0.051)	0.41 \pm (0.029)	0.25 \pm (0.037)
(b)	6.33 \pm (0.34)	4.11 \pm (0.14)	1.61 \pm (0.17)	83	89	100	0.81 \pm (0.042)	0.77 \pm (0.051)	0.37 \pm (0.055)
(c)	7.62 \pm (0.97)	5.36 \pm (0.17)	1.95 \pm (0.41)	84	92	99	0.92 \pm (0.075)	0.88 \pm (0.081)	0.41 \pm (0.066)

TABLE 7: P values produced by Wilcoxon's test comparing CAB to GA and BFOA over the averaged ME from Table 2.

Image	P value	
	CAB versus GA	CAB versus BFOA
Synthetic images		
(a)	1.8061e – 004	1.8288e – 004
(b)	1.7454e – 004	1.9011e – 004
(c)	1.7981e – 004	1.8922e – 004
Natural images		
(a)	1.7788e – 004	1.8698e – 004
(b)	1.6989e – 004	1.9124e – 004
(c)	1.7012e – 004	1.9081e – 004

the detection results, the detection rate (DR) is introduced as a performance index. DR is defined as the percentage of reaching detection success after a certain number of trials. For “success” it does mean that the compared algorithm is able to detect all circles contained in the image, under the restriction that each circle must hold the condition $Es < 1$. Therefore, if at least one circle does not fulfil the condition of $Es < 1$, the complete detection procedure is considered as a failure.

In order to use an error metric for multiple-circle detection, the averaged Es produced from each circle in the image is considered. Such criterion, defined as the multiple error (ME), is calculated as follows:

$$ME = \left(\frac{1}{NC} \right) \cdot \sum_{R=1}^{NC} Es_R, \quad (18)$$

where NC represents the number of circles within the image according to a human expert.

Figure 6 shows three synthetic images and the resulting images after applying the GA-based algorithm [41], the BFOA method [45], and the proposed approach. Figure 7 presents experimental results considering three natural images. The performance is analyzed by considering 35 different executions for each algorithm. Table 6 shows the averaged execution time, the detection rate in percentage, and the averaged multiple error (ME), considering six test images (shown by Figures 6 and 7). The best entries are boldfaced in

Table 6. Close inspection reveals that the proposed method is able to achieve the highest success rate keeping the smallest error, still requiring less computational time for most cases.

In order to statistically analyze the results in Table 6, a nonparametric significance proof known as the Wilcoxon's rank test [64–66] for 35 independent samples has been conducted. Such proof allows assessing result differences among two related methods. The analysis is performed considering a 5% significance level over multiple error (ME) data. Table 7 reports the P values produced by Wilcoxon's test for a pairwise comparison of the multiple error (ME), considering two groups gathered as CAB versus GA and CAB versus BFOA. As a null hypothesis, it is assumed that there is no difference between the values of the two algorithms. The alternative hypothesis considers an existent difference between the values of both approaches. All P values reported in Table 7 are less than 0.05 (5% significance level) which is a strong evidence against the null hypothesis, indicating that the best CAB mean values for the performance are statistically significant which has not occurred by chance.

Figure 8 demonstrates the relative performance of CAB in comparison with the RHT algorithm as it is described in [40]. All images belonging to the test are complicated and contain different noise conditions. The performance analysis is achieved by considering 35 different executions for each algorithm over the three images. The results, exhibited in Figure 8, present the median-run solution (when the runs were ranked according to their final ME value) obtained throughout the 35 runs. On the other hand, Table 4 reports the corresponding averaged execution time, detection rate (in %) and average multiple error (using (10)) for CAB and RHT algorithms over the set of images (the best results are boldfaced). Table 8 shows a decrease in performance of the RHT algorithm as noise conditions change. Yet the CAB algorithm holds its performance under the same circumstances.

7. Conclusions

In recent years, several metaheuristic optimization methods have been inspired from nature-like phenomena. In this paper, a new multimodal optimization algorithm known as the collective animal behavior algorithm (CAB) has been

TABLE 8: Average time, detection rate, and averaged error for CAB and HT, considering three test images.

Image	Average time \pm standard deviation (s)		Success rate (DR) (%)		Average ME \pm standard deviation	
	RHT	CAB	RHT	CAB	RHT	CAB
(a)	7.82 \pm (0.34)	0.30 \pm (0.10)	100	100	0.19 \pm (0.041)	0.11 \pm (0.017)
(b)	8.65 \pm (0.48)	0.22 \pm (0.13)	64	100	0.47 \pm (0.037)	0.13 \pm (0.019)
(c)	10.65 \pm (0.48)	0.25 \pm (0.12)	11	100	1.21 \pm (0.033)	0.15 \pm (0.014)

introduced. In CAB, the searcher agents emulate a group of animals that interact with each other depending on simple behavioral rules which are modeled as mathematical operators. Such operations are applied to each agent considering that the complete group hold a memory to store its own best positions seen so far, using a competition principle.

CAB has been experimentally evaluated over a test suite consisting of 8 benchmark multimodal functions for optimization. The performance of CAB has been compared to some other existing algorithms including deterministic crowding [17], probabilistic crowding [18], sequential fitness sharing [15], clearing procedure [20], clustering-based niching (CBN) [19], species conserving genetic algorithm (SCGA) [21], elitist-population strategies (AEGA) [22], clonal selection algorithm [24], and the artificial immune network (aiNet) [25]. All experiments have demonstrated that CAB generally outperforms all other multimodal metaheuristic algorithms regarding efficiency and solution quality, typically showing significant efficiency speedups. The remarkable performance of CAB is due to two different features: (i) operators allow a better exploration of the search space, increasing the capacity to find multiple optima; (ii) the diversity of solutions contained in the M_h memory in the context of multimodal optimization is maintained and even improved through of the use of a competition principle (dominance concept).

The proposed algorithm is also applied to the engineering problem of multicircle detection. Such a process is faced as a multimodal optimization problem. In contrast to other heuristic methods that employ an iterative procedure, the proposed CAB method is able to detect single or multiple circles over a digital image by running only one optimization cycle. The CAB algorithm searches the entire edge map for circular shapes by using a combination of three noncollinear edge points as candidate circles (animal positions) in the edge-only image. A matching function (objective function) is used to measure the existence of a candidate circle over the edge map. Guided by the values of such matching function, the set of encoded candidate circles is evolved using the CAB algorithm so that the best candidate can be fitted into an actual circle. After the optimization has been completed, an analysis of the embedded memory is executed in order to find the significant local minima (remaining circles). The overall approach generates a fast subpixel detector which can effectively identify multiple circles in real images despite that some circular objects exhibit a significant occluded portion.

In order to test the circle detection performance, both speed and accuracy have been compared. Score functions are defined by (17) and (18) in order to measure accuracy and effectively evaluate the mismatch between manually detected and machine-detected circles. We have demonstrated that the

CAB method outperforms both the GA (as described in [41]) and the BFOA (as described in [45]) within a statistically significant framework (Wilcoxon test). In contrast to the CAB method, the RHT algorithm [40] shows a decrease in performance under noisy conditions. Yet the CAB algorithm holds its performance under the same circumstances. Finally, Table 6 indicates that the CAB method can yield better results on complicated and noisy images compared with the GA and the BFOA methods.

References

- [1] A. Ahrari, M. Shariat-Panahi, and A. A. Atai, "GEM: a novel evolutionary optimization method with improved neighborhood search," *Applied Mathematics and Computation*, vol. 210, no. 2, pp. 376–386, 2009.
- [2] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence through Simulated Evolution*, John Wiley & Sons, Chichester, UK, 1966.
- [3] K. de Jong, *Analysis of the behavior of a class of genetic adaptive systems [Ph.D. thesis]*, The University of Michigan Press, Ann Arbor, Mich, USA, 1975.
- [4] 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, Palo Alto, Calif, USA, 1990.
- [5] J. H. Holland, *Adaptation in Natural and Artificial Systems*, The University of Michigan Press, Ann Arbor, Mich, USA, 1975.
- [6] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Boston, Mass, USA, 1989.
- [7] 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.
- [8] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
- [9] Ş. İ. Birbil and S.-C. Fang, "An electromagnetism-like mechanism for global optimization," *Journal of Global Optimization*, vol. 25, no. 3, pp. 263–282, 2003.
- [10] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: a gravitational search algorithm," *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.
- [11] 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.
- [12] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, no. 36–38, pp. 3902–3933, 2005.
- [13] Z. W. Geem, "Novel derivative of harmony search algorithm for discrete design variables," *Applied Mathematics and Computation*, vol. 199, no. 1, pp. 223–230, 2008.

- [14] X. Z. Gao, X. Wang, and S. J. Ovaska, "Uni-modal and multi-modal optimization using modified Harmony Search methods," *International Journal of Innovative Computing, Information and Control*, vol. 5, no. 10, pp. 2985–2996, 2009.
- [15] 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.
- [16] 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 (ICEC '96)*, pp. 786–791, May 1996.
- [17] S. W. Mahfoud, *Niching methods for genetic algorithms [Ph.D. dissertation]*, Illinois Genetic Algorithm Laboratory, University of Illinois, Urbana, Ill, USA, 1995.
- [18] O. J. Mengshoel and D. E. Goldberg, "Probability crowding: deterministic crowding with probabilistic replacement," in *Proceedings of the International Conference on Genetic and Evolutionary Computation Conference (GECCO '99)*, W. Banzhaf, Ed., pp. 409–416, Orlando, Fla, USA, 1999.
- [19] X. Yin and N. Gernay, "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.
- [20] 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, IEEE Press, Nagoya, Japan, May 1996.
- [21] 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.
- [22] 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.
- [23] L. Y. Wei and M. Zhao, "A niche hybrid genetic algorithm for global optimization of continuous multimodal functions," *Applied Mathematics and Computation*, vol. 160, no. 3, pp. 649–661, 2005.
- [24] L. N. Castro and F. J. Zuben, "Learning and optimization using the clonal selection principle," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 3, pp. 239–251, 2002.
- [25] L. N. Castro and J. Timmis, "An artificial immune network for multimodal function optimization," in *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp. 699–704, IEEE Press, Honolulu, Hawaii, 2002.
- [26] 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.
- [27] 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.
- [28] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.
- [29] D. B. Chen and C. X. Zhao, "Particle swarm optimization with adaptive population size and its application," *Applied Soft Computing Journal*, vol. 9, no. 1, pp. 39–48, 2009.
- [30] D. Sumper, "The principles of collective animal behaviour," *Philosophical Transactions of the Royal Society B*, vol. 361, no. 1465, pp. 5–22, 2006.
- [31] O. Petit and R. Bon, "Decision-making processes: the case of collective movements," *Behavioural Processes*, vol. 84, no. 3, pp. 635–647, 2010.
- [32] A. Kolpas, J. Moehlis, T. A. Frewen, and I. G. Kevrekidis, "Coarse analysis of collective motion with different communication mechanisms," *Mathematical Biosciences*, vol. 214, no. 1–2, pp. 49–57, 2008.
- [33] I. D. Couzin, "Collective cognition in animal groups," *Trends in Cognitive Sciences*, vol. 13, no. 1, pp. 36–43, 2008.
- [34] I. D. Couzin and J. Krause, "Self-organization and collective behavior in vertebrates," *Advances in the Study of Behavior*, vol. 32, pp. 1–75, 2003.
- [35] N. W. F. Bode, D. W. Franks, and A. Jamie Wood, "Making noise: emergent stochasticity in collective motion," *Journal of Theoretical Biology*, vol. 267, no. 3, pp. 292–299, 2010.
- [36] I. D. Couzin, J. Krause, R. James, G. D. Ruxton, and N. R. Franks, "Collective memory and spatial sorting in animal groups," *Journal of Theoretical Biology*, vol. 218, no. 1, pp. 1–11, 2002.
- [37] I. D. Couzin, "Collective minds," *Nature*, vol. 445, no. 7129, pp. 715–728, 2007.
- [38] S. Bazazi, J. Buhl, J. J. Hale et al., "Collective motion and cannibalism in locust migratory bands," *Current Biology*, vol. 18, no. 10, pp. 735–739, 2008.
- [39] T. J. Atherton and D. J. Kerbyson, "Using phase to represent radius in the coherent circle Hough transform," in *Proceedings of the IEE Colloquium on the Hough Transform*, IEE, London, UK, 1993.
- [40] L. Xu, E. Oja, and P. Kultanen, "A new curve detection method: randomized Hough transform (RHT)," *Pattern Recognition Letters*, vol. 11, no. 5, pp. 331–338, 1990.
- [41] V. Ayala-Ramirez, C. H. Garcia-Capulin, A. Perez-Garcia, and R. E. Sanchez-Yanez, "Circle detection on images using genetic algorithms," *Pattern Recognition Letters*, vol. 27, no. 6, pp. 652–657, 2006.
- [42] E. Cuevas, N. Ortega-Sánchez, D. Zaldivar, and M. Pérez-Cisneros, "Circle detection by harmony search optimization," *Journal of Intelligent and Robotic Systems*, vol. 66, no. 3, pp. 359–376, 2012.
- [43] E. Cuevas, D. Oliva, D. Zaldivar, M. Pérez-Cisneros, and H. Sossa, "Circle detection using electro-magnetism optimization," *Information Sciences*, vol. 182, no. 1, pp. 40–55, 2012.
- [44] E. Cuevas, D. Zaldivar, M. Pérez-Cisneros, and M. Ramírez-Ortegón, "Circle detection using discrete differential evolution optimization," *Pattern Analysis and Applications*, vol. 14, no. 1, pp. 93–107, 2011.
- [45] S. Dasgupta, S. Das, A. Biswas, and A. Abraham, "Automatic circle detection on digital images with an adaptive bacterial foraging algorithm," *Soft Computing*, vol. 14, no. 11, pp. 1151–1164, 2010.
- [46] N. W. F. Bode, A. J. Wood, and D. W. Franks, "The impact of social networks on animal collective motion," *Animal Behaviour*, vol. 82, no. 1, pp. 29–38, 2011.
- [47] B. H. Lemasson, J. J. Anderson, and R. A. Goodwin, "Collective motion in animal groups from a neurobiological perspective: the adaptive benefits of dynamic sensory loads and selective attention," *Journal of Theoretical Biology*, vol. 261, no. 4, pp. 501–510, 2009.

- [48] M. Bourjade, B. Thierry, M. Maumy, and O. Petit, "Decision-making in przewalski horses (*equus ferus przewalskii*) is driven by the ecological contexts of collective movements," *Ethology*, vol. 115, no. 4, pp. 321–330, 2009.
- [49] A. Bang, S. Deshpande, A. Sumana, and R. Gadagkar, "Choosing an appropriate index to construct dominance hierarchies in animal societies: a comparison of three indices," *Animal Behaviour*, vol. 79, no. 3, pp. 631–636, 2010.
- [50] Y. Hsu, R. L. Earley, and L. L. Wolf, "Modulation of aggressive behaviour by fighting experience: mechanisms and contest outcomes," *Biological Reviews of the Cambridge Philosophical Society*, vol. 81, no. 1, pp. 33–74, 2006.
- [51] M. Broom, A. Koenig, and C. Borries, "Variation in dominance hierarchies among group-living animals: modeling stability and the likelihood of coalitions," *Behavioral Ecology*, vol. 20, no. 4, pp. 844–855, 2009.
- [52] K. L. Bayly, C. S. Evans, and A. Taylor, "Measuring social structure: a comparison of eight dominance indices," *Behavioural Processes*, vol. 73, no. 1, pp. 1–12, 2006.
- [53] L. Conradt and T. J. Roper, "Consensus decision making in animals," *Trends in Ecology and Evolution*, vol. 20, no. 8, pp. 449–456, 2005.
- [54] A. Okubo, "Dynamical aspects of animal grouping," *Advances in Biophysics*, vol. 22, pp. 1–94, 1986.
- [55] C. W. Reynolds, "Flocks, herds and schools: a distributed behavioural model," *ACM SIGGRAPH Computer Graphics*, vol. 21, no. 4, pp. 25–33, 1987.
- [56] S. Gueron, S. A. Levin, and D. I. Rubenstein, "The dynamics of herds: from individuals to aggregations," *Journal of Theoretical Biology*, vol. 182, no. 1, pp. 85–98, 1996.
- [57] A. Czirók and T. Vicsek, "Collective behavior of interacting self-propelled particles," *Physica A*, vol. 281, no. 1, pp. 17–29, 2000.
- [58] M. Ballerini, "Interaction ruling collective animal behavior depends on topological rather than metric distance: evidence from a field study," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 105, pp. 1232–1237, 2008.
- [59] X.-S. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, Beckington, UK, 2008.
- [60] 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.
- [61] H. Zang, S. Zhang, and K. Hapeshi, "A review of nature-inspired algorithms," *Journal of Bionic Engineering*, vol. 7, pp. S232–S237, 2010.
- [62] A. Gandomi and A. Alavi, "Krill herd: a new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, pp. 4831–4845, 2012.
- [63] J. E. Bresenham, "A linear algorithm for incremental digital display of circular arcs," *Communications of the ACM*, vol. 20, no. 2, pp. 100–106, 1977.
- [64] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.
- [65] S. Garcia, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 special session on real parameter optimization," vol. 15, no. 6, pp. 617–644, 2009.
- [66] J. Santamaría, O. Cordón, S. Damas, J. M. García-Torres, and A. Quirin, "Performance evaluation of memetic approaches in 3D reconstruction of forensic objects," *Soft Computing*, vol. 13, no. 8–9, pp. 883–904, 2009.

Research Article

Solving Unconstrained Global Optimization Problems via Hybrid Swarm Intelligence Approaches

Jui-Yu Wu

Department of Business Administration, Lunghwa University of Science and Technology, No. 300, Section 1, Wanshou Road, Guishan, Taoyuan County 33306, Taiwan

Correspondence should be addressed to Jui-Yu Wu; jywu@mail.lhu.edu.tw

Received 7 September 2012; Revised 3 December 2012; Accepted 4 December 2012

Academic Editor: Baozhen Yao

Copyright © 2013 Jui-Yu Wu. 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.

Stochastic global optimization (SGO) algorithms such as the particle swarm optimization (PSO) approach have become popular for solving unconstrained global optimization (UGO) problems. The PSO approach, which belongs to the swarm intelligence domain, does not require gradient information, enabling it to overcome this limitation of traditional nonlinear programming methods. Unfortunately, PSO algorithm implementation and performance depend on several parameters, such as cognitive parameter, social parameter, and constriction coefficient. These parameters are tuned by using trial and error. To reduce the parametrization of a PSO method, this work presents two efficient hybrid SGO approaches, namely, a real-coded genetic algorithm-based PSO (RGA-PSO) method and an artificial immune algorithm-based PSO (AIA-PSO) method. The specific parameters of the internal PSO algorithm are optimized using the external RGA and AIA approaches, and then the internal PSO algorithm is applied to solve UGO problems. The performances of the proposed RGA-PSO and AIA-PSO algorithms are then evaluated using a set of benchmark UGO problems. Numerical results indicate that, besides their ability to converge to a global minimum for each test UGO problem, the proposed RGA-PSO and AIA-PSO algorithms outperform many hybrid SGO algorithms. Thus, the RGA-PSO and AIA-PSO approaches can be considered alternative SGO approaches for solving standard-dimensional UGO problems.

1. Introduction

An unconstrained global optimization (UGO) problem can generally be formulated as follows:

$$\text{Minimize } f(\mathbf{x}), \quad \mathbf{x} = [x_1, x_2, \dots, x_N]^T \in \mathcal{R}^N, \quad (1)$$

where $f(\mathbf{x})$ is an objective function and \mathbf{x} represents a decision variable vector. Additionally, $\mathbf{x} \in S, S \subseteq \mathcal{R}^N$ denotes search space (S), which is N dimensional and bounded by parametric constraints as follows:

$$x_n^l \leq x_n \leq x_n^u, \quad n = 1, 2, \dots, N, \quad (2)$$

where x_n^l and x_n^u are the lower and upper boundaries of the decision variables x_n , respectively.

Many conventional nonlinear programming (NLP) techniques, such as the golden search, quadratic approximation,

Nelder-Mead, steepest descent, Newton, and conjugate gradient methods, have been used to solve UGO problems [1]. Unfortunately, such NLP methods have difficulty in solving UGO problems when an objective function of an UGO problem is nondifferential. Many stochastic global optimization (SGO) approaches developed to overcome this limitation of the traditional NLP methods include genetic algorithms (GAs), particle swarm optimization (PSO), ant colony optimization (ACO), and artificial immune algorithms (AIAs). For instance, Hamzaçebi [2] developed an enhanced GA incorporating a local random search algorithm for eight continuous functions. Furthermore, Chen [3] presented a two-layer PSO method to solve nine UGO problems. Zhao [4] presented a perturbed PSO approach for 12 UGO problems. Meanwhile, Toksari [5] developed an ACO algorithm for solving UGO problems. Finally, Kelsey and Timmis [6] presented an AIA method based on the clonal selection principle for solving 12 UGO problems.

This work focuses on a PSO algorithm, based on it is being effective, robust and easy to use in the SGO methods. Research on the PSO method has considered many critical issues such as parameter selection, integration of the PSO algorithm with the approaches of self-adaptation, and integration with other intelligent optimizing methods [7]. This work surveys two issues: first is a PSO approach that integrates with other intelligent optimizing methods and second is parameter selection for use in a PSO approach.

Regarding the first issue, the conventional PSO algorithm lacks evolution operators of GAs, such as crossover and mutation operations. Therefore, PSO has premature convergence, that is, a rapid loss of diversity during optimization [4]. To overcome this limitation, many hybrid SGO methods have been developed to create diverse candidate solutions to enhance the performance of a PSO approach. Hybrid algorithms have some advantages; for instance, hybrid algorithms outperform individual algorithms in solving certain problems and thus can solve general problems more efficiently [8]. Kao and Zahara [9] presented a hybrid GA and PSO algorithm to solve 17 multimodal test functions. Their study used the operations of GA and PSO methods to generate candidate solutions to improve solution quality and convergence rates. Furthermore, Shelokar et al. [10] presented a hybrid PSO and ACO algorithm to solve multimodal continuous optimization problems. Their study used an ACO algorithm to update the particle positions to enhance a PSO algorithm performance. Chen et al. [11] presented a hybrid PSO and external optimization based on the Bak-Sneppen model to solve unimodal and multimodal benchmark problems. Furthermore, Thangaraj et al. [12] surveyed many algorithms that combine the PSO algorithm with other search techniques and compared the performances obtained using hybrid differential evolution PSO (DE-PSO), adaptive mutation PSO (AMPSO), and hybrid GA and PSO (GA-PSO) approaches to solve nine conventional benchmark problems.

Regarding the second issue, a PSO algorithm has numerous parameters that must be set, such as cognitive parameter, social parameter, inertia weight, and constriction coefficient. Traditionally, the optimal parameter settings of a PSO algorithm are tuned based on trial and error. The abilities of a PSO algorithm to explore and exploit are constrained to optimum parameter settings [13, 14]. Therefore, Jiang et al. [15] used a stochastic process theory to analyze the parameter settings (e.g., cognitive parameter, social parameter, and inertia weight) of a standard PSO algorithm.

This work focuses on the second issue related to the application of a PSO method. Fortunately, the optimization of parameter settings for a PSO algorithm can be viewed as an UGO problem. Moreover, real-coded GA (RGA) and AIA are efficient SGO approaches for solving UGO problems. Based on the advantage of a hybrid algorithm [8], this work develops two hybrid SGO approaches. The first approach is a hybrid RGA and PSO (RGA-PSO) algorithm, while the second one is a hybrid AIA and PSO (AIA-PSO) algorithm. The proposed RGA-PSO and AIA-PSO algorithms are considered as a means of solving the two optimization problems simultaneously. The first UGO problem (optimization of cognitive parameter, social parameter, and constriction coefficient) is

optimized using external RGA and AIA approaches, respectively. The second UGO problem is then solved using the internal PSO algorithm. The performances of the proposed RGA-PSO and AIA-PSO algorithms are evaluated using a set of benchmark UGO problems and compared with those of many hybrid algorithms [9, 10, 12].

The rest of this paper is organized as follows. Section 2 describes RGA, PSO, and AIA approaches. Section 3 then presents the proposed RGA-PSO and AIA-PSO methods. Next, Section 4 compares the experimental results of the proposed RGA-PSO and AIA-PSO approaches with those of many hybrid methods. Conclusions are finally drawn in Section 5.

2. Related Works

The SGO approaches such as RGA, PSO, and AIA [16] are described as follows.

2.1. Real-Coded Genetic Algorithm. GAs are based on the concepts of natural selection and use three genetic operations, that is, selection, crossover, and mutation, to explore and exploit the solution space. In solving continuous function optimization problems, RGA method outperforms binary-coded GA approach [17]. Therefore, this work describes operators of a RGA method [18].

2.1.1. Selection Operation. A selection operation picks up strong individuals from a current population based on their fitness function values and then reproduces these individuals into a crossover pool. Many selection operations developed include the roulette wheel, the ranking, and the tournament methods [17, 18]. This work employs the normalized geometric ranking method as follows:

$$p_j = q'(1 - q)^{\text{rank} - 1}, \quad j = 1, 2, \dots, \text{ps}_{\text{RGA}}, \quad (3)$$

where p_j = probability of selecting individual j , q = probability of choosing the best individual (here $q = 0.35$), $q' = q/(1 - (1 - q)^{\text{ps}_{\text{RGA}}})$, and rank = individual ranking based on fitness value, where 1 represents the best, rank = 1, 2, ..., ps_{RGA} , and ps_{RGA} = population size of the RGA method.

2.1.2. Crossover Operation. While exploring the solution space by creating new offspring, the crossover operation randomly chooses two parents from the crossover pool and then uses these two parents to create two new offspring. This operation is repeated until the $\text{ps}_{\text{RGA}}/2$ is satisfied. The whole arithmetic crossover is easily performed as follows:

$$\begin{aligned} \mathbf{v}'_1 &= \beta \times \mathbf{v}_1 + (1 - \beta) \times \mathbf{v}_2, \\ \mathbf{v}'_2 &= (1 - \beta) \times \mathbf{v}_1 + \beta \times \mathbf{v}_2, \end{aligned} \quad (4)$$

where \mathbf{v}_1 and \mathbf{v}_2 = parents (decision variable vectors), \mathbf{v}'_1 and \mathbf{v}'_2 = offspring (decision variable vectors), and β = uniform random number in the interval $[0, 1.5]$.

2.1.3. Mutation Operation. Mutation operation can improve the diversity of individuals (candidate solutions). Multi-non-uniform mutation is described as follows:

$$x_{\text{trial},n} = \begin{cases} x_{\text{current},n} + (x_n^u - x_{\text{current},n}) \text{pert}(g_{\text{RGA}}) & \text{if } U_1(0, 1) < 0.5, \\ x_{\text{current},n} - (x_{\text{current},n} - x_n^l) \text{pert}(g_{\text{RGA}}) & \text{if } U_1(0, 1) \geq 0.5, \end{cases} \quad (5)$$

where $\text{pert}(g_{\text{RGA}}) = [U_2(0, 1)(1 - g_{\text{RGA}}/g_{\text{max,RGA}})]^2$, perturbed factor, $U_1(0, 1)$ and $U_2(0, 1)$ = uniform random variable in the interval $[0, 1]$, $g_{\text{max,RGA}}$ = maximum generation of the RGA method, g_{RGA} = current generation of the RGA method, $x_{\text{current},n}$ = current decision variable x_n , and $x_{\text{trial},n}$ = trial decision variable (candidate solution) x_n .

2.2. Particle Swarm Optimization. Kennedy and Eberhart [19] first presented a standard PSO algorithm, which is inspired by the social behavior of bird flocks or fish schools. Like GAs, a PSO method is a population-based algorithm. A population of candidate solutions is called a particle swarm. The particle velocities can be updated by (6) as follows:

$$\begin{aligned} v_{j,n}(g_{\text{PSO}} + 1) &= v_{j,n}(g_{\text{PSO}}) + c_1 r_{j,1}(g_{\text{PSO}}) \\ &\times [p_{j,n}^{\text{lb}}(g_{\text{PSO}}) - x_{j,n}(g_{\text{PSO}})] \\ &+ c_2 r_{j,2}(g_{\text{PSO}}) [p_{j,n}^{\text{gb}}(g_{\text{PSO}}) - x_{j,n}(g_{\text{PSO}})] \\ j &= 1, 2, \dots, \text{ps}_{\text{PSO}}, \quad n = 1, 2, \dots, N \end{aligned} \quad (6)$$

where $v_{j,n}(g_{\text{PSO}} + 1)$ = particle velocity of decision variable x_n of particle j at generation $g_{\text{PSO}} + 1$, $v_{j,n}(g_{\text{PSO}})$ = particle velocity of decision variable x_n of particle j at generation g_{PSO} , c_1 = cognitive parameter, c_2 = social parameter, $x_{j,n}(g_{\text{PSO}})$ = particle position of decision variable x_n of particle j at generation g_{PSO} , $r_{j,1}(g_{\text{PSO}})$, $r_{j,2}(g_{\text{PSO}})$ = independent uniform random numbers in the interval $[0, 1]$ at generation g_{PSO} , $p_{j,n}^{\text{lb}}(g_{\text{PSO}})$ = best local solution at generation g_{PSO} , $p_{j,n}^{\text{gb}}(g_{\text{PSO}})$ = best global solution at generation g_{PSO} , and ps_{PSO} = population size of the PSO algorithm.

The particle positions can be obtained using (7) as follows:

$$\begin{aligned} x_{j,n}(g_{\text{PSO}} + 1) &= x_{j,n}(g_{\text{PSO}}) + v_{j,n}(g_{\text{PSO}} + 1), \\ j &= 1, 2, \dots, \text{ps}_{\text{PSO}}, \quad n = 1, 2, \dots, N. \end{aligned} \quad (7)$$

Shi and Eberhart [20] introduced a modified PSO algorithm by incorporating an inertia weight (ω_{in}) into (8) to

control the exploration and exploitation capabilities of a PSO algorithm as follows:

$$\begin{aligned} v_{j,n}(g_{\text{PSO}} + 1) &= \omega_{\text{in}} v_{j,n}(g_{\text{PSO}}) + c_1 r_{j,1}(g_{\text{PSO}}) \\ &\times [p_{j,n}^{\text{lb}}(g_{\text{PSO}}) - x_{j,n}(g_{\text{PSO}})] \\ &+ c_2 r_{j,2}(g_{\text{PSO}}) [p_{j,n}^{\text{gb}}(g_{\text{PSO}}) - x_{j,n}(g_{\text{PSO}})] \\ j &= 1, 2, \dots, \text{ps}_{\text{PSO}}, \quad n = 1, 2, \dots, N. \end{aligned} \quad (8)$$

A constriction coefficient (χ) in (9) is used to balance the exploration and exploitation tradeoff [21–23] as follows:

$$\begin{aligned} v_{j,n}(g_{\text{PSO}} + 1) &= \chi \{v_{j,n}(g_{\text{PSO}}) + \tau_1(g_{\text{PSO}}) \\ &\times [p_{j,n}^{\text{lb}}(g_{\text{PSO}}) - x_{j,n}(g_{\text{PSO}})] \\ &+ \tau_2(g_{\text{PSO}}) [p_{j,n}^{\text{gb}}(g_{\text{PSO}}) - x_{j,n}(g_{\text{PSO}})]\} \\ j &= 1, 2, \dots, \text{ps}_{\text{PSO}}, \quad n = 1, 2, \dots, N, \end{aligned} \quad (9)$$

where

$$\chi = \frac{2U_3(0, 1)}{|2 - \tau - \sqrt{\tau(\tau - 4)}|}, \quad (10)$$

$U_3(0, 1)$ = uniform random variable in the interval $[0, 1]$, $\tau = \tau_1 + \tau_2$, $\tau_1 = c_1 r_{j,1}$, $\tau_2 = c_2 r_{j,2}$.

This work considers parameters ω_{in} and χ to modify the particle velocities as follows:

$$\begin{aligned} v_{j,n}(g_{\text{PSO}} + 1) &= \chi \{ \omega_{\text{in}} v_{j,n}(g_{\text{PSO}}) + c_1 r_{j,1}(g_{\text{PSO}}) \\ &\times [p_{j,n}^{\text{lb}}(g_{\text{PSO}}) - x_{j,n}(g_{\text{PSO}})] \\ &+ c_2 r_{j,2}(g_{\text{PSO}}) \\ &\times [p_{j,n}^{\text{gb}}(g_{\text{PSO}}) - x_{j,n}(g_{\text{PSO}})] \} \\ j &= 1, 2, \dots, \text{ps}_{\text{PSO}}, \quad n = 1, 2, \dots, N, \end{aligned} \quad (11)$$

where $\omega_{\text{in}} = ((g_{\text{max,PSO}} - g_{\text{PSO}})/g_{\text{max,PSO}})$, increased g_{PSO} value reduces the ω_{in} , and $g_{\text{max,PSO}}$ = maximum generation of the PSO algorithm.

According to (11), the optimal values of parameters c_1 , c_2 , and χ are difficult to obtain through trial and error. This work thus optimizes these parameter settings by using RGA and AIA approaches.

2.3. Artificial Immune Algorithm. Wu [24] presented an AIA approach based on clonal selection and immune network theories to solve constrained global optimization problems. The AIA method consists of selection, hypermutation, receptor editing, and bone marrow operations. The selection operation is performed to reproduce strong antibodies (**Ab**s). Also, diverse **Ab**s are created using hypermutation, receptor editing, and bone marrow operations, as described in the following subsections.

2.3.1. Ab and Ag Representation. In the human system, an antigen (**Ag**) has multiple epitopes (antigenic determinants), which can be recognized by many **Abs** with paratopes (recognizers), on its surface. In the AIA approach, an **Ag** represents known parameters of a solved problem. The **Abs** are the candidate solutions (i.e., decision variables x_n , $n = 1, 2, \dots, N$) of the solved problem. The quality of a candidate solution is evaluated using an **Ab-Ag** affinity that is derived from the value of an objective function of the solved problem.

2.3.2. Selection Operation. The selection operation controls the number of antigen-specific **Abs**. This operation is defined according to **Ab-Ag** and **Ab-Ab** recognition information as follows:

$$p_{j,\text{rec}} = \frac{1}{N} \sum_{n=1}^N \frac{1}{e^{d_{j,n}}},$$

$$d_{j,n} = \left| \frac{x_n^* - x_{j,n}}{x_n^*} \right|, \quad j = 1, 2, \dots, \text{rs}_{\text{AIA}}, \quad n = 1, 2, \dots, N, \quad (12)$$

where $p_{j,\text{rec}}$ = probability that **Ab_j** recognizes **Ab*** (the best solution), x_n^* = the best **Ab*** with the highest **Ab-Ag** affinity, $x_{j,n}$ = decision variables x_n of **Ab_j**, and rs_{AIA} = repertoire (population) size of the AIA approach.

The **Ab*** is recognized by other **Ab_j** in a current **Ab** repertoire. Large $p_{j,\text{rec}}$ implies that **Ab_j** can effectively recognize **Ab***. The **Ab_j** with $p_{j,\text{rec}}$ that is equivalent to or larger than the threshold degree of AIA approach $p_{\text{rt,AIA}}$ is reproduced to generate an intermediate **Ab** repertoire.

2.3.3. Hypermutation Operation. The somatic hypermutation operation can be expressed as follows:

$$x_{\text{trial},n} = \begin{cases} x_{\text{current},n} + (x_n^u - x_{\text{current},n}) \text{pert}(g_{\text{AIA}}), & \text{if } U_4(0, 1) < 0.5, \\ x_{\text{current},n} - (x_{\text{current},n} - x_n^l) \text{pert}(g_{\text{AIA}}), & \text{if } U_4(0, 1) \geq 0.5, \end{cases} \quad (13)$$

where $\text{pert}(g_{\text{AIA}}) = \{U_5(0, 1)(1 - g_{\text{AIA}}/g_{\text{max,AIA}})\}^2$ = perturbation factor, g_{AIA} = current generation of the AIA method, $g_{\text{max,AIA}}$ = maximum generation number of the AIA method, and $U_4(0, 1)$ and $U_5(0, 1)$ = uniform random number in the interval $[0, 1]$.

This operation has two tasks, that is, a uniform search and local fine tuning.

2.3.4. Receptor-Editing Operation. A receptor-editing operation is developed based on the standard Cauchy distribution $C(0, 1)$, in which the local parameter is zero and the scale parameter is one. Receptor editing is implemented using

Cauchy random variables that are created from $C(0, 1)$, owing to their ability to provide a large jump in the **Ab-Ag** affinity landscape to increase the probability of escaping from the local **Ab-Ag** affinity landscape. Cauchy receptor editing can be defined by

$$x_{\text{trial}} = x_{\text{current}} + U_6(0, 1)^2 \times \sigma, \quad (14)$$

where $\sigma = [\sigma_1, \sigma_2, \dots, \sigma_N]^T$, vector of Cauchy random variables, and $U_6(0, 1)$ = uniform random number in the interval $[0, 1]$.

This operation is used in local fine-tuning and large perturbation.

2.3.5. Bone Marrow Operation. The paratope of an **Ab** can be created by recombining gene segments $V_H D_H J_H$ and $V_L J_L$ [25]. Therefore, based on this metaphor, diverse **Abs** are synthesized using a bone marrow operation. This operation randomly selects two **Abs** from the intermediate **Ab** repertoire and a recombination point from the gene segments of the paratope of the selected **Abs**. The selected gene segments (e.g., gene x_1 of **Ab₁** and gene x_1 of the **Ab₂**) are reproduced to create a library of gene segments. The selected gene segments in the paratope are then deleted. The new **Ab₁** is formed by inserting the gene segment, which is gene x_1 of the **Ab₂** in the library plus a random variable created from standard normal distribution $N(0, 1)$, at the recombination point. The details of the implementation of the bone marrow operation can be found [24].

3. Methods

This work develops the RGA-PSO and AIA-PSO approaches for solving UGO problems. The implementation of the RGA-PSO and AIA-PSO methods is described as follows.

3.1. RGA-PSO Algorithm. Figure 1 shows the pseudocode of the proposed RGA-PSO algorithm. The best parameter setting of the internal PSO algorithm is obtained by using the external RGA method. Benchmark UGO problems are solved by using the internal PSO algorithm.

External RGA

Step 1 (initialize the parameter settings). Parameter settings such as ps_{RGA} , crossover probability of the RGA method $p_{c,\text{RGA}}$, mutation probability of the RGA approach $p_{m,\text{RGA}}$, ps_{PSO} as well as lower and upper boundaries of the parameters (c_1 , c_2 , and χ) for a PSO algorithm are given. The candidate solution (individual) of a RGA method represents the optimized parameters of internal PSO algorithm. Figure 2 illustrates the candidate solution of the RGA method.

Step 2 (calculate the fitness function value). The fitness function value fitness_j of the external RGA method is the

Procedure external RGA method

```

begin
   $g_{RGA} \leftarrow 0$ 
  count  $\leftarrow 0$ 
  Step 1: Initialize the parameter settings
    (a) parameter settings
    (b) generate an initial population
  while  $g_{RGA} \leq g_{max,RGA}$  do
    Step 2: Calculate the fitness function value
    candidate solution  $j, j = 1, 2, \dots, ps_{RGA}$ 
    fitnessj =  $f(x_{PSO}^*)$ 
    Step 3: Perform a selection operation
    For each candidate solution  $j, j = 1, 2, \dots, ps_{RGA}/2$  do
      if rand()  $\leq P_{c,RGA}$  then
        Step 4: Implement a crossover operation
      endif
    endFor
    For each candidate solution  $j, j = 1, 2, \dots, ps_{RGA}$  do
      if rand()  $\leq P_{m,RGA}$  then
        Step 5: Conduct a mutation operation
      endif
    endFor
    Step 6: Perform an elitist strategy
    if fitness( $x_n, \chi, c_1, c_2, g_{RGA} + 1$ ) - fitness( $x_n, \chi, c_1, c_2, g_{RGA}$ )  $\leq 1 \times 10^{-9}$  then
      count  $\leftarrow$  count + 1
    endif
    if count  $\leq 5$  then
      break
    endif
     $g_{RGA} \leftarrow g_{RGA} + 1$ 
  endwhile
end

```

Procedure internal PSO Algorithm

```

begin
   $g_{PSO} \leftarrow 0$ 
  (1) Generate an initial particle swarm
  (a) parameter settings obtained from external RGA
  (b) generate an initial particle swarm
  while  $g_{PSO} \leq g_{max,PSO}$  do
    (2) Compute the objective function value
    (3) Update the particle velocity and position
    (4) Perform an elitist strategy
     $g_{PSO} \leftarrow g_{PSO} + 1$ 
  endwhile
end

```

FIGURE 1: The pseudocode of the proposed RGA-PSO algorithm.

TABLE 1: The parameter settings for the proposed RGA-PSO and AIA-PSO approaches.

Methods	Parameter settings	Search space
External RGA	$p_{c,RGA} = 1$	$[\chi^l, \chi^u] = [0.1, 1]$ $[c_1^l, c_1^u] = [0.1, 5]$ $[c_2^l, c_2^u] = [0.1, 5]$
	$p_{m,RGA} = 0.15$	
	$ps_{RGA} = 20$	
	$g_{max,RGA} = 20$	
External AIA	$p_{rt,AIA} = 0.9$	
	$rs_{AIA} = 20$	
	$g_{max,AIA} = 20$	
Internal PSO	$ps_{PSO} = 20$	$[x_n^l, x_n^u]$ for a UGO problem
	$g_{max,PSO} = 1000$ for $N \leq 3$	
	$g_{max,PSO} = 3000$ for $5 \leq N \leq 30$	

Candidate solution j			
χ	c_1	c_2	Fitness _j

FIGURE 2: Candidate solution of the RGA method.

best objective function value $f(x_{PSO}^*)$ obtained from the best solution x_{PSO}^* of each internal PSO algorithm execution as follows:

$$\text{fitness}_j = f(x_{PSO}^*), \quad j = 1, 2, \dots, ps_{RGA}. \quad (15)$$

The candidate solution j of the external RGA method is incorporated into the internal PSO algorithm and, then, the internal PSO algorithm is used to solve an UGO problem. The internal PSO algorithm is executed as follows.

Internal PSO Algorithm

- (1) *Generate an initial particle swarm.* An initial particle swarm is created based on ps_{PSO} from $[x_n^l, x_n^u]$ of an UGO problem. A particle represents a candidate solution of an UGO problem.

Procedure external AIA method

```

begin
   $g_{AIA} \leftarrow 0$ 
  count  $\leftarrow 0$ 
  Step 1: Initialize the parameter settings
  while  $g_{AIA} < g_{\max, AIA}$  do
    Step 2: Evaluate the Ab-Ag affinity  $\Rightarrow$   $\begin{matrix} \chi & c_1 & c_2 \\ -1 \times x_{PSO}^* \end{matrix} \Leftarrow$ 
     $Ab^* \leftarrow \max(\text{affinity}_j), j = 1, 2, \dots, rs$ 
    Step 3: Perform a clonal selection operation
    for each  $Ab_j, j = 1, 2, \dots, rs_{AIA}$  do
      if  $p_{j, rec} \geq p_{rt, AIA}$  then
        promote (clone)
      else
        suppress
      endIf
    endFor
    Step 4: Implement an Ab-Ag affinity maturation operation
    for each promoted  $Ab_j$  do
      if rand()  $\leq 0.5$  do
        somatic hypermutation
      else
        receptor editing
      endIf
    endFor
    Step 5: Introduce diverse Abs
    Step 6: Update an Ab repertoire
    if  $|\text{affinity}(x_n, \chi, c_1, c_2, g_{AIA} + 1) - \text{affinity}(x_n, \chi, c_1, c_2, g_{AIA})| \leq 1 \times 10^{-9}$  then
      count  $\leftarrow$  count + 1
    endIf
    if count  $\leq 5$  then
      break
    endIf
     $g_{AIA} \leftarrow g_{AIA} + 1$ 
  endWhile
end
end

```

Procedure internal PSO algorithm

```

begin
   $g_{PSO} \leftarrow 0$ 
  (1) Generate an initial particle swarm
  (a) parameter settings obtained from external AIA
  (b) generate an initial particle swarm
  while  $g_{PSO} \leq g_{\max, PSO}$  do
    (2) Compute the fitness value
    (3) Update the particle velocity and position
    (4) Perform an elitist strategy
     $g_{PSO} \leftarrow g_{PSO} + 1$ 
  endWhile
end
end

```

FIGURE 3: The pseudocode of the proposed AIA-PSO algorithm.

- (2) *Compute the objective function value.* The objective function value of the internal PSO algorithm of particle j ($j = 1, 2, \dots, ps_{PSO}$) is the objective function value of an UGO problem.
- (3) *Update the particle velocity and position.* Equations (7) and (11) can be used to update the particle position and velocity.
- (4) *Perform an elitist strategy.* A new particle swarm is generated from internal step (3). Notably, $f(x_{j, PSO})$ of a candidate solution j (particle j) in the particle swarm is evaluated. This work makes a pairwise comparison between the $f(x_{j, PSO})$ of candidate solutions in the new particle swarm and that in the current particle swarm. A situation in which the candidate solution j ($j = 1, 2, \dots, ps_{PSO}$) in the new particle swarm is better than candidate solution j in the current particle swarm implies that the strong candidate solution j in the new particle swarm replaces the candidate solution j in the current particle swarm. The elitist strategy guarantees that the best candidate solution is always preserved in the next generation. The current particle swarm is updated to the particle swarm of the next generation.

Internal steps from (2) to (4) are repeated until the maximum generation number of the PSO method $g_{\max, PSO}$ of the internal PSO algorithm is satisfied.

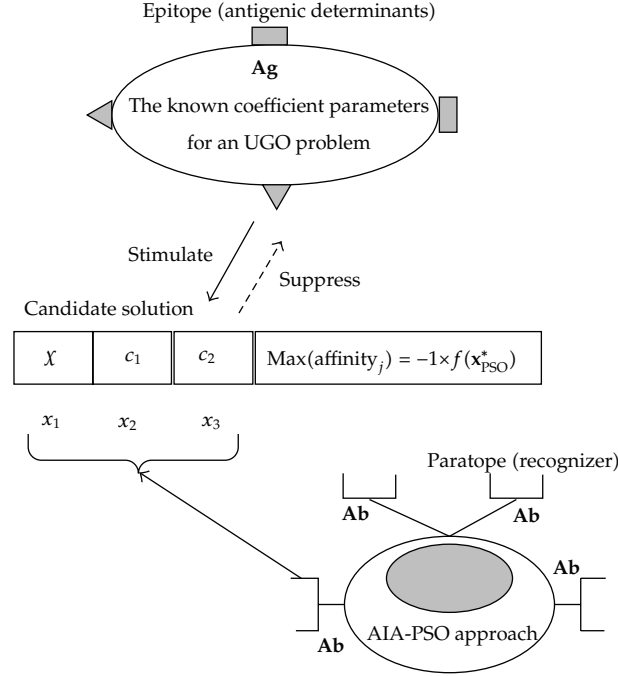
End

Step 3 (perform a selection operation). Equation (3) is used to select the parents into a crossover pool.

Step 4 (implement a crossover operation). The crossover operation performs a global search. The candidate solutions are created by using (4).

Step 5 (conduct a mutation operation). The mutation operation implements a local search. A solution space is exploited using (5).

Step 6 (perform an elitist strategy). This work presents an elitist strategy to update the population. A situation in which the fitness _{j} of candidate solution j in the new population is larger than the fitness _{j} of candidate solution j in the current population suggests that a replacement of the weak candidate solution j takes place. Additionally, a situation in which the fitness _{j} of candidate solution j in the new population is equal to or worse than that in the current population implies that the candidate solution j in the current population survives. In addition to maintaining the strong candidate solutions, this strategy effectively eliminates weak candidate solutions.

FIGURE 4: **Ab** and **Ag** representation.

External steps 2 to 6 are repeated until the $g_{\max, \text{RGA}}$ of the external RGA method is met.

3.2. AIA-PSO Algorithm. Figure 3 shows the pseudocode of the proposed AIA-PSO algorithm. The external AIA method is used to optimize the parameter settings of the internal PSO method, which is employed to solve benchmark UGO problems.

External AIA

Step 1 (initialize the parameter settings). Several parameters must be predetermined. These include repertoire (population) size rs_{AIA} and $p_{\text{rt, AIA}}$. An available **Ab** repertoire (population) is randomly generated using rs_{AIA} from the lower and upper boundaries of parameters χ [χ^l, χ^u], c_1 [c_1^l, c_1^u], and c_2 [c_2^l, c_2^u]. Figure 4 shows the **Ab** and **Ag** representation.

Step 2 (evaluate the **Ab-Ag** affinity).

Internal PSO Algorithm. The external AIA approach offers parameter settings c_1 , c_2 , and χ for the internal PSO algorithm. Subsequently, internal steps (1)–(4) of the PSO algorithm are implemented. The internal PSO method returns the best fitness value of $\text{PSO } f(\mathbf{x}_{\text{PSO}}^*)$ to the external AIA method.

- (1) *Generate an initial particle swarm.* An initial particle swarm is created based on ps_{PSO} from [x_n^l, x_n^u] of an UGO problem. A particle represents a candidate solution of an UGO problem.
- (2) *Compute the fitness value.* The fitness value of the internal PSO algorithm $f(\mathbf{x}_{\text{PSO}, j})$ $j = 1, 2, \dots, \text{ps}_{\text{PSO}}$ is the objective function value of an UGO problem.

- (3) *Update the particle velocity and position.* Equations (7) and (11) can be used to update the particle position and velocity.

- (4) *Perform an elitist strategy.* A new particle swarm (population) is generated from internal step (3). Notably, $f(\mathbf{x}_{j, \text{PSO}})$ of a candidate solution j (particle j) in the particle swarm is evaluated. This work makes a pairwise comparison between the $f(\mathbf{x}_{j, \text{PSO}})$ of candidate solutions in the new particle swarm and that in the current particle swarm. The elitist strategy guarantees that the best candidate solution is always preserved in the next generation. The current particle swarm is updated to the particle swarm of the next generation.

Internal steps from (2) to (4) are repeated until the $g_{\max, \text{PSO}}$ of the internal PSO algorithm is satisfied.

End

Consistent with the **Ab-Ag** affinity metaphor, an **Ab-Ag** affinity is determined using (16) as follows:

$$\max(\text{affinity}_j) = -1 \times f(\mathbf{x}_{\text{PSO}}^*) \quad j = 1, 2, \dots, rs_{\text{AIA}}. \quad (16)$$

Following the evaluation of the **Ab-Ag** affinities of **Ab**s in the current **Ab** repertoire, the **Ab** with the highest **Ab-Ag** affinity (**Ab***) is chosen to undergo clonal selection in external Step 3.

Step 3 (perform a clonal selection operation). To control the number of antigen-specific **Ab**s, (12) is employed.

Step 4 (implement an **Ab-Ag** affinity maturation operation). The intermediate **Ab** repertoire that is created in external

TABLE 2: Numerical results obtained from the proposed RGA-PSO algorithm for solving 14 UGO problems.

TP number	Function name	Global minimum	Required accuracy	Success rate (%)	$f(\mathbf{x}_{\text{RGA-PSO}}^*)$	$f(\mathbf{x}_{\text{RGA-PSO}}^{\text{mean}})$	$f(\mathbf{x}_{\text{RGA-PSO}}^{\text{worst}})$	ME	MCCT (sec.)
1	SHCB	-1.0316	1E-4	100	-1.0316	-1.0316	-1.0316	0.0000000	40.56
2	GP	3	1E-3	100	3	3	3	0.0000000	38.29
3	ES	-1	1E-3	100	-1	-1	-1	0.0000000	41.29
4	B2	0	1E-2	100	0.0000000	0.0000000	0.0000000	0.0000000	40.26
5	DJ	0	1E-4	100	0.0000000	5.533E-226	5.533E-226	5.533E-226	40.50
6	Booth	0	1E-4	100	0.0000000	0.0000000	0.0000000	0.0000000	37.57
7	RC	5/4 π	1E-3	100	0.39788735	0.39788735	0.39788735	0.0000000	40.97
8	RA	-2	1E-4	100	-2	-2	-2	0.0000000	39.31
9	RS ₂	0	1E-3	100	0.0000000	8.709E-28	8.686E-26	8.709E-28	37.47
10	RS ₅	0	1E-3	100	9.587E-12	9.452E-06	9.058E-05	9.452E-06	306.15
11	SH	-186.7309	1E-2	100	-186.7309	-186.7309	-186.7309	0.0000000	54.11
12	ZA ₂	0	1E-4	100	0.0000000	0.0000000	0.0000000	0.0000000	40.74
13	ZA ₅	0	1E-4	100	0.0000000	2.660E-117	2.660E-115	2.660E-117	157.09
14	ZA ₁₀	0	1E-3	100	2.453E-120	4.375E-26	2.933E-24	4.375E-26	194.73

$f(\mathbf{x}_{\text{RGA-PSO}}^*)$: the objective value obtained using the best RGA-PSO solution.

$f(\mathbf{x}_{\text{RGA-PSO}}^{\text{mean}})$: mean objective value obtained using the RGA-PSO solutions.

$f(\mathbf{x}_{\text{RGA-PSO}}^{\text{worst}})$: the worst objective value obtained using the worst RGA-PSO solution.

Step 3 is divided into two subsets. These **Abs** undergo somatic hypermutation operation by using (13) when the random number is 0.5 or less. Notably, these **Abs** suffer receptor-editing operation using (14) when the random number exceeds 0.5.

Step 5 (introduce diverse **Abs**). Based on the bone marrow operation, diverse **Abs** are created to recruit the **Abs** suppressed in external Step 3.

Step 6 (Update an **Ab** repertoire). A new **Ab** repertoire is generated from external Steps 3–5. The **Ab-Ag** affinities of the **Abs** in the generated **Ab** repertoire are evaluated. This work presents a strategy for updating the **Ab** repertoire. A situation in which the **Ab-Ag** affinity of Ab_j in the new **Ab** repertoire exceeds that in the current **Ab** repertoire implies that a strong **Ab** in the new **Ab** repertoire replaces the weak **Ab** in the current **Ab** repertoire. Additionally, a situation in which the **Ab-Ag** affinity of Ab_j in the new **Ab** repertoire equals to or is worse than that in the current **Ab** repertoire implies that the Ab_j in the current **Ab** repertoire survives. In addition to maintaining the strong **Abs**, this strategy eliminates nonfunctional **Abs**

Repeat external Steps 2–6 until the termination criterion $g_{\max, AIA}$ is satisfied.

4. Results

The proposed RGA-PSO and AIA-PSO algorithms were applied to a set of benchmark UGO problems taken from other studies [9, 10, 17, 26], as detailed in the Appendix. The proposed RGA-PSO and AIA-PSO approaches were coded in MATLAB software and run on a Pentium D 3.0 (GHz) personal computer. One-hundred independent runs were conducted for each test problem (TP). To have comparable numerical results, the accuracy was chosen based on the numerical results reported in [9, 10, 17, 26]. Numerical results were summarized, including rate of successful minimizations (success rate %), best mean worst, mean computational CPU time (MCCT), and mean error ME (average value of the gap between the objective function values calculated using the AIA-PSO and RGA-PSO solutions and the known global minimum value). Table 1 lists the parameter settings for the proposed RGA-PSO and AIA-PSO approaches. The table shows 20,000 (20×1000) objective function evaluations of the internal PSO approach for an UGO problem with $N \leq 3$ decision variables and 60,000 (20×3000) objective function evaluations of the internal PSO for an UGO problem with $5 \leq N \leq 30$ decision variables. Moreover, the external AIA and RGA methods stop when $g_{\max, RGA} = 20$ and $g_{\max, AIA} = 20$ are met or the best fitness value of the RGA approach (or the best **Ab-Ag** affinity of the AIA method) does not significantly change for the past five generations.

4.1. Numerical Results Obtained Using the RGA-PSO and AIA-PSO Algorithms for Low-Dimensional UGO Problems ($2 \leq N \leq 10$). Table 2 lists the numerical results obtained using the proposed RGA-PSO algorithm. The numerical results indicate that the RGA-PSO algorithm can obtain the global minimum for each test UGO problem since these MEs equal

or closely approximate “0,” and the RGA-PSO algorithm has an acceptable MCCT for each TP. Table 3 lists the optimal parameter settings obtained using the proposed RGA-PSO algorithm to solve 14 UGO problems.

Table 4 lists the numerical results obtained using the proposed AIA-PSO algorithm. Numerical results indicate that the AIA-PSO algorithm can obtain the global minimum for each test UGO problem since these MEs equal or closely approximate “0,” and that the AIA-PSO algorithm has an acceptable MCCT for each TP. Table 5 lists the optimal parameter settings obtained using the proposed AIA-PSO algorithm for solving 14 UGO problems.

4.2. Numerical Results Obtained Using the RGA-PSO and AIA-PSO Algorithms for a Standard-Dimensional UGO Problem ($N = 30$). To investigate the effectiveness of the RGA-PSO and AIA-PSO methods for solving a standard-dimensional UGO problem, the Zakharov problem with 30 decision variables (ZA_{30}), as described in the Appendix, has been solved using the RGA-PSO and AIA-PSO approaches. Fifty independent runs were performed to solve the UGO problem. To increase the diversity of candidate solutions for use in the external RGA method, the parameter $p_{m, RGA}$ was set from 0.15 to “1.” Table 6 lists the numerical results obtained using the RGA-PSO and AIA-PSO approaches. This table indicates that the two approaches converge to the global optimum value, since the MEs closely approximate “0,” and the MCCT of the RGA-PSO method is larger than that of the AIA-PSO method. Moreover, the success rates of the proposed RGA-PSO and AIA-PSO approaches are 100%. The Wilcoxon test is performed to the difference of median values of the MEs obtained using the RGA-PSO and AIA-PSO methods. The P value of the Wilcoxon test is 0.028, which is smaller than the significance level of 0.05, indicating that the performance of the RGA-PSO method is statistically different to that of the AIA-PSO method. Table 7 summarizes the optimal parameter settings obtained using the proposed RGA-PSO and AIA-PSO algorithms for the UGO problem ZA_{30} .

The UGO problem ZA_{50} was solved using the RGA-PSO and AIA-PSO methods. The RGA-PSO and AIA-PSO methods fail to solve the UGO problem, since the diversity of the particle swarm in the internal PSO method cannot be maintained. Hence, future work will focus on improving the diversity of the particle swarm by applying mutation operations.

4.3. Comparison. Table 8 lists the results of the Wilcoxon test for the MEs obtained using the proposed RGA-PSO and AIA-PSO methods for 14 UGO problems. In this table, the “**” represents the P value of Wilcoxon test which cannot be obtained, since the MEs obtained using the RGA-PSO and AIA-PSO methods for a TP are identical. Moreover, the median values of the MEs obtained using the RGA-PSO and AIA-PSO methods for TP 10 are not statistically different, since their P value is larger than the significance level of 0.05. Overall, the performances obtained using the RGA-PSO and AIA-PSO methods are statistically identical.

TABLE 3: The optimal parameter settings obtained using the proposed RGA-PSO algorithm for solving 14 UGO problems.

TP number	Function name	χ	c_1	c_2
1	SHCB	0.36497798	1.79972890	2.36717639
2	GP	0.86138264	3.74964714	0.10000000
3	ES	0.36038653	0.42043804	4.78367612
4	B2	0.75486133	0.17875492	0.95144705
5	DJ	0.39357143	3.17898242	4.72512283
6	Booth	0.56703400	4.84740898	2.58109482
7	RC	0.44153938	2.71748076	3.81076900
8	RA	0.93533829	0.76530619	3.53402840
9	RS ₂	0.87003776	1.28577955	2.77423057
10	RS ₅	0.62553666	0.10000000	5.00000000
11	SH	0.45903159	1.07452631	4.15038556
12	ZA ₂	0.75044069	0.50883133	1.93860581
13	ZA ₅	0.52013863	1.05471011	4.61197390
14	ZA ₁₀	0.56126516	0.39456605	4.95735921

Table 9 compares the numerical results obtained using the RGA-PSO and AIA-PSO methods with those obtained using the hybrid algorithms for 11 TPs. Specifically, the table lists the numerical results obtained using the Nelder-Mead simplex search and PSO (NM-PSO) and GA-PSO taken from [9], those obtained using particle swarm ant colony optimization (PSACO), continuous hybrid algorithm (CHA) and continuous tabu simplex search (CTSS) taken from [10], and those obtained using DE-PSO, AMPSO1, and AMPSO2 taken from [12]. Table 9 indicates that the RGA-PSO and AIA-PSO methods yield superior accuracy of *MEs* obtained using the NM-PSO, GA-PSO, DE-PSO, AM-PSO1, AM-PSO2, CHA and CTSS approaches for TPs 2, 3, 4, 5, 7, 9, 11, and 12 and that RGA-PSO and AIA-PSO approaches yield superior accuracy of *MEs* obtained using the PSACO method for TPs 4, 5, 7, 9, 10, 11, 12, 13 and 14. Table 10 compares the percentage success rates of the proposed RGA-PSO and AIA-PSO approaches and those of the hybrid algorithms for 11 TPs, indicating that all algorithms except for the CHA and CTSS methods achieved identical performance (100% success rate) for all TPs.

4.4. Summary of Results. The proposed RGA-PSO and AIA-PSO algorithms have the following benefits.

- (1) Parameter manipulation of the internal PSO algorithm is based on the solved UGO problems. Owing to their ability to efficiently solve UGO problems, the external RGA and AIA approaches are substituted for trial and error to manipulate the parameters (χ , c_1 , and c_2).
- (2) Besides obtaining the optimum parameter settings of the internal PSO algorithm, the RGA-PSO and AIA-PSO algorithms can yield a global minimum for an UGO problem.
- (3) Beside, outperforming some published hybrid SGO methods, the proposed RGA-PSO and AIA-PSO

approaches reduce the parametrization for the internal PSO algorithm, despite being more complex than individual SGO approaches.

The proposed RGA-PSO and AIA-PSO algorithms are limited in that they cannot solve high-dimensional UGO problems (such as $N = 50$). Future work will focus on increasing the diversity of the particle swarm of the internal PSO method by applying mutation to solve high-dimensional UGO problems.

5. Conclusions

This work developed RGA-PSO and AIA-PSO algorithms. Performances of the proposed RGA-PSO and AIA-PSO approaches were evaluated using a set of benchmark UGO problems. Numerical results indicate that the proposed RGA-PSO and AIA-PSO methods can converge to global minimum for each test UGO problem and obtain the best parameter settings of the internal PSO algorithm. Moreover, the numerical results obtained using the RGA-PSO and AIA-PSO algorithms are superior to those obtained using many alternative hybrid SGO methods. The RGA-PSO and AIA-PSO methods can thus be considered efficient SGO approaches for solving standard-dimensional UGO problems.

Appendix

(1) *Six-hump camel back (SHCB) (two variables) [17]:*

$$f(\mathbf{x}) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3} \right) x_1^2 + x_1x_2 + (-4 + 4x_2^2) x_2^2 \quad (\text{A.1})$$

search domain: $-3 \leq x_1 \leq 3$; $-2 \leq x_2 \leq 2$

one global minimum at two different points: $\mathbf{x}^* = (-0.0898, 0.7126)$ and $\mathbf{x}^* = (0.0898, -0.7126)$, $f(\mathbf{x}^*) = -1.0316$.

TABLE 4: Numerical results obtained from the proposed AIA-PSO algorithm for solving 14 UGO problems.

TP number	Function name	Global minimum	Required accuracy	Success rate (%)	$f(\mathbf{x}_{\text{AIA-PSO}}^*)$	$f(\mathbf{x}_{\text{AIA-PSO}}^{\text{mean}})$	$f(\mathbf{x}_{\text{AIA-PSO}}^{\text{worst}})$	ME	MCCT (sec)
1	SHCB	-1.0316	$1E-4$	100	-1.0316	-1.0316	-1.0316	0.00000000	39.69
2	GP	3	$1E-3$	100	3	3	3	0.00000000	37.80
3	ES	-1	$1E-3$	100	-1	-1	-1	0.00000000	42.08
4	B2	0	$1E-2$	100	0.00000000	0.00000000	0.00000000	0.00000000	40.43
5	DJ	0	$1E-4$	100	0.00000000	$2.544E-215$	$2.544E-213$	$2.544E-215$	39.43
6	Booth	0	$1E-4$	100	0.00000000	0.00000000	0.00000000	0.00000000	37.48
7	RC	$5/4\pi$	$1E-3$	100	0.39788735	0.39788735	0.39788735	0.00000000	40.53
8	RA	-2	$1E-4$	100	-2	-2	-2	0.00000000	39.17
9	RS ₂	0	$1E-3$	100	0.00000000	$2.341E-27$	$2.129E-25$	$2.341E-27$	37.49
10	RS ₅	0	$1E-3$	100	$4.342E-10$	$7.291E-06$	$9.313E-05$	$7.291E-06$	304.23
11	SH	-186.7309	$1E-2$	100	-186.7309	-186.7309	-186.7309	0.00000000	54.70
12	ZA ₂	0	$1E-4$	100	0.00000000	0.00000000	0.00000000	0.00000000	40.31
13	ZA ₅	0	$1E-4$	100	0.00000000	$7.2591E-184$	$7.259E-182$	$7.259E-184$	144.38
14	ZA ₁₀	0	$1E-3$	100	$1.472E-120$	$3.287E-42$	$2.856E-40$	$3.287E-42$	192.06

$f(\mathbf{x}_{\text{AIA-PSO}}^*)$: the objective value obtained using the best AIA-PSO solution.

$f(\mathbf{x}_{\text{AIA-PSO}}^{\text{mean}})$: mean objective value obtained using the AIA-PSO solutions.

$f(\mathbf{x}_{\text{AIA-PSO}}^{\text{worst}})$: the worst objective value obtained using the worst AIA-PSO solution.

TABLE 5: The optimal parameter settings obtained using the proposed AIA-PSO algorithm for solving 14 UGO problems.

TP number	Function name	χ	c_1	c_2
1	SHCB	0.59403239	1.75960560	4.12826475
2	GP	0.29584147	2.62468918	4.41787729
3	ES	0.39500323	0.84589484	2.64979988
4	B2	0.64806710	1.78727677	1.83995777
5	DJ	0.45630341	1.11829588	4.10382418
6	Booth	0.60889449	2.53158038	1.32827344
7	RC	1.00000000	1.48772085	0.43921335
8	RA	0.38157302	4.93248942	4.08732827
9	RS ₂	0.39872504	4.65992304	4.84228028
10	RS ₅	0.55519288	0.10000000	4.52483714
11	SH	0.54293231	2.10545153	4.52889414
12	ZA ₂	0.44530811	1.05474921	3.20092987
13	ZA ₅	0.51683760	1.17129205	4.94965004
14	ZA ₁₀	0.53688659	0.53752883	4.99881022

TABLE 6: Numerical results obtained from the proposed RGA-PSO and AIA-PSO algorithms for solving a standard-dimensional UGO problem.

Function name	Global minimum	Required accuracy	Methods	Success rate (%)	$f(\mathbf{x}_{\text{RGA-PSO}}^*)$	$f(\mathbf{x}_{\text{RGA-PSO}}^{\text{mean}})$	$f(\mathbf{x}_{\text{RGA-PSO}}^{\text{worst}})$	ME	MCCT (sec)
ZA ₃₀	0	$1E-3$	RGA-PSO	100	$4.186E-13$	$4.537E-06$	$5.331E-05$	$4.537E-06$	1753.86
			AIA-PSO	100	$6.520E-12$	$8.099E-06$	$9.175E-05$	$8.099E-06$	1453.98

(2) *Goldstein-price (GP) (two variables)* [9, 10]:

$$f(\mathbf{x}) = \left[1 + (x_1 + x_2 + 1)^2 \right. \\ \times (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2) \\ \times [30 + (2x_1 - 3x_2)^2 \\ \times (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)] \quad (\text{A.2})$$

search domain: $-2 \leq x_n \leq 2, n = 1, 2$
four local minima; one global minimum: $\mathbf{x}^* = (0, -1)$,
 $f(\mathbf{x}^*) = 3$.

(3) *Easom (ES) (two variables)* [9]:

$$f(\mathbf{x}) = -\cos(x_1) \cos(x_2) e^{-(x_1 - \pi)^2 - (x_2 - \pi)^2} \quad (\text{A.3})$$

search domain: $-100 \leq x_n \leq 100, n = 1, 2$
several local minima (exact number unspecified in
usual literature);
one global minimum: $\mathbf{x}^* = (\pi, \pi)$, $f(\mathbf{x}^*) = -1$.

(4) *B2 (two variables)* [9, 10, 17]:

$$f(\mathbf{x}) = x_1^2 + 2x_1^2 - 0.3 \cos(3\pi x_1) - 0.4 \cos(4\pi x_2) + 0.7 \quad (\text{A.4})$$

search domain: $-100 \leq x_n \leq 100, n = 1, 2$

several local minima (exact number unspecified in
usual literature);

one global minimum $\mathbf{x}^* = (0, 0)$, $f(\mathbf{x}^*) = 0$.

(5) *De Jong (DJ) (three variables)* [9, 10]:

$$f(\mathbf{x}) = \sum_{n=1}^3 x_n^2 \quad (\text{A.5})$$

search domain: $-5.12 \leq x_n \leq 5.12, n = 1, 2, 3$

one global minimum: $\mathbf{x}^* = (0, 0, 0)$, $f(\mathbf{x}^*) = 0$.

(6) *Booth (BO) (two variables)* [26]:

$$f(\mathbf{x}) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2 \quad (\text{A.6})$$

search domain: $-10 \leq x_n \leq 10, n = 1, 2$

one global minimum: $\mathbf{x}^* = (1, 3)$, $f(\mathbf{x}^*) = 0$.

(7) *Branin RCOC (RC) (two variables)* [9, 10]:

$$f(\mathbf{x}) = \left(x_2 - \frac{5.1}{4\pi^2} + \frac{5}{\pi} x_1 - 6 \right)^2 \\ + 10 \left(1 - \frac{1}{8\pi} \right) \cos(x_1) + 10 \quad (\text{A.7})$$

TABLE 7: The optimal parameter settings obtained using the proposed RGA-PSO and AIA-PSO algorithms for solving a standard-dimensional UGO problem.

Function name	Methods	χ	c_1	c_2
ZA ₃₀	RGA-PSO	0.58721675	0.56183025	4.91401312
	AIA-PSO	0.61082112	0.38556559	5.00000000

TABLE 8: Results of Wilcoxon test for the MEs obtained using the proposed RGA-PSO and AIA-PSO methods for 14 UGO problems.

TP number	Function name	RGA-PSO versus AIA-PSO P value
1	SHCB	**
2	GP	**
3	ES	**
4	B2	**
5	DJ	**
6	Booth	**
7	RC	**
8	RA	**
9	RS ₂	**
10	RS ₅	0.833
11	SH	**
12	ZA ₂	**
13	ZA ₅	**
14	ZA ₁₀	**

search domain: $-5 \leq x_1 \leq 10, 0 \leq x_2 \leq 15$

no local minimum;

three global minima: $\mathbf{x}^* = (-\pi, 12.275)$, $\mathbf{x}^* = (\pi, 2.275)$, $\mathbf{x}^* = (3\pi, 2.475)$, $f(\mathbf{x}^*) = 5/4\pi$.

(8) *Rastrigin (RA) (two variables) [10]:*

$$f(\mathbf{x}) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2) \quad (\text{A.8})$$

search domain: $-1 \leq x_n \leq 1, n = 1, 2$

50 local minima;

One global minimum: $f(\mathbf{x}^*) = -2$, $\mathbf{x}^* = (0, 0)$.

(9) *Rosenbrock (RSn) (N variables) [9, 10]:*

$$f(\mathbf{x}) = \sum_{n=1}^{N-1} \left[100(x_n^2 - x_{n+1})^2 + (x_n - 1)^2 \right] \quad (\text{A.9})$$

Two functions were considered: RS₂, RS₅

search domain: $-5 \leq x_n \leq 10, n = 1, 2, \dots, N$

several local minima (exact number unspecified in usual literature);

global minimum: $\mathbf{x}^* = (1, 1)$, $f(\mathbf{x}^*) = 0$.

(10) *Shubert (SH) (two variables) [9, 10]:*

$$f(\mathbf{x}) = \sum_{n=1}^{N-1} \left[100(x_n^2 - x_{n+1})^2 + (x_n - 1)^2 \right] \quad (\text{A.10})$$

search domain: $-10 \leq x_n \leq 10, n = 1, 2$

760 local minima;

18 global minima;

$f(\mathbf{x}^*) = -186.7309$.

(11) *Zakharov (ZAn) (N variables) [9, 10]:*

$$f(\mathbf{x}) = \sum_{n=1}^N x_n^2 + \left(\sum_{n=1}^N 0.5nx_n \right)^2 + \left(\sum_{n=1}^N 0.5nx_n \right)^4. \quad (\text{A.11})$$

Three functions were considered: ZA₂, ZA₅, ZA₁₀, ZA₃₀

search domain: $-5 \leq x_n \leq 10, n = 1, 2, \dots, N$

several local minima (exact number unspecified in usual literature);

global minimum: $\mathbf{x}^* = (0, \dots, 0)$, $f(\mathbf{x}^*) = 0$.

Conflict of Interests

The author confirm that he does not has a conflict of interest with the MATLAB software.

Acknowledgment

The author would like to thank the National Science Council of the Republic of China, Taiwan for financially supporting this research under Contract no. NSC 100-2622-E-262-006-CC3.

TABLE 9: Comparison of the results of the proposed RGA-PSO and AIA-PSO approaches and those of the hybrid algorithms for 11 TPs.

TP number	Function name	MEs									
		NM-PSO [9]	PSACO [10]	GA-PSO [9]	DE-PSO [12]	AM-PSO1 [12]	AM-PSO2 [12]	CHA [10]	CTSS [10]	AIA-PSO (this work)	RGA-PSO (this work)
2	GP	0.00003	0.00000000	0.00012	3.67E-5	3.81E-5	9.89E-6	0.0010	0.001	0.00000000	0.00000000
3	ES	0.00004	0.00000000	0.00003	1.67E-5	4.10E-5	3.71E-5	0.0010	0.005	0.00000000	0.00000000
4	B2	0.00003	5.5511E-17	0.00001	1.11E-5	1.17E-6	2.64E-5	0.0000002	0.000005	0.00000000	0.00000000
5	DJ	0.00003	7.6900E-29	0.00004	1.67E-5	4.23E-5	3.34E-5	0.0002	0.0002	2.544E-215	5.5331E-226
7	RC	0.00003	2.6185E-13	0.00009	2.29E-6	1.52E-5	1.49E-5	0.0001	0.005	0.00000000	0.00000000
9	RS ₂	0.00003	1.7152E-10	0.00064	2.46E-5	2.16E-5	3.33E-5	0.0040	0.0040	2.341E-27	8.709E-28
10	RS ₅	0.0056	1.8538E-04	0.00013	7.74E-5	9.73E-5	9.93E-5	—	—	7.291E-06	9.452E-06
11	SH	0.00002	1.09239E-09	0.00007	3.03E-6	1.72E-5	2.54E-7	0.0050	0.001	0.00000000	0.00000000
12	ZA ₂	0.00003	5.7061E-27	0.00005	1.12E-5	2.54E-5	2.61ZE-6	3E-6	3E-7	0.00000000	0.00000000
13	ZA ₅	0.00026	3.6352E-17	0.00000	3.49E-5	3.44E-5	5.07E-5	—	—	7.259E-184	2.660E-117
14	ZA ₁₀	—	2E-08	0.00000	6.91E-5	7.41E-5	9.11E-5	1E-6	—	3.287E-42	4.3756E-26

("—" denotes unavailable information).

TABLE 10: Comparison of the success rate % of the proposed RGA-PSO and AIA-PSO approaches and those of the hybrid algorithms for 11 TPs.

TP number	Function name	Success rate %									
		NM-PSO [9]	PSACO [10]	GA-PSO [9]	DE-PSO [12]	AM-PSO1 [12]	AM-PSO2 [12]	CHA [10]	CTSS [10]	AIA-PSO (this work)	RGA-PSO (this work)
2	GP	100	100	100	100	100	100	100	100	100	100
3	ES	100	100	100	100	100	100	100	100	100	100
4	B2	100	100	100	100	100	100	100	100	100	100
5	DJ	100	100	100	100	100	100	100	100	100	100
7	RC	100	100	100	100	100	100	100	100	100	100
9	RS ₂	100	100	100	100	100	100	100	100	100	100
10	RS ₅	100	100	100	100	100	100	—	—	100	100
11	SH	100	100	100	100	100	100	100	100	100	100
12	ZA ₂	100	100	100	100	100	100	100	100	100	100
13	ZA ₅	100	100	100	100	100	100	—	—	100	100
14	ZA ₁₀	100	100	100	100	100	100	100	—	100	100

("—" denotes unavailable information).

References

- [1] W. Y. Yang, W. Cao, T.-S. Chung, and J. Morris, *Applied Numerical Methods Using MATLAB*, John Wiley & Sons, Hoboken, NJ, USA, 2005.
- [2] C. Hamzaçebi, "Improving genetic algorithms' performance by local search for continuous function optimization," *Applied Mathematics and Computation*, vol. 196, no. 1, pp. 309–317, 2008.
- [3] C. C. Chen, "Two-layer particle swarm optimization for unconstrained optimization problems," *Applied Soft Computing Journal*, vol. 11, no. 1, pp. 295–304, 2011.
- [4] X. Zhao, "A perturbed particle swarm algorithm for numerical optimization," *Applied Soft Computing Journal*, vol. 10, no. 1, pp. 119–124, 2010.
- [5] M. D. Toksari, "Minimizing the multimodal functions with Ant Colony Optimization approach," *Expert Systems with Applications*, vol. 36, no. 3, pp. 6030–6035, 2009.
- [6] J. Kelsey and J. Timmis, "Immune inspired somatic contiguous hypermutation for function optimisation," in *Proceedings of the Genetic and Evolutionary Computation (GECCO '03)*, pp. 207–208, Chicago, Ill, USA, 2003.
- [7] M. Gang, Z. Wei, and C. Xiaolin, "A novel particle swarm optimization algorithm based on particle migration," *Applied Mathematics and Computation*, vol. 218, no. 11, pp. 6620–6626, 2012.
- [8] H. Poorzahedy and O. M. Rouhani, "Hybrid meta-heuristic algorithms for solving network design problem," *European Journal of Operational Research*, vol. 182, no. 2, pp. 578–596, 2007.
- [9] Y. T. Kao and E. Zahara, "A hybrid genetic algorithm and particle swarm optimization for multimodal functions," *Applied Soft Computing Journal*, vol. 8, no. 2, pp. 849–857, 2008.
- [10] P. S. Shelokar, P. Siarry, V. K. Jayaraman, and B. D. Kulkarni, "Particle swarm and ant colony algorithms hybridized for improved continuous optimization," *Applied Mathematics and Computation*, vol. 188, no. 1, pp. 129–142, 2007.
- [11] M. R. Chen, X. Li, X. Zhang, and Y. Z. Lu, "A novel particle swarm optimizer hybridized with extremal optimization," *Applied Soft Computing Journal*, vol. 10, no. 2, pp. 367–373, 2010.
- [12] R. Thangaraj, M. Pant, A. Abraham, and P. Bouvry, "Particle swarm optimization: hybridization perspectives and experimental illustrations," *Applied Mathematics and Computation*, vol. 217, no. 12, pp. 5208–5226, 2011.
- [13] Z. H. Zhan, J. Zhang, Y. Li, and H. S. H. Chung, "Adaptive particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 39, no. 6, pp. 1362–1381, 2009.
- [14] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.
- [15] M. Jiang, Y. P. Luo, and S. Y. Yang, "Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm," *Information Processing Letters*, vol. 102, no. 1, pp. 8–16, 2007.
- [16] J. Y. Wu, "Solving constrained global optimization problems by using hybrid evolutionary computing and artificial life approaches," *Mathematical Problems in Engineering*, vol. 2012, Article ID 841410, 36 pages, 2012.
- [17] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, New York, NY, USA, 1999.
- [18] C. R. Houck, J. A. Joines, M. G. Kay, and in: "A genetic algorithm for function optimization: a MATLAB implementation," in *NSCU-IE TR 95-09*, North Carolina State University, Raleigh, NC, USA, 1995.
- [19] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, WA, Australia, December 1995.
- [20] Y. Shi and R. Eberhart, "Modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, Anchorage, Alaska, USA, May 1998.
- [21] M. Clerc, "The swarm and the queen: towards a deterministic and adaptive particle swarm optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1951–1957, Washington, DC, USA, 1999.
- [22] 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.
- [23] A. P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, John Wiley & Sons, 2005.
- [24] J. Y. Wu, "Solving constrained global optimization via artificial immune system," *International Journal on Artificial Intelligence Tools*, vol. 20, no. 1, pp. 1–27, 2011.
- [25] L. N. de Castro and F. J. Von Zuben, "Artificial Immune Systems—Part I—Basic Theory and Applications," *FEEC/Universidade Estadual de Campinas, Campinas, Brazil*, 1999, ftp://ftp.dca.fee.unicamp.br/pub/docs/vonzuben/tr_dca/trdca0199.pdf.
- [26] S. K. S. Fan and E. Zahara, "A hybrid simplex search and particle swarm optimization for unconstrained optimization," *European Journal of Operational Research*, vol. 181, no. 2, pp. 527–548, 2007.

Research Article

Escape-Route Planning of Underground Coal Mine Based on Improved Ant Algorithm

Guangwei Yan and Dandan Feng

*School of Control and Computer Engineering, North China Electric Power University,
Beijing 102206, China*

Correspondence should be addressed to Guangwei Yan; yan_guang_wei@126.com

Received 2 October 2012; Revised 1 December 2012; Accepted 4 December 2012

Academic Editor: Baozhen Yao

Copyright © 2013 G. Yan and D. Feng. 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.

When a mine disaster occurs, to lessen disaster losses and improve survival chances of the trapped miners, good escape routes need to be found and used. Based on the improved ant algorithm, we proposed a new escape-route planning method of underground mines. At first, six factors which influence escape difficulty are evaluated and a weight calculation model is built to form a weighted graph of the underground tunnels. Then an improved ant algorithm is designed and used to find good escape routes. We proposed a tunnel network zoning method to improve the searching efficiency of the ant algorithm. We use max-min ant system method to optimize the meeting strategy of ants and improve the performance of the ant algorithm. In addition, when a small part of the mine tunnel network changes, the system may fix the optimal routes and avoid starting a new processing procedure. Experiments show that the proposed method can find good escape routes efficiently and can be used in the escape-route planning of large and medium underground coal mines.

1. Introduction

In mining, water, fire, gas, and other natural disasters often occur. The disasters have a heavy effect to mines' safety production. Statistics show the coal mine industry has the most serious casualty accidents in China [1]. From 1991 to 2001, 86000 people died in coal mine accident, accounts for 85% of deaths in the mining industry. According to statistics in 2006–2008, in high-risk industries of China, the proportion of coal mine accidents and deaths equal 21.3% and 28.5%, respectively, top the list of industrial and mining business enterprises [2].

When mine disaster occurs, it is very important to find good escape routes. Escape routes planning can be realized based on computers, sensor networks, and relevant data. This problem had been studied at home and abroad, literature [3–7] use the Dijkstra algorithm or its improved algorithm to solve the problem. These algorithms are different in time complexity, space complexity, and so forth. The Dijkstra algorithm has three shortages. First, large amounts of calculation are required when number of network nodes and edges reaches several hundred, because these algorithms usually

need to traverse all vertices of the network. Second, the Dijkstra algorithm only obtains the best route and in fact the second-best route is needed sometimes. Third, the Dijkstra algorithm is static, if mine states change, it needs to be recalculated.

Put forward by Italian scholar Dorigo et al., the ant algorithm [8] is a new heuristic evolutionary algorithm. Biological ant colony can produce a chemical substance called pheromone for communication and coordination of ants. Pheromone can form positive feedback and make individual ants gradually gathered in the shortest route between the food source and the nest. The algorithm has advantages of strong robustness, distributed computing, positive feedback mechanism and it is easy to combine with other algorithms. When solving the shortest route in large-scale network, it has excellent feasibility and adaptability.

In this paper, factors which influence passing difficulty in tunnels are led into. Based on these factors and the actual lengths of the tunnels, equivalent lengths of the tunnels are calculated and acted as weights of the underground tunnel graph. This graph acts as the input of the improved ant algorithm.

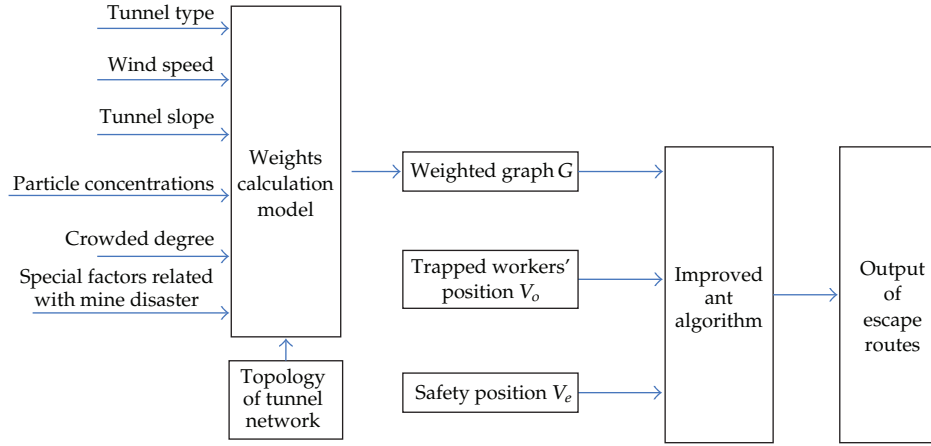


FIGURE 1: System Framework.

In this paper, an important improvement to the ant algorithm is introducing tunnel zoning. Underground mine networks include many faces and main tunnels. This paper divides all mine zones into two types: the backbone-zone S_0 and the nonbackbone-zone S_n ($n = 1, 2, 3 \dots$). The backbone-zone S_0 is the main searching area, containing backbone tunnels and the safety point. The nonbackbone-zone S_n containing several zones, each face is a nonbackbone-zone. They connect to the main road (See Figure 2). By zoning the weighted graph of tunnels, the ants' behaviors can be affected and the searching efficiency can be improved.

The experimental results in this paper show the improved algorithm are suitable for escape routes planning of large and medium sized mines.

2. System Framework

To calculate the optimal escape routes in underground mines, tunnels' status and data should be provided. And these can be obtained by sensor network in tunnels. In different mine disasters, factors that influence men passing difficulty of the tunnel are various and complex. Based on methods proposed by literature [9, 10], these factors are generalized for six classes including tunnel type, wind speed, tunnel slope, particle concentrations, crowded degree, and special factors related with mine disaster. Firstly, these factors are converted to tunnel equivalent length according to the formula (1). Secondly, a graph denoted by $G(V, E)$ is built based on the tunnel network topology and the normalized tunnel equivalent lengths which act as the graph weights. Thirdly, a group of underground workers is set as node V_0 and a safety point or exit point is set as V_e in G . Finally, improved ant algorithm is used to plan the optimal route and several alternative routes between the terminals (V_0, V_e) in G . The system implementation framework is shown in Figure 1.

3. Calculation of Tunnel Weights

3.1. Calculation of Tunnel Equivalent Length. Status of underground tunnels are various and complex. Factors influencing passing difficulty of tunnels are generalized for six classes

including tunnel type, wind speed, tunnel slope, particle concentrations, crowded degree, and special factors related to mine disasters. These factors influence people's escape speed and they can increase or decrease the escape speed v . The influences of these factors to the escape speed v are transformed to the tunnels' equivalent length. The smaller the value of the speed v is, the longer the tunnel equivalent length is and vice versa. Formula (1) gives the calculation method of the tunnel equivalent lengths. And these equivalent lengths are as weights of the graph G after normalizing:

$$l_i = (k_{ti} \cdot k_{wi} \cdot k_{gi} \cdot k_{vi} \cdot k_{mi} \cdot k_{di}) l_{ri}. \quad (1)$$

In (1), l_{ri} indicates the actual length of the tunnel i and l_i is the equivalent length of the tunnel i ; $k_{ti}, k_{wi}, k_{gi}, k_{vi}, k_{mi}, k_{di}$ are tunnel type influence coefficient, tunnel wind speed influence coefficient, tunnel slope influence coefficient, tunnel particle concentrations influence coefficient, tunnel crowded degree influence coefficient, special factors related to mine disasters influence coefficient of the tunnel i , respectively. Below we explain the relationship of these coefficients and the escape speed v .

We denote the people's normal walking speed by v_0 . Under the influence of the tunnel type coefficient, the people escape speed v equals to v_0/k_{ti} . These six coefficients are independent, and under their common influence, the escape speed v equals to $v_0/(k_{ti} \cdot k_{wi} \cdot k_{gi} \cdot k_{vi} \cdot k_{mi} \cdot k_{di})$. Through formula (1), we can know the escape time t (which equals to l_{ri}/v) also equals to l_i/v_0 . Below we will explain in detail how these coefficients are calculated and how they influence the escape speed v separately.

3.1.1. Tunnel Type Influence Coefficient. Underground tunnels can generally be divided into working surface, transport tape lane, contact lane, rail lane, shaft, air leakage branch, and ventilation borehole, in which the air leakage branch and the ventilation borehole are forbidden for people to pass. Tunnel type may influence the people's escape speed. For example, in the rail lane, the passing speed is equal to the speed of

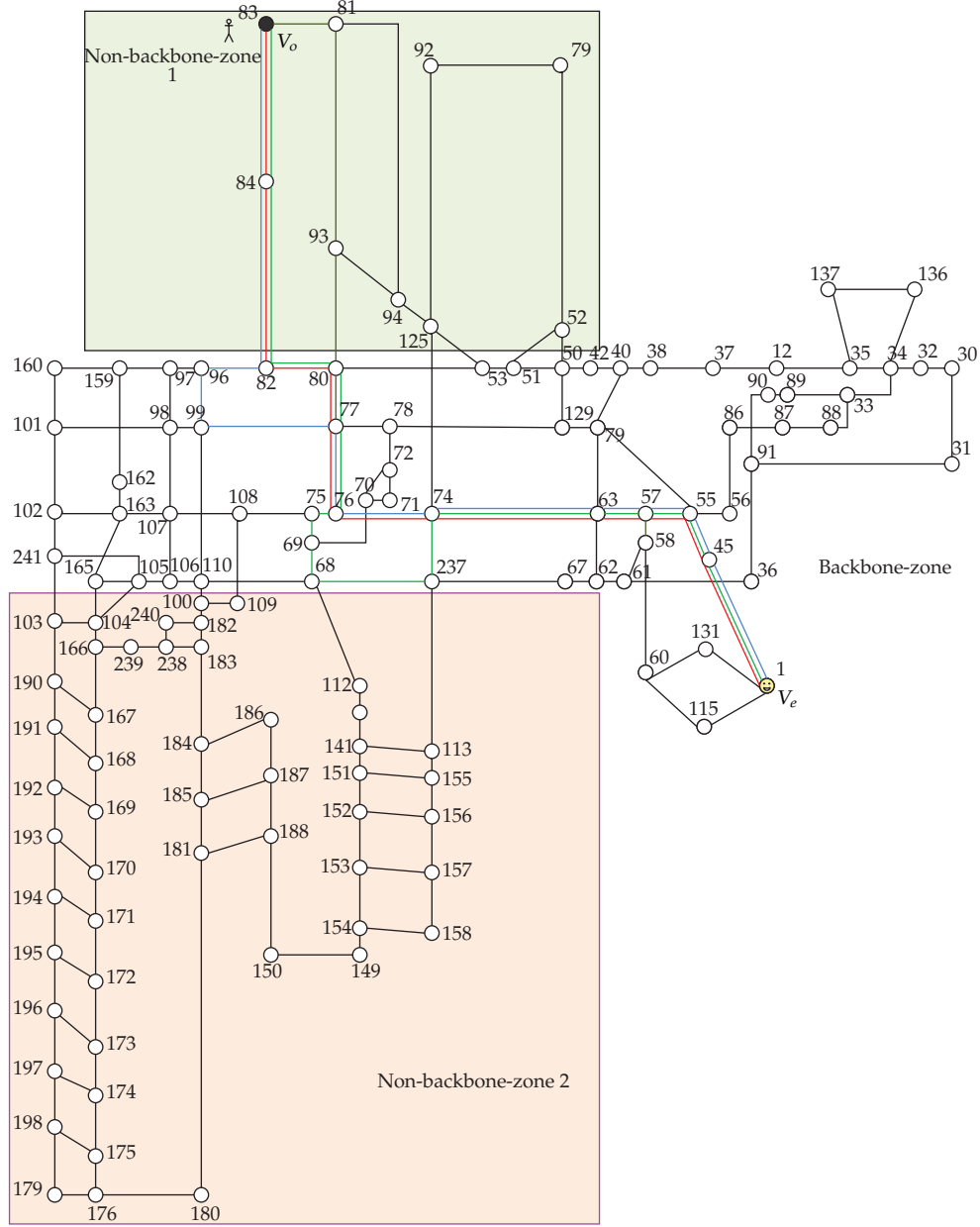


FIGURE 2: The network topology of the mine.

transportation tool. Formula (2) gives the calculating method of the tunnel type influence coefficient.

$$k_{ti} = \begin{cases} \frac{v_0}{v_{\text{vehicle}}}, & \text{tunnel } i \text{ is the rail lane type,} \\ +\infty, & \text{tunnel } i \text{ is the air leakage branch} \\ & \text{or the ventilation borehole type,} \\ 1, & \text{other tunnel type.} \end{cases} \quad (2)$$

The parameter v_{vehicle} represents the speed of the transformation tool. People's moving speed equals to the speed of the vehicle when people are in the vehicle. When the value of v_{vehicle} is greater than the value of v_0 , the value of k_{ti} is smaller

than 1. And this indicates the equivalent length of the tunnel i will decrease according to the formula (1) and the people will spend less time to pass through the tunnel.

3.1.2. Wind Speed Influence Coefficient. The mine networks is a thorough ventilation system itself. Generally in the total return lanes the wind speed is big. The wind speed may influence people's escape speed.

Its influence degree is in direct ratio with the actual roadway length. We assume the human's walking power is constant and denote it by P_0 , and only taking stable states into account, we can derive the formula under against wind situation as follows:

$$P_0 = Fv_0 = (F + F_w)v, \quad (3)$$

where F is the power of the human and v_0 is the normal human walking speed without any outside influence. And F_w is the power of wind and v is the human's speed after affected by wind.

Assume the person is a cuboid object, and F_w can be derived from the Bernoulli's equation [11]:

$$F_w = \frac{c_d \rho S v_w^2}{2}. \quad (4)$$

The parameter c_d is a resistance coefficient, experience shows that it is related to the Reynolds number; ρ is the density of the gas; S is the touching surface area between the human body and the wind; v_w is the wind speed in the current roadway.

This paper concludes (5) from (3) and (4):

$$v = \frac{2v_0 P_0}{2P_0 + c_d \rho S v_0 v_w^2}. \quad (5)$$

The wind speed influence coefficient under against wind situation is shown as

$$k_{wi} = \frac{l_i}{l_{ri}} = \frac{v_0}{v} = \frac{2P_0 + c_d \rho S v_0 v_w^2}{2P_0}. \quad (6)$$

Similarly, the coefficient under following wind situation is shown as follows:

$$k_{wi} = \frac{2P_0 - c_d \rho S v_0 v_w^2}{2P_0}. \quad (7)$$

Formula (7) has a limitation: when the wind speed is too big, this formula is not applicable.

3.1.3. Tunnel Slope Influence Coefficient. Tunnel gradient influences the peoples' walking speed. The greater the slope is, the greater the resistance is. With the same analysis method as Section 3.1.2, we get the formula as follows under climbing situation:

$$P_0 = Fv_0 = mgv \sin \theta_i + Fv \cos \theta_i, \quad (8)$$

where m is the standard human mass, g is the gravity acceleration, θ_i is the current tunnel's angle of slope. The tunnel upslope influence coefficient can be represented as follows:

$$k_{gi} = \frac{v_0}{v} = \frac{mgv_0 \sin \theta_i}{P_0} + \cos \theta_i. \quad (9)$$

When workers pass down slope tunnels, we assume their speed will still be v_0 (i.e., $k_{gi} = 1$).

The parameter values in Sections 3.1.2 and 3.1.3 are shown in Table 1.

3.1.4. Tunnel Particle Concentrations Influence Coefficient. Tunnel particle concentrations mainly include tunnel visibility, the height between the gas critical layer and the floor. They have important effects on the walking speed of people who

TABLE 1: Parameters' values.

Parameter name	Value
ρ	1.2 (kg/m ³)
S	0.7225 (m ²)
c_d	0.8
P_0	200 (w)
m	80 (kg)
v_0	5 (m/s)

are escaping [12]. This coefficient is denoted by k_{vi} , and its calculation method is

$$k_{vi} = (1 + \alpha_h + L_r). \quad (10)$$

In (10), α_h is the effecting coefficient of the height of gas layer and its empirical value is shown in Table 2 [12]; L_r is the affecting coefficient of visibility; its empirical value is shown in Table 3.

3.1.5. Tunnel Crowded Degree. The denseness of the crowd affects the walking speed to some extent. Thompson proposed a speed calculation method which utilized the crowd denseness factor in Simulex model [13]. This method can be expressed as a coefficient calculation formula as follows:

$$k_{mi} = \begin{cases} \frac{1}{\sin \{90^\circ \times ((d-b)/(t_d-b))\}}, & b < d < t_d, \\ 1, & d \geq t_d. \end{cases} \quad (11)$$

In (11), d represents the body interval between people, here the body interval means the distance from one person's body center to another person's body center (we assume the people have the same height); t_d represents the upper limit of the moving constraint interval; b represents the size of the body in horizontal direction.

3.1.6. Special Factors Related to Mine Disaster Influence Coefficient. In order to create a simple and unified mathematical model for various tunnel disasters, we consider special factors related to mine disaster, which refers to the extreme cases of different tunnel disasters, denoted by k_{di} . Its value is in $\{1, +\infty\}$, and 1 indicates that this tunnel is normal and there is not exist mine disaster's influence, and $+\infty$ means this tunnel cannot be passed. There are a lot of factors which can cause the tunnel cannot be passed, for example:

- (1) there is serious landslide in the tunnel and people cannot go through it.
- (2) There exists high temperature (or large amount of smoke, toxic gas) in the tunnel, people cannot bear it.
- (3) There is flood in the tunnel and the water height is more than people's bear limitation.

3.2. The Normalization of Tunnel Weights. According to the above calculation model, the tunnel equivalent length's range is $(0, +\infty]$, in which $+\infty$ indicates that the tunnel cannot

TABLE 2: Value of parameter α_h .

Height of gas layer (m)	Coefficient α_h	Height of gas layer (m)	Coefficient α_h
>6	0	1.6–1.8	2
4–6	0.1	1.4–1.6	5
2–4	0.5	1–1.4	10
1.8–2	1	<1	100

TABLE 3: Value of parameter L_r .

Visibility distance (m)	Coefficient L_r
>20	1
10–20	1.25
5–10	2.95
3–5	6.25
<3	12.5

be got through. For the convenience of data processing, the data is mapped to range (0, 1]. The linear mapping is used as follows:

$$w_i = 0.1 + \frac{(l_i - \min(l))}{(\max(l) - \min(l))} * (0.9 - 0.1). \quad (12)$$

In which $\min(l)$ refers to the minimum equivalent length of all tunnels, $\max(l)$ is the maximum equivalent length of all tunnels (except for the length value $+\infty$). This function means the closer to 0 the values after mapping is, and the easier to pass through it. When the value is 1, it means the tunnel cannot be passed.

Based on calculated weights and the tunnel topology graph $G(V, E)$, the following improved ant algorithm can do a good route planning processing.

4. Searching the Optimal Escape Routes with Improved Ant Algorithm

This paper presents an improved ant algorithm which can quickly and effectively find the best route and several sub optimal routes from V_o (the trapped personnel position) to V_e (the safety point) in the tunnel network. There are two improvements to the ant algorithm: firstly, tunnel zoning characteristics are used to influence the ants' behaviors and improve the searching efficiency of ants; secondly, the strategy of meeting of ants and the ant death rules are modified to improve the ant algorithm's performance.

4.1. Calculation Steps of the Improved Ant Algorithm. Basic steps of the improved ant algorithm are as follows.

Step 1. Based on the above tunnel weights calculation model, weighted graph G is built. The trapped personnel position is denoted by V_o and the exit point (safety point) is denoted by V_e in graph G . The zoning attribute information is loaded.

Step 2. Initializes the ant algorithm's parameters including α , β , iter, C_{\max} , the pheromone trail matrix and the ant routes record table, and so forth; assigns the zoning number to each

node according to the zoning attribute information acquired; arranges the ants' position with the distribution requirements of Section 4.2.

Step 3. In the graph G , every ant is constructed an accessible node set in accordance with the moving rules in Section 4.2, and the tunnel zoning may influence the ants' accessible node set. The ant selects the next node by formula (14), and puts the selected node into its route record. After every live ant has selected a node, the system will judge whether the ant is dead or not (based on the rules described in Section 4.3). The procedure is repeated until all ants are dead and by this manner an iteration step is finished. The whole ant algorithm calculating procedure requires C_{\max} times iteration.

Step 4. Updates the pheromone trail matrix and the parameter α , β in accordance with the update rules in Section 4.4. Records the optimal route and L alternative routes in this iteration, and denotes it by OP_i^j (i is the number of current iteration; $j = 0, 1, 2, \dots, L$; $j = 0$ indicates the optimal route). According to the information of the current underground tunnels, weighted graph G' is reconstructed. The change rate of ΔG is calculated according to the formula (19). If ΔG is less than a small constant δ , it changes optimal routes of all executed iterations if these routes contain changed tunnels. If ΔG is greater than δ , a new ant algorithm is restarted (go to Step 1).

Step 5. Initializes ant routes recording table; fixes and controls pheromone trail matrix by the MMAS method; updates parameter iter as (iter + 1). If current iter's value is less than C_{\max} , go to Step 2.

Step 6. From the records of the iterations' routes OP_i^j ($i = 1, 2, \dots, C_{\max}$, $j = 0, 1, 2, \dots, L$), select the optimal route and L suboptimal routes for the current underground trapped workers.

4.2. Ants' Distribution and Moving Rules. The V_o and V_e nodes of the graph G are set m ants separately. For the ant k , its current node i 's accessible nodes set is built as

$$\text{allow}_i^k = A_i - B_k - (1 - p) C_k, \quad p = \{0, 1\}. \quad (13)$$

In (13), if the node i is in the backbone zone S_0 or expected zone S_e , p 's value is 1, else p 's value is 0. The set A_i is {nodes | the node which connected with node i }, and the set B_k is {nodes | the node that the ant k has passed}, and the set C_k is {nodes | the node which does not belong to zone S_0 and zone S_e }. Through the use of the set C_k , unnecessary detour can be avoided and the searching efficiency can be improved.

In accordance with the probability value calculated through (14), the ant k selects the next node in allow_i^k . Formula (14) is described as follows:

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha w_{ij}^\beta}{\sum_{s \in \text{allow}_i^k} \tau_{is}^\alpha w_{is}^\beta}, & j \in \text{allow}_i^k, \\ 0, & \text{else.} \end{cases} \quad (14)$$

In (14), α , β are factors of importance of the pheromone trail and the heuristic function respectively. And τ_{ij} , w_{ij} are pheromone trail concentration and weights respectively from the node i to the node j . The heuristic function is defined as weights between the current nodes.

4.3. The Rules of Meeting and Death of Ants. Every ant has attributes of a start node and an expected node. There are two improvements in meeting and death strategy showed in rule (2) and (4).

These rules are as follows.

- (1) If there is no way for the ant, the ant dies and is marked with no available route status.
- (2) When two ants meet, if the expecting point of one ant and the start point of the other ant intersect, the ant having short route gives its route to another, and then the latter dies and is marked with an available route status. The former moves on.
- (3) If the ant reaches the expected point, the ant dies and is marked with an available route status.
- (4) Sets a physical power parameter to the trapped workers group i and denotes it as PF_i which will reduce gradually. The PF_i parameters are assigned to ants. The ant will die when PF_i of the ant is reduced to zero no matter what status the ant is in and then it is marked with no available route status.

4.4. Update of Pheromone Trail and Main Parameters. When one iteration processing is completed, the pheromone matrix and the parameter α and β need to be updated. To prevent premature problem, the MMAS method [10] is used in the pheromone updating procedure. After an iteration processing, only the pheromone of this iteration optimal route is updated, according to (15) and (16). In addition, in order to improve the convergence of the later iteration and the early intelligence, factors α and β are dynamically changed at a fixed length between two iterations. At the later stage, α acts as the major factor; at the early stage, β acts as the major factor. See (17) and (18):

$$\tau_{ij}(t+1) = (1 - \rho) \tau_{ij}(t) + \Delta \tau_{ij}(t), \quad (15)$$

$$\Delta \tau_{ij} = \begin{cases} \frac{Q}{\sum w}, & \text{arc}(i, j) \text{ in optimal route,} \\ 0, & \text{arc}(i, j) \text{ not in optimal route,} \end{cases} \quad (16)$$

$$\alpha(t+1) = \alpha(t) + \Delta \alpha, \quad \Delta \alpha = \frac{(\alpha_{\max} - \alpha_{\min})}{c_{\max}}, \quad (17)$$

$$\beta(t+1) = \beta(t) - \Delta \beta, \quad \Delta \beta = \frac{(\beta_{\max} - \beta_{\min})}{c_{\max}}. \quad (18)$$

Formula (15) gives the pheromone relationship between the iteration $(t+1)$ and the iteration t , and ρ is the volatile factor of the pheromone, and $\Delta \tau_{ij}$ is the incremental pheromone.

Formula (16) is built based on MMAS strategy, only if the $\text{arc}(i, j)$ is in the optimal route of the iteration, the pheromone

TABLE 4: Parameters value of the ant algorithm.

Parameters	Value
α	1-2
β	2-1
ρ	0.3
Q (in formula (16))	1
C_{\max}	150
m (the number of ants)	16
L (the number of alternative routes)	2

of the $\text{arc}(i, j)$ is updated. $\sum w$ is the length of the current optimal route.

In (17), α_{\max} is the maximum of α , and α_{\min} is the minimum of α , c_{\max} is the total iteration number. The meaning of formula (18) is similar to the meaning of formula (17).

4.5. Condition of the Algorithm Stopping. When one iteration calculation is completed, according to current information, underground tunnel diagram G' is reconstructed, and the diagram's change rate can be represented as

$$\Delta G(t) = \frac{|G'(t) - G|}{G}. \quad (19)$$

In (19), $|G'(t) - G|$ indicates the changing number of arcs between G' of iteration t and the original G , and G indicates the total number of arcs of the original diagram.

When ΔG is less than the small value δ , weights of the related arcs in the optimal routes of executed iterations are updated. Under this situation, the optimal length of executed iterations changes, but the order of passed node does not change. When ΔG is more than δ , current calculation is stopped and a new algorithm calculation is restarted.

5. Implementation of the Algorithm and the Experimental Results

5.1. Parameter Setting. Different α , β , ρ parameter values have different influence to the stability and convergence of the ant algorithm. Based on literature [14] and relative experiments, to an underground tunnel network which has 210 tunnels and 147 nodes, these parameters' values are set in Table 4.

5.2. Experiment Data. The underground tunnel data come from a large sized coal mine in China. Partial data are showed in Table 5.

In Table 5, some simulated factors (such as slope, tunnel particle concentrations influence, tunnel crowded degree and special factors influence) are created based on original data. The network topology of the mine and its zoning result is shown in Figure 2.

The Matlab tool is used to realize the data processing and the ant algorithm. The node 83 is set as V_o and the node 1 is set as V_e . The iteration curve of the ant algorithm is shown in Figure 3.

TABLE 5: Partial data and weights of underground tunnel network.

Start point	End point	Tunnel type influenc k_{ti}	Wind speed (m/s)	Angle of slope ($^{\circ}$)	Special factors influence k_{di}	The lowest height of smoke layer (m)	Visibility distance (m)	Body interval (m)	Length (m)
31	30	∞	0	0	1	6	20	1	40
30	32	1	10.3	10	1	6	20	0.8	35
32	33	1	5.6	0	1	6	20	0.8	242
35	12	0.5	9.4	0	1	5	20	0.8	80
12	37	1	10.3	10	∞	5	20	0.8	281
...

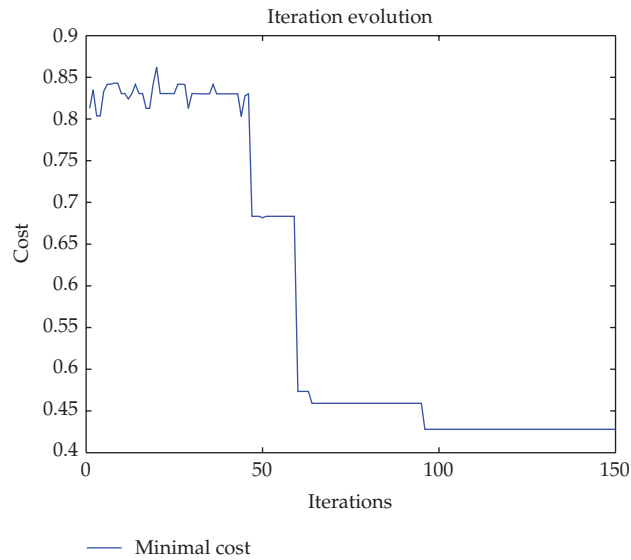


FIGURE 3: Iteration curve of the ant algorithm.

In Figure 3, the x -axis represents the number of iterations, and the ordinate represents the minimum cost of the current iteration. The cost is the sum of arc weights in the optimal route of the current iteration. From the chart we can see the algorithm has better convergence ability: after three convergences the global optimal route is found in 100 iterations about.

The algorithm procedure loads related parameters and experimental data firstly and then does calculation. Five calculation results are shown in Table 6. The experimental data is showed in Table 7.

In Table 6, what in small brackets is the sequence of passed nodes. To observe the calculated routes intuitively, the first running result is marked in Figure 2. The red line indicates the optimal route, and the green line is the first alternative route and the blue line is the second alternative route.

In fact, through the manual calculation we know the shortest cost between node 83 and node 1 is 0.42799 and the route is (83, 84, 82, 80, 77, 76, 74, 63, 57, 55, 45, 1). From

results in Table 6 we can see the best route is found steadily, meanwhile, two alternative routes are found but not stable.

6. Conclusion

An improved ant algorithm is proposed in this paper to solve the escape routes planning problem of underground coal mines. A simple unified tunnel weights calculation model is designed, which considers various impact factors. By the tunnel zoning method, searching efficiency of the ant algorithm has been improved effectively. The MMAS method is used to optimize the meeting strategy to improve the performance of the ant algorithm. In the algorithm's processing procedure, when small part of mine networks changes, the system may fix the optimal routes and avoid starting a new processing procedure. The algorithm was achieved in the Matlab platform and the best route was worked out steadily. The algorithm obtained good results when treating the data of a large coal mine of China. The proposed algorithm can be used in underground mine disaster rescue.

TABLE 6: Five running results of the ant algorithm.

NO.	Computing time (s)	The optimal route	The first alternative route	The second alternative route
1	3.935	(83, 84, 82, 80, 77, 76, 74, 63, 57, 55, 45, 1) cost: 0.42799	(83, 84, 82, 80, 77, 76, 75, 69, 68, 237, 74, 63, 57, 55, 45, 1) cost: 0.45920	(83, 84, 82, 96, 99, 77, 76, 74, 63, 57, 55, 45, 1) cost: 0.47337
2	3.443	(83, 84, 82, 80, 77, 76, 74, 63, 57, 55, 45, 1) cost: 0.42799	(83, 84, 82, 96, 99, 77, 76, 74, 63, 57, 55, 45, 1) cost: 0.47337	(83, 84, 82, 96, 99, 77, 76, 75, 69, 68, 237, 74, 63, 57, 55, 45, 1) cost: 0.49651
3	3.365	(83, 84, 82, 80, 77, 76, 74, 63, 57, 55, 45, 1) cost: 0.42799	(83, 84, 82, 80, 77, 76, 75, 69, 68, 237, 74, 63, 57, 55, 45, 1) cost: 0.45920	(83, 84, 82, 96, 99, 77, 76, 75, 69, 68, 237, 74, 63, 57, 55, 45, 1) cost: 0.49651
4	3.337	(83, 84, 82, 80, 77, 76, 74, 63, 57, 55, 45, 1) cost: 0.42799	(83, 84, 82, 80, 77, 76, 75, 69, 68, 237, 74, 63, 57, 55, 45, 1) cost: 0.45920	(83, 84, 82, 80, 77, 78, 72, 71, 70, 69, 68, 237, 74, 63, 57, 55, 45, 1) cost: 0.58666
5	3.935	(83, 84, 82, 80, 77, 76, 74, 63, 57, 55, 45, 1) cost: 0.42799	(83, 84, 82, 80, 77, 76, 75, 69, 68, 237, 74, 63, 57, 55, 45, 1) cost: 0.45920	(83, 84, 82, 80, 77, 76, 75, 69, 68, 237, 74, 63, 62, 61, 58, 57, 55, 45, 1) cost: 0.51041
Average	3.603	Average cost: 0.42799	Average cost: 0.462034	Average cost: 0.512692

TABLE 7: Experimental data.

Start point	End point	Tunnel type influence k_{ti}	Wind speed (m/s)	Angle of slope ($^{\circ}$)	Special factors influence k_{di}	The lowest height of smoke layer (m)	Visibility distance (m)	Body interval (m)	Length (m)
31	30	∞	0	0	1	6	20	1	40
30	32	1	10.3	10	1	6	20	0.8	35
32	33	1	5.6	0	1	6	20	0.8	242
35	12	0.5	9.4	0	1	5	20	0.8	80
12	37	1	10.3	10	∞	5	20	0.8	281
37	38	1	10.4	0	1	5	20	0.8	135
38	40	1	10.4	0	1	5	20	0.8	43
40	42	0.5	10.2	0	1	5	20	0.8	28
45	46	1	0	0	1	5	20	0.8	136.4
46	40	1	0.3	-20	1	5	20	1	70.6
46	38	1	0.3	-20	1	6	20	1	27.6
42	129	1	6.4	0	1	6	20	1	194
50	51	0.5	10.9	0	1	6	20	1	40
52	51	1	0	0	∞	6	20	1	56
51	53	1	10.9	0	1	6	20	1	120
50	52	1	0	0	1	6	20	1	40
1	45	0.5	3.3	10	1	6	20	1	670
45	55	0.5	2.5	0	1	6	20	1	67
56	37	0.5	0.6	0	1	3	20	1	40
56	55	0.5	4.9	0	1	3	20	1	131.5
55	57	0.5	0.1	0	1	3	20	1	50
57	58	0.5	10.9	0	1	3	20	1	100
59	58	0.5	0.6	0	1	3	20	1	100
58	60	0.5	4.9	15	1	3	20	1	580
61	59	0.5	0	0	1	3	20	1	45
36	61	0.5	10.5	0	1	3	20	1	517
61	62	1	10.6	0	1	6	20	1	28
62	63	1	0.2	0	1	6	20	1	40
63	57	1	10.4	0	1	6	20	1	85
65	59	1	0.6	0	1	6	20	1	26
126	65	1	0	0	1	6	20	1	20
66	65	1	0.4	-15	1	6	20	1	40
67	66	1	0.5	0	1	6	20	1	72
62	67	1	10	0	1	6	20	1	27
67	237	1	10.8	0	1	6	20	1	348
69	68	0.5	1.9	0	1	6	20	1	20
71	70	0.5	0	0	1	6	15	1	15
72	70	0.5	1.6	0	∞	6	20	0.7	30
74	63	0.5	11	0	1	3	20	0.7	375
69	75	0.5	0	0	1	3	20	0.7	20
76	74	0.5	10.4	20	1	3	20	0.7	50
75	76	0.5	10.4	0	1	3	20	0.7	22
77	76	0.5	0	20	1	3	20	0.7	40
78	77	0.5	2.8	0	1	3	20	0.7	25

TABLE 7: Continued.

Start point	End point	Tunnel type influence k_{ti}	Wind speed (m/s)	Angle of slope ($^{\circ}$)	Special factors influence k_{di}	The lowest height of smoke layer (m)	Visibility distance (m)	Body interval (m)	Length (m)
129	78	1	6.4	0	1	3	20	0.7	223
52	79	1	0	0	1	3	20	0.7	460
125	92	1	0	0	1	3	20	0.7	35
80	93	1	0	20	1	6	20	0.7	90
82	84	0.5	2.7	0	1	6	20	0.7	40
83	81	0.5	2.6	0	1	6	20	1.5	240
84	83	0.5	2.7	0	1	6	20	1.5	630
128	56	0.5	5.6	0	1	6	20	1.5	300
87	86	0.5	4.3	0	1	5	20	1.5	25
88	87	0.5	3	0	1	5	20	1.5	20
33	88	0.5	2.1	0	1	5	20	1.5	47
33	89	1	3.4	0	1	5	20	1.5	27
89	90	1	2.5	0	1	5	20	1.5	20
90	86	1	1.6	0	1	5	20	1.5	45
90	87	1	0	0	1	5	20	1.5	20
89	88	1	1.3	0	1	5	20	1.5	20
31	91	1	4	-20	1	5	20	1.5	286
81	93	1	2.6	0	1	6	20	1.5	640
93	94	1	2.7	0	1	6	20	1.5	35
53	80	1	10.9	0	1	6	20	1.5	90
80	82	1	10.9	0	1	6	20	1.5	204
77	99	1	4.7	0	1	6	20	1.5	254
80	77	1	2	0	1	6	20	1.5	40
82	96	0.5	9.1	0	1	6	20	1.5	61
96	97	1	8.5	15	1	6	20	1.5	40
97	98	1	0	0	1	6	20	1.5	196
99	98	1	3.5	0	1	6	20	1.2	80
96	99	1	0.6	15	1	3	20	1.2	40
99	110	1	1.9	0	∞	3	20	1.2	29.2
98	101	1	3.4	15	1	3	20	1.2	40
101	102	1	0.1	0	1	3	20	1.2	265.2
103	241	0.5	3.1	0	1	3	20	1.2	163.3
104	103	1	5.3	-20	1	3	20	1.2	21.2
105	104	0.5	3.7	0	1	3	20	1.2	110
106	105	0.5	10.8	0	1	3	20	1.2	40
106	107	1	0.1	0	1	3	20	1.2	75
98	107	1	0.1	-15	1	3	20	1.2	50
102	107	1	10.1	0	1	6	20	1.2	85
107	108	1	10.1	0	1	6	20	1.2	218.2
109	108	0.5	8.1	0	1	6	20	1.2	50
110	106	0.5	10.8	0	1	6	20	1.2	70
68	110	1	10.2	0	1	6	20	1.2	243
110	100	1	1.1	10	1	6	5	1.2	5
71	112	0.5	1.7	0	1	6	20	1.2	128

TABLE 7: Continued.

Start point	End point	Tunnel type influence k_{ti}	Wind speed (m/s)	Angle of slope ($^{\circ}$)	Special factors influence k_{di}	The lowest height of smoke layer (m)	Visibility distance (m)	Body interval (m)	Length (m)
72	71	0.5	2.4	0	1	6	14	1.2	14
112	68	0.5	0.8	0	1	6	11	1.2	11
237	113	1	4.3	0	1	5	20	0.8	65.1
113	141	1	2	10	1	5	20	0.8	248
1	131	1	0.1	-15	1	5	20	0.8	50
60	115	1	10.3	0	1	5	20	0.8	50
105	241	1	8.1	0	1	5	20	0.8	130
35	91	1	9.5	10	1	5	20	0.8	40
91	138	0.5	10.4	0	1	5	20	0.8	40
108	75	0.5	10.7	20	1	5	20	0.8	95
78	72	0.5	3	0	1	5	7.6	0.8	7.6
81	94	1	0	0	1	5	20	0.8	690
94	125	1	2.7	0	1	5	20	0.8	61.2
53	125	1	0	0	1	5	20	0.8	57
125	74	1	2.7	-20	1	5	20	0.8	530
127	126	0.5	0	0	1	5	20	0.8	40
66	127	0.5	0	20	1	5	15	0.8	15
86	128	0.5	5.9	0	1	6	20	0.8	24.7
50	129	1	0	0	1	6	20	0.8	40
70	69	1	1.8	20	1	6	20	0.8	65
131	60	1	0	0	1	6	20	0.8	20
1	115	0.5	10.4	-15	1	6	20	0.8	25
32	34	0.5	10	0	1	6	20	0.8	100
136	34	0.5	0	0	1	6	20	0.8	46.1
136	137	1	10.1	0	1	6	20	0.8	210
137	35	1	11	0	1	6	20	0.8	41.4
138	36	1	10.4	0	1	6	20	0.8	39
112	139	1	0	0	1	6	20	0.8	26
100	109	1	4.3	0	1	6	20	0.8	40
142	145	0.5	0	0	1	6	20	0.8	52.2
144	145	0.5	0	0	1	6	20	0.8	71
143	144	0.5	0	0	1	6	20	0.8	71
141	143	0.5	0.4	0	∞	6	20	0.8	71
148	140	1	1.4	0	1	6	20	0.8	52.2
147	148	1	0	-10	1	6	20	0.6	71
146	147	1	0	-10	1	5	20	0.6	71
139	146	1	0.4	0	1	5	20	0.6	71
143	146	1	0	0	1	5	20	0.6	25
144	147	1	0	0	1	5	20	0.6	25
145	148	1	0	0	1	5	20	0.6	25
149	150	0.5	4.4	0	1	5	20	0.6	265.2
141	151	0.5	2.2	20	1	5	20	0.6	150
152	153	0.5	2.3	20	1	5	20	0.6	165
153	154	0.5	2.3	0	1	5	20	0.6	160

TABLE 7: Continued.

Start point	End point	Tunnel type influence k_{ti}	Wind speed (m/s)	Angle of slope ($^{\circ}$)	Special factors influence k_{di}	The lowest height of smoke layer (m)	Visibility distance (m)	Body interval (m)	Length (m)
154	149	1	4.3	0	1	5	20	0.6	20
113	155	1	2.1	0	1	6	10	0.6	10
155	156	1	2.1	0	1	6	20	0.6	159.6
156	157	1	2.1	10	1	6	20	0.6	165.8
157	158	1	2	0	1	6	20	0.6	155.1
158	154	1	1.8	0	1	6	20	0.6	25.5
157	153	1	0	0	1	6	20	0.6	26.9
156	152	1	0	0	1	6	20	0.6	26.6
155	151	1	0	0	1	6	20	0.6	27.9
140	109	1	3.7	0	1	6	20	1	25.7
142	140	1	2.6	0	1	6	20	1	25.2
97	159	1	8.6	-10	1	6	20	1	240
159	160	1	4.1	-10	1	6	20	1	44
159	162	1	4.7	0	1	6	20	1	47
162	163	1	2.6	0	1	6	14.4	1	14.4
163	164	1	0	0	1	3	19	1	19
162	164	1	1.7	0	1	3	20	1	33.4
163	165	1	2.2	0	1	3	20	1	61.8
164	165	1	2.4	0	∞	3	20	1	51.2
165	104	1	4.6	20	1	3	20	1	92.3
104	166	1	3	20	1	3	20	1	120
166	167	1	2.7	0	1	3	20	1	157.9
167	168	1	2.2	0	1	3	20	1	225.9
168	169	1	2.2	0	1	3	20	1	201.4
169	170	0.5	2.2	0	1	3	20	1	212.5
170	171	0.5	2.2	0	1	3	20	1	202.8
171	172	0.5	2.2	10	1	3	20	1	175.5
172	173	0.5	2.2	10	1	3	20	1	197.5
173	174	0.5	2.2	0	1	3	20	1	216.8
174	175	1	2.2	0	1	3	20	1	203.9
175	176	1	2.4	0	1	3	20	1	51.9
176	177	1	0	0	∞	3	20	1	24.2
177	178	1	0	0	1	6	20	1	23.5
179	178	1	2.4	0	1	6	20	1	24.3
179	176	1	0	0	1	6	20	1	23.5
176	180	1	1.4	20	1	6	20	1	266.2
180	181	1	1.4	0	1	6	20	1	1285.6
182	100	1	3	0	1	6	20	1	9.2
183	182	1	1.9	0	1	6	20	1	24.9
184	183	1	2.6	0	1	6	20	1	151.2
185	184	1	1.3	0	1	6	20	1	203.5
181	185	1	1.4	0	1	6	20	1	207.6
186	142	1	3.1	0	1	6	20	1	132.7
187	186	1	4.4	0	∞	6	20	1	203.6

TABLE 7: Continued.

Start point	End point	Tunnel type influence k_{ti}	Wind speed (m/s)	Angle of slope ($^{\circ}$)	Special factors influence k_{di}	The lowest height of smoke layer (m)	Visibility distance (m)	Body interval (m)	Length (m)
188	187	1	4.4	0	1	6	20	1	209
150	188	1	4.4	0	1	6	20	1	112.9
181	188	1	0	-15	1	6	20	1	29.2
187	185	1	0	0	1	6	20	1	30.3
186	184	1	1.8	0	1	6	20	1	30.3
101	160	1	1.2	0	1	6	20	1	240
103	190	1	2.1	10	1	6	20	1	25
167	190	1	0	10	1	6	20	1	29.3
190	191	0.5	2.7	0	1	5	20	1	215.6
191	168	0.5	0	0	1	5	20	1	35.8
191	192	0.5	2.6	0	1	5	20	1	211.5
169	192	0.5	0	0	1	5	20	1	29.2
192	193	1	2.7	0	∞	5	20	1	216.4
170	193	1	0	0	1	5	20	1	27.1
193	194	1	2.6	0	1	5	20	1	200
171	194	1	0	0	1	5	20	1	28.4
194	195	1	2.7	0	1	5	20	1	174.2
195	172	1	0	0	1	5	20	1	29.1
195	196	1	2.7	0	1	6	20	1	195
196	173	1	0	0	1	6	20	1	30.6
196	197	1	2.7	0	1	6	20	1	219.3
174	197	1	0	0	1	6	20	1	29.2
197	198	1	2.7	-10	1	6	20	1	197.3
198	175	1	0	-10	1	6	20	1	33.1
198	179	1	2.4	0	1	6	20	1.4	74.6
237	68	0.5	8.5	0	1	6	20	1.4	72
183	238	0.5	0	0	1	6	20	1.4	77
166	239	0.5	0.4	0	∞	6	20	1.4	101.5
239	238	0.5	0	0	1	6	20	1.4	86.1
239	240	0.5	0	0	1	6	20	1.4	112.9
238	240	1	0	0	1	6	20	1.4	25.1
240	182	1	0.8	0	1	6	20	1.4	77
237	74	1	0	20	1	6	20	1.4	40
241	102	1	0.1	0	1	6	20	1.4	40

References

- [1] L. Yun, "Current safety situation analysis in china," *China's Development Observer*, vol. 1, no. 3, pp. 33–37, 2005.
- [2] State Administration of Work Safety, "The national production safety supervision and administration bureau dispatching statistics 2006–2008," pp. 40–41, 2009.
- [3] L. Shuling, "The confirmation of the best route for evading accident under the pit," *Journal of LiaoNing Technical University*, vol. 18, no. 1, pp. 27–29, 1999.
- [4] S. Jia, S. Diange, and J. Zhong'an, "The improvement of Dijkstra algorithm used for choosing the best escape route in mine's emergency response," *China Mining Magazine*, vol. 14, no. 6, pp. 46–48, 2005.
- [5] H. Yunbing, X. Xing, Y. Xu et al., "Improved shortest path algorithm based on mine geographic network model," *Coal Science and Technology*, vol. 39, no. 2, pp. 103–105, 2011.
- [6] Y. Wanqing, D. heng, and H. Jiejun, "Mine safety relief system based on the improved Dijkstra algorithm," *Mining Safety & Environmental Protection*, vol. 35, no. 3, pp. 40–44, 2008.
- [7] S. E. Jalali and M. Noroozi, "Determination of the optimal escape routes of underground mine networks in emergency cases," *Safety Science*, vol. 47, no. 8, pp. 1077–1082, 2009.
- [8] M. Dorigo, V. Maniezzo, and A. Coloni, "Ant system: optimization by a colony of cooperating agents," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 26, no. 1, pp. 29–41, 1996.
- [9] D. Jiping, *Mining*, Chinese Mining University Press, 2009.
- [10] W. Deming, "Selecting the optimum escape routes by a computer during a mine fire," *Journal of China University of Mining & Technology*, vol. 23, no. 3, pp. 27–32, 1994.
- [11] Y. Liao, *Fluid Mechanics*, China Railway, Beijing, China, 1987.
- [12] Z. Yunming, "Study on safe evacuation with performance-based analytical method in public gathering place," *Journal of Safety Science and Technology*, vol. 7, no. 10, pp. 70–74, 2011.
- [13] P. Thompson and E. Marehant, "A computer model for the evacuation of large building populations," *Fire Safety Journal*, vol. 24, no. 2, pp. 131–148, 1995.
- [14] T. Stützle and H. H. Hoos, "Max–Min Ant system," *Future Generation Computer Systems*, vol. 16, no. 8, pp. 889–914, 2000.

Research Article

A Constructive Data Classification Version of the Particle Swarm Optimization Algorithm

Alexandre Szabo and Leandro Nunes de Castro

Natural Computing Laboratory, Mackenzie University, Rua da Consolação 930, 01302-907 São Paulo, Brazil

Correspondence should be addressed to Leandro Nunes de Castro; lnunes@mackenzie.br

Received 2 October 2012; Revised 1 November 2012; Accepted 8 November 2012

Academic Editor: Baozhen Yao

Copyright © 2013 A. Szabo and L. N. de Castro. 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 particle swarm optimization algorithm was originally introduced to solve continuous parameter optimization problems. It was soon modified to solve other types of optimization tasks and also to be applied to data analysis. In the latter case, however, there are few works in the literature that deal with the problem of dynamically building the architecture of the system. This paper introduces new particle swarm algorithms specifically designed to solve classification problems. The first proposal, named Particle Swarm Classifier (PSCl), is a derivation of a particle swarm clustering algorithm and its architecture, as in most classifiers, is pre-defined. The second proposal, named Constructive Particle Swarm Classifier (cPSCl), uses ideas from the immune system to automatically build the swarm. A sensitivity analysis of the growing procedure of cPSCl and an investigation into a proposed pruning procedure for this algorithm are performed. The proposals were applied to a wide range of databases from the literature and the results show that they are competitive in relation to other approaches, with the advantage of having a dynamically constructed architecture.

1. Introduction

Data classification is one of the most important tasks in data mining, which is applied to databases in order to label and describe characteristics of new input objects whose labels are unknown. Generally, this task requires a classifier model, obtained from samples of objects from the database, whose labels are known *a priori*. The process of building such a model is called *learning* or *training* and refers to the adjustment of parameters of the algorithm so as to maximize its generalization ability.

The prediction for unlabeled data is usually done using models generated in the training step or some lazy learning mechanism able to compare objects classified *a priori* with new objects yet to be classified. Thus, not all algorithms generate an explicit classification model (classifier), as is the case of k -NN (k -nearest neighbors) [1, 2], and Naïve Bayes [3], which use historical data in a comparative process of distance or probability estimation, respectively, to classify new objects. Examples of algorithms that generate an explicit classification model are the decision trees [4, 5], artificial neural networks

[6, 7], and learning vector quantization (LVQ) algorithms [8–11].

This paper proposes two algorithms for data classification—Particle Swarm Classifier (PSCl) and its successor Constructive Particle Swarm Classifier (cPSCl), both based on the particle swarm optimization algorithm (PSO) [12]. These algorithms were developed from adaptations of the PSO and other bioinspired techniques, and were evaluated in seven databases from the literature. Their performance was compared with that of other swarm algorithms and also with some well-known methods from the literature, such as k -NN, naïve Bayes, and an MLP neural network.

In the PSCl algorithm, two steps are necessary to construct the classifier. In the first step, a number of prototypes are positioned, in an unsupervised way, on regions of the input space with some density of data; for this, the Particle Swarm Clustering (PSC) [13] algorithm is used. In the next step, the prototypes are adjusted by an LVQ1 method [14] in order to minimize the percentage of misclassification. Thus, the PSCl automatically positions the prototypes in the respective classes of objects, defining the decision boundaries

between classes and increasing the efficiency of the algorithm during the construction of the classifier.

The cPSClazz, in turn, improved its unsupervised step by inserting mechanisms inspired by the immune system to automatically build the classifier model, more specifically, to automatically set the number of cells (prototypes) in the swarm. At this step, the PSC has been replaced by the Constructive Particle Swarm Clustering (cPSC) [15], which uses the clonal selection principle [16] to iteratively control the number of particles in the swarm. Furthermore, a pruning phase was introduced so as to avoid the explosion in the number of particles. Thus, cPSClazz eliminates the need for the user to define the swarm size a priori, a critical parameter required for many data classification algorithms.

The paper is organized as follows. As the cPSClazz algorithm borrows ideas from swarm intelligence and the immune system, Section 2 is dedicated to a brief review of the biological concepts necessary for a proper understanding of the algorithm. Section 3 presents the PSC and cPSC algorithms, which form the basis for designing PSClazz and cPSClazz. In Section 4, two classifiers are proposed based on the PSO—PSClazz and cPSClazz. Section 5 provides a literature review emphasizing algorithms based on the PSO to solve classification problems and PSO versions with dynamic population. The PSClazz and cPSClazz algorithms are evaluated in Section 6 and a parametric sensitivity analysis for cPSClazz was performed to evaluate the growth of the swarm. The paper is concluded in Section 7, with discussions and proposals for future works.

2. Biological Background

Biological and behavioral mechanisms are some of the natural inspirations that motivate scientists and engineers to construct intelligent computational tools. One of the pioneer computational bioinspired tools was proposed by McCulloch and Pitts [17], with their logic model of the neuron. After that, several other lines of research have emerged, with new bioinspirations, such as evolutionary computing with the Darwinian laws of evolution [18–20], swarm intelligence [12, 21–23] inspired by the emergent behavior of social agents (most often insects), and artificial immune systems, inspired by the vertebrate immune system [24, 25]. These four approaches constitute one of the three major areas of natural computing [26]. The following sections briefly review swarm intelligence concepts and their main lines of research, as well as an overview of the vertebrate immune system and its main defense mechanisms.

2.1. Swarm Intelligence. Collective systems (swarms) are composed of agents that interact with each other and with the environment in order to achieve a common objective. These agents can be formed by a flock of birds, a school of fish, or a colony of insects, which are able to learn from their own experience and with the social influence and adapt to changes in the environment where they live [27]. These agents individually have limited cognitive abilities that allow them to perform only simple tasks. However, when put together and

interacting with one another, they can perform substantially complex tasks [28]. The emergent behavior of this social interaction is called swarm intelligence.

This terminology was first used in the work of Beni and Wang [21], which described the behavior of a group of robots that interacted with each other in a particular environment, respecting a set of local rules inspired by the behavior of ants. According to Kennedy et al. [29], any collective behavior, like a flock of birds or an immune system, can be named a swarm.

There are basically two approaches in Swarm Intelligence: the works based on the behavior of social insects [30] and those based on the human ability to process knowledge [31]. In both lines of research, there is a group of agents that interact with one another and with the environment.

Among the swarm intelligence algorithms, a great deal of attention has been given to the PSO algorithm, introduced by Kennedy and Eberhart [12]. PSO was inspired by the social behavior of flocks of birds or schools of fish to solve complex optimization problems. It uses a population of solutions, called particles, which interact with one another exchanging experience in order to optimize its ability within a certain environment. The main bioinspiration in the PSO is that one behaves according to its own past experience and that of other interacting agents. Since its introduction, PSO has also been improved and adapted to be applied to various tasks [32].

2.2. Immune System. Living organisms have cells and molecules that protect their body against the onslaught of disease-causing agents (*pathogens*). Specific cells and their mechanisms, such as identification, signaling, reproduction, and attack, are parts of a complex system called the immune system [24], responsible for keeping diseases at bay. The vertebrate immune system is divided mainly into *innate immune system* and *adaptive immune system*.

The innate immune system does not evolve, remaining the same throughout the life time of the individual. It recognizes many infectious diseases and is responsible for combating infectious agents while the adaptive immune system is preparing to act. The innate immune system, by itself, cannot remove most pathogens [24]. The main role of the innate immune system is to signal other immune system cells, since most pathogens do not directly stimulate the cells of the adaptive immune system.

The adaptive immune system is also called specific immune system, because some pathogens are recognized only by cells and molecules of the adaptive immune system. One of its main features is the ability to learn from infections and develop an immune memory, in other words, to recognize an antigen when it is presented recursively to the body. Thus, the adaptability of the adaptive immune system renders it more capable of recognizing and combating specific antigens each time it tries to reinfect the body.

The main components of the innate immune system are the macrophages and granulocytes, while in the adaptive immune system these are the lymphocytes. Both systems depend on the activity of white blood cells (leukocytes).

The organs that make up the immune system are called lymphoid organs consisting of the following two subsystems:

- (i) primary lymphoid organs are responsible for the production, growth, development, and maturation of lymphocytes.
- (ii) secondary lymphoid organs are sites in which lymphocytes recognize and fight pathogens.

Some lymphocytes are generated, developed, and matured in the bone marrow. These lymphocytes are called B cells. However, some lymphocytes generated in the bone marrow migrate to an organ called *thymus*, where they are matured and come to be called the T cells. The B cells and T cells are the lymphocytes of the immune system.

The main mechanisms of recognition and activation of the immune system are described by the *clonal selection principle* or the *clonal selection theory* [16], which explains how the adaptive immune system responds to pathogens. Only cells that recognize antigens are selected for reproduction, whilst the remainder die after some time due to a lack of stimulation.

The immune cells subjected to high concentrations of antigens (pathogens) are selected as candidates for reproduction. If the affinity between this cell receptor (called *antibody*) and an antigen of greater affinity to it exceeds a certain threshold, the cell is cloned. The immune recognition, thus, triggers the proliferation of antibodies as one of the main mechanisms of immune response to an antigenic attack, a process called clonal expansion. During clonal expansion B cells are subjected to an affinity proportional mutation process, resulting in variations in the repertoire of immune cells and, thus, antibodies, which are B-cell receptors free in solution.

The clonal selection theory is considered the core of the immune response system, since it describes the dynamics of the adaptive immune system when stimulated by disease-causing agents. Therefore, it is used in the design of adaptive problem solving systems [24] and was used in the design of the cPSC algorithm to be proposed in this paper.

3. Clustering Using Particle Swarm

The idea of using the PSO algorithm to solve clustering problems was initially proposed in [33], so that each particle corresponds to a vector containing the centroid of each group of the database. Since then, several clustering algorithms based on the PSO have been proposed [13, 15, 31, 33–56]. Among them, the PSC and cPSC algorithms, form the basis for the PSCl and cPSCl classification algorithms, respectively, proposed in this paper. In this section, the precursors of PSCl and cPSCl are reviewed.

3.1. Particle Swarm Clustering (PSC). The PSC, proposed in [13], is an adaptation of the PSO to solve data clustering problems. In the PSC, particles interact with one another and with the environment (database) so that they become representatives of a natural group from the database. The convergence criterion of the algorithm is determined by the stabilization of the path of the particles, and the number of particles in the swarm is initialized empirically. The dimension of a particle is given by the dimension of the input objects, where each vector position is an attribute of the object.

The main structural differences between the PSC and PSO algorithms are as follows.

- (i) In PSC, the particles altogether compose a solution to the data clustering problem.
- (ii) The PSC does not use an explicit cost function to evaluate the quality of the particles. Instead, the Euclidean distance is used as a measure to assess the dissimilarity between a particle and an object, and particles move around the space in order to represent statistical regularities of the input data.
- (iii) A self-organizing term, which moves the particle towards the input object, was added to the velocity equation.
- (iv) In the PSO algorithm, all the particles in the swarm are updated at each iteration. In the PSC, the particles to be updated are defined by input objects (i.e., only the *winner*—the one closest to the input datum—is updated according to (1) and (2)).

For each input object, there is a particle of greater similarity to it, obtained by the Euclidean distance between the particle and the object. This particle is called *winner*, and is updated following the proposed velocity equation (1) as

$$v_i(t+1) = \omega * v_i(t) + \varphi_1 \otimes (p_i^j(t) - x_i(t)) + \varphi_2 \otimes (g^j(t) - x_i(t)) + \varphi_3 \otimes (y^j - x_i(t)). \quad (1)$$

In (1), the parameter ω , called *inertia moment*, is responsible for controlling the convergence of the algorithm. The cognitive term, $\varphi_1 \otimes (p_i^j(t) - x_i(t))$, associated with the experience of the particle, represents the best particle's position $p_i^j(t)$, in relation to the input object so far, that is, the smallest distance between the input object (y^j) and the *winner* particle (x_i). The social term, $\varphi_2 \otimes (g^j(t) - x_i(t))$, is associated with the particle $g^j(t)$ closest to the input object, that is, the particle that had the smallest distance in relation to the input object (y^j) so far. The self-organizing term, $\varphi_3 \otimes (y^j - x_i(t))$, moves the particle towards the input object.

$$x_i(t+1) = x_i(t) + v_i(t+1). \quad (2)$$

Thus, the particles converge to the centroid of the groups, or regions of higher density of objects, becoming prototypes representatives of groups from the database.

The pseudocode of the PSC algorithm is described in Pseudocode 1.

Step 13 of Pseudocode 1 assigns a label (**PLABELS**) to each input object, which is given by a label (**CLABELS**) that represents the dominant class of objects for which each particle was the *winner*. Generally, in real world problems the correct labels are not known *a priori*. So, each label (**CLABELS**) must be given by each particle in the swarm.

Step 26 of Pseudocode 1 updates all those particles that did not move at iteration t . Thus, after all objects were presented to the swarm, the algorithm verifies whether some particle x_i did not *win* in that iteration ($x_i! = \text{win}$). If yes, then

```

1. procedure [X, V, P, g,  $\omega$ , PLABELS] = PSC (Y,  $v_{\max}$ , m,  $\omega$ , CLABELS)
2. Y//dataset
3. initialize X//initialize at random each particle  $x_i \in [0,1]$ 
4. initialize  $v_i$ //initialize at random each  $v_i \in [-v_{\max}, v_{\max}]$ 
5. initialize dist
6.  $t = 1$ 
7. while stopping criterion is not met
8.   for  $j = 1$  to  $n$ //for each input datum
9.     for  $i = 1$  to  $m$ //for each particle
10.      dist ( $i$ ) = distance ( $x_i, y^j$ )
11.    end for
12.     $I = \text{index}(\min(\mathbf{dist}))$ 
13.    [PLABELS] = label ( $x_I$ , CLABELS ( $y^j$ ))//predicted label
14.    if distance ( $x_I, y^j$ ) < distance ( $p_I^j, y^j$ )
15.       $p_I^j = x_I$ 
16.    end if
17.    if distance ( $x_I, y^j$ ) < distance ( $g^j, y^j$ )
18.       $g^j = x_I$ 
19.    end if
20.     $v_I(t+1) = \omega * v_I(t) + \varphi_1 \otimes (p_I^j(t) - x_I(t)) + \varphi_2 \otimes (g^j(t) - x_I(t)) + \varphi_3 \otimes (y^j - x_I(t))$ 
21.     $v_I \in [-v_{\max}, v_{\max}]$ 
22.     $x_I(t+1) = x_I(t) + v_I(t+1)$ 
23.     $x_I \in [0, 1]$ 
24.  end for
25.  for  $i = 1$  to  $m$ 
26.    if ( $x_i \neq \text{win}$ )//particles did not win
27.       $v_i(t+1) = \omega * v_i(t) + \varphi_4 \otimes (x_{\text{most\_win}} - x_i(t))$ 
28.       $v_i \in [-v_{\max}, v_{\max}]$ 
29.       $x_i(t+1) = x_i(t) + v_i(t+1)$ 
30.    end if
31.  end for
32.   $t = t + 1$ 
33.   $\omega = 0.95 * \omega$ 
34.  Test the stopping criterion
35. end while
36. end procedure

```

PSEUDOCODE 1: Particle Swarm Clustering (PSC).

these particles are updated in relation to the particle that was elected the winner more often at iteration t . Such particle is called $x_{\text{most_win}}$ (step 27 in Pseudocode 1), where φ_4 is a random vector within the interval (0,1).

3.2. Constructive Particle Swarm Clustering (cPSC). To dynamically determine the number of particles in PSC, Prior and de Castro [15] proposed a successor, called cPSC, which eliminated the need for the user to input the number of particles (prototypes) in the swarm. cPSC automatically finds a suitable number of prototypes in databases by employing strategies borrowed from the PSC algorithm with the addition of three new features inspired by the antibody network named ABNET [57]: growth of the particle swarm, pruning of particles, and automatic stopping criterion. Furthermore, the cPSC algorithm uses an affinity threshold (ε) as a criterion to control the growth of the swarm. The growth, pruning and stopping functions are described below and are evaluated at every two iterations.

3.2.1. Swarm Growth. The growth stage is based on the immune cell reproduction mechanism during clonal selection and expansion [16, 24], as described previously. The cells that are subjected to high concentrations of antigens are chosen as candidates to reproduce. If the affinity between the antibodies of these cells in relation to the antigens of higher affinities to them is greater than a threshold ε , then these cells are cloned.

These principles of selection and reproduction of antibodies inspired the design of the constructive particle swarm clustering algorithm. In the cPSC, particles are analogs to immune cells and objects from the database to antigens. The algorithm starts with only one particle (immune cell), with position and velocity initially random. At every two iterations the algorithm evaluates the necessity of growing the swarm as follows: the particle that was elected the winner more times (cell submitted to the highest concentration of antigen) is selected. The algorithm evaluates the degree of affinity between the particle and the object (antigen) of higher affinity to it, using threshold ε . If the affinity between them is greater

```

1. Procedure [X, V, P, g,  $\omega$ , PLABELS] = cPSC (Y,  $m$ ,  $\omega$ ,  $v_{\max}$ ,  $\varepsilon$ , CLABELS)
2. Y//dataset
3. initialize X//initialize at random only one particle  $x_i \in [0, 1]$ 
4. initialize V//initialize at random,  $v_i \in [-v_{\max}, v_{\max}]$ 
5. initialize dist
6.  $t = 1$ 
7. while stopping criterion is not met
8.   for  $j = 1$  to  $n$ //for each input datum
9.     for  $i = 1$  to  $m$ //for each particle
10.      dist ( $i$ ) = distance ( $x_i, y^j$ )
11.    end for
12.     $I = \text{index}(\min(\mathbf{dist}))$ 
13.    [PLABELS] = label ( $x_I, \text{CLABELS}(y^j)$ )//predicted label
14.    if distance ( $x_I, y^j$ ) < distance ( $p_I^j, y^j$ )
15.       $p_I^j = x_I$ 
16.    end if
17.    if distance ( $x_I, y^j$ ) < distance ( $g^j, y^j$ )
18.       $g^j = x_I$ 
19.    end if
20.     $v_I(t+1) = \omega * v_I(t) + \varphi_1 \otimes (p_I^j(t) - x_I(t)) + \varphi_2 \otimes (g^j(t) - x_I(t)) + \varphi_3 \otimes (y^j - x_I(t))$ 
21.     $v_I \in [-v_{\max}, v_{\max}]$ 
22.     $x_I(t+1) = x_I(t) + v_I(t+1)$ 
23.     $x_I \in [0, 1]$ 
24.  end for
25.  if mod( $t, 2$ )==0
26.    Eliminate particles from the swarm if necessary
27.    Test the stopping criterion
28.    Clone particles if necessary
29.  end if
30.   $t = t + 1$ 
31.   $\omega = 0.95 * \omega$ 
32. end while
33. end procedure

```

PSEUDOCODE 2: Constructive particle swarm clustering.

than ε , then a new particle is created in the swarm. This new particle is positioned in the middle between the winner and the object of higher affinity to it.

3.2.2. Pruning of Particles. At every two iterations, the algorithm evaluates the need for pruning particles. If a particle has not moved at all in two iterations, then it is deleted from the swarm. A new step, called *suppression*, was added right after the pruning of particles step. If the particles are close to one another (Euclidean distance < 0.3), they are eliminated. It is a metaphor based on the immune system: cells and molecules recognize each other even in the absence of antigens. If the cell recognizes an antigen (*positive response*), then a clonal immune response is started; otherwise, (*negative response*) a *suppression*, which refers to the death of cells recognized as self, happens.

3.2.3. Stopping Criterion. At every two iterations the algorithm evaluates the stopping criterion by the average Euclidean distance between the current position and the position of the memory particles. Thus, the algorithm stops when this distance is less than or equal to 10^{-3} or 200 iterations.

The pseudocode of the cPSC algorithm is described as in Pseudocode 2.

4. Classification Using Particle Swarm

This paper proposes two data classification algorithms based on particle swarms: (1) PSClass, that uses the LVQ1 heuristics to adjust the position of prototypes generated by a clustering swarm-based algorithm; and (2) cPSClass, an improved version of PSClass that uses ideas from the immune system to dynamically determine the number of particles in the swarm.

The training process of PSClass consists of the iterative adjustment of the position of particles (prototypes). After training, there is a predictor model formed by a set of prototypes able to describe and predict the class to which a new input object from the database must belong. The testing step assesses the generalization capability of the classifier. At this stage, a number of test objects are presented to the classifier and their classes are predicted.

Two methods are combined to generate the predictor: the PSC algorithm, and an LVQ1 model. The LVQ1 heuristics was adopted for its simplicity and allows, through simple

```

1. procedure[X, PLABELS] = PSClass (Y,  $v_{\max}$ ,  $m$ ,  $\omega$ , CLABELS)
2. Y//dataset
3. CLABELS//correct_labels
4. [X, V, P, g,  $\omega$ , PLABELS] = PSC (Y,  $v_{\max}$ ,  $m$ ,  $\omega$ , CLABELS)
5. while stopping criterion is not met
6.   for  $j = 1$  to  $n$ //for each object
7.     for  $i = 1$  to  $m$ //for each particle
8.       dist ( $i$ ) = distance ( $x_i$ ,  $y^j$ )
9.     end for
10.     $I = \text{index}(\min(\mathbf{dist}))$ 
11.    if distance ( $x_I$ ,  $y^j$ ) < distance ( $p_I^j$ ,  $y^j$ )
12.       $p_I^j = x_I$ 
13.    end if
14.    if distance ( $x_I$ ,  $y^j$ ) < distance ( $g^j$ ,  $y^j$ )
15.       $g^j = x_I$ 
16.    end if
17.     $v_I(t+1) = \omega * v_I(t) + \varphi_1 \otimes (p_I^j(t) - x_I(t)) + \varphi_2 \otimes (g^j(t) - x_I(t)) + \varphi_3 \otimes (y^j - x_I(t))$ 
18.     $v_I \in [-v_{\max}, v_{\max}]$ 
19.    if (PLABELS ( $I$ ) == CLABELS ( $I$ ))
20.       $x_I(t+1) = x_I(t) + v_I(t+1)$ 
21.    else
22.       $x_I(t+1) = x_I(t) - v_I(t+1)$ 
23.    end if
24.  end for
25.   $\omega = 0.95 * \omega$ 
26.  Test the stopping criterion
27. end while
28. end procedure

```

PSEUDOCODE 3: Particle Swarm Classifier (PSCClass).

procedures of position adjustment, the correction of misplaced prototypes in the data space [10].

Within PSCClass, the PSC algorithm is run to find groups of objects in the database by placing the particles (prototypes) on the natural groups of the database. The number of particles must be informed by the user and this number should be at least equal to the number of the existing classes in the database. The algorithm places the prototypes in the input objects space in order to map each object in a representative prototype of each class. Thus, the algorithm generates a decision boundary between classes based on the prototypes that represent the classes. Then, the LVQ1 heuristics is used to adjust the position of the prototypes so as to minimize the classification error.

In its classification version, the PSC algorithm was modified such that the number of iterations for convergence is not determined by the user. Instead, its stopping criterion is defined by the stabilization of the prototypes around the input objects.

Two steps are required to generate the PSCClass classifier.

- (i) *Unsupervised Step*. In this stage, the PSC algorithm is run in order to position the particles in regions of the input space that are representative of the natural clusters of data.
- (ii) *Supervised Step*. Some steps are added to the PSC algorithm such that the prototypes are adjusted by

the LVQ1 method so as to minimize the classification error, as shown in (3) and (4).

For each object j in the database, there is a prototype i with greater similarity to it, determined by a nearest neighbor method. This prototype is updated considering the PSC equations combined with the LVQ1 method as

$$x_i(t+1) = x_i(t) + v_i(t+1), \quad (3)$$

if the prototype x_i and the object y^j belong to the same class; and

$$x_i(t+1) = x_i(t) - v_i(t+1), \quad (4)$$

if the prototype x_i and the object y^j do not belong to the same class.

Thus, when a particle labels correctly an object from the database, the particle is moved toward this object (3); otherwise, it is moved in the opposite direction to the object (4).

The following pseudocode presents the supervised step of the PSCClass classifier (see Pseudocode 3).

As in most data classification algorithms, the user must define the architecture of the system, for example, number of particles in the swarm or neurons in the neural network; some changes in the PSCClass classifier were proposed so that it could dynamically determine the number of prototypes in the swarm, and, thus, the automatic construction of a classifier model, giving rise to cPSCClass. The cPSCClass algorithm was

inspired by the work of Prior and de Castro [15], with the proposal of the cPSC, discussed in Section 3.2.

Like its predecessor, PSClass, two steps are necessary to generate the cPSCClass classifier.

- (i) *Unsupervised Step*. In this stage, the cPSC algorithm is run in order to position the particles in regions of the input space that are representative of the natural clusters of data and also to determine a suitable number of particles in the swarm (Pseudocode 2).
- (ii) *Supervised Step*. Some steps are added to the PSC algorithm such that the prototypes are adjusted by the LVQ1 method so as to minimize the classification error, as shown in (3) and (4).

5. Related Works

As the contributions of this paper emphasize a constructive particle swarm classification algorithm, the related works to be reviewed here include the use of the PSO algorithm for data classification and PSO techniques with dynamic population.

5.1. PSO for Data Classification. There are several works in the literature involving data classification with PSO, such as [34–41]. These will be briefly reviewed in this section.

In [34], two approaches to the binary PSO are applied to classification problems: one called Pittsburgh PSO (PPSO) and the other called Michigan PSO (MPSO). In the Pittsburgh approach, each particle represents one or more prediction rules. Thus, a single particle is used to solve a classification problem. The classification is done based on the nearest neighbors rule (NN). In the Michigan PSO, by contrast, each particle represents a single prototype. Thus, all particles are used to solve a classification problem. A refinement of the MPSO is presented in [35] with the adaptive Michigan PSO (AMPSO), where the population is dynamic and the whole swarm is used to solve the problem. The fitness of each particle is used to calculate its growth probability.

In [36], the authors proposed an extension of the binary PSO, called DPSO, to discover classification rules. Each attribute may or may not be included in the resulting rule. An improvement of DPSO was proposed in [37], culminating in the hybrid algorithm PSO/ACO. The proposal starts with an empty set of rules, and for each class from the database it returns the best rule for the class evaluated.

Hybrid algorithms are common, as is the case of PSORS [38], which is used to cluster and classify objects. It combines the PSO, rough sets (RS), and a modified form of the Huang Index function to optimize the number of groups found in the database. The number of groups for each attribute of the particle is limited by a range defined by the Huang index, which is applied to the database. Attributes are fuzzified by the fuzzy *c*-means [42], and the index function is applied to each object to determine the group to which it belongs.

In [39] it was proposed a hybrid algorithm, named hybrid particle swarm optimization tabu search (HPSOTS), for selecting genes for tumor classification. The HPSOTS combines PSO with tabu search (TS) to maintain population

diversity and prevent deceptive local optimum. The algorithm initializes a population of individuals represented by binary strings. Ninety percent of the neighbors of an individual are assessed according to mechanisms from the modified discrete PSO [43, 44]. The algorithm selects a new individual of the neighborhood according to the tabu conditions and updates the population.

According to Wang et al. [40], many classification techniques, such as decision trees [4, 5] and artificial neural networks [6, 7], do not produce acceptable predictive accuracy when applied to real problems. In this sense, the authors proposed the use of the PSO algorithm [12] to discover classification rules. Their method initializes a population of individuals (rules) with the dimension given by the number of attributes of the objects evaluated. A fitness function is defined to evaluate the solution (classification rules) for the problem in question. The smaller the rule set, the easier to understand it.

The works presented in [40, 45] also cite the quality of the results produced when using classical techniques to solve real world problems. For Wang et al. [40], decision trees, artificial neural networks, and naive Bayes are some of the classic techniques applied to classification problems and they work well in linearly separable problems. The authors proposed a classification method based on a multiple linear regression model (MRLM) and the PSO algorithm, called the PSO-MRLM, which is able to learn the relationship between objects from a database and also express it mathematically. The MRLM technique builds a mathematical model able to represent the relationship between variables, associating the value of each independent variable with the value of the dependent variable. The set of equations (rules) contemplate this relationship using coefficients that act on each of the attributes of each rule. The proposed method uses the PSO algorithm to estimate the value of the coefficients.

A classifier based on PSO is proposed in [41] and applied to power systems operations. Pattern recognition based on PSO (PSO-PR) evaluates a condition of operation and predicts whether this is safe or unsafe. The first step to obtain the classifier is to generate the patterns (data) necessary to the training process. As the number of variables describing the power system state is very large, the next step in this process involves a feature selection procedure, responsible for eliminating redundant and irrelevant variables. In the next step PSO is used to minimize the percentage of misclassification.

Compared with the proposed PSClass and cPSCClass, none of the works available in the literature work by using a clustering algorithm followed by a vector quantization approach. The proposals here initially operate in a completely self-organized manner, and only after the particles are positioned in regions of the space that represent the input data, their positions are corrected so as to minimize the classification error. Despite these differences, in the present paper the performances of PPSO, MPSO, and AMPSO algorithms are compared with that of PSClass and cPSCClass.

5.2. PSO with Dynamic Population. The original PSO has also been modified to dynamically determine the number of particles in the swarm. According to [46], there are few

publications dealing with the issue of dynamic population size in PSO, and the main ones are briefly reviewed below.

In [47] two dynamic population approaches are proposed to improve the PSO speed: expanding population PSO (EP-PSO) and diminishing population PSO (DP-PSO). According to the experiments shown, both approaches reduce the run time of PSO by 60% on average.

In [48], it was proposed the dynamic population PSO (DPPSO), where the number of particles is controlled by a function that contains an attenuation item (responsible for reducing the number of particles) and a waving item (particles are generated to avoid local optimum and the ones considered inefficient die and are removed from the swarm) to control the population size.

In [46] it was proposed the dynamic multiobjective particle swarm optimization (DMOPSO) algorithm, which uses the particle growth strategy inspired by the algorithm incrementing multiobjective evolutionary (IMOEA) [49] in which particles of best fitness are selected to generate new particles.

In [50], the authors proposed the ladder particle swarm optimization (LPSO) algorithm, where the population size is determined based on its diversity.

In [51], two approaches to multiobjective optimization with dynamic population are proposed: dynamic multiobjective particle swarm optimization (DPSMO) and dynamic particle swarm evolutionary algorithms (DPSEA), which combine PSO and genetic algorithm mechanisms to regulate the number of particles in the swarm.

All PSO versions with dynamic population present growth strategies for the number of particles to ensure the diversity of solutions and pruning strategies to reduce the processing cost. Their applications, however, are focused on optimization problems, not on data classification problems, as proposed in the present paper. Therefore, no direct performance comparisons will be made with these methods.

6. Performance Assessment

The PSClass and cPSClass algorithms were implemented in MATLAB 7.0, and their performances were compared with those three algorithms based on the PSO: PPSO, MPSO and AMPSO, and also with the three well-known classification algorithms from the literature: naïve Bayes, k -NN and a multi-layer perceptron (MLP) neural network trained with the backpropagation algorithm [11]. The parametric configurations for the MLP were as follows: *learning rate* equals to 0.3, maximum number of *epochs* equals to 500, and number of nodes in the hidden layers equals to 4.0. The number of hidden layers was given by (number of attributes + number of classes)/2. The classic algorithms were run using the Weka 3.6 [58] tool, and the results of the PSO-based algorithms were taken from the literature. A k -fold cross-validation procedure was used to train the algorithms and estimate the prediction error, and the algorithms were run 30 times for a validation with $k = 10$ folders each. The objects of each class were distributed evenly and with stratification among the 10 folders. For *benchmarking* we

TABLE 1: Main characteristics of the databases used for performance assessment.

Databases	Objects	Attributes	Classes
Iris	150	4	3
Yeast	205	20	4
Wine	178	13	3
Glass identification	214	9	6
Haberman's survival	306	3	2
Ruspini	75	2	4
<i>E.coli</i>	336	8	5

used seven databases available in the UCI Data Repository (<http://archive.ics.uci.edu/ml/datasets.html>). The main features of these databases are listed in Table 1.

The parametric configurations used in the two algorithms proposed here have been inherited from their predecessor, the PSC. The vectors φ_1 , φ_2 , and φ_3 are random in the interval (0, 1). The inertia moment (ω) has an initial value of 0.90, with a decay of 0.95 iteratively to 0.01. The number of particles used in PSCClass was twice the number of classes present in the respective database in Table 1. The domain of the vector space was limited to [0, 1] and the velocity of the particles was also controlled and was set in the range [-0.1, 0.1] to avoid particle explosion.

In its original version, the PSC stops after a fixed number of iterations. For the construction of the PSCClass, the PSC was modified such that the number of iterations required for their convergence is not defined by the user. To do that, a stopping criterion had to be proposed: the stabilization of the swarm; that is, if the average Euclidean distance between the current position and the position of the memory prototypes is less than a given threshold, then the algorithm is assumed to have converged. This stopping criterion is assessed every two iterations.

In cPSCClass, the pruning of particles occurs when they do not move during two consecutive iterations. When the similarity between the prototype and the object of greater similarity to it is greater than the affinity threshold, the prototype is cloned and the new prototype is positioned in the middle between it and the object evaluated. A suppression step was added right after the pruning step so as to minimize the number of prototypes generated.

The threshold ε depends on the dataset, for which it must be defined empirically. A number of particles added to the swarm much different from the number of classes in the database compromise the effectiveness of the algorithm. To understand the influence of ε in the cPSCClass algorithm, a sensitivity analysis of ε was performed using the databases from Table 1. The following values for ε were tested: 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, and 0.50. The results in terms of accuracy (percentage of correct classification—Pcc), number of particles (NP), and number of iterations (NI) for each value of ε tested are shown in Table 2. The results presented are the average over 30 runs, and the best results of Pcc, NP and NI on average were made in bold.

Increasing the value of ε is the same as increasing the affinity degree between the winner particle and the input

TABLE 2: Sensitivity analysis of the cPSClass algorithm in relation to ε for the seven datasets to be evaluated. Percentage of correct classification (Pcc), number of prototypes (NP), and number of iterations (NI) for convergence.

ε	0.15	0.20	0.25	0.30	0.35	0.40	0.45	0.50
Iris								
Pcc	88.89 \pm 3.20	88.67 \pm 3.11	89.78 \pm 3.38	87.78 \pm 4.98	87.56 \pm 4.87	87.78 \pm 6.57	88.22 \pm 5.16	86.67 \pm 7.43
NP	3.0 \pm 0.0	3.07 \pm 0.25	3.03 \pm 0.18	3.0 \pm 0.0	2.97 \pm 0.18	2.97 \pm 0.18	3.0 \pm 0.26	2.93 \pm 0.37
NI	214.63 \pm 5.16	214.57 \pm 5.36	212.10 \pm 4.49	221.27 \pm 34.06	215.50 \pm 5.20	215.47 \pm 5.61	218.63 \pm 25.75	215.10 \pm 6.61
Yeast								
Pcc	95.09 \pm 1.34	99.65 \pm 1.34	99.65 \pm 1.34	88.22 \pm 5.16	100.0 \pm 0.0	99.82 \pm 0.96	100.0 \pm 0.0	100.0 \pm 0.0
NP	3.08 \pm 0.25	3.93 \pm 0.25	3.93 \pm 0.25	3.0 \pm 0.26	4.0 \pm 0.0	3.97 \pm 0.18	4.0 \pm 0.0	4.0 \pm 0.0
NI	281.23 \pm 52.78	242.40 \pm 20.41	236.57 \pm 9.37	219.33 \pm 34.39	247.63 \pm 14.70	241.60 \pm 20.45	236.67 \pm 8.90	239.67 \pm 10.59
Wine								
Pcc	43.75 \pm 0.0	57.50 \pm 21.49	93.75 \pm 0.0	94.17 \pm 1.59	94.58 \pm 2.16	95.0 \pm 2.54	93.96 \pm 1.14	93.96 \pm 1.14
NP	1.0 \pm 0.0	1.63 \pm 1.0	5.93 \pm 0.74	6.90 \pm 0.76	6.90 \pm 0.80	7.23 \pm 0.86	7.10 \pm 0.66	7.27 \pm 0.74
NI	377.87 \pm 40.95	391.80 \pm 30.63	382.83 \pm 39.19	343.80 \pm 61.27	332.0 \pm 58.82	335.87 \pm 62.33	330.23 \pm 56.90	323.50 \pm 61.14
Glass Identification								
Pcc	52.81 \pm 8.90	51.40 \pm 9.44	52.46 \pm 7.88	51.05 \pm 7.46	50.70 \pm 7.63	51.05 \pm 8.97	52.63 \pm 8.85	50.18 \pm 9.14
NP	6.90 \pm 0.40	6.90 \pm 0.48	6.97 \pm 0.32	6.93 \pm 0.45	6.93 \pm 0.52	6.87 \pm 0.35	6.83 \pm 0.38	6.90 \pm 0.55
NI	321.97 \pm 82.96	324.47 \pm 82.58	308.93 \pm 77.06	298.60 \pm 86.12	321.0 \pm 87.56	286.27 \pm 79.80	304.63 \pm 83.32	276.23 \pm 78.83
Haberman's Survival								
Pcc	91.11 \pm 3.31	92.0 \pm 3.57	92.44 \pm 3.15	92.0 \pm 2.41	90.33 \pm 3.43	91.33 \pm 2.98	91.44 \pm 2.72	92.22 \pm 3.07
NP	8.13 \pm 0.82	8.10 \pm 0.61	8.13 \pm 0.63	8.0 \pm 0.83	8.13 \pm 0.78	8.13 \pm 0.73	7.97 \pm 0.76	8.17 \pm 0.91
NI	222.87 \pm 6.14	222.10 \pm 4.06	222.27 \pm 5.11	221.80 \pm 5.25	221.30 \pm 4.91	222.10 \pm 8.79	220.80 \pm 7.46	221.40 \pm 6.67
Ruspini								
Pcc	100.0 \pm 0.0	100.0 \pm 0.0	100.0 \pm 0.0	100.0 \pm 0.0	100.0 \pm 0.0	100.0 \pm 0.0	100.0 \pm 0.0	100.0 \pm 0.0
NP	4.0 \pm 0.0	4.0 \pm 0.0	4.0 \pm 0.0	4.0 \pm 0.0	4.0 \pm 0.0	4.0 \pm 0.0	4.0 \pm 0.0	4.0 \pm 0.0
NI	132.87 \pm 65.99	151.53 \pm 62.29	160.73 \pm 54.09	157.90 \pm 53.33	132.97 \pm 62.06	120.10 \pm 67.77	159.77 \pm 56.46	156.50 \pm 58.0
E. coli								
Pcc	82.04 \pm 9.50	84.73 \pm 6.45	82.26 \pm 7.52	84.30 \pm 7.56	81.83 \pm 10.11	83.66 \pm 8.96	80.97 \pm 10.42	82.69 \pm 9.26
NP	5.43 \pm 1.04	5.73 \pm 0.78	5.30 \pm 0.95	5.53 \pm 1.01	5.43 \pm 0.82	5.60 \pm 0.89	5.73 \pm 0.83	5.37 \pm 0.93
NI	231.07 \pm 28.27	224.17 \pm 10.91	236.10 \pm 36.74	232.30 \pm 36.84	232.40 \pm 25.94	223.60 \pm 16.40	232.87 \pm 35.04	231.97 \pm 41.43

TABLE 3: Accuracy (Pcc) and number of prototypes (NP) of cPSClazz and ScPSClazz.

Database	cPSClazz		ScPSClazz	
	Pcc	NP	Pcc	NP
Iris	95.56 \pm 4.74	11.87 \pm 3.25	89.78 \pm 3.38	3.03 \pm 0.18
Yeast	100.0 \pm 0.0	23.90 \pm 5.19	100.0 \pm 0.0	4.0 \pm 0.0
Wine	93.54 \pm 2.59	15.87 \pm 3.43	95.0 \pm 2.54	7.23 \pm 0.86
Glass	60.35 \pm 7.66	22.60 \pm 5.06	52.63 \pm 8.85	6.83 \pm 0.38
Haberman	93.67 \pm 3.08	22.10 \pm 3.79	92.44 \pm 3.15	8.13 \pm 0.63
Ruspini	100.0 \pm 0.00	7.97 \pm 1.13	100.0 \pm 0.0	4.0 \pm 0.0
<i>E.coli</i>	89.68 \pm 2.86	17.97 \pm 3.58	84.73 \pm 6.45	5.73 \pm 0.78

TABLE 4: Number of prototypes for PSClazz, cPSClazz, PPSO, MPSO, and AMPSO algorithms.

Database	PSClazz	cPSClazz	PPSO	MPSO	AMPSO
Iris	6	3.03 \pm 0.18	10	16	18
Yeast	8	4.0 \pm 0.0	—	—	—
Wine	6	7.23 \pm 0.86	—	—	—
Glass	12	6.83 \pm 0.38	10	16	19
Haberman	4	8.13 \pm 0.63	—	—	—
Ruspini	8	4.0 \pm 0.0	—	—	—
<i>E.coli</i>	10	5.73 \pm 0.78	—	—	—

TABLE 5: Classification accuracy for PSClazz, cPSClazz, PPSO, MPSO, AMPSO, naïve Bayes, K-NN, and MLP (ANN).

Database	PSClazz	cPSClazz	PPSO	MPSO	AMPSO	Naïve Bayes	k-NN	ANN
Iris	91.78 \pm 5.16	89.78 \pm 3.38	90.89 \pm 0.0	96.70 \pm 0.0	96.89 \pm 0.0	96.00 \pm 0.0	96.00 \pm 0.0	97.36 \pm 0.0
Yeast	100.0 \pm 0.0	100.0 \pm 0.0	—	—	—	97.56 \pm 0.0	98.05 \pm 0.0	96.59 \pm 0.0
Wine	93.96 \pm 1.14	95.0 \pm 2.54	—	—	—	96.63 \pm 0.0	97.75 \pm 0.0	97.19 \pm 0.0
Glass	59.82 \pm 6.84	52.63 \pm 8.85	74.34 \pm 0.0	86.27 \pm 0.0	86.94 \pm 0.0	49.53 \pm 0.0	71.63 \pm 0.0	68.37 \pm 0.0
Haberman	86.11 \pm 4.88	92.44 \pm 3.15	—	—	—	74.51 \pm 0.0	69.28 \pm 0.0	74.18 \pm 0.0
Ruspini	99.44 \pm 3.04	100.0 \pm 0.0	—	—	—	98.67 \pm 0.0	100.0 \pm 0.0	100.0 \pm 0.0
<i>E.coli</i>	79.78 \pm 10.44	84.73 \pm 6.45	—	—	—	87.46 \pm 0.0	84.71 \pm 0.0	86.85 \pm 0.0

object. Thus, when the threshold increases, the number of clones tends to increase, which can compromise the effectiveness of the algorithm. As the accuracy of the algorithm is, in principle, its most important measure of interest, the value of ϵ detached in bold in Table 2 as the most suitable one for each dataset was that with higher Pcc. It can be observed, though, that the algorithm presents some robustness in relation to ϵ , in the sense that its performance in terms of Pcc, NP, and NI changes little even with a high variation in ϵ .

To evaluate the performance of the suppression step in cPSClazz, its performance was compared with that without using it. The results are shown in Table 3 and the cPSClazz with the suppression step is identified by ScPSClazz to differentiate it from the standard cPSClazz. It can be observed that the accuracy of cPSClazz with the suppression step was worse for the Iris, Glass and *E. coli* datasets, but the number of prototypes generated was substantially smaller. By contrast, the performance of ScPSClazz was equivalent or increased even with a substantial reduction in the number of particles in the swarm.

The parametric configurations of the PPSO, MPSO and AMPSO algorithms are available in [34, 35]. The number

of prototypes of such algorithms, as well as of PSClazz and cPSClazz, is shown in Table 4. For the MLP network, the number of output neurons used was equal to the number of classes in the database (Table 1), as well as the value of k for the k -NN.

Table 5 shows the performance of PPSO, MPSO, AMPSO, PSClazz, cPSClazz algorithms and the classic algorithms from the literature when applied to the databases in Table 1. The best absolute results, on average, are made in bold in the table. The PSClazz and cPSClazz algorithms showed similar performances, on average, for the databases of Table 1. For the Yeast and Ruspini databases, the cPSClazz algorithm presented maximal accuracy, whilst no other algorithm was capable of presenting such performance. It is worth noting that the number of particles generated by the cPSClazz is greater than that used in the PSClazz for the Wine and Haberman databases, as can be observed in Table 4. Naïve Bayes, k -NN, and MLP presented a worse performance than PSClazz and cPSClazz for the Haberman databases but were competitive for the other databases. PPSO, MPSO, and AMPSO performed quite well for the Glass database, being competitive with PSClazz and cPSClazz.

A Shapiro-Wilk test [59] was used to determine whether the behavior presented by the algorithms had a normal distribution. Assuming a confidence level equals to 0.95, the test of normality revealed that the null hypothesis (H_0) should be rejected and, thus, a nonparametric test should be used to assess the statistical significance of performances. In order to determine whether the difference in performance among the evaluated algorithms is significant, we used the Friedman test [60, 61], a nonparametric method analogous to the parametric ANOVA (analysis of variance) [62]. The Friedman test is based on the ranking of the results obtained for each sample (database) j to all k algorithms. The value of the degrees of freedom is obtained by $k - 1$, being $k = 8$, so there are 7 degrees of freedom. Thus, according to the χ^2 table [63], the critical values of probability, for $\alpha = 5\%$ and $\alpha = 1\%$, are 14.07 and 18.48, respectively. As χ^2 calculated is less than the critical values, the null hypothesis H_0 is not rejected. In other words, the difference in performance between the algorithms is not statistically significant for the Haberman databases tested.

7. Conclusion

This paper presented two algorithms based on the original particle swarm optimization algorithm—PSClazz and cPSClazz—to solve data classification problems. The PSClazz initially finds natural groups within the database, in an unsupervised way, and then adjusts the prototypes' position using an LVQ1 method, in a supervised way, in order to minimize the misclassification error. The cPSClazz algorithm is similar to PSClazz, except in its unsupervised phase, where it dynamically determines the number of particles in the swarm using the immune clonal selection metaphor. A parametric sensitivity analysis for cPSClazz was also performed to evaluate the relation between the growth of the swarm and ϵ . For cPSClazz, a suppression step was added right after the pruning step to reduce the number of prototypes generated by the algorithm.

The algorithms were applied to seven data classification problems and their performance was compared with that of algorithms well known in the literature— k -NN, MLP, and Naïve Bayes, in addition to three algorithms based on the original PSO-PPSO, MPSO, and AMPSO. It was used a k -fold cross-validation to train the algorithms and estimate the prediction error. The algorithms were run 30 times for $k = 10$ folders. The results showed that cPSClazz was the best algorithm, on average, for the Haberman database, whilst AMPSO was the best for Glass, naïve Bayes was best for *E. coli*, k -NN was the best for Wine, and the MLP was the best for Iris. The PSClazz and cPSClazz algorithms showed similar results with each other for all the databases evaluated. However, the cPSClazz has the advantage of automatically determining the number of prototypes (particles) in the swarm.

Acknowledgments

The authors thank CNPq, Fapesp, Capes, and MackPesquisa for the financial support.

References

- [1] G. E. A. P. A. Batista and M. C. Monard, "An analysis of four missing data treatment methods for supervised learning," *Applied Artificial Intelligence*, vol. 17, no. 5-6, pp. 519–533, 2003.
- [2] Y. Yang and X. Liu, "A re-examination of text categorization methods," in *Proceedings of the 22nd ACM International Conference on Research and Development in Information Retrieval*, pp. 42–49, Berkeley, Calif, USA, 1999.
- [3] X. Yi and Y. Zhang, "Privacy-preserving naive Bayes classification on distributed data via semi-trusted mixers," *Information Systems*, vol. 34, no. 3, pp. 371–380, 2009.
- [4] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufman Press, 2000.
- [5] C. Westphal and T. Blaxton, *Data Mining Solutions: Methods and Tools for Solving Real-World Problems*, Wiley Computer Publishing, New York, NY, USA, 1998.
- [6] L. M. Silva, J. Marques de Sá, and L. A. Alexandre, "Data classification with multilayer perceptrons using a generalized error function," *Neural Networks*, vol. 21, no. 9, pp. 1302–1310, 2008.
- [7] B. Gabrys and A. Bargiela, "General fuzzy min-max neural network for clustering and classification," *IEEE Transactions on Neural Networks*, vol. 11, no. 3, pp. 769–783, 2000.
- [8] T. Kohonen, *Self-Organizing Maps*, vol. 30, Springer, Berlin, Germany, 2nd edition, 1997.
- [9] T. Kohonen, *Improved Versions of Learning Vector Quantization*, Helsinki University of Technology, Helsinki, Finland, 1990.
- [10] G. R. Lloyd, R. G. Brereton, R. Faria, and J. C. Duncan, "Learning vector quantization for multiclass classification: application to characterization of plastics," *Journal of Chemical Information and Modeling*, vol. 47, no. 4, pp. 1553–1563, 2007.
- [11] R. K. Madyastha and B. Aazhang, "Algorithm for training multilayer perceptrons for data classification and function interpolation," *IEEE Transactions on Circuits and Systems I*, vol. 41, no. 12, pp. 866–875, 1994.
- [12] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks (Perth, Australia)*, pp. 1942–1948, Piscataway, NJ, USA, December 1995.
- [13] S. C. M. Cohen and L. N. de Castro, "Data clustering with particle swarms," in *Proceedings of the World Congress on Computational Intelligence*, pp. 6256–6262, Vancouver, Canada, 2006.
- [14] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.
- [15] A. K. F. Prior and L. N. de Castro, "Um Algoritmo de Enxame Construtivo Para Agrupamento de Dados," in *Proceedings of the 18th Congresso Brasileiro de Automação*, pp. 3300–3307, Bonito, Brazil, 2010.
- [16] F. M. Burnet, *The Clonal Selection of Acquired Immunity*, Cambridge University Press, London, UK, 1959.
- [17] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133, 1943.
- [18] T. Bäck, D.B. Fogel, and Z. Michalewicz, *Evolutionary Computation: Basic Algorithms and Operators*, Institute of Physics Publishing, Bristol, UK, 2000.
- [19] T. Bäck, "Evolutionary algorithms: comparison of approaches," in *Computing with Biological Metaphors*, R. Paton, Ed., pp. 227–243, Chapman & Hall, 1994.

- [20] D. B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*, IEEE Press, Piscataway, NJ, USA, 2nd edition, 2000.
- [21] G. Beni and J. Wang, "Swarm intelligence," in *Proceedings of the 7th Annual Meeting of the Robotics Society of Japan*, pp. 425–428, 1989.
- [22] E. Bonabeau, G. Theraulaz, J. L. Deneubourg, S. Aron, and S. Camazine, "Self-organization in social insects," *Trends in Ecology and Evolution*, vol. 12, no. 5, pp. 188–193, 1997.
- [23] N. R. Franks, "Army ants: a collective intelligence," *American Scientist*, vol. 77, no. 2, pp. 138–145, 1989.
- [24] L. N. de Castro and J. Timmis, *Artificial Immune Systems: A New Computational Intelligence Approach*, Springer, 2002.
- [25] E. Hart and J. Timmis, "Application areas of AIS: the past, the present and the future," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 191–201, 2008.
- [26] L. N. de Castro, "Fundamentals of natural computing: an overview," *Physics of Life Reviews*, vol. 4, no. 1, pp. 1–36, 2007.
- [27] E. O. Wilson, *Sociobiology: The New Synthesis*, Belknap Press, Cambridge, Mass, USA, 1975.
- [28] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, New York, NY, USA, 1999.
- [29] J. Kennedy, R. Eberhart, and Y. Shi, *Swarm Intelligence*, Morgan Kaufman Publishers, 2001.
- [30] B. J. Zhao, "An ant colony clustering algorithm," in *Proceedings of the 6th International Conference on Machine Learning and Cybernetics (ICMLC '07)*, pp. 3933–3938, Hong Kong, August 2007.
- [31] A. Szabo, A. K. F. Prior, and L. N. De Castro, "The proposal of a velocity memoryless clustering swarm," in *6th IEEE World Congress on Computational Intelligence (WCCI '10)*, Barcelona, Spain, July 2010.
- [32] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization: an overview," *Swarm Intelligence*, vol. 1, pp. 33–57, 2007.
- [33] D. W. van der Merwe and A. P. Engelbrecht, "Data clustering using particle swarm optimization," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC '03)*, pp. 215–220, Canberra, Australia, 2003.
- [34] A. Cervantes, I. Galván, and P. Isasi, "A Comparison between the Pittsburgh and Michigan Approaches for the Binary PSO Algorithm," in *Proceedings of IEEE Congress on Evolutionary Computation (CEC '05)*, pp. 290–305, September 2005.
- [35] A. Cervantes, I. M. Galván, and P. Isasi, "AMPSO: a new particle swarm method for nearest neighborhood classification," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 39, no. 5, pp. 1082–1091, 2009.
- [36] T. Sousa, A. Silva, and A. Neves, "Particle Swarm based Data Mining Algorithms for classification tasks," *Parallel Computing*, vol. 30, no. 5-6, pp. 767–783, 2004.
- [37] N. Holden and A. A. Freitas, "A hybrid particle swarm/ant colony algorithm for the classification of hierarchical biological data," in *Proceedings of IEEE Swarm Intelligence Symposium (SIS '05)*, pp. 100–107, Pasadena, Calif, USA, June 2005.
- [38] K. Y. Huang, "A hybrid particle swarm optimization approach for clustering and classification of datasets," *Knowledge-Based Systems*, vol. 24, no. 3, pp. 420–426, 2011.
- [39] Q. Shen, W. M. Shi, and W. Kong, "Hybrid particle swarm optimization and tabu search approach for selecting genes for tumor classification using gene expression data," *Computational Biology and Chemistry*, vol. 32, no. 1, pp. 52–59, 2008.
- [40] Z. Wang, X. Sun, and D. Zhang, "A PSO-based classification rule mining algorithm," in *Advanced Intelligent Computing Theories and Applications*, vol. 4682, pp. 377–384, Springer, Berlin, Germany, 2007.
- [41] S. Kalyani and K. S. Swarup, "Classifier design for static security assessment using particle swarm optimization," *Applied Soft Computing Journal*, vol. 11, no. 1, pp. 658–666, 2011.
- [42] 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.
- [43] Q. Shen, J. H. Jiang, C. X. Jiao, G. L. Shen, and R. Q. Yu, "Modified particle swarm optimization algorithm for variable selection in MLR and PLS modeling: QSAR studies of antagonism of angiotensin II antagonists," *European Journal of Pharmaceutical Sciences*, vol. 22, no. 2-3, pp. 145–152, 2004.
- [44] Q. Shen, J. H. Jiang, C. X. Jiao, S. Y. Huan, G. L. Shen, and R. Q. Yu, "Optimized partition of minimum spanning tree for piecewise modeling by particle swarm algorithm. QSAR studies of antagonism of angiotensin II antagonists," *Journal of Chemical Information and Computer Sciences*, vol. 44, no. 6, pp. 2027–2031, 2004.
- [45] S. C. Satapathy, J. V. R. Murthy, P. V. G. D. Prasad Reddy, B. B. Misra, P. K. Dash, and G. Panda, "Particle swarm optimized multiple regression linear model for data classification," *Applied Soft Computing Journal*, vol. 9, no. 2, pp. 470–476, 2009.
- [46] W. F. Leong and G. G. Yen, "PSO-based multiobjective optimization with dynamic population size and adaptive local archives," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 38, no. 5, pp. 1270–1293, 2008.
- [47] B. Soudan and M. Saad, "An evolutionary dynamic population size PSO implementation," in *Proceedings of the 3rd International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA '08)*, pp. 1–5, April 2008.
- [48] S. Sun, G. Ye, Y. Liang, Y. Liu, and Q. Pan, "Dynamic population size based particle swarm optimization," *Lecture Notes in Computer Science, Advances in Computation and Intelligence*, vol. 4683, pp. 382–392, 2007.
- [49] K. C. Tan, T. H. Lee, and E. F. Khor, "Evolutionary algorithms with dynamic population size and local exploration for multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 6, pp. 565–588, 2001.
- [50] D. Chen and C. Zhao, "Particle swarm optimization with adaptive population size and its application," *Applied Soft Computing Journal*, vol. 9, no. 1, pp. 39–48, 2009.
- [51] G. G. Yen and H. Lu, "Dynamic population strategy assisted particle swarm optimization," in *Proceedings of IEEE International Symposium on Intelligent Control*, pp. 697–702, October 2003.
- [52] X. Cui and T. E. Potok, "Document clustering analysis based on hybrid PSO+K-means algorithm," *The Journal of Computer Science*, Special issue on Efficient heuristics for information organization, Science Publications, New York, NY, USA, 2005.
- [53] J. Dong and M. Qi, "A new algorithm for clustering based on particle swarm optimization and K-means," in *Proceedings of the International Conference on Artificial Intelligence and Computational Intelligence (AICI '09)*, vol. 4, pp. 264–268, November 2009.
- [54] F. T. Niknam and M. Nayeripour, "A new evolutionary algorithm for cluster analysis," *World Academy of Science, Engineering and Technology*, vol. 46, pp. 584–588, 2008.

- [55] S. Alam, G. Dobbie, and P. Riddle, "An evolutionary particle swarm optimization algorithm for data clustering," in *Proceedings of IEEE Swarm Intelligence Symposium (SIS '08)*, pp. 1–6, September 2008.
- [56] M. G. H. Omran, A. P. Engelbrecht, and A. Salman, "Dynamic clustering using particle swarm optimization with application in unsupervised image classification," *Transactions on Engineering, Computing and Technology*, vol. 9, pp. 199–204, 2005.
- [57] L. N. de Castro, F. J. Von Zuben, and G. A. de Deus, "The construction of a Boolean competitive neural network using ideas from immunology," *Neurocomputing*, vol. 50, pp. 51–85, 2003.
- [58] I. H. Witten and E. Frank, *Data Mining—Practical Machine Learning Tools and Techniques*, Morgan Kaufman, San Francisco, Calif, USA, 2005.
- [59] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality: complete samples," *Biometrika*, vol. 52, pp. 591–611, 1965.
- [60] 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, pp. 675–701, 1937.
- [61] 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.
- [62] R. A. Fisher, *Statistical Methods and Scientific Inference*, Hafner Publishing Company, New York, NY, USA, 2nd edition, 1959.
- [63] F. R. Pisani and R. Purves, *Statistics*, W.W. Norton & Company, New York, NY, USA, 3rd edition, 1998.

Research Article

Particle Swarm Optimization Algorithm for Unrelated Parallel Machine Scheduling with Release Dates

Yang-Kuei Lin

Department of Industrial Engineering and Systems Management, Feng Chia University, P.O. Box 25-097, Taichung 40724, Taiwan

Correspondence should be addressed to Yang-Kuei Lin; yklin@mail.fcu.edu.tw

Received 6 September 2012; Revised 11 December 2012; Accepted 25 December 2012

Academic Editor: Baozhen Yao

Copyright © 2013 Yang-Kuei Lin. 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.

We consider the NP-hard problem of minimizing makespan for n jobs on m unrelated parallel machines with release dates in this research. A heuristic and a very effective particle swarm optimization (PSO) algorithm have been proposed to tackle the problem. Two lower bounds have been proposed to serve as a basis for comparison for large problem instances. Computational results show that the proposed PSO is very accurate and that it outperforms the existing metaheuristic.

1. Introduction

This research considers the problem of scheduling n jobs on m unrelated parallel machines in the presence of release dates. The performance measure, makespan, is defined as $\max(C_1, \dots, C_n)$, where C_j is the completion time of job j . Minimizing makespan not only completes all jobs as quickly as possible but also is a surrogate for maximizing the utilization of machines. Following the three-field notation of Graham et al. [1], we refer to this problem as $R|r_j|C_{\max}$. This problem is NP hard [2].

Chen and Vestjens [3] used the largest processing time (LPT) to minimize makespan for identical parallel machines with release dates ($P|r_j|C_{\max}$). The release date of a job is not known in advance, and its processing time becomes known at its arrival. Kellerer [4] proposed algorithms for the $P|r_j|C_{\max}$ problem and $P|r_j|C_{\min}$ problem. Koulamas and Kyparis [5] considered uniform parallel machine scheduling problems ($Q|r_j|C_{\max}$). They proposed a heuristic and derived a tight worse-case ratio bound for this heuristic. Centeno and Armacost [6] showed that the LPT rule performed better than the least flexible job (LFJ) rule for the problem with machine eligibility restrictions ($P|r_j, M_j|C_{\max}$). Lancia [7] applied a branch-and-bound (b & b) procedure to solve scheduling problems with release dates and tails on two unrelated parallel machines ($R_2|r_j, q_j|C_{\max}$). Similarly, Gharbi and Haouari [8]

also presented a b & b procedure to solve the $P|r_j, q_j|C_{\max}$ problem. Carlier and Pinson [9] reported new results on the structures of Jackson's pseudopreemptive scheduling applied to the $P|r_j, q_j|C_{\max}$ problem. Li et al. [10] used a polynomial time approximation scheme for scheduling identical parallel batch machines ($P|r_j, B|C_{\max}$). Li and Wang [11] proposed an efficient algorithm for scheduling with inclusive processing set restrictions and job release times ($P|r_j, incl. proc. sets|C_{\max}$).

To the best of our knowledge, no research has yet been published that develops an efficient algorithm to minimize makespan for unrelated parallel machines with release dates. The rest of this paper is organized as follows. Section 2 presents our proposed lower bounds. Section 3 presents the proposed PSO. In Section 4, the computational results are reported. Section 5 presents our conclusions and suggestions for future research.

2. Lower Bounds to $R|r_j|C_{\max}$

We propose two straightforward and easily implementable lower bounds for the studied problem. LB_1 is the maximum value of each job's release date plus the minimum processing time (across all machines). LB_2 is set to the minimum release date (among all jobs) plus the sum of all jobs' minimum processing times (across machines) divided by the number

TABLE 1: The matrix of processing times for example 1.

P_{ij}	j_1	j_2	j_3	j_4	j_5	j_6	j_7
m_1	15	29	40	32	46	8	44
m_2	41	29	40	32	31	24	49
r_j	27	4	3	9	13	17	1

of machines. We set lower bound LB equal to the maximum value of LB_1 and LB_2 :

$$\begin{aligned}
 LB_1 &= \max_{1 \leq j \leq n} \left(r_j + \min_{1 \leq i \leq m} p_{ij} \right) \\
 LB_2 &= \min_{1 \leq j \leq n} r_j + \frac{\left(\sum_{j=1}^n \min_{1 \leq i \leq m} p_{ij} \right)}{m} \\
 LB &= \max \{ LB_1, LB_2 \}.
 \end{aligned} \tag{1}$$

To illustrate the proposed lower bounds, we consider example 1, which has 2 machines and 7 jobs. The matrix of processing times for example 1 is given in Table 1:

$$\begin{aligned}
 LB_1 &= \max \{ 27 + 15, 4 + 29, 3 + 40, 9 \\
 &\quad + 32, 13 + 31, 17 + 8, 1 + 44 \} = 45 \\
 LB_2 &= 1 + \frac{(15 + 29 + 40 + 32 + 31 + 8 + 44)}{2} \\
 &= 1 + \frac{199}{2} = 100.5 \\
 LB &= \max \{ 45, 100.5 \} = 100.5.
 \end{aligned} \tag{2}$$

3. The Proposed PSO Algorithm

PSO was first introduced by Kennedy and Eberhart [12] for solving continuous nonlinear function optimization problems. PSO is based on the metaphor of social interaction and communication in flocks of birds or schools of fish. In these groups, there is a leader (the one with the best performance) who guides the movement of the whole swarm. In a PSO, each individual is called a “particle,” and each particle flies around the search space with some velocity. In each iteration, a particle moves from its previous location to a new location at its newly updated velocity, which is calculated based on the particle’s own experience and the experience of the whole swarm.

A population of M particles are assumed to evolve in an N -dimensional vector search R^N such that each particle k is assigned the position vector $X_k^t = (x_{k1}^t, x_{k2}^t, \dots, x_{kN}^t)$ and velocity vector $V_k^t = (v_{k1}^t, v_{k2}^t, \dots, v_{kN}^t)$, where x_{kd}^t represents the location and v_{kd}^t represents the velocity of particle k in the d th dimension of the search space at the t th iteration and $k \in \{1, 2, \dots, M\}$, $d \in \{1, 2, \dots, N\}$. Each particle knows its position and the corresponding objective

7	6	1	4	*	3	2	5
---	---	---	---	---	---	---	---

FIGURE 1: Representation of a particle’s solution.

function. The local best position for each particle k is encoded in the variables $P_k^t = (P_{k1}^t, P_{k2}^t, \dots, P_{kN}^t)$, and the global best position among all particles is encoded in the variable $P_g^t = (P_{g1}^t, P_{g2}^t, \dots, P_{gN}^t)$. The standard PSO equations can be described as follows [12]:

$$\begin{aligned}
 v_{kd}^{t+1} &= wv_{kd}^t + c_1 r_1 (P_{kd}^t - x_{kd}^t) + c_2 r_2 (P_{gd}^t - x_{kd}^t) \\
 x_{kd}^{t+1} &= x_{kd}^t + v_{kd}^{t+1},
 \end{aligned} \tag{3}$$

where w is the weight that controls the impact of the previous velocities on the current velocity, c_1 is the cognition learning factor, c_2 is the social learning factor, and r_1 and r_2 are random numbers uniformly distributed in $[0, 1]$.

PSO has been successfully applied to a variety of continuous nonlinear optimization problems. In recent years, considerable effort has been expended on solving scheduling problems by PSO algorithms. Articles [13, 14] used PSO algorithms to solve scheduling problems similar to the problems in this paper. Reference [13] provided a PSO algorithm for scheduling identical parallel machines to minimize makespan ($P||C_{\max}$). Reference [14] presented a PSO algorithm for scheduling nonidentical parallel batch processing machines to minimize makespan ($P|batch, s_j, S_m|C_{\max}$). This research uses PSO for the $R|r_j|C_{\max}$ problem. The following five headings describe the PSO algorithm used in this research: particle representation, initial population generation, particle velocity and sequence metric operators, local search, stopping criteria, and parameter settings.

3.1. Particle Representation. A coding scheme developed in [15] is used to represent a solution to the problem at hand. The coding scheme uses a list of job symbols and partitioning symbols. A sequence of job symbols, denoted by integers, represents a possible sequence of jobs. The partitioning symbol, an asterisk, designates the partition of jobs to machines. Generally, for an m -machine n -job problem, a solution contains $m - 1$ partitioning symbols and n job symbols, resulting in a total size of $(m + n - 1)$. For example, for a schedule with 7 jobs and 2 machines, the particle can be represented as shown in Figure 1.

The completed schedule is thus jobs 7, 6, 1, and 4 on machine 1; jobs 3, 2, and 5 on machine 2. This coding scheme specifies not only which jobs are assigned to which machine but also the order of the jobs on each machine. These pieces of

information are important, since we are scheduling unrelated parallel machines with release dates.

3.2. Initial Population Generation. In order to give the PSO algorithm good initial solutions and to increase the chances of getting closer to regions that yield good objective functions, we propose a heuristic, named SRD_Reassign. The proposed heuristic SRD_Reassign is described as follows.

3.2.1. Heuristic SRD_Reassign

Step 1. Let U be the set of unscheduled jobs; let t_i be the sum of the processing times of the jobs that have already been scheduled on machine i , $i = 1, \dots, m$. Initially, set $U = \{1, \dots, n\}$ and $t_i = 0$, for $i = 1, \dots, m$.

Step 2. Arrange the jobs in the order of the shortest release date (SRD) first, and then assign the job to machine i^* that has the minimum processing time, that is, $p_{i^*j} = \min_{1 \leq i \leq m} p_{ij}$. Repeat until all jobs have been scheduled to generate a complete schedule.

Step 3. Let A_i be the set of scheduled jobs on machine i , $i = 1, \dots, m$; let $C_{\max} = \max_{i=1, \dots, m} \{t_i\}$ represent the maximum completion time and denote the set of candidate jobs for reassignment as B . Initially, set $B = \{\text{null}\}$.

Step 4. Identify machine i for which $t_i = C_{\max}$. For every job j , $j \in A_i$, search for machine h ($h \neq i$), such that if job j was reassigned to machine h and the jobs on machine h were sorted in SRD, the new calculated t_h would be smaller than C_{\max} , that is, $t_h = \max(t_h, r_j) + p_{hj, j \in A_h} < C_{\max}$. If job j and machine h can be found, update the candidate set by adding job j on machine h to the candidate set B by setting $B = B \cup \{(h, j)\}$. If $B = \{\text{null}\}$, then go to Step 6.

Step 5. Select machine h and job j from B for the reassignment that has maximum Gain, where $\text{Gain} = \max\{C_{\max} - t_h\}$. Reassign job j to machine h by setting $A_h = A_h \cup \{j\}$. Sort the jobs on machine h in SRD and update $t_h = \max(t_h, r_j) + p_{hj, j \in A_h}$. Remove job j from machine i by setting $A_i = A_i \setminus \{j\}$. Sort the jobs on machine i in SRD and update $t_i = \max(t_i, r_j) + p_{ij, j \in A_i}$. Set $C_{\max} = \max_{i=1, \dots, m} \{t_i\}$ and $B = \{\text{null}\}$. Go to Step 4.

Step 6. Terminate the procedure.

The first two particles are generated by first-come, first-serve (FCFS) rule and SRD_Reassign. The remaining particles are generated by applying local search to the solution found by SRD_Reassign. For FCFS rule, we consider all unscheduled jobs and schedule each one on the first available machine according to FCFS. Local search is done by randomly choosing two jobs j_1 and j_2 from the solution found by SRD_Reassign and then interchanging jobs j_1 and j_2 to generate a new solution. From the initial population pool, we identify the best current solution C_{\max}^* and update the global best location (P_g^t).

3.3. Particle Velocity and Sequence Metric Operators. Kashan and Karimi [13] worked on the classical PSO equations to provide a discrete PSO algorithm that maintained all major characteristics of the original continuous PSO equations when solving parallel machine scheduling problems. In this research, we use the two equations proposed in [13] to update the particle velocity and the sequence metric operators as shown in (4) and (5):

$$V_k^{t+1} = V_k^t \overset{+}{\circ} \left(\left(R_1 \overset{\times}{\circ} \left(P_k^t \overset{-}{\circ} X_k^t \right) \right) \overset{+}{\circ} \left(R_2 \overset{\times}{\circ} \left(P_g^t \overset{-}{\circ} X_k^t \right) \right) \right) \quad (4)$$

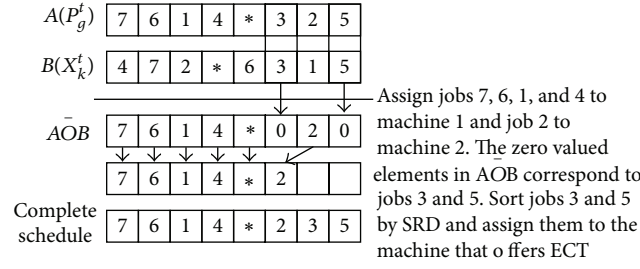
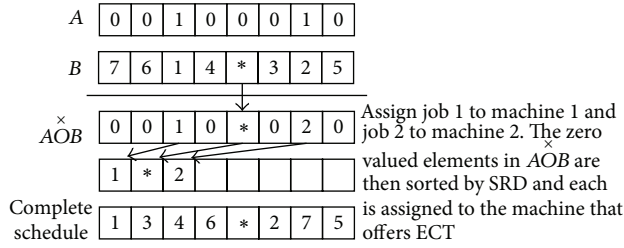
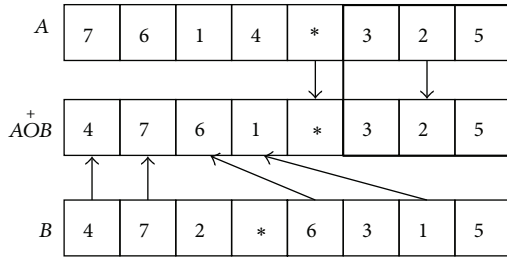
$$X_k^{t+1} = X_k^t \overset{+}{\circ} V_k^{t+1}. \quad (5)$$

In (4), V_k^t and X_k^t represent the velocity and position arrays of particle k at the t th iteration, respectively. P_k^t and P_g^t represent the local best position for each particle k and the global best position among all particles visited so far. R_1 and R_2 are 1-by- $(m + n - 1)$ arrays in which each digit is 0 or 1. These random arrays are generated from a Bernoulli distribution. $\overset{-}{\circ}$, $\overset{\times}{\circ}$, and $\overset{+}{\circ}$ are subtraction, multiplication, and addition operators, respectively. The definitions of the operators are as follows.

The subtraction operator ($\overset{-}{\circ}$) defines the differences between the current position of the k th particle, X_k^t , and a desired position P_k^t (or P_g^t). It first finds elements that do not have the same content in X_k^t and P_k^t (or P_g^t). It schedules those elements based on their orders in P_k^t (or P_g^t). Next, it finds elements that have the same content in X_k^t and P_k^t (or P_g^t) and gives those elements zero values. It puts zero-valued elements in SRD and then schedules them on whatever machine that offers the earliest completion time (ECT). Figure 2 demonstrates the manner in which the $\overset{-}{\circ}$ operator works for example 1.

The multiplication operator ($\overset{\times}{\circ}$) can enhance our PSO algorithm's exploration. It first generates a 1-by- $(m + n - 1)$ binary vector for a solution vector, and then it does a multiplication process where the asterisk positions from solution are kept. These random binary arrays perform subroutines within PSO that use random numbers to enhance the exploration ability of PSO. Figure 3 demonstrates the manner in which the $\overset{\times}{\circ}$ operator works for example 1. The nonzero-valued elements in $A \overset{\times}{\circ} B$ are scheduled first. Then, all zero-valued elements in $A \overset{\times}{\circ} B$ are sorted in SRD, and then each zero-valued element is assigned to the machine that offers the ECT.

The addition operator ($\overset{+}{\circ}$) is a crossover operator that is commonly used in genetic algorithms. Here, we used a crossover that was proposed in [15]. The crossover scheme has three main steps: (1) it obtains asterisk positions from the first parent A ; (2) it obtains a randomly selected subschedule from the first parent A ; (3) it scans the second parent B from left to right and fills the gaps in the child's ($A \overset{+}{\circ} B$) schedule

FIGURE 2: The subtraction operator (\bar{O}) applied to example 1.FIGURE 3: The multiplication operator ($\times O$) applied to example 1.FIGURE 4: The addition operator (^+O) applied to example 1.

with jobs taken from the second parent B. Figure 4 shows an illustration of this crossover scheme.

During the execution of the \bar{O} , $\times O$, and ^+O operators, whenever a complete schedule generates a better solution than the best current solution C_{\max}^* , we will update the best current solution C_{\max}^* and the global best location (P_g^t).

3.4. Local Search. It is well known that evolutionary memetic algorithms can be improved by hybridization with local search. For each particle P_k^t , we do the following local search procedure (LSP) to further improve the current solution.

3.4.1. Local Search Procedure (LSP)

Step 1. Initially, set $l = 1$.

Step 2. Identify machine i that has maximum completion time (C_{\max}). Randomly choose one job j_1 from machine i and randomly choose one job j_2 from machine h ($h \neq i$). Insert job

j_1 after job j_2 on machine h . If a better $C_{\max}(P_k^t)$ is found, update P_k^t and go to Step 2; otherwise, go to Step 3.

Step 3. Randomly choose two jobs j_1 and j_2 from P_k^t ; j_1 and j_2 can be on the same machine or on two different machines. Interchange jobs j_1 and j_2 . If a better $C_{\max}(P_k^t)$ is found, update P_k^t and go to Step 2; otherwise, go to Step 4.

Step 4. If $l = L$, stop; otherwise set $l = l + 1$ and go to Step 2.

Again, during the execution of LSP, whenever a complete schedule generates a better solution than the best current solution C_{\max}^* , we will update the best current solution C_{\max}^* and the global best location (P_g^t).

3.5. Stopping Criteria and Parameter Settings. We studied the effects of five important parameters (R_1 , R_2 , local search moves L , population size, and number of iterations) on the performance of our proposed PSO. The model was tested and parameterized through a factorial study. The selected PSO parameters were $R_1 = 0.9$, $R_2 = 0.1$, $L = 10$, population size = 73, and number of iterations = 4044. The appendix includes a detailed description of our parameterization study.

4. Computational Results

In this section, we present several computational results of the proposed PSO algorithm. We compare our proposed PSO algorithm with a mixed integer programming (MIP) model developed in our previous research [16] on the $R|r_j|C_{\max}$ problem. The MIP model [16] was coded in AMPL and implemented in CPLEX 11.2. The proposed heuristic, SRD_Reassign, and the proposed PSO algorithm were implemented in C. The MIP model, heuristic, and PSO algorithm were executed on a computer with a 2.5 GHz CPU and 2 GB of memory. Processing times p_{ij} were generated from the uniform distribution [1, 100]. Release dates were generated in a manner similar to that of Mönch et al. [17]. We generated release dates r_j from the uniform distribution $[0, (\alpha/m)((\sum_{i=1}^m \sum_{j=1}^n p_{ij})/m)]$. α controlled the range of release dates. High values of α tend to produce widely separated release dates. α values were set at 0.1, 0.25, and 0.5. We used 4 machines with 18 jobs ($4m18n$) to represent small problem instances and 10 machines with 100 jobs ($10m100n$) to represent large problem instances. For each α , 20 problem

TABLE 2: The performance of heuristic SRD_Reassign for small problem instances.

$4m18n$ α	MIP		SRD_Reassign		FCFS	
	Mean	Avg. time	Mean	Avg. time	Mean	Avg. time
0.10	1	1176.32	1.11	0.003	1.46	0.000
0.25	1	239.19	1.09	0.002	1.43	0.000
0.50	1	58.02	1.04	0.002	1.34	0.000
Average	1	491.18	1.08	0.002	1.41	0.000

Mean = average ratios of heuristic/MIP obtained from 20 instances.

TABLE 3: The performance of heuristic SRD_Reassign for large problem instances.

$10m100n$ α	SRD_Reassign		FCFS	
	Mean	Avg. time	Mean	Avg. time
0.10	1.43	0.005	2.03	0.001
0.25	1.17	0.003	1.62	0.000
0.50	1.05	0.005	1.36	0.000
Average	1.22	0.004	1.67	0.000

Mean = average ratios of heuristic/LB obtained from 20 instances.

instances were randomly generated. The effectiveness of each algorithm was evaluated by the mean performance and the required computation time (labeled as “Avg. time”). For small problem instances, a ratio was calculated by dividing the algorithm’s makespan by the optimal MIP makespan. For large problem instances, a ratio was calculated by dividing the algorithm’s makespan by the makespan from LB. The mean performance of the algorithm for each α was the average ratio obtained from 20 runs of the algorithm.

4.1. Comparison of Heuristics for $R|r_j|C_{\max}$ Problem. We compared the proposed heuristic SRD_Reassign with the optimal solutions obtained from the MIP model [16] and FCFS. FCFS is a dispatching rule that is commonly used for practical problems with release dates. Computational results for small and large problem instances are given in Tables 2 and 3, respectively. The results show that the proposed SRD_Reassign outperformed FCFS in terms of makespan. For small problem instances, the average SRD_Reassign makespan was 1.08 times the optimum, and the average FCFS makespan was 1.41 times the optimum. Both heuristics outperformed the MIP model in terms of computation time. When α was small, both heuristics had larger ratios to the optimal solutions than they had when α was large. Also, the MIP took more computation time to find optimal solutions when α was small. This probably indicates that problems with small release date ranges are harder to solve than problems with large release date ranges. For large problem instances, the average SRD_Reassign makespan was 1.22-times greater than the lower bound (LB), and the average FCFS makespan was 1.67 times the LB. Both heuristics were calculated very quickly (in less than 1 second) even for large problem instances.

4.2. Comparison of Metaheuristics for $R|r_j|C_{\max}$ Problem. We compared the proposed PSO with an existing metaheuristic,

namely, the version of simulated annealing (SA) described by Lee et al. [18]. This SA variant was originally designed for solving the $P||C_{\max}$ problem. SA is a metaheuristic, and it can be used without any problem-dependent knowledge; therefore, it can be used to solve the $R|r_j|C_{\max}$ problem. In order to provide a fair comparison, we used the same initial solution (SRD_Reassign) for both PSO and SA. We also adjusted the SA parameters to ensure that both PSO and SA ran for similar computation times. The termination criterion for SA was set to run for 12 seconds for small problem instances and 83 seconds for large problem instances. If SA found a solution equal to LB, the program would terminate earlier. Computational results for small and large problem instances are given in Tables 4 and 5, respectively.

Computational results show that the proposed PSO outperformed the SA in terms of makespan. For small problem instances, the PSO found optimal solutions at all three α settings. The average SA makespan was 1.05 times the optimum. Both metaheuristics outperformed the MIP model in terms of computation time. The last column in Table 4 reports how many times a given algorithm produced a better makespan than the other algorithm. For instance, a value of c/d in column PSO/SA means that, out of 20 problems, there were c problems for which PSO yielded a better solution than SA, d problems for which SA performed better, and $20-c-d$ problems for which PSO and SA yielded the same makespan.

For large problem instances, the average PSO makespan was 1.08-times greater than the LB, and the average SA makespan was 1.15 times the LB. The last column in Table 5 shows how many times out of 20 the LB was obtained by LB_1 and how many times the LB was obtained by LB_2 . When α was small, LB_2 provided a better lower bound than LB_1 ; however, when α was large, LB_1 provided a better lower bound than LB_2 . This suggests that LB_2 performs better for problems with narrow release date ranges, and LB_1 performs better for problems with wide release date ranges.

TABLE 4: The performance of PSO for small problem instances.

$4m18n$ α	MIP		PSO		SA		PSO/SA
	Mean	Avg. time	Mean	Avg. time	Mean	Avg. time	
0.10	1	1176.32	1.00	11.73	1.05	12.79	11/0
0.25	1	239.19	1.00	10.69	1.06	12.17	18/0
0.50	1	58.02	1.00	8.69	1.03	10.28	10/0
Average	1	491.18	1.00	10.37	1.05	11.75	13/0

Mean = average ratios of algorithm/MIP obtained in 20 instances.

TABLE 5: The performance of PSO for large problem instances.

$10m100n$ α	PSO		SA		PSO/SA	LB_1/LB_2
	Mean	Avg. time	Mean	Avg. time		
0.10	1.20	82.61	1.32	84.1	20/0	0/20
0.25	1.03	49.37	1.11	85.54	20/0	10/10
0.50	1.00	25.34	1.01	55.18	10/0	20/0
Average	1.08	52.44	1.15	74.94	16.7/0	10/10

Mean = average ratios of algorithm/LB obtained in 20 instances.

4.3. The Effects of the Proposed PSO. Next, since the proposed PSO effectively incorporates a number of ideas (initial solutions, SRD, ECT, and LSP), we examine which parts are essential to its functionality. We examine these effects by disabling a single component, running the proposed PSO without that component, and observing performance. We choose to study large problems. These experiments are described as follows.

PSO-Initial Heuristics: instead of generating an initial population by heuristics, we randomly generated an initial set of solutions to make up the initial population.

PSO-SRD: instead of sorting elements by SRD and then scheduling them on whatever machine that offered the ECT within PSO operators (\bar{O} and \bar{O}^x), we randomly chose unscheduled jobs and then scheduled them on whatever machine that offered the ECT.

PSO-ECT: instead of sorting elements by SRD and then scheduling them on whatever machine that offered the ECT within PSO operators (\bar{O} and \bar{O}^x), we sorted elements by SRD and then scheduled them on the first available machine.

PSO-LSP: local search procedure was disabled.

PSO-LSP + LS [13]: local search procedure was disabled and a local search algorithm used in [13] was applied. Since the formulation $0 < p_a - p_b < FT_i - FT_j$ in step 4 [13] was not suitable for the unrelated parallel machines environment, we modified it to find two jobs from M_i and M_j such that an exchange of those two jobs was able to improve the current best makespan.

Table 6 lists the average makespan ratio of PSO-variant to standard PSO (which has initial heuristics, SRD, ECT, and LSP by default). Table 6 shows that the PSO performed poorly when the initial heuristics were not applied and when the initial population was randomly generated. The PSO also performed poorly when the ECT strategy was not applied within the PSO operators. The average ratio of PSO without

initial heuristics to standard PSO was 1.019, the average of PSO without SRD to standard PSO was 1.006, the average of PSO without ECT to standard PSO was 1.020, and the average of PSO without LSP to standard PSO was 1.007. In all, all of the proposed PSO versions without any one of the parts (heuristic initial solutions, SRD, ECT, and LSP) performed worse than the PSO with all of them.

Moreover, we compared our proposed PSO with another existing PSO. The closest existing PSO that we were able to find was the hybridized discrete PSO (HDPSO) proposed in [13]. The HDPSO [13] was designed to minimize makespan for identical parallel machines ($P||C_{\max}$). The proposed PSO and the HDPSO both are designed to minimize makespan for parallel machines and they both use formulas (4)-(5) to update each particle's velocity and position. However, the HDPSO is still quite different from our PSO. The HDPSO considers problems without release dates; hence, its coding scheme does not consider the order of jobs on the same machine. Also, HDPSO considers an identical parallel machine environment; it uses the LPT rule to assign jobs with zero-valued elements within \bar{O} , \bar{O}^x , and \bar{O}^+ operators in formulas (4)-(5). It is well known that the LPT rule does not perform well in unrelated parallel machine environments. If HDPSO is used to solve the $R|r_j|C_{\max}$ problem without modifications, it will not perform very well. Table 7 shows a comparison between HDPSO and our PSO. Since the \bar{O} , \bar{O}^x , and \bar{O}^+ operators within formulas (4)-(5) are quite different in both PSOs, there is no point in comparing them. We focus our comparison on initial heuristics and local searches. The first column in Table 6 indicates that our PSO with an initial heuristic performs better than a version without an initial heuristic. HDPSO might exhibit similar performance differences. The last column in Table 6 indicates that our proposed LSP performs better than the local search algorithm used in [13]. The standard PSO (LSP is embedded)

TABLE 6: The effects of the proposed PSO.

10m100n	PSO-initial heuristics	PSO-SRD	PSO-ECT	PSO-LSP	PSO-LSP + LS [13]
α	Mean	Mean	Mean	Mean	Mean
0.10	1.054	1.017	1.055	1.019	1.023
0.25	1.001	1.001	1.004	1.002	1.003
0.50	1.001	1.001	1.001	1.001	1.001
Average	1.019	1.006	1.020	1.007	1.009
Ave. time	52.55	51.14	51.29	48.44	1624.31

Mean = average ratios of PSO-variant/standard PSO obtained in 20 instances.

TABLE 7: Comparison between two PSOs.

	Problem	Initial heuristic	\bar{O} , \bar{O}^* , and \bar{O}^+ operators within formulas (4)-(5)	Local search
HDPSO	$P C_{\max}$	n/a	Coding scheme and LPT rule are not suitable for the $R r_j C_{\max}$ problem	Local search algorithm
PSO	$R r_j C_{\max}$	SRD_Reassign	Coding scheme, SRD, and ECT are designed for the $R r_j C_{\max}$ problem	LSP

TABLE 8: Factors and levels of PSO parameter study.

Factor	Design units	
	-1	+1
A: R_1	0.1	0.9
B: R_2	0.1	0.9
C: L	10	50
D: population size	10	100
E: iteration	1000	5000

versus PSO-LSP+LS [13] is 1.000 versus 1.009. Moreover, the proposed LSP outperforms the local search algorithm used in [13] in terms of average computation time. Therefore, we can conclude that our proposed PSO provides better and more efficient strategies for parallel machine makespan minimization problems than what HDPSO provides. Our PSO is more likely to provide promising results than HDPSO.

5. Conclusions and Future Work

We studied the problem of scheduling jobs on unrelated parallel machines with release dates to minimize makespan. In this research, we proposed two lower bounds for the studied problem. We also proposed a heuristic, SRD_Reassign, and a metaheuristic, PSO, to tackle the problem. Computational results showed that SRD_Reassign outperformed the commonly used heuristic, FCFS, in terms of makespan. The proposed PSO outperformed a comparable variant of SA in terms of makespan. Future work can extend our approach for other performance criteria or even for multiobjective parallel machine scheduling problems.

Appendix

We studied the effects of five important parameters (R_1 , R_2 , local search moves L , population size, and number of iterations) on the performance of our proposed PSO. In order to test the significance of each parameter, 10m100n was chosen as a representative problem instance; the objective was to minimize C_{\max} in an unrelated parallel machine environment with release dates. Because problems with small ranges of release dates are harder to solve than problems with large ranges of release dates, the release date factor α was set to 0.1. In order to obtain information about the importance of each of the factors, we conducted an initial screening experiment. Each parameter was categorized as being at a high or low level, as shown in Table 8. We conducted a 2^{5-2}_{III} screening experiment to determine which factors were significant. The results of this experiment are shown in Table 9 where each C_{\max} value is the average of 20 problem instances. The half normal plot provided by Design Expert indicates that R_1 (factor A), population size (factor D), and iterations (factor E) were significant to the C_{\max} response in the screening experiment. The model in terms of coded factors is $\hat{y} = 116.43 - 1.13A - 0.45D - 0.59E$. The model shows that increasing the A, D, and E values could decrease makespan.

Next, we used the method of steepest descent (Myers et al. [19]) to provide information about the region of improved response. The path of steepest descent is shown in Table 10. The results show that a reduction in C_{\max} was experienced after Run 2. Although Run 2 improved C_{\max} by 2.4% compared with Run 0, its computation time was very long. Run 1 improved C_{\max} by 2.1% relative to Run 0 and used less computation time. We chose the settings of Run 1 as our final set of parameters. Hence, PSO parameters were set to $R_1 = 0.9$, population size = 73, and iterations = 4044. Since

TABLE 9: Results of 2^{5-21}_{III} factorial design for parameters of PSO.

Run	A: R_1	B: R_2	C: L	D: Pop-size	E: Iteration	C_{\max}
1	0.1	0.9	50	10	1000	118.65
2	0.9	0.1	50	10	5000	115.45
3	0.9	0.9	10	100	1000	115.00
4	0.9	0.1	10	10	1000	116.30
5	0.1	0.1	10	100	5000	116.35
6	0.9	0.9	50	100	5000	114.45
7	0.1	0.9	10	10	5000	117.10
8	0.1	0.1	50	100	1000	118.10

TABLE 10: The path of steepest descent for PSO.

Run		Coded units				Natural units		C_{\max}	Improv.	Time
		A	D	E	R_1	Pop. size	Iteration			
0	Base	0	0	0	0.5	55	3000	117.30	—	50.68
	Increment = Δ	1	0.40	0.52	0.4	18	1044.4	—	—	—
1	Base + Δ	1	0.40	0.52	0.90	73	4044	114.80	0.021	82.61
2	Base + 2Δ	2	0.80	1.04	0.90	91	5089	114.40	0.024	129.06
3	Base + 3Δ	3	1.20	1.57	0.90	109	6133	114.95	0.002	193.28

the R_2 and L were not significant, they were kept at low levels ($R_2 = 0.1$ and $L = 10$) to save computation time.

References

- [1] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan, "Optimization and approximation in deterministic sequencing and scheduling: a survey," *Annals of Discrete Mathematics*, vol. 5, pp. 287–326, 1979.
- [2] M. L. Pinedo, *Scheduling Theory, Algorithms, and Systems*, Springer, New York, NY, USA, 4th edition, 2012.
- [3] B. Chen and A. P. A. Vestjens, "Scheduling on identical machines: how good is LPT in an on-line setting?" *Operations Research Letters*, vol. 21, no. 4, pp. 165–169, 1997.
- [4] H. Kellerer, "Algorithms for multiprocessor scheduling with machine release times," *IIE Transactions*, vol. 30, no. 11, pp. 991–999, 1998.
- [5] C. Koulamas and G. J. Kyparisis, "Makespan minimization on uniform parallel machines with release times," *European Journal of Operational Research*, vol. 157, no. 1, pp. 262–266, 2004.
- [6] G. Centeno and R. L. Armacost, "Minimizing makespan on parallel machines with release time and machine eligibility restrictions," *International Journal of Production Research*, vol. 42, no. 6, pp. 1243–1256, 2004.
- [7] G. Lancia, "Scheduling jobs with release dates and tails on two unrelated parallel machines to minimize the makespan," *European Journal of Operational Research*, vol. 120, no. 2, pp. 277–288, 2000.
- [8] A. Gharbi and M. Haouari, "Minimizing makespan on parallel machines subject to release dates and delivery times," *Journal of Scheduling*, vol. 5, no. 4, pp. 329–355, 2002.
- [9] J. Carlier and E. Pinson, "Jackson's pseudo-preemptive schedule and cumulative scheduling problems," *Discrete Applied Mathematics*, vol. 145, no. 1, pp. 80–94, 2004.
- [10] S. Li, G. Li, and S. Zhang, "Minimizing makespan with release times on identical parallel batching machines," *Discrete Applied Mathematics*, vol. 148, no. 1, pp. 127–134, 2005.
- [11] C.-L. Li and X. Wang, "Scheduling parallel machines with inclusive processing set restrictions and job release times," *European Journal of Operational Research*, vol. 200, no. 3, pp. 702–710, 2010.
- [12] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the 1995 IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, Australia, December 1995.
- [13] A. H. Kashan and B. Karimi, "A discrete particle swarm optimization algorithm for scheduling parallel machines," *Computers and Industrial Engineering*, vol. 56, no. 1, pp. 216–223, 2009.
- [14] P. Damodaran, D. A. Diyadawagamage, O. Ghayeb, and M. C. Véllez-Gallego, "A particle swarm optimization algorithm for minimizing makespan of nonidentical parallel batch processing machines," *International Journal of Advanced Manufacturing Technology*, vol. 58, pp. 1131–1140, 2012.
- [15] R. Cheng, M. Gen, and T. Tozawa, "Minmax earliness/tardiness scheduling in identical parallel machine system using genetic algorithms," *Computers and Industrial Engineering*, vol. 29, no. 1–4, pp. 513–517, 1995.
- [16] Y. K. Lin and C. W. Lin, "Dispatching rules for unrelated parallel machine scheduling with release dates," *International Journal of Advanced Manufacturing Technology*, 2013.
- [17] L. Mönch, H. Balasubramanian, J. W. Fowler, and M. E. Pfund, "Heuristic scheduling of jobs on parallel batch machines with incompatible job families and unequal ready times," *Computers and Operations Research*, vol. 32, no. 11, pp. 2731–2750, 2005.
- [18] W. C. Lee, C. C. Wu, and P. Chen, "A simulated annealing approach to makespan minimization on identical parallel machines," *International Journal of Advanced Manufacturing Technology*, vol. 31, no. 3–4, pp. 328–334, 2006.

- [19] R. H. Myers, D. C. Montgomery, and C. M. Anderson-Cook, *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*, John Wiley & Sons, New York, NY, USA, 3rd edition, 2009.

Research Article

Dynamic Route Guidance Using Improved Genetic Algorithms

Zhanke Yu,¹ Mingfang Ni,¹ Zeyan Wang,¹ and Yanhua Zhang²

¹ Institute of Communication Engineering, PLA University of Science and Technology, NanJing 210007, China

² Lightning Protection Center, Meteorology Bureau of Jiangsu Province, NanJing 210009, China

Correspondence should be addressed to Zhanke Yu; jackty_2004@163.com

Received 27 September 2012; Accepted 30 December 2012

Academic Editor: Rui Mu

Copyright © 2013 Zhanke Yu 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 improved genetic algorithm (IGA) for dynamic route guidance algorithm. The proposed IGA design a vicinity crossover technique and a greedy backward mutation technique to increase the population diversity and strengthen local search ability. The steady-state reproduction is introduced to protect the optimized genetic individuals. Furthermore the junction delay is introduced to the fitness function. The simulation results show the effectiveness of the proposed algorithm.

1. Introduction

Online route guidance is one of the most desirable features in intelligent transportation systems (ITSs) [1]. Dynamic route guidance algorithm compute routes with minimum travel time by taking into account the rapid changes in the network traffic conditions and guide the behavior of travelers by providing them with optimal route based on real-time traffic information. As a result the travel time can be saved and the traffic congestion can be avoided.

Various algorithms are available for computing the optimal route. The most popular algorithm is the Dijkstra's algorithm. Many variations to the Dijkstra's algorithm such as bidirectional search and binary heap implementation have been proposed to improve its response time. The A* algorithm, which is widely used in vehicle navigation, is an improved version of the Dijkstra's algorithm [2, 3]. It makes use of an appropriate heuristic function to search the most promising nodes first thereby reducing the computation time. However, the traditional optimal algorithms often cannot be used because they are too computationally intensive to be feasible for real-time operations. A number of heuristic search strategies have been developed for increasing the computational efficiency of route search. Most of these heuristic search algorithms originated in the artificial intelligence field, such as genetic algorithm (GA) [4]. GA is an intelligence search algorithm based on the hypothesis of

natural selections and natural genetics and has been successfully applied to various areas of optimization [5–10]. GA-based approaches have several advantages. Naturally, they cannot only treat the discrete variables but also overcome the dimensionality problem. In addition, they have the capability to search for the global optimum or quasi optimums within a reasonable computation time. But in engineering practice, premature convergence often happens, and sometimes the speed of convergence is very slow. Accordingly, researchers have been designing the improved genetic algorithms. Zou proposes an improved genetic algorithm for dynamic route guidance algorithm that generate initial population by random A* algorithm to increase the population diversity [8]. Kanoh proposes a method for finding the quasishortest route within a given time using a genetic algorithm. In the proposed method, he improves the search rate and quality of solutions by giving direction to the search, using viruses [9].

In this paper, an improved genetic algorithm (IGA) for dynamic route guidance, which can overcome the aforementioned problems of the conventional GA to some extents, is developed. The proposed IGA incorporates the following three main features. First, a vicinity crossover scheme is devised to increase the diversity of population. Second, a greedy backward mutation scheme is developed to strengthen the local search ability. Third, the steady-state reproduction is introduced to protect the optimized genetic individuals.

In order to test the performance of the proposed algorithm, we implement the algorithm in matlab 2009. Simulation experiments demonstrate the effectiveness of the algorithm by evaluating it on random road topologies.

The paper is organized as follows. Problem definition and mathematical formulation is introduced in Section 2. The improved genetic algorithm for dynamic route guidance algorithm is developed in Section 3. The numerical simulation is reported in Section 4. Finally, conclusions are presented in Section 5.

2. Problem Definition and Mathematical Formulation

A road traffic network is represented by a digraph $G(V, E)$ that consists of a set of nodes V and a set of links E . Denote the number of nodes $|V| = n$ and the number of links $|E| = m$. A link $E_{ij} \in E$ is directed from node i to node j . Each link has an associated average travel time function $t(E_{ij})$ and delay function $d(E_{ij})$. Let $T(E_{ij})$ denote the weight of link E_{ij} . It can be expressed as

$$T(E_{ij}) = t(E_{ij}) + d(E_{ij}). \quad (1)$$

A path from an origin (o) to destination (d) may be defined as a sequential list of links: $(o, j), \dots, (i, d)$ and the weight of the path is the sum of weights on the individual links. The problem is to find the path P that has the minimum total weight from the origin node to the destination node

$$\min \sum_{E_{ij} \in P} T(E_{ij}), \quad (2)$$

where $i, j \in V$ and $i \neq j$.

2.1. Link Travel Time Function. Using BPR model the average travel time for a vehicle can be expressed as

$$t(E_{ij}) = t_0 \left(1 + 0.15 \left(\frac{q(E_{ij})}{c(E_{ij})} \right)^4 \right), \quad (3)$$

where t_0 = free flow travel time on link E_{ij} , in minute; $q(E_{ij})$ = volume of traffic on link E_{ij} per hour; $c(E_{ij})$ = capacity of link E_{ij} per hour; $t(E_{ij})$ = the average travel time for a vehicle on link E_{ij} , in minute.

2.2. Delay Functions for Signalized Intersections. Over the past few decades, a number of delay formulas have been proposed to account for delay at signalized intersections. Among the better known of these delay formulas are Webster and HCM 2000.

As for nonsaturated intersections, Webster model can be used. Average delay at intersection can be expressed as

$$d(E_{ij}) = 0.9 \left(\frac{T(1 - \lambda^2)}{2(1 - \lambda X)} + \frac{X^2}{2Q(1 - X)} \right), \quad (4)$$

where $d(E_{ij})$ = average delay at intersection, in seconds; T = cycle length of the traffic light, in seconds; λ = green time

TABLE 1: Characteristics of the IGA.

Encoding:	Path string, no duplicate individuals
Selection:	Roulette
Crossover:	Vicinity crossover
Mutation:	Greedy backward mutation Vicinity crossover
Improvements:	Greedy backward mutation Steady-state reproduction

of the traffic light/cycle length of the traffic light; Q = traffic volume on entering link, in vehicle per hour; X = intersection saturation degree.

This model can be applied under saturated ($X < 0.8$) conditions only.

As for saturated intersections ($X > 0.8$), average delay at intersection is estimated as

$$d(E_{ij}) = d_1(E_{ij}) + d_2(E_{ij}),$$

$$d_1 = 0.38T \frac{(1 - \lambda)^2}{(1 - \lambda X)}, \quad (5)$$

$$d_2 = 173X^2 \left((X - 1) + \sqrt{(X - 1)^2 + \frac{16X}{S}} \right),$$

where $d(E_{ij})$ = average delay at intersection, in seconds; $d_1(E_{ij})$ = uniform delay at intersection, in seconds; $d_2(E_{ij})$ = oversaturation delay at intersection, in seconds; S = saturate volume of intersection, in vehicle per hour; X = intersection saturation degree.

3. The Improved Genetic Algorithm

By introducing the following new techniques, the performance of a simple genetic algorithm for dynamic route guidance algorithm has been improved essentially. Various characteristics of this proposed IGA are shown in Table 1.

3.1. Coding and Initial Population. We regard each route from the origin node to the destination node as a chromosome and express it as a sequence of nodes. The first gene in the chromosome is always the origin node, and the last gene in the chromosome is always the destination node. Since different paths may have different number of intermediate nodes, the chromosomes will be of variable length. However, the maximum length of a chromosome cannot exceed the total number of nodes in the network. Any repeated nodes in the chromosome signify that the path represented by the chromosome contains a loop and in network routing, any loop should be eliminated. At the beginning, the population is filled with chromosomes that represent random paths.

The algorithm to generate the random path is adapted from [10]. The algorithm goes as follows.

Step 1. Start from the origin node.

Step 2. Randomly choose, with equal probability, one of the nodes that are connected to the current node.

Step 3. If the chosen node has not been visited before, mark that node as the next node in the path. Otherwise, find another node.

Step 4. If all the neighboring nodes have been visited, go back to Step 1.

Step 5. Otherwise, repeat Step 2 by using the next node as the current node. Do this until the destination node is found.

3.2. Fitness Function. For the dynamic route guidance problem, the lower the total weight of the path, the better the solution. The fitness function is defined as follows:

$$f = \frac{1}{\sum_{E_{ij} \in P} T(E_{ij})}. \quad (6)$$

3.3. Selection. In this GA, roulette wheel (Monte Carlo) method is adopted as the selection operator. Chromosomes with better fitness will have a higher probability to be selected.

3.4. Steady-State Reproduction. To avoid the highest fitness chromosome destroyed in the process of crossover and mutation, we adopt steady-state reproduction and replace the worst chromosome produced by the genetic operators with the best chromosome in the previous population.

3.5. Vicinity Crossover Strategy. The common way of crossover is the two chromosomes selected must have at least one common node other than the origin and destination nodes. The common node is called the crossover point. Crossover operation will be carried out on crossover point. In this paper, we propose a new vicinity crossover strategy. The main idea is crossover operation will be carried out not on common point but on vicinity point. The vicinity point is the two adjacent points that the distance between them less than the given maximal vicinity value in the networks. The distance between vicinity points is vicinity value. To ensure that the paths generated by the crossover operation are still valid paths, the two chromosomes selected must have at least one vicinity node other than the origin and destination nodes. If more than one vicinity node exists, one of them will be randomly chosen with equal probability. For the sake of clearly describing the concept of vicinity crossover strategy, we consider the following example. The topology of network is shown in Figure 1(a). Assume origin node is 1, destination node is 8 and maximal vicinity value is 3.

Assume we have the following parent chromosomes.

Parent chromosome1 = [1 5 | 4 8].

Parent chromosome2 = [1 3 | 2 8].

where node 1 and node 8 are the origin node and destination node respectively. In this example, the vicinity nodes are node 5 of parent chromosome1 and node 2 of parent chromosome2, node 3 of parent chromosome2 and node 4 of parent chromosome1. Therefore, crossover operation will exchange the first portion of chromosome1 with the second portion of chromosome2 and vice versa. The vicinity value

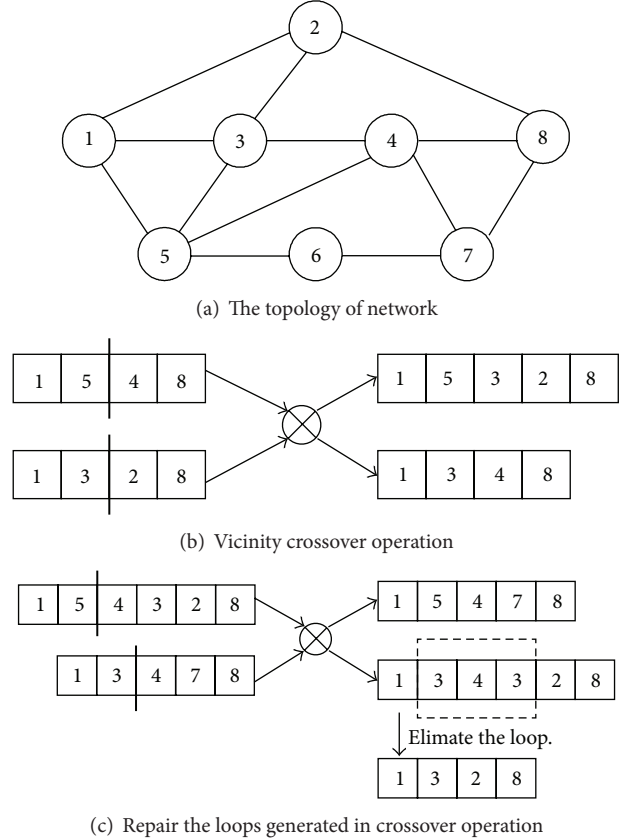


FIGURE 1: The process of vicinity crossover operation.

of node 5 and node 2 is 2. Hence $5 \rightarrow 3 \rightarrow 2$ is a generated vicinity path. The vicinity value of node 3 and node 4 is 1.

As a result, the following child chromosomes will be generated.

Child chromosome1: [1 5 3 2 8].

Child chromosome2: [1 3 4 8].

These two chromosomes would then become new members of the population as shown in Figure 1(b). It is possible that loops may occur after crossover operation is performed. Loops in a chromosome can be repaired by performing a search along the chromosome to find repeated nodes. The nodes in between the repeated nodes are then eliminated. For example, in Figure 1(c), assume that we have the following chromosome that contains a loop.

Chromosome with loop: [1 3 4 3 2 8].

In this case, there are two node 3 in the chromosome which signifies that the path contains a loop. This chromosome can be fixed by eliminating one of the node 3's and all the other nodes in between the two node 3's. The fixed chromosome would become like this.

Fixed chromosome: [1 3 2 8].

The fixed chromosome can be searched again just in case there are multiple loops in the chromosome. The vicinity crossover strategy will increase population diversity.

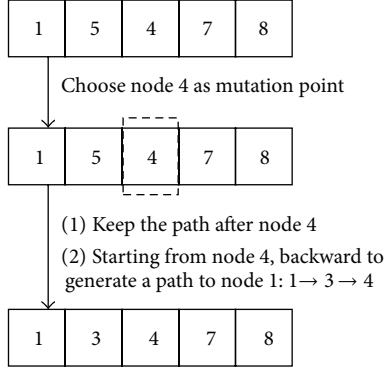


FIGURE 2: The process of greedy backward mutation.

3.6. Greedy Backward Mutation Strategy. Instead of bitwise mutation, a new greedy backward mutation strategy is used. The main idea is the following. For each chromosome that is chosen to be mutated, if the correlation degree of intermediate nodes in the path from origin to destination (i.e., the origin and destination node cannot be chosen as the mutation point) is greater than 2, then the intermediate nodes will be chosen as candidate nodes. A mutation point will be chosen randomly, with equal probability, among the candidate nodes. Once the mutation point is chosen, the chromosome will be changed starting from the node before the mutation point and backwards. To explain this procedure, we still use the network as shown in Figure 1(a) and the procedure of greedy backward mutation is shown in Figure 2.

Assume that the following chromosome has been chosen to be mutated.

Original chromosome: [1 5 4 7 8].

Where 1 and 8 are the origin node and the destination node, respectively. Assume also that the node 4 has been chosen as the mutation point. The mutated chromosome would become like this.

Mutated chromosome: [1 x_1 x_2 ... 4 7 8].

The mutated chromosome now contains a new backward path from 4 to 1, where x_i is the i th new node in the path. The new path is generated randomly; the same way as the paths in the initial population is generated.

4. Simulation Results

To test the effectiveness of the proposed IGA for dynamic route guidance, we implement the algorithm and conduct a series of simulation experiments. The algorithm has been coded in Matlab 2009 and implemented on an Intel Core 2, CPU 2.53 Ghz, RAM 2 GB, Windows XP System. Several main performance metrics are considered: number of generation, population size, convergence ability, and convergence speed.

We performed a set of experiments on a virtual square matrix map. The virtual maps of square matrix had sizes of 4×4 , 8×8 , 16×16 , and 32×32 . As shown in Figure 3, the

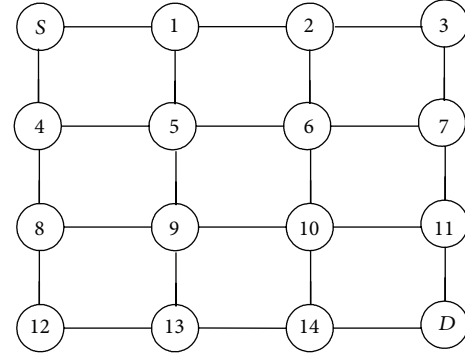


FIGURE 3: Virtual map of square matrix.

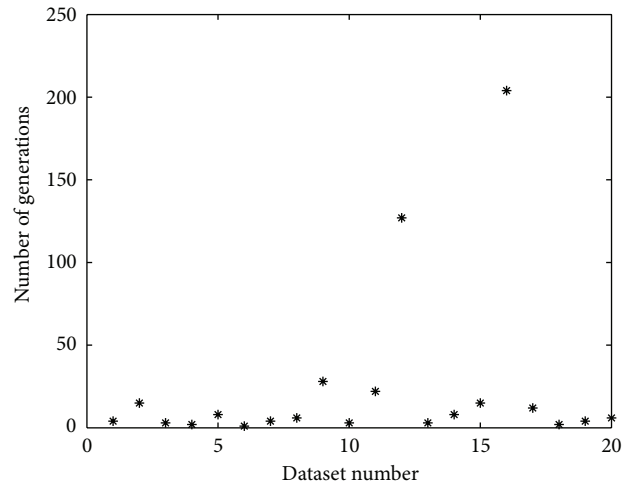


FIGURE 4: The number of generations required to find a feasible solution.

origin was at the upper left corner and the destination was at the lower right corner. The distances between nodes were varied from 10 to 50.

We generate 20 random datasets using the 4×4 virtual square matrix maps. Figure 4 shows the number of generations required to find a feasible solution. From the numerical results, we observe that all 20 datasets are able to converge to a feasible solution and the proposed IGA is correct. From the result, it can also be seen that most of time, the number of generations required to find a feasible solution is quite low and acceptable. However, in some cases, the number of generations required to find a feasible solution can exceed 200. The difference is quite large. The fact is probably caused by the state of initial population. The quality of initial population may be very low and takes many generations to find a feasible solution.

To solve the problem above, we can adjust the population size. Figure 5 shows the effect of population size on the average number of generations needed to find a solution. We execute the algorithm 10 times and record the average value. From Figure 5, we can conclude that the average number of

TABLE 2: Simulation parameters for IGA and SGA.

	IGA	SGA
Population size	60	60
Selection mode	Roulette	Roulette
Crossover operator	Vicinity crossover Crossover rate $p_c = 0.9$ Maximal vicinity value = 3	One-point crossover Crossover rate $p_c = 0.9$
Mutation operator	Greedy backward mutation Mutation rate $p_m = 0.01$	One-point mutation Mutation rate $p_m = 0.01$
Termination	500	500
Execution run	20	20

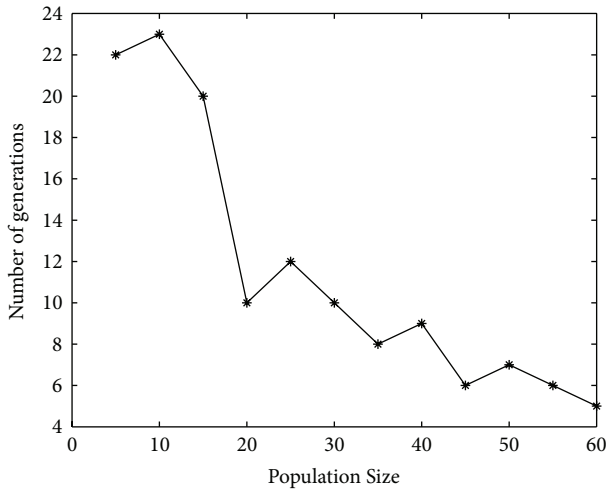


FIGURE 5: Average number of generations for different population size.

TABLE 3: Average generation to find optimal path.

Number of nodes	IGA	SGA
16	4.5	4.5
64	5.5	6.5
256	10	15.5
1024	18.5	39

generations become less with the increase of population size and is acceptable most of time.

We compare our improved genetic algorithm (IGA) with simple genetic algorithm (SGA) on virtual square matrix maps. Table 2 showed all of the simulation parameters for IGA and SGA.

Table 3 gives the average number of generational processes to find the optimal path. It shows that our IGA converges very fast, less than 20 generations even when the number of nodes growing to 1024. Also for a complex map with thousands of nodes, the average number of generational processes for the genetic algorithm beginning to converge is still small enough for real time applications.

Table 4 shows the experiment results of the minimal path weight and average path weight for our IGA and SGA for

TABLE 4: The comparison of IGA and SGA.

Number of nodes	IGA Mini_weight	SGA Mini_weight	IGA Avg_weight	SGA Avg_weight
16	126	126	128.2	129.6
64	201	207	212.4	224.2
256	392	396	403.9	418.3
1024	784	791	795.5	824.3

different network sizes. IGA achieves better minimum weight and average weight than SGA for all scenarios.

5. Conclusion

This paper presented an improved genetic algorithm for dynamic route guidance algorithm. Several details of the genetic algorithm such as vicinity crossover scheme, greedy backward mutation scheme, and steady-state reproduction have been specifically designed for solving this problem while other details are adapted from previous works. The simulation results show that the proposed algorithm works well in finding an optimal path for real-time applications and converges much faster to better solutions. The performance gets better with larger population size.

Acknowledgments

This work supported by the National Nature Science Foundation of China (no. 70971136) and supported by the Youth Foundation of Institute of Sciences, PLA University of Science and Technology (no. KYLYZL001235).

References

- [1] G. Dimitrakopoulos and P. Demestichas, "Intelligent transportation systems: systems based on cognitive networking principles and management functionality," *IEEE Vehicular Technology Magazine*, vol. 5, no. 1, pp. 77–84, 2010.
- [2] I. Chabini and S. Lan, "Adaptations of the A* algorithm for the computation of fastest paths in deterministic discrete-time dynamic networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 1, pp. 60–74, 2002.

- [3] G. R. Jagadeesh, T. Srikanthan, and K. H. Quek, "Heuristic techniques for accelerating hierarchical routing on road networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 3, no. 4, pp. 301–309, 2002.
- [4] L. Fua, D. Sunb, and L. R. Rilettc, "Heuristic shortest path algorithms for transportation applications: state of the art," *Computers and Operations Research*, vol. 33, pp. 3324–3343, 2006.
- [5] S. Yussof and O. H. See, "QoS routing for multiple additive QoS parameters using genetic algorithm," in *Proceedings of the 13th IEEE International Conference on Networks jointly held with the 7th IEEE Malaysia International Conference on Communications*, pp. 99–104, November 2005.
- [6] H. R. Varia and S. L. Dhingra, "Dynamic optimal traffic assignment and signal time optimization using genetic algorithms," *Computer-Aided Civil and Infrastructure Engineering*, vol. 19, no. 4, pp. 260–273, 2004.
- [7] N. Huimin, C. Mingming, and Z. Minghui, "Optimization theory and method of train operation scheme for urban rail transit," *China Railway Science*, vol. 32, pp. 128–133, 2011 (Chinese).
- [8] L. Zou, J. Xu, and L. Zhu, "Application of genetic algorithm in dynamic route guidance system," *Journal of Transportation Systems Engineering and Information Technology*, vol. 7, no. 3, pp. 45–48, 2007 (Chinese).
- [9] H. Kanoh, "Dynamic route planning for car navigation systems using virus genetic algorithms," *International Journal of Knowledge-Based and Intelligent Engineering Systems*, vol. 11, no. 1, pp. 65–78, 2007.
- [10] C. W. Ahn and R. S. Ramakrishna, "A genetic algorithm for shortest path routing problem and the sizing of populations," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 6, pp. 566–579, 2002.

Research Article

A Hybrid Algorithm of Traffic Accident Data Mining on Cause Analysis

Jianfeng Xi,^{1,2} Zhenhai Gao,¹ Shifeng Niu,³ Tongqiang Ding,² and Guobao Ning⁴

¹ State Key Laboratory of Automobile Dynamic Simulation, Jilin University, Changchun 130022, China

² College of Traffic, Jilin University, Changchun 130022, China

³ School of Automobile, Chang'an University, Xi'an 710064, China

⁴ School of Automotive Engineering, Tongji University, Shanghai 201804, China

Correspondence should be addressed to Guobao Ning; guobao.ning@tongji.edu.cn

Received 6 October 2012; Revised 30 December 2012; Accepted 31 December 2012

Academic Editor: Baozhen Yao

Copyright © 2013 Jianfeng Xi 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.

Road traffic accident databases provide the basis for road traffic accident analysis, the data inside which usually has a radial, multidimensional, and multilayered structure. Traditional data mining algorithms such as association rules, when applied alone, often yield uncertain and unreliable results. An improved association rule algorithm based on Particle Swarm Optimization (PSO) put forward by this paper can be used to analyze the correlation between accident attributes and causes. The new algorithm focuses on characteristics of the hyperstereo structure of road traffic accident data, and the association rules of accident causes can be calculated more accurately and in higher rates. A new concept of Association Entropy is also defined to help compare the importance between different accident attributes. T-test model and Delphi method were deployed to test and verify the accuracy of the improved algorithm, the result of which was a ten times faster speed for random traffic accident data sampling analyses on average. In the paper, the algorithms were tested on a sample database of more than twenty thousand items, each with 56 accident attributes. And the final result proves that the improved algorithm was accurate and stable.

1. Introduction

In recent years, with the growth of the volume and travel speed of road traffic, the number of traffic accidents, especially severe crashes, has been increasing rapidly on a yearly basis. The issue of traffic safety has raised great concerns across the globe, and it has become one of the key issues challenging the sustainable development of modern traffic and transportation. Therefore, it is crucial for engineers to be able to extract useful information from existing data to analyze the causes of traffic accidents, so that traffic administrations can be more accurately informed and better policies can be introduced [1–3].

Traffic conditions are a complex system due to many stochastic factors [4–6], and traffic accident data has long been known to be very difficult to process. Many attempts have been made in recent years through applying different methodologies and algorithms. Association rules has

captured wide attentions and careful studies because of its adoptability and the nature of being easily understood, the focus of study of which is how to increase the accuracy and efficiency of the calculation. Among the researches to date, Geurts et al. [7] used association rules to identify accident circumstances that frequently occur together at high frequency accident locations; Tesema et al. [8] developed an adaptive regression trees to build a decision support system to handle road traffic accident analysis; Marukatat [9] has made noticeable attempts at identifying the degree of importance of Information Entropy for road traffic accident analyses. Dong et al. [10], Lee et al. [11], Hassan and Tazaki [12], Zhang et al. [13], and other researchers have achieved multileveled data mining of traffic accidents by means of a comprehensive application of data mining techniques. The researches above all achieved the mining of accident data on a certain level; however, the overall calculating processes are largely too complicated and cannot be applied

to all types of data, especially the multiattribute ones. On the other hand, the PSO algorithm has been applied in many fields. Shi and Eberhart [14] studied the parameters' optimization, based on particle swarm optimizer. Wang et al. [15] propose an association rules algorithm based on particle swarm optimization algorithm to mining the transaction data in the stock market. Moreover, others [16–20] improved and applied PSO algorithm to their purpose. So far, there have been a lot of researches targeting at different types of data, and due to the “capricious” nature of real-world data, coupled with the innate shortcomings of the algorithm, the association rules still falls short of people's expectations in being less complicated, less time and space-consuming, and more efficient.

In this paper, a new method of traffic accident data mining, based on PSO, association rules, and Information Entropy theories and through a comparative analysis of a variety of traffic accident data mining techniques, is put forward to identify the importance of different attributes and their respective values. The method is an attempt to come up with a multidimensional, all-inclusive method of data analysis to simplify existing algorithms as well as apply computational intelligence algorithms such as PSO to road traffic data analyses.

2. Characteristics of Road Traffic Accident Data

Road traffic accident is under the influence of many factors, which makes it a complicated, and as far as information is concerned, an unfinished, uncertain gray system. There are different databases of traffic accident in different countries [8, 21, 22]. At present, roughly 60 items of information are contained in the China “Database of Road Traffic Accident” which is used by Chinese traffic administrative agencies, spawning off approximately 130 items of single-unit information, which can be used to reconstruct the whole process of the accident in a relatively full and objective manner. It provides more than adequate the information and references for road traffic accident analyses.

Road traffic accidents have their innate, random nature but are also subject to other factors. If the connections of those factors could be identified, through manual control and interference, the rates of traffic accidents could be lower.

Traffic accident data is the foundation of traffic accident analysis, the form and structure of which determine the form and structure of the analysis model. From in-depth analysis of the traffic accident database operated by the Ministry of Public Security, the data could be regards as a radial, multidimensional, and multilayered structure, as shown in Figure 1.

The structure of the data determines the structure of the causation-analysis model. This paper designs a double-layered analysis model and provides an improved algorithm according to the hyperstereo structure of the data. The purpose is to analyze the importance of each value on the attribute value layer with the association rules method and to compare the importance of each attribute on the attribute value layer with the Information Entropy method.

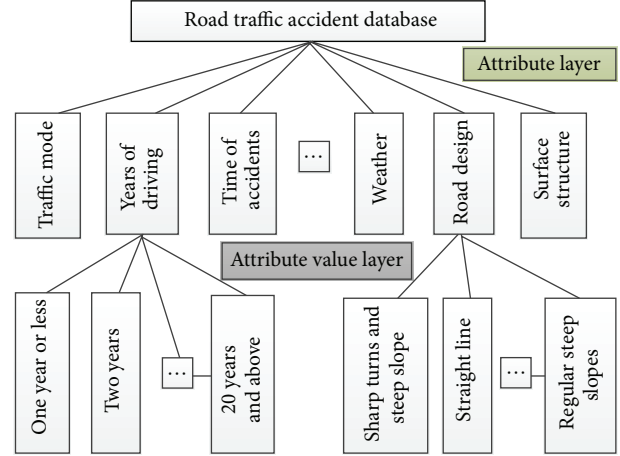


FIGURE 1: Structure of the traffic accidents data.

3. Characteristics of Data Analysis Algorithms

3.1. Association Rules. Association rules is a data mining method for investigating the associative property of different events, which can be used in traffic accident data mining to mine the importance of attributes, that is, the associative relationship of events with certain types of accident. Its basic idea is to treat each characteristic as an item. Accident site, number of death, and so on can all be called an item. The higher the association, the more likely one event is directly linked to the cause of a certain type of accident.

To decide how related two items are, we need to identify how many times some characteristics appear at the same time in a large number of similar events. If items show up at the same time frequently, indicating that there is a statistic pattern behind it, we can start to believe that the items are relevant.

In the association rules algorithm, X and Y are two random events, which can be thought as relevant and also can be thought as seemingly irrelevant. Assume there is a causal relationship between X and Y , which means when X happens, Y also happens (and vice versa). “ $X \rightarrow Y$ ” is used to indicate this relationship, while “support” and “confidence” are used to measure the degree of it.

Agrawal et al. first put forward the problem of mining the association rules of datasets in consumer transaction databases and designed a simple algorithm [7, 23], the basic idea of which is based on the recursive method of frequent set theory. The classical frequent set algorithm is as follows.

- (1) The basic idea. First, find out all the frequent sets and create the association rules from them.
- (2) The procedure. The Apriori algorithm [23] is originated by Agrawal in 1994, the basic idea of which is to scan the database repeatedly. A brief description of the essential program is as follows:

$L_1 = \{\text{large 1-itemsets}\};$

for $(k = 2; L_{k-1} \neq \emptyset; k++)$ do begin

$C_k = \text{apriori_gen}(L_{k-1});$ //New candidate set

```

for all transactions  $t \in D$  do begin
 $C_t = \text{subset}(C_{k,t});$  //candidate sets that contained in
event  $t$  do
for all candidates  $c \in C_t$ 
 $c.\text{count}++;$ 
end
 $L_k = \{c \in C_k \mid c.\text{count} \geq \text{min\_sup}\}$ 
end
Answer =  $\bigcup_k L_k$ .

```

But the main drawback of the Apriori algorithm is that it produces a large number of candidate sets during the computation process and the database may need to be scanned once again. In order to avoid the repetitiveness, an easier method must be developed.

3.2. PSO Algorithm. PSO was originated by Kennedy and Kberhart [15] and was intended for simulating social behavior, which has many advantages such as higher convergence rates and being more applicable than other algorithms. However, it generally has a lower accuracy than genetic algorithms and under certain initial conditions it can only reach optima in a subset of the problem.

PSO is a population-based search algorithm and is initialized with a population of random solutions, called particles. Unlike in the other evolutionary computation techniques, each particle in PSO is also associated with a velocity. Particles fly through the search space with velocities which are dynamically adjusted according to their historical behaviors. Therefore, the particles have the tendency to fly towards the better search area over the course of search process. The PSO was first designed to simulate birds seeking food which is defined as a “cornfield vector.”

3.3. Definition of Information Entropy. Traffic accidents are events with strong randomness, while entropy is the mathematical method for analyzing an event’s uncertainty. From the perspective of statistical mathematics, entropy is a measure of system randomness. Information Entropy is a value for characterizing the statistical characteristics of random variables. It is a measure of the average uncertainty of random variables, an objective description of statistical characteristics of the population.

In the traffic accident data gathering process, due to the influence and limitations from many factors, the number of traffic accident data items is usually insufficient, which cannot be used to analyze the statistical characteristics. The use of Information Entropy as a statistical measure of the uncertainty information does not require the probability distribution of the data to be known and does not require the distribution to be single humped; that is, no prior information is needed. So Information Entropy is very suitable for testing the degree of discreteness of the population. As far as Information Entropy is concerned, the decrease of uncertainty means the reduction of entropy value.

According to the definition of Information Entropy by Shannon [13],

$$H = - \sum_{k=1}^m p_k \log_2 p_k, \quad (1)$$

where m is the dimension of state space and p_k is the probability of the k th state.

If n^k is the number of times when k happens, and the number of samples is N , then $p_k = n_k/N$.

Information Entropy is a measure of the degree of how sequenced a system is. Data measuring with Information Entropy does not require the probability distribution to be known, and the probability distribution of the data does not have to be in a single-hump shape; therefore, it is suitable for examining the discrete level of the distribution. The value indicates how certain factor is affecting the system. The lower the value, the more important factor in the system. Applying the Information Entropy theory to traffic accident, when the attribute spreads evenly across the distribution, indicates that the accident depends weakly on that attribute, which suggests that the importance of the accident is lower.

4. The Method of Traffic Accident Data Preprocessing

Due to human factors, in the actual accident data gathering process, some data items may be unavailable [10], affecting the integrity of the data and causing the results of the causation analysis to be considered not convincing. Data preprocessing is an essential step in any of the data mining processes. Researches on data preprocessing techniques mainly focus on the preprocessing of data which follows obvious patterns, the widely used method of which is to find the patterns, characteristics, and properties so that data can be preprocessed in a certain way. In contrast, it is rarely seen that data in nondigital form is processed in the same manner. The process includes the following parts: data washing, data filling, data integration, and data transformation.

- (1) This paper adopts the data-washing methodology according to the characteristics of traffic accidents to build a traffic accident database using a form which resembles how “antivirus” software works, which has an “antivirus” definition library of its own, that is, an error database that is artificially created, and to predefine a set of rules for the “software” to use in order to hunt down all the “viruses.” However, before the data-washing can be initiated, a database of all the errors needs to be created. The error database comprises a gathering of area-specific knowledge and common sense, through which data that contains errors are marked through comparison. Of course, the data washing only could correct and delete the error data which is detected in logic, not all error types.
- (2) Due to the fact that data format and structure in data mining are not entirely identical with those in the database, data needs to be transformed before it

can be mined, so that existing data can be changed into proper format or form to be mined through data mining techniques. The current data inside the traffic accident database happens to be coded; only part of the statistical attributes is in coded format which is required by the association rules; therefore, the data in the database should be kept as close to their original form as possible. Attributes whose attribute values are not coded but sequential ought to be scattered to a certain extend and coded in orders (seen in Table 1).

5. The Improved Algorithm in Traffic Accident Data Mining

5.1. Fundamental Principle of the Algorithm. Focusing on the characteristics of road traffic accident data, causation analyses need to examine the double-leveled structure. Unfortunately, although they can analyze the causes of accident from different angles and each method has its advantages, none of the data mining methods currently being widely used can accomplish an overall, multiangled, multilayered data mining task on its own.

With the accumulation of traffic accidents database, the data quantity is more and more huge, so how to obtain the effective knowledge, hiding rules, and fundamental causes is changing into one key issue for road traffic administration.

To meet the demand of better accuracy and efficient analysis of traffic accident causes, this paper combines the binary PSO algorithm to improve the association rules. The reason is that the speed of the PSO algorithm does not decrease with the increase of the number of datasets. To solve the problem that accident data needs to be analyzed in different layers, this paper introduces the Information Entropy theory into road traffic accident analysis, with the help of the Association Rules theory, and puts forward the concept of Association Entropy and its algorithm.

With the introduction of PSO and Association Entropy, traffic accident causes can be analyzed from all angles and on all layers, satisfying the requirement that the association rules have to be within a certain support level. In the meanwhile, causes on different levels can provide references for different traffic administrations at different levels, so that more effective preventative measures can be taken.

5.2. Importance of Traffic Accident Attribute Value. Conducting data mining with association rules, first large item sets must be found from the original information datasets. Later, the association rules are made up of all the large item sets. It is required that the frequency of any item must be greater than the \min_sup , otherwise the item is considered not common enough because it falls short of the frequency requirement, which may render it meaningless. Meanwhile, the rules calculated from the large item sets must be greater than the minimum confidence. Otherwise the results from the rule cannot be trusted. However, due to the fact that the information in the database does not include all types of traffic accident information, the ratios of the samples themselves do not match reality. Therefore, traditional data

TABLE 1: Coding schedule of attributes on road condition.

The name of attributes	Coding of association rule	
	Attribute value	Coding
Road surface condition	Intact	1
	In construction	2
	Concave-convex	3
	Collapsed	4
	Roadblock	5
	Others	6
Road physical separation	Not separated	1
	Median separated (1)	2
	Separated between vehicles and nonvehicles (2)	3
	(1) and (2)	4
Road alignment	Straight line	1
	Common turn	21
	Sharp turn	22
	Common slope	31
	Steep slope	32
	Continuous downward slope	33
	Regular slope	41
	Sharp turn and steep slope	42
	Regular slope turn	51
	Regular turn slope	52
Roadside safeguarding types	Guard rail	1
	Barrier wall	2
	Barrier barrel	3
	Others	4
	None	5

mining methods often lead to large sample volume, that is, high importance mistakes. To solve this problem, this paper made some modifications to the traditional methods, as shown in the following.

- (1) According to the Association Rules theory, it is when confidence level reaches the minimum confidence \min_conf and support level reaches the minimum threshold \min_sup that the importance of attribute value layer starts to make sense. The function for calculating the importance of the association rules is

$$c_{ijk}^z = \begin{cases} \frac{s_{ijk}^g}{s_{ij}^t} = \frac{B_{ijk}/A}{D_{ij}/A} = \frac{B_{ijk}}{D_{ijk}}, & s_{ij}^t > \min_sup_1, s_{ijk}^g > \min_sup_2 \\ 0, & \text{others,} \end{cases} \quad (2)$$

where c_{ijk}^z is the attribute i, j 's value of importance; s_{ijk}^g is the attribute i, j 's support on k level of traffic accident severity; s_{ij}^t is the attribute i, j 's support; B_{ijk} is the entry number of i and j on k level in the database; D_{ij} is the entry number in the database that contains I and j ; A is the entry number in the database; \min_sup_1 is the minimum support of i and j ; \min_sup_2 is the minimum support of i and j on k level;

The function for conditional support is

$$s_{ij}^t = \begin{cases} \frac{D_{ij}}{A}, & \frac{D_{ij}}{A} > \min_sup_1 \\ 0, & \text{others.} \end{cases} \quad (3)$$

In order to tell the severity, traffic accidents are graded according to the number of deaths. Therefore, the function for calculating the association rule support is

$$s_{ijk}^g = \begin{cases} \frac{B_{ijk}}{A}, & \frac{B_{ijk}}{A} > \min_sup_2 \\ 0, & \text{others.} \end{cases} \quad (4)$$

- (2) In the traffic accident database, the following attributes are used to describe the severity of the accident: number of deaths, number of injuries, and property loss. Through surveys on experts, this model takes number of death as the determining attribute to indicate severity. The group of experts consists of 4 researchers of traffic safety at university, 4 staffs of traffic management, and 4 policemen of traffic accidents, who are at least 5 years of work experience. In order to make sure that the attribute value meets the requirement of support level, the model divides traffic accident into different levels based on the graded idea. It used the Delphi method to calculate the grade standards and the degree of importance of traffic accident p^k , shown in Table 2.

According to the calculation method of association rules, with reference of the importance value in Table 2, the importance of attribute j with respect to severity is calculated as such:

$$\rho^j = \sum_{k=1}^4 p_i^k \cdot c_{ijk}^z - 1, \quad (5)$$

where p^k is the relative degree of importance with regard to the k th severity level; ρ^j is the attribute j 's degree of importance with regard to accident severity under the association rules.

As shown in the above models, in the calculations of attribute value importance according to the association rules, it has the advantage of being able to effectively shield out certain attribute values of high importance because they have higher frequencies. It is a method of quantifying severity

TABLE 2: Graded severity of accident and importance.

Grade	Set	Importance p^i
1	[0, 1)	1
2	[1, 3)	2
3	[3, 10)	6
4	[10, ∞)	24

importance of traffic accidents from single datum under the requirement of confident level.

In the calculation results of association rule, the value of degree of importance with regard to accident severity, the association of attributes with consequence of traffic accident, can be used to indicate the relationship of attributes and the causes of traffic accident; that is, factors with higher value of importance are the major causes in traffic accidents.

5.3. The Application of PSO Algorithm. Assume there are N particles, and each individual is treated as a volume-less particle (a point) in the D -dimensional search space. The speed, location, individual best position, and swarm best position of the i th particle at " t " moment are represented as $v_i(t)$, $x_i(t)$, $p_i(t)$, and $g_i(t)$, respectively, from which we can have the following recursive equations of the speed and location of the i th particle:

$$\begin{aligned} v_{id}(t+1) &= w \cdot v_{id}(t) + c_1 r_1 (p_{id}(t) - x_{id}(t)) \\ &\quad + c_2 r_2 (g_d(t) - x_{id}(t)), \\ v_{id}(t+1) &= \begin{cases} 1, & r_3 < \text{Sig}(v_{id}(t+1)) \\ 0, & r_3 \geq \text{Sig}(v_{id}(t+1)) \end{cases} \end{aligned} \quad (6)$$

while $i = 1, \dots, N$, N is the size of the group, usually assigned 20; $d = 1, \dots, D$, represents the number of dimensions of each individual, which is decided by specific problems; r_1, r_2, r_3 are random numbers within the range (0,1); c_1, c_2 are two learning factors, usually given as $c_1 = c_2 = 2$; w is greater or equal to 0 and is called the inertia factor, usually given the value of 1; $\text{Sig}(\cdot)$ is the sigmoid function; that is, $\text{Sig}(x) = 1/(1 + \exp(-x))$. The X_{id} of each particle in the search space is given the value of 0 or 1; $v_{id} \in [-v_{\max}, v_{\max}]$ is the probability of X_{id} equals 1; hence, when $\text{Sig}(v_{id}) = 0$, X_{id} equals 0.

Because $\text{Sig}(x) = 1/(1 + \exp(-x))$ when x is from the range $[-10, 10]$, we usually has a result within the range $[0, 1]$; therefore, the maximum speed v_{\max} is usually less than 10.

With PSO, determining attributes of traffic accident can be sequenced to form a series of attributes all at once, no longer needing to change the sequence during the mining process, which conforms to the procedure of discrete binary PSO algorithm.

In the binary space, the moving of particles is done through switching the position values, and the speed of the particle is the change of digits after each recursive calculation. Attribute swarm is for finding out frequent item sets; each individual particle in a swarm is represented by m (m is the number of accident determining attributes) digits of 1's and 0's. And in the determining attribute swarm, each digit of

the particle is used to indicate whether the corresponding attributes appear or not; 1 means the attribute appears and 0 not.

The adaptability function is for measuring the quality of particles' rule sets. In the competition of all the rules, only the ones with high confidence and credibility may survive.

Construct a series of rule structures with all the determining and task attributes in the form of $\{B_1, B_2, \dots, B_m, A_1, A_2, \dots, A_n\}$, where A_i represents determining attributes, which is the severity of traffic accidents and B_i is task attributes, which are driving years, weather, and so forth.

The rule support is

$$\text{Sup}(R) = \text{Cover}(A + B) \quad (7)$$

while R is the rule; $\text{Cover}(A + B)$ is the ratio of two events in a database.

The rule confidence is

$$\text{Conf}(R) = \frac{\text{cover}(A + B)}{\text{cover}} = \frac{\text{Sup}(R)}{\text{Sup}(R_A)} \quad (8)$$

while R_A represents the datasets in R that match the attributes of rule R .

Then the adaptability function of the swarm is defined as such: $\text{Fitness} = a \text{Sup}(R) + b \text{Conf}(R)$, while a and b are constants, and $0 \leq a \leq 1, 0 \leq b \leq 1$; the value of a and b can be adjusted according to specific problems.

5.4. Calculation of Accident Causation Attributes Association Entropy. In the calculation of Association Entropy, using each attribute value's association rule importance as the source of information analyzes the entropy to compare the importance between different attributes. Borrowing from the calculation method of Information Entropy, the Association Entropy of traffic accident is calculated as follows:

$$\tilde{H} = \frac{H}{H_{\max}} \quad (9)$$

while H is the Association Entropy of attributes; H_{\max} is the maximum association entropy value from m traffic accident attributes; According to the definition of Shannon entropy [4, 8, 10],

$$H = - \sum_{k=1}^m p_k \log_2 p_k, \quad (10)$$

where m is the attribute number under attribute i 's values; p_{ij} is the degree of importance probability under attribute j ; p_{ij} is calculated as follows:

$$p_{ij} = \frac{\rho_{ij}}{\sum_{j=1}^m \rho_{ij}} \quad (11)$$

while ρ_{ij} is the association rule importance of attribute j 's relative determinative attribute; ($j = 1, 2, \dots, m$).

From (10) and (11), association importance probability reaches maximum entropy value when distributed according

to importance. And the maximum entropy value is calculated as follows:

$$\begin{aligned} H_{\max} &= - \sum_{i=1}^m p_i \log_2 p_i = - \sum_{k=1}^m \frac{1}{m} \log_2 \left(\frac{1}{m} \right) \\ &= -m \frac{1}{m} \log_2 \left(\frac{1}{m} \right) = \log_2 m. \end{aligned} \quad (12)$$

Carry (10) and (12) into (9), the entropy value of traffic accident attribute importance is calculated as such:

$$\tilde{H} = \frac{H}{H_{\max}} = - \frac{\sum_{j=1}^m p_{ij} \log_2 p_{ij}}{\log_2 m}. \quad (13)$$

Sequence the results according to entropy value. The larger the entropy, the more evenly distributed the association among all the attributes and the more uncertain the result; that is, the lower the importance of traffic accident attribute, and in contrast, the higher the importance of accident attributes.

6. Application and Validation of the Algorithm

6.1. Sample Data Declaration. The sample data sets in the model and algorithm validation of the paper are more than twenty thousand in total and each of them contains 56 accident attributes. And all of them are randomly collected from traffic accident database, which come from traffic accident data in Northeast China, North China, East China, South Central China, Southwest China, and Northwest China based on equivalent sampling principle, so that it can reflect the whole picture of road traffic accident nationwide better.

6.2. Validation Method. It is needed to know the computation result and standard answer of a method to test and verify the accuracy of the new model method and to compare and analyze the data. While in the validation of the traffic accident factor analysis model, because the standard answer is unavailable, accurate analysis cannot be implemented using the traditional method. But we can use another completely different and widely accepted method to carry on the analysis for the same target and then use the consistency of two kinds of methods to verify the relative accuracy of the existing model calculation results. Therefore, the Delphi method is used to compare the computation result and standard answer exactly, which is a common method in testing uncertain event. We can design expert experience questionnaire by using Delphi method to get the attribute value of pavement condition and the importance weight of road traffic accidents attribute through the way of expert experience questionnaire.

The degree of importance of attribute and attribute value are calculated through different methods in the model of the paper, so it will be conducted into two parts when analyzing the results. First, test and verify the part of attribute value which is calculated through improved association rules.

- (1) For attribute value, use formulae (1) and (2) to conduct support test to the data. In consideration of the sample size in the model, we have $\min_sup_1 = 0.0001$ and $\min_sup_2 = 0.005$.

TABLE 3: Graded severity of accident in relation to importance.

Attribute value	The average weight by expert	The degree of importance of associated regulation	The checking result $\alpha = 0.025$ $t = 2.3646$
Common bending	0.163	0.083	Zero difference
Sharp turn	0.138	0.099	Zero difference
Common bending slope	0.053	0.145	Different
Sharp turn abrupt slope	0.159	0.151	Zero difference
Common slope sharp turn	0.102	0.171	Zero difference
Common bending abrupt slope	0.052	0.351	Zero difference

TABLE 4: Part of road traffic accidents attributes importance for comparative table.

Attribute	The average weight by expert	The degree of importance of associated regulation	The checking result $\alpha = 0.025$ $t = 2.3646$
Driving years	0.048	0.206	Zero difference
Road alignment	0.177	0.196	Zero difference
Weather	0.109	0.202	Zero difference
The safeguard types on roadside	0.071	0.204	Zero difference
Vehicle safety condition	0.196	0.193	Zero difference

- (2) Use formulae (3) and (4) to calculate the importance of attribute value and make normalization.
- (3) Use the degree of importance of attribute value as the source of information and use formulae (10) and (12) to analyze the entropy value of the attribute.

Because of the complexity of the factors of traffic accident, in data mining process if we want to get the result smoothly, we should consider the calculation results of multigroup equal precision sample and then get their average value. And the more the sample groups, the closer the final average results to the truth. So, in order to test and verify the accuracy of the computing method of the model, make random sampling analysis ten times for the data and conduct check consistency with matching t -test for the model calculating results and Delphi method results.

6.3. Calculation Results and Verification Analysis. By applying the algorithm in Sections 5.2 and 5.3 and then using the verification method in Section 6.2, the final results are listed as follows.

From Table 3 we can see that general bend and sharp bend steep slopes have the most influence on accident severity, respectively, 0.163 and 0.159. The main reason is although the former alignment condition is good, but driver has a lower safety cognitive level, often because over speed driving led to accident, the later alignment condition and road side security level are relatively low; once a vehicle loses control in this section, the result must be serious. However, real traffic accident analysis result showed that, except for general bend slope factor, other road sections have no obvious difference to the influence on accident severity. This shows that the expert's subjective impression not necessarily represents the truth. In the meantime, the accident influencing factors' absolute value

of importance from association rules algorithm in different in numbers, but before significance check, also cannot represent that it really has difference between accident influence factor.

Same as the result shown in Table 3, as far as people are concerned, it is clearly different how years of driving, road geometrics, weather, and so forth affect traffic accident severity; however, the result of data mining does not show the same degree of obviousness as how the corresponding factors differ, which suggests the limitedness of people's experiences; for example, years of driving is indeed one of the major factors triggering accidents, but that does not mean relatively inexperienced drivers tend to have more severe accidents than more experienced drivers.

According to the results in Tables 3 and 4, most of the data of importance in the two methods have no difference from a statistical point of view, and only the importance of common bending slope has difference. The reason may come from the difference of the data between the experts' judgment and the model use which are limited in sample size. But in general, the importance from the two methods has almost no difference, and the relative precision of the computing results from the method can meet requirements.

7. Conclusion

This paper aims at the hierarchically structured characteristics of road traffic accident databases, mixed using the method of associated rules, PSO, and Information Entropy to analyze the degree of importance of traffic accidents. Through a modification of traditional methodologies and algorithms, a PSO algorithm and associated entropy model are built for calculating the degree of importance of road accidents. Through applying the improved algorithm on

both the attribute and the attribute value layers, respectively, each accident-triggering factor's influence on the severity of accident is calculated. The algorithm this paper introduced has the advantage of better accuracy and higher mining rates over the traditional association rules and PSO algorithms, the result of which is quite different from what the experts concluded, which indicates when facing a large amount of random information, people's experiences and how people perceive things are limited. Of course, traffic accident data as a type of data possesses certain physical meanings—whether there really exist connections between certain types of data, and that certain types of data were manually gathered so they may not be error-free, as well as whether they can be fully applied to the models and algorithms of this paper in their entirety questions as those that are yet to be discussed in future researches. However, approved by real applications and tests of effectiveness, this type of data mining method which is based on traffic accident database provides yet another powerful tool to quantify data in traffic accident analysis, which is going to be helpful to accident experts and traffic administrative agencies to clarify how much of role different factors play in investigations of traffic severity.

Acknowledgment

This project is supported by NSFC (Grant no. 50808093).

References

- [1] H. Nabi, L. R. Salmi, S. Lafont, M. Chiron, M. Zins, and E. Lagarde, "Attitudes associated with behavioral predictors of serious road traffic crashes: results from the GAZEL cohort," *Injury Prevention*, vol. 13, no. 1, pp. 26–31, 2007.
- [2] E. R. Green, K. R. Agent, and J. G. Pigman, "Evaluation of auto incident recording system (AIRS)," Tech. Rep. KTC-05-09/SPR277-04-1F, Kentucky Transportation Center, 2005.
- [3] M. Hirasawa, "Development of traffic accident analysis system using GIS," *Proceedings of the Eastern Asia Society for Transportation Studies*, vol. 10, no. 4, pp. 1193–1198, 2005.
- [4] B. Yu, W. H. K. Lam, and M. L. Tam, "Bus arrival time prediction at bus stop with multiple routes," *Transportation Research Part C*, vol. 19, no. 6, pp. 1157–1170, 2011.
- [5] B. Yu, Z. Yang, and S. Li, "Real-time partway deadheading strategy based on transit service reliability assessment," *Transportation Research Part A*, vol. 46, no. 8, pp. 1265–1279, 2012.
- [6] B. Yu, Z. Z. Yang, P.-H. Jin, S.-H. Wu, and B. Z. Yao, "Transit route network design using ant colony optimization," *Transportation Research Part C*, vol. 22, pp. 58–75, 2012.
- [7] K. Geurts, G. Wets, T. Brijs, and K. Vanhoof, "Profiling of high-frequency accident locations by use of association rules," *Journal of the Transportation Research Board*, no. 1840, pp. 123–130, 2003.
- [8] T. Tesema, A. Abraham, and C. Grosan, "Rule mining and classification of road traffic accidents using adaptive regression trees. I," *Journal of Simulation*, vol. 6, no. 10, pp. 80–94, 2005.
- [9] R. Marukatat, "Structure-based rule selection framework for association rule mining of traffic accident data," in *Computational Intelligence and Security*, vol. 4456, pp. 231–239, 2007.
- [10] L.-Y. Dong, G.-Y. Liu, S.-M. Yuan, Y.-L. Li, and Z.-H. Wu, "Application of data mining to traffic accidents analysis," *Journal of Jilin University Science Edition*, vol. 44, no. 6, pp. 951–955, 2006.
- [11] D.-H. Lee, S.-T. Jeng, and P. Chandrasekar, "Applying data mining techniques for traffic incident analysis," *Journal of the Institution of Engineers*, vol. 44, no. 2, pp. 90–101, 2004.
- [12] Y. Hassan and E. Tazaki, "Emergence decision using hybrid rough sets/cellular automata," *Kybernetes*, vol. 35, no. 6, pp. 797–813, 2006.
- [13] Y. Zhang, F. Wang, J. Dai, W. Huang, and Y. Chen, "New comprehensive evaluation algorithm based on fuzzy clustering and information entropy," *Journal of Changchun Post and Telecommunication Institute*, vol. 22, no. 6, pp. 643–647, 2004.
- [14] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, May 1998.
- [15] X. Wang, X. Liu, and F. Dai, "Method and application of mining association rules based on PSO algorithm," *Information Technology and Informatization*, vol. 3, pp. 85–87, 2009.
- [16] Q. K. Pan, M. F. Tasgetiren, and Y. C. Liang, "A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem," *Computers and Operations Research*, vol. 35, no. 9, pp. 2807–2839, 2008.
- [17] T. J. Ai and V. Kachitvichyanukul, "A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery," *Computers and Operations Research*, vol. 36, no. 5, pp. 1693–1702, 2009.
- [18] P. Pongchairerks and V. Kachitvichyanukul, "A particle swarm optimization algorithm on job-shop scheduling problems with multi-purpose machines," *Asia-Pacific Journal of Operational Research*, vol. 26, no. 2, pp. 161–184, 2009.
- [19] A. W. Mohemmed, N. C. Sahoo, and T. K. Geok, "Solving shortest path problem using particle swarm optimization," *Applied Soft Computing Journal*, vol. 8, no. 4, pp. 1643–1653, 2008.
- [20] M. Lovbjerg, T. K. Rasmussen, and T. Krink, "Hybrid particle swarm optimizer with breeding and subpopulations," *Institute of Electrical and Electronics Engineers*, no. 7, pp. 115–118, 2001.
- [21] I. B. Gundogdu, F. Sari, and O. Esen, "A new approach for geographical information system-supported mapping of traffic accident data," *TSIB-GIS in Urban Planning and Management*, 6:03–11, 2008.
- [22] ESTC, *EU Transport Accident, Incident and Casualty Databases: Current Status and Future Needs*, European Transport Safety Council, 2001.
- [23] A. Savasere, E. Omiecinski, and S. Navathe, "An efficient algorithm for mining association rules in large databases," in *Proceedings of the 21th International Conference on Very Large Database (VLDB '95)*, pp. 432–443, Zurich, Switzerland, 1995.

Research Article

Improved Barebones Particle Swarm Optimization with Neighborhood Search and Its Application on Ship Design

Jingzheng Yao and Duanfeng Han

College of Shipbuilding Engineering, Harbin Engineering University, Harbin 150001, China

Correspondence should be addressed to Jingzheng Yao; yaojz_113@126.com

Received 30 August 2012; Revised 17 November 2012; Accepted 25 November 2012

Academic Editor: Baozhen Yao

Copyright © 2013 J. Yao and D. Han. 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.

Barebones particle swarm optimization (BPSO) is a new PSO variant, which has shown a good performance on many optimization problems. However, similar to the standard PSO, BPSO also suffers from premature convergence when solving complex optimization problems. In order to improve the performance of BPSO, this paper proposes a new BPSO variant called BPSO with neighborhood search (NSBPSO) to achieve a tradeoff between exploration and exploitation during the search process. Experiments are conducted on twelve benchmark functions and a real-world problem of ship design. Simulation results demonstrate that our approach outperforms the standard PSO, BPSO, and six other improved PSO algorithms.

1. Introduction

Particle swarm optimization (PSO), developed by Kennedy and Eberhart [1], is a new optimization technique inspired by swarm intelligence. Like other evolutionary algorithms (EAs), PSO is also a population-based stochastic search algorithm, but it does not contain any crossover or mutation operator. During the search process, each particle adjusts its search behavior according to the search experiences of its previous best position (p_{best}) and the global best position (g_{best}). Due to its simplicity and easy implementation, PSO has been successfully applied to various practical optimization problems [2–5].

However, like other stochastic algorithms, PSO also suffers from premature convergence when handling complex multimodal problems. The main reason is that the attraction search pattern of PSO greatly depends on p_{best} and g_{best} . Once these best particles (p_{best} and g_{best}) get stuck, all particles in the swarm will quickly converge to the trapped position. In order to enhance the performance of PSO, different versions of PSO have been proposed in the past decades. Shi and Eberhart [6] introduced an inertia weight w into the original PSO to achieve a balance between the

global and local search. Reported results show that a linearly decreased w is a parameter setting. Parsopoulos and Vrahaitis [7] proposed a unified PSO (UPSO) which is a hybrid algorithm by combining the global and local versions of PSO. In [8], a fully informed PSO, called FIPS, is proposed by employing a modified velocity model. van den Bergh and Engelbrecht [9] used a cooperative mechanism to improve the performance of PSO on multimodal optimization problems. In the standard PSO, particles are attracted by their corresponding previous best particles and the global best particle. This search pattern is a greedy method which may result in premature convergence. To tackle this problem, Liang et al. [10] proposed a comprehensive learning PSO (CLPSO), in which each particle can be attracted by different previous best positions. Computational results on a set of multimodal problems demonstrate the effectiveness of CLPSO. In [11], Wang et al. proposed a new PSO algorithm (NSPSO) to search the neighbors of particles. This can provide more chances of finding better candidate solutions. Simulation studies show that NSPSO outperforms UPSO, FIPS, CPSO-H, and CLPSO. In [12], another version of NSPSO is proposed by employing neighborhood search and diversity enhanced mechanism.

Similar to other EAs, the performance of PSO also greatly depends on its control parameters, w , c_1 , and c_2 . The first parameter is known as inertia weight, and the last two are acceleration coefficients. Slight differences of these parameters may result in significantly different performance. To tackle this problem, some PSO variants based on adaptive parameters have been proposed to minimize the dependency of these parameters [13, 14]. Compared to these adaptive PSO algorithms, Kennedy developed a novel PSO called barebones PSO (BPSO) [15], which eliminates the velocity term and does not contain the parameters w , c_1 , and c_2 . In BPSO, a Gaussian sampling is used to generate new positions of particles. Empirical studies demonstrate that the performance of BPSO is competitive to the standard PSO and some improved PSO algorithms. Inspired by the idea of BPSO, some new algorithms have been proposed. In [16], Omran et al. combined BPSO with differential evolution (DE) and the proposed barebones DE (BBDE). The reported results show that BBDE outperforms the standard DE and BPSO and it also achieves promising solutions for unsupervised image classification. Krohling and Mendel [17] introduced Gaussian and Cauchy mutations into BPSO to improve its performance. Experimental studies on a suite of well-known multimodal benchmark functions demonstrate the effectiveness of this approach. Blackwell [18] presented a theoretical analysis of BPSO. A series of experimental trials confirmed that the BPSO situated at the edge of collapse is comparable to other PSO algorithms and that performance can be still further improved with the use of an adaptive distribution. In [19], Wang embedded opposition-based learning (OBL) into BPSO to solve constrained nonlinear optimization problems. In addition, a new boundary search strategy is utilized. Simulation studies on thirteen constrained benchmark functions show that the new approach outperforms PSO, BPSO, and six other improved PSO algorithms.

In this paper, we also propose an improved barebones PSO called NSBPSO which employs a global and local neighborhood search strategies to make a balance between exploration and exploitation during the search process. In order to verify the performance of NSBPSO, twelve well-known benchmark functions and a real-world problem on ship design are used in the experiments. Computational results show that our approach outperforms PSO, BPSO, and several other improved PSO variants in terms of the quality of solutions.

The rest of the paper is organized as follows. The standard PSO and barebones PSO are given in Section 2. In Section 3, our approach NSBPSO is proposed. Experimental studies are presented in Section 4. Section 5 presents a real-world application on ship design. Finally, the work is summarized in Section 6.

2. Barebones Particle Swarm Optimization

PSO is a population-based stochastic search algorithm, which simulates the behaviors of fish schooling or birds flocking. Each particle has a velocity and a position vectors. During

the search space, a particle dynamically adjusts its velocity to generate a new position as follows:

$$\begin{aligned} V_i(t+1) &= w \cdot V_i(t) + c_1 \cdot r_1 \cdot (p_{\text{best}_i} - X_i(t)) \\ &\quad + c_2 \cdot r_2 \cdot (g_{\text{best}} - X_i(t)), \\ X_i(t+1) &= X_i(t) + V_i(t+1), \end{aligned} \quad (1)$$

where X_i and V_i are the position and velocity vector for the i th particle, respectively. p_{best_i} is the previous best particle of the i th particle and g_{best} is the global best particle. r_1 and r_2 are two independently generated random numbers within $[0, 1]$. The parameter w is known as inertia weight. c_1 and c_2 are acceleration coefficients.

A recent study [20] proved that the particles in PSO converge to the weighted position of p_{best} and g_{best} as follows:

$$\lim_{t \rightarrow +\infty} X_i(t) = \frac{c_1 \cdot p_{\text{best}_i} + c_2 \cdot g_{\text{best}}}{c_1 + c_2}. \quad (2)$$

Based on the convergence characteristic of PSO, Kennedy [15] proposed a new PSO variant called barebones PSO (BPSO), in which each particle only has a position vector and eliminates the velocity vector. Therefore, BPSO does not contain the parameters w , c_1 , and c_2 . In BPSO, a new position is updated by Gaussian sampling as follows:

$$X_i(t+1) = N\left(\frac{g_{\text{best}} + p_{\text{best}_i}}{2}, |g_{\text{best}} - p_{\text{best}_i}|\right), \quad (3)$$

where $N(\cdot)$ indicates a Gaussian distribution with mean $(g_{\text{best}} + p_{\text{best}_i})/2$ and standard deviation $|g_{\text{best}} - p_{\text{best}_i}|$.

3. Barebones PSO with Neighborhood Search

Due to the intrinsic randomness, both PSO and EAs suffer from premature convergence when solving complex multimodal problems. Sometimes, the suboptima are near to the global optimum and the neighborhoods of trapped particles may contain the global optimum. For this case, searching the neighbors of particles is beneficial for finding better solutions. Based on this idea, some neighborhood search strategies have been successfully applied to various algorithms.

In [11], Wang et al. proposed a new PSO algorithm called PSO with neighborhood search strategies (NSPSO), which utilizes one local and two global neighborhood search strategies. The NSPSO includes two operations. First, for each particle, three trial particles are generated by the above neighborhood search strategies, respectively. Second, the best one among the three trial particles and the current particle is chosen as the new current particle. Simulation studies on twelve unimodal and multimodal benchmark problems show that NSPSO achieves better results than standard PSO and five other improved PSO algorithms.

Although NSPSO has shown good search abilities, its performance is still seriously influenced by its control parameters, w , c_1 , and c_2 . In [11], NSPSO used an empirical parameter settings, $c_1 = c_2 = 1.49618$ and $w = 0.72984$. In order to minimize the effects of the control parameters

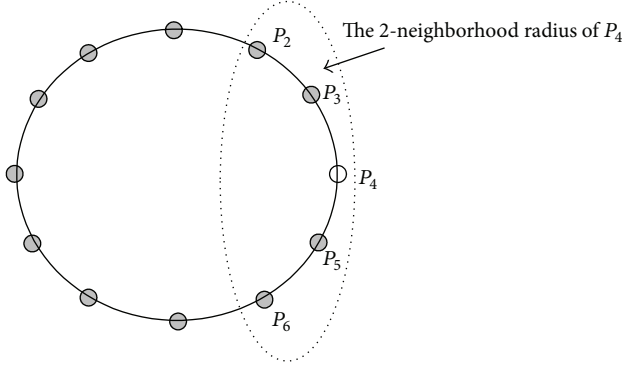


FIGURE 1: The ring topology and 2-neighborhood radius.

on the performance of NSPSO, this paper proposes an improved PSO algorithm by combining barebones PSO and the neighborhood search strategies.

There are various population topologies, such as ring, wheel, star, Von Neumann, and random. A recent study shows that the complexity of population topology affects the performance of PSO. A population topology with few connections (low complexity) may perform well on multimodal problems, while a highly interconnected population topology may perform well on unimodal problems. In this paper, a ring topology is used by the suggestions of [11].

The ring topology assumes that particles are organized as a ring. In [21], a special ring topology is proposed by connecting the indices of particles. For example, the fourth particle P_4 is connected by the third one P_3 and the fifth one P_5 . In other words, P_3 and P_5 are two immediate neighbors of P_4 . Figure 1 shows the employed ring topology. Based on the ring topology, a k -neighborhood radius is defined, where k is a predefined integer number. For each particle P_i , its k -neighborhood radius consists of $2k + 1$ particles (include itself), which are $P_{i-k}, \dots, P_{i-1}, P_i, P_{i+1}, \dots, P_{i+k}$. It is obvious that the parameter k satisfies $0 \leq k \leq (N - 1)/2$. Figure 1 shows the 2-neighborhood radius of P_4 , where 5 particles are covered by the neighborhood. By the suggestions of [11], $k = 2$ is used in this paper.

Based on the k -neighborhood radius, a local neighborhood search strategy is proposed. For each particle P_i , a local particle L_i is generated as follows [11]:

$$LX_i(t+1) = a_1 \cdot X_i(t) + a_2 \cdot p_{\text{best}_i} + a_3 \cdot (X_{i1}(t) - X_{i2}(t)), \quad (4)$$

where X_{i1} and X_{i2} are the position vectors of two particles, P_{i1} and P_{i2} , randomly selected from the k -neighborhood neighborhood, $i1, i2 \in [i - k, i + k] \wedge i1 \neq i2 \neq i$, a_1, a_2, a_3 are three random numbers within $(0, 1)$, and $a_1 + a_2 + a_3 = 1$. In [11], the velocity of L_i keeps the same with P_i . Although the velocity mechanism is simple, it may not be beneficial for the next flight of L_i . Therefore, we use a similar method to generate LV_i :

$$LV_i(t+1) = a_1 \cdot V_i(t) + a_2 \cdot p_{\text{best}_{V_i}} + a_3 \cdot (V_{i1}(t) - V_{i2}(t)), \quad (5)$$

where $p_{\text{best}_{V_i}}$ is the velocity vector of p_{best_i} and V_{i1}, V_{i2} are the velocity vectors of P_{i1} and P_{i2} , respectively.

Beside the local neighbor strategy, a global neighborhood search strategy is proposed. For each particle P_i , a global particle G_i is generated as follows [11]:

$$GX_i(t+1) = a_1 \cdot X_i(t) + a_2 \cdot g_{\text{best}} + a_3 \cdot (X_{i3}(t) - X_{i4}(t)), \quad (6)$$

where X_{i3} and X_{i4} are the position vectors of two particles, P_{i3} and P_{i4} , randomly selected from the current swarm, $i3, i4 \in [1, N] \wedge i3 \neq i4 \neq i$, a_1, a_2, a_3 are three random numbers within $(0, 1)$, and $a_1 + a_2 + a_3 = 1$. In [11], the velocities of G_i keeps the same with P_i . So, the velocity of L_i, G_i , and P_i are the same, but L_i (local) and G_i (global) are two different types of particles. Like (5), this paper uses a new method to generate GV_i :

$$LV_i(t+1) = a_1 \cdot V_i(t) + a_2 \cdot g_{\text{best}_V} + a_3 \cdot (V_{i3}(t) - V_{i4}(t)), \quad (7)$$

where g_{best_V} is the velocity vector of g_{best} and V_{i3}, V_{i4} are the velocity vectors of P_{i3} and P_{i4} , respectively.

After generating two new particles L_i and G_i , a greedy selection mechanism is used. Among P_i, L_i , and G_i , we select the best one as the new P_i .

In our approach NSBPSO, it embeds the local and global neighborhood search strategies into barebones PSO. The neighborhood search strategies focus on searching the neighbors of particles and provide different search behaviors. The BPSO concentrates on minimizing the dependency of the control parameters (without w, c_1 , and c_2). By hybridization of BPSO and the neighborhood strategies, NSBPSO is almost a parameter-free algorithm (except for the probability of the neighborhood search), which achieves a tradeoff between exploration and exploitation.

The main steps of NSBPSO are listed as follows.

Step 1. Randomly initialize the swarm, and evaluate the fitness values of all particles.

Step 2. Initialize p_{best} and g_{best} .

Step 3. For each particle P_i , calculate its new position vector X_i according to (3). Evaluate the fitness value of P_i . If needed, update p_{best_i} and g_{best} .

Step 4. For each particle P_i , if r and $(0, 1) < P_{\text{ns}}$, where r and $(0, 1)$ is a random number within $[0, 1]$ and P_{ns} is the probability of conducting neighborhood search, then go to Step 5; otherwise go to Step 6.

Step 5. Generate a local particle L_i according to (4) and (5). Generate a local particle G_i according to (6) and (7). Evaluate the fitness values of L_i and G_i . Among P_i, L_i , and G_i , we select the best one as the new P_i . If needed, update p_{best_i} and g_{best} .

Step 6. If the stop criterion is satisfied, then stop the algorithm and output the results; otherwise go to Step 3.

TABLE 1: The twelve benchmark problems.

Problems	D	Search range
$f_1(x) = \sum_{i=1}^D x_i^2$	30	$[-100, 100]$
$f_2(x) = \sum_{i=1}^{D-1} [100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2]$	30	$[-2.048, 2.048]$
$f_3(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i)\right) + 20 + e$	30	$[-32.768, 32.768]$
$f_4(x) = \sum_{i=1}^D \frac{x_i^2}{4000} - \prod_{i=1}^D \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]$
$f_5(x) = \sum_{i=1}^D \left(\sum_{k=0}^{k_{\max}} [a^k \cos(2\pi b^k (x_i + 0.5))] \right) - D \sum_{k=0}^{k_{\max}} [a^k \cos(2\pi b^k \cdot 0.5)]$ $a = 0.5, b = 3, k_{\max} = 20$	30	$[-0.5, 0.5]$
$f_6(x) = \sum_{i=1}^D (x_i^2 - 10 \cos(2\pi x_i) + 10)$	30	$[-5.12, 5.12]$
$f_7(x) = \sum_{i=1}^D (y_i^2 - 10 \cos(2\pi y_i) + 10)$ $y_i = \begin{cases} x_i, & x_i < \frac{1}{2} \\ \frac{\text{round}(2x_i)}{2}, & x_i \geq \frac{1}{2} \end{cases}$	30	$[-5.12, 5.12]$
$f_8(x) = 418.9829 \cdot D - \sum_{i=1}^D (x_i \sin(\sqrt{ x_i }))$	30	$[-500, 500]$
$f_9(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D y_i^2}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^D \cos(2\pi y_i)\right) + 20 + e$ $y = \mathbf{M} * x$	30	$[-32.768, 32.768]$
$f_{10}(x) = \sum_{i=1}^D \frac{y_i^2}{4000} - \prod_{i=1}^D \cos\left(\frac{y_i}{\sqrt{i}}\right) + 1$ $y = \mathbf{M} * x$	30	$[-600, 600]$
$f_{11}(x) = \sum_{i=1}^D \left(\sum_{k=0}^{k_{\max}} [a^k \cos(2\pi b^k (y_i + 0.5))] \right) - D \sum_{k=0}^{k_{\max}} [a^k \cos(2\pi b^k \cdot 0.5)]$ $y = \mathbf{M} * x$	30	$[-0.5, 0.5]$
$f_{12}(x) = \sum_{i=1}^D (y_i^2 - 10 \cos(2\pi y_i) + 10)$ $y = \mathbf{M} * x$	30	$[-5.12, 5.12]$

4. Experimental Study

4.1. Test Problems. In order to verify the performance of our approach, there are twelve well-known benchmark problems used in the following experiments [10]. According to the properties of these problems, they are divided into two three types: unimodal problems (f_1 – f_2), unrotated multimodal problems (f_3 – f_8), and rotated multimodal problems (f_9 – f_{12}). For rotated problems, the original variable x is left multiplied by the orthogonal matrix \mathbf{M} to get the new rotated variable $y = \mathbf{M} * x$. For all test problems, they are to be minimized and their global optima are zero. The specific descriptions of these problems are presented in Table 1.

4.2. Effects of the Parameter P_{ns} . The main contribution of this paper is to minimize the effects of the control parameters and improve the performance of BPSO. Although NSBPSO eliminates the control parameters, w , c_1 , and c_2 , it introduces two new parameters k and P_{ns} . The parameter k is the size of neighborhood radius. The ring population topology used in this paper assumes that particles are connected by their indices. Although P_2 and P_3 are two neighbors of P_1 , they may not be the nearest one to P_1 (Euclidean distance). So, the size of the neighborhood radius does not affect the selection of particles in the local neighborhood search. Our empirical studies also confirm it (here we do not list the results of NSBSO with different k -neighborhood radius). According to the suggestions of [11], k is set to 2 in this paper.

TABLE 2: Results achieved by NSBPSO with different P_{ns} .

Problems	$P_{ns} = 0.0$ Mean	$P_{ns} = 0.1$ Mean	$P_{ns} = 0.3$ Mean	$P_{ns} = 0.5$ Mean	$P_{ns} = 0.7$ Mean	$P_{ns} = 1.0$ Mean
f_1	$3.60E - 87$	$1.47E - 285$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$
f_2	$1.84E + 01$	$1.81E + 01$	$2.01E + 01$	$2.08E + 01$	$2.13E + 01$	$2.23E + 01$
f_3	$1.12E - 14$	$5.89E - 16$	$5.89E - 16$	$5.89E - 16$	$5.89E - 16$	$5.89E - 16$
f_4	$9.86E - 03$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$
f_5	$5.65E - 05$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$
f_6	$3.38E + 01$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$
f_7	$2.50E + 01$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$
f_8	$2.13E + 03$	$1.80E + 03$	$1.01E + 03$	$2.78E + 03$	$2.63E + 03$	$4.17E + 03$
f_9	$1.90E + 00$	$5.89E - 16$	$5.89E - 16$	$5.89E - 16$	$5.89E - 16$	$5.89E - 16$
f_{10}	$2.95E - 02$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$
f_{11}	$7.95E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$
f_{12}	$6.07E + 01$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$	$0.00E + 00$

The parameter P_{ns} controls the probability of conducting neighborhood search. A larger P_{ns} will result in more neighborhood search operations, while a smaller P_{ns} will have less. This may affect the performance of NSBPSO. To investigate the effects of P_{ns} , this section presents an experimental study. In the experiment, the P_{ns} is set to 0.0, 0.1, 0.3, 0.5, 0.7, and 1.0, respectively. The performance of NSBPSO with different P_{ns} is compared.

For other parameters of NSBPSO, we use the following settings by the suggestions of [10]. The population size N is set to 40. When the number of fitness evaluations (FEs) reaches the maximum value MAX_FEs, the algorithm stops running. In the experiment, MAX_FEs is set to $2.0e + 05$. For each test problem, NSBPSO is run 30 times and the mean fitness error values are reported.

Table 2 presents the computational results of NSBPSO under different P_{ns} , where “Mean” represents the mean fitness error values. The best results among the comparison are shown in boldface. As seen, the performance of NSBPSO is not sensitive to the parameter P_{ns} . A smaller ($P_{ns} < 0.3$) or larger ($P_{ns} > 0.5$) value of P_{ns} almost achieves similar results. For $P_{ns} = 0.0$, NSBPSO is equal to the original BPSO, because the neighborhood search operations are not conducted. For this case, the algorithm shows poor performance and falls into local minima on most test functions. When $P_{ns} = 0.1$, NSBPSO significantly outperforms NSBPSO with $P_{ns} = 0.0$. It demonstrates that the neighborhood search strategies are very effective. Even if we use a small P_{ns} , NSBPSO can also obtain promising results.

The value of P_{ns} does not affect the performance of NSBPSO, and $P_{ns} > 0$ is applicable for all test problems. In this paper, $P_{ns} = 0.3$ is used in the following experiments.

Figure 2 presents the convergence processes of NSBPSO with different P_{ns} . Although different P_{ns} of NSBPSO can find the global optimum on the majority of test functions, they show different convergence characteristics. For problem f_1 , larger P_{ns} converges faster than smaller P_{ns} . For problem f_2 , $P_{ns} = 0.1$ converges fastest than other values. For f_8 , $P_{ns} = 0.3$ shows the fastest convergence speed.

4.3. Comparison of NSBPSO with Other PSO Algorithms. In this section, experiments are conducted to compare nine PSO algorithms including the proposed NSBPSO on the 12 test problems. The involved algorithms are listed as follows.

- (1) standard PSO,
- (2) barebones PSO (BPSO),
- (3) unified PSO (UPSO) [7],
- (4) fully informed PSO (FIPS) [8],
- (5) cooperative PSO (CPSO-H) [9],
- (6) comprehensive learning PSO (CLPSO) [10],
- (7) adaptive learning PSO (APSO) [13],
- (8) pSO with neighborhood search (NSPSO) [11],
- (9) our approach (NSBPSO).

For the sake of fair comparison, we use the same settings for the same parameters. For all algorithms, the population size N is set to 40, and the maximum number of fitness evaluations (MAX_FEs) is set to $2.0e + 05$. For standard PSO, w is linearly decreased from 0.9 to 0.4, and $c_1 = c_2 = 1.49618$. For NSPSO, the probability of neighborhood search P_{ns} is set to 0.3. The parameter settings of UPSO, CPSO-H, FIPS, and CLPSO are described in [10]. For NSPSO and APSO, the same parameters are used by the literature [11, 13], respectively. For each test problem, each algorithm is conducted 30 times and the mean fitness error values are reported.

Table 3 lists the comparison results of NSBPSO with other eight PSO algorithms, where “Mean” represents the mean fitness error values. The best results are shown in boldface. From the results, it can be seen that NSBPSO outperforms PSO, BPSO, and FIPS on all test problems. UPSO and APSO perform better than NSBPSO on f_2 , while NSBPSO achieves better results for the rest 11 problems. CPSO-H obtains better solution than NSBPSO on f_2 , while NSBPSO outperforms CPSO-H on 10 problems. For problem f_6 , both NSBPSO and CPSO-H can find the global optimum. NSPSO performs

TABLE 3: Results achieved by the nine PSO algorithms.

Algorithms	f_1 Mean	f_2 Mean	f_3 Mean	f_4 Mean	f_5 Mean	f_6 Mean	f_7 Mean	f_8 Mean	f_9 Mean	f_{10} Mean	f_{11} Mean	f_{12} Mean
PSO	9.78E-30	2.93E+01	3.94E-14	8.13E-03	1.30E-04	2.90E+01	2.97E+01	1.10E+03	2.80E-01	1.64E-01	6.66E-01	9.90E+00
BPSO	3.60E-87	1.84E+01	1.12E-14	9.86E-03	5.65E-05	3.38E+01	2.50E+01	2.13E+03	1.90E+00	2.95E-02	7.95E+00	6.07E+01
UPSO	4.17E-87	1.51E+01	1.22E-15	1.66E-03	9.60E+00	6.59E+01	6.34E+01	4.84E+03	1.00E+00	7.76E-02	2.61E+00	1.52E+01
FIPS	2.69E-12	2.45E+01	4.81E-07	1.16E-06	1.54E-01	7.30E+01	6.08E+01	2.05E+03	2.25E-15	1.70E-01	5.93E-14	1.20E+01
CPSO-H	1.16E-113	7.08E+00	4.93E-14	3.63E-02	7.82E-15	0.00E+00	1.00E-01	1.08E+03	1.36E+00	1.20E-01	4.35E+00	2.67E+01
CLPSO	1.46E-14	2.01E+01	0.00E+00	3.14E-10	3.45E-07	4.85E-10	4.36E-10	1.27E-12	3.65E-05	4.50E-02	3.72E-10	5.97E+00
APSO	9.60E-66	1.83E+01	1.09E-14	1.20E-02	4.77E-02	6.27E+00	2.27E+00	2.13E+03	1.22E+00	1.38E-02	8.40E+00	7.09E+01
NSPSO	0.00E+00	4.08E+00	5.89E-16	0.00E+00	6.72E-13	0.00E+00	0.00E+00	1.07E+03	9.55E-13	0.00E+00	1.54E-13	0.00E+00
NSBPSO	0.00E+00	2.01E+01	5.89E-16	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1.01E+03	5.89E-16	0.00E+00	0.00E+00	0.00E+00

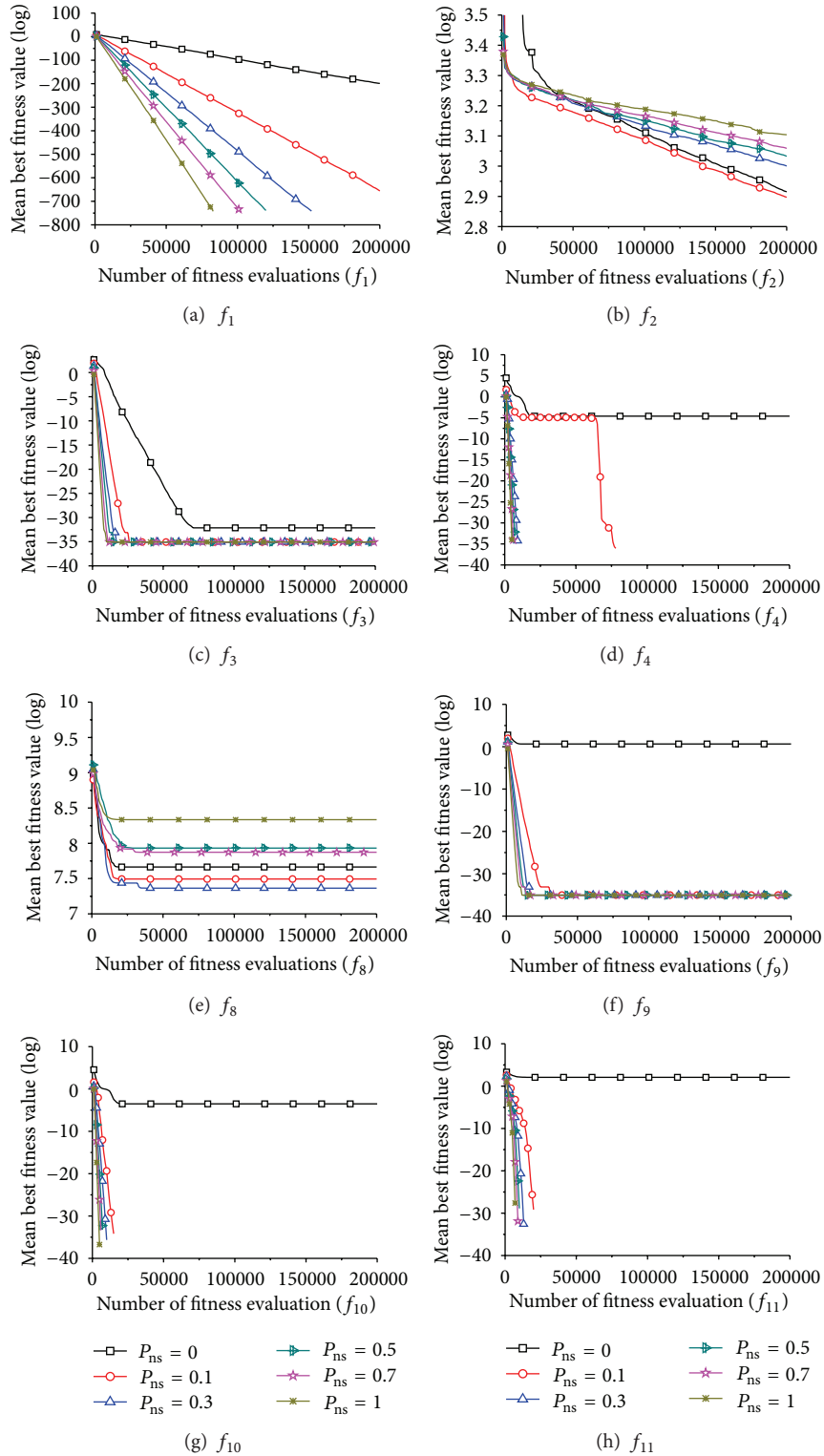
FIGURE 2: The convergence curves of NSBPSO with different P_{ns} .

TABLE 4: Results of Wilcoxon signed-rank test between NSBPSO and other eight PSO algorithms.

NSBPSO versus	<i>P</i> values
PSO	4.88E – 04
BPSO	6.05E – 03
UPSO	6.35E – 03
FIPS	1.17E – 02
CPSO-H	4.88E – 04
CLPSO	6.54E – 02
APSO	6.35E – 03
NSPSO	3.75E – 01

better than NSBPSO on f_2 , while NSBPSO outperforms NSPSO on 4 problems. Both of them can converge to the global optimum on 7 problems.

From the comparison of BPSO and PSO, BPSO outperforms PSO on 6 problems, while PSO achieves better results than BPSO for the rest 6 problems. The results demonstrate that the performance of BPSO is similar to PSO on these problems. Compared to PSO, BPSO is more competitive, because BPSO does not contain any control parameter (except for the population size), while PSO employs empirical parameter settings. By hybridization of BPSO (or PSO) and the neighborhood search, NSBPSO (or NSPSO) achieves significant improvements on the performance of BPSO (or PSO). Compared to NSPSO, NSBPSO not only achieves better results, but also has less control parameters.

In order to compare the performance differences between NSBPSO and the other eight PSO algorithms, we conduct the Wilcoxon signed-rank test by the suggestions of [22]. Table 4 shows the *P*-values achieved by the Wilcoxon test. The *P* values below 0.05 are shown in boldface. As shown, NSBPSO is significantly better than all other algorithms except for CLPSO and NSPSO. Though NSBPSO is not significantly better than them, it outperforms them in the majority of test problems.

5. Application on Ship Design

5.1. Problem Description. This section investigates the performance of our approach NSBPSO for a conceptual ship design. The original optimization statements are presented in [23, 24]. The ship design optimization problem used in this paper has six design variables, three objectives, and 9 inequality constraints. The design variables are length (L), beam (B), depth (D), draft (T), block coefficient (C_B), and speed in knots (V_k). The ship design problem aims to minimize transportation cost (T_c) and lightship weight (L_s) and maximize annual cargo (A_c) [24]:

$$\begin{aligned} &\text{Minimize} \quad \{T_c, L_s\} \\ &\text{Maximize} \quad A_c, \end{aligned} \quad (8)$$

where $T_c = C_A/A_c$, $A_c = (DWT - F_c - DWT_M) \cdot RTPA$, and $L_s = W_s + W_o + W_m$. The specific model definition is described

in Table 5 [25]. The search ranges of the six variables L , B , D , T , C_B , and V_k are listed in Table 6.

There are 9 inequality constraints listed as follows:

$$\begin{aligned} 6 - \frac{L}{B} &\leq 0, \\ \frac{L}{D} - 15 &\leq 0, \\ \frac{L}{T} - 19 &\leq 0, \\ T - 0.45DWT^{0.31} &\leq 0, \\ T - 0.7D + 0.7 &\leq 0, \\ DWT - 500000 &\leq 0, \\ 25000 - DWT &\leq 0, \\ F_n - 0.32 &\leq 0, \\ 0.07B - KB - BMT + KG &\leq 0. \end{aligned} \quad (9)$$

5.2. Constraint Handling. In order to deal with the constraints, an adaptive penalty method is employed by the suggestions of [19]. Let $f(x)$ be the objective function (T_c , L_s , or A_c). The fitness evaluation function $F(x)$ is defined by

$$F(x) = \begin{cases} f(x), & \text{if } x \text{ is feasible,} \\ \bar{f}(x) + \sum_{j=1}^m k_j \cdot CV_j, & \text{otherwise,} \end{cases} \quad (10)$$

where m is the number of inequality constraints, $CV_j = \min\{0, c_j(x)\}$ is the constraint violation for the j th constraint, $c_j(x)$ is the j th constraint, $\bar{f}(x)$ is the mean objective function value in the current swarm, and k_j is a penalty coefficient defined as follows:

$$k_j = \frac{|f(x)| \cdot \overline{CV_j}(x)}{\sum_{i=1}^m [\overline{CV_i}(x)]^2}, \quad (11)$$

where $\overline{CV_j}(x)$ is the average violation of the j th constraint for all particles in the swarm.

5.3. Computational Results. The ship design problem is a multiobjective optimization problem which has three objectives. By the suggestions of [24, 25], this paper only considers single objective optimization. Therefore, the whole problem is divided into three single objective optimization problems: (1) minimize transportation cost (T_c), (2) minimize lightship weight (L_s), and (3) maximize annual cargo (A_c).

In this section, we conduct three series of experiments for the three single optimization problems. In order to verify the performance of our approach NSBPSO, we compare it with four other algorithms. The involved algorithms are listed as follows:

- (1) parsons and Scott's method [24],
- (2) standard PSO,

TABLE 5: Model definition of the ship design problem.

Parameter	Definition	Parameter	Definition
Steel weight (W_s)	$0.034 L^{1.7} B^{0.7} D^{0.4} C_B^{0.5}$	Port cost (C_p)	$6.3 \text{ DWT}^{0.8}$
Outfit weight (W_o)	$1.0 L^{0.8} B^{0.6} D^{0.4} C_B^{0.5}$	Fuel carried (F_c)	$D_c(D_s + 0.5)$
Displacement (Δ)	$1.025 L B T C_B$	Misc. deadweight (DWT_M)	$2.0 \text{ DWT}^{0.5}$
Machinery weight (W_m)	$0.17 P^{0.9}$	Cargo deadweight (DWT_c)	$\text{DWT} - F_c - \text{DWT}_M$
Capital cost (C_c)	$0.2 C_s$	Port days (D_p)	$2 (\text{DWT}_c / H_R + 0.5)$
Running cost (C_R)	$40000 \text{ DWT}^{0.3}$	Round trips per year (RTPA)	$350 / (D_s + D_p)$
Daily cost (D_c)	$(0.19 \cdot 24 \cdot P / 1000) + 0.2$	Voyage cost (C_v)	$(C_F + C_p) \text{ RTPA}$
Handling rate (H_R)	8000 (t/day)	Round trip miles (RTM)	5000 (nm)
Fuel price (F_p)	100 (£/t)	Sea days (D_s)	$\text{RTM} / 24 V_k$
Power (P)	$\sqrt[3]{\Delta^2 V_k^3 / (a + b F_n)}$	Fuel cost (C_F)	$1.05 D_c D_s F_p$
Froude number (F_n)	$V / \sqrt{gL}, V = 0.5144 V_k$	g	9.8065
Annual cost (C_A)	$(\text{DWT}_c) \text{ RTPA}$	KB	$0.53 T$
KG	$1.0 + 0.52 D$	BMT	$(0.085 C_B - 0.002) B^2 / (T C_B)$
Ship cost (C_s)			$1.3 (2000 W_s^{0.85} + 3500 W_o + 2400 P^{0.8})$

TABLE 6: Search ranges of the six design variables.

$150 \leq L \leq 274.32$	$10 \leq T \leq 11.71$
$20 \leq B \leq 32.31$	$0.63 \leq C_B \leq 0.75$
$13 \leq D \leq 25$	$11 \leq V_k \leq 20$

- (3) barebones PSO (BPSO),
- (4) PSO with neighborhood search (NSPSO),
- (5) our approach (NSBPSO).

To have a fair comparison, the same parameter settings are used for common parameters. For all algorithms, the population size and the maximum number of fitness evaluations (MAX_FEs) are set to 100 and $1.0e+06$. For standard PSO, w is linearly decreased from 0.9 to 0.4 and $c_1 = c_2 = 1.49618$. For NSBPSO and NSPSO, P_{ns} is set to 0.3. For each optimization problem, each algorithm is run 10 times and the best results among these runs are presented.

Tables 7–9 show the computational results for the three problems. For Table 7, NSBPSO achieves the minimal transportation cost among the five algorithms; but it also obtains the minimal value of annual cargo. For the objective of T_c , NSBPSO is the best among the five algorithms, however, it could not obtain the best results for all three objectives. For annual cargo A_c , Parsons and Scott's [24] method is the best. Tables 8 and 9 can also get similar conclusions. The results demonstrate that NSBPSO shows better performance than the other three algorithms for single objective optimization problem of the ship design. When considering all objectives, we cannot conclude which algorithm is the best. To perfectly solve this problem, we may use multiobjective optimization algorithms. Figures 3, 4, and 5 present the convergence curves of PSO, BPSO, NSPSO, and NSBPSO for the three single objective problems. For minimizing transportation cost, NSBPSO shows faster convergence speed at the last stage of the evolution. For minimizing lightship weight, NSBPSO converges faster than other three PSO algorithms.

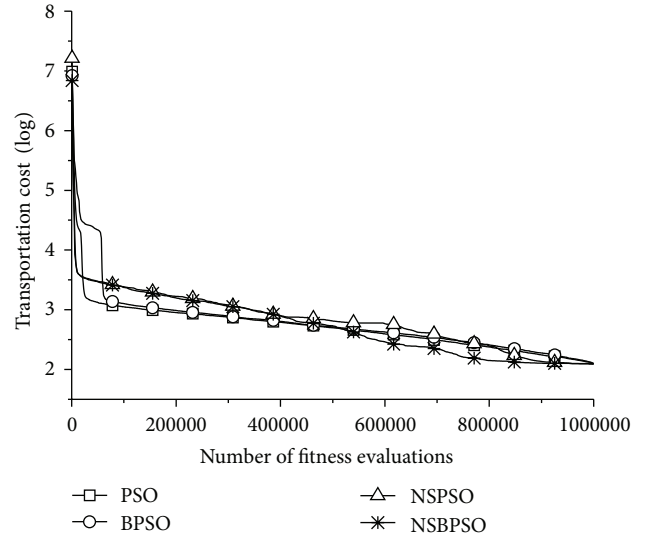


FIGURE 3: The convergence curves of PSO, BPSO, NSPSO and NSBPSO for minimizing transportation cost.

For maximizing annual cargo, both NSBPSO and NSPSO show similar convergence characteristics.

6. Conclusions

Barebones PSO (BPSO) is a new variant of PSO which eliminates the velocity term. Although some reported results show that BPSO is better than PSO, it still gets stuck when solving complex multimodal problems. In order to enhance the performance of BPSO, this paper proposes an improved version called BPSO with neighborhood search (NSBPSO). The new approach embeds one local and one global neighborhood search strategies into the original BPSO to achieve a tradeoff between exploration and exploitation. Compared to other improved PSO algorithms, NSBPSO is

TABLE 7: Computational results for minimizing transportation cost.

Objective	Parsons and Scott [24]	PSO	BPSO	NSPSO	NSBPSO
Transportation cost (T_c)	8.377	8.213	8.195	8.176	8.162
Lightship weight (L_s)	9029.0	8420.1	8486.3	8452.8	8355.2
Annual cargo (A_c)	551265	509078	508069	505738	503261
L	193.86	190.63	191.72	193.04	192.36
B	32.31	30.15	30.13	29.57	29.34
D	15.73	15.54	15.49	15.42	15.39
T	11.71	11.64	11.50	11.51	11.48
C_B	0.681	0.729	0.730	0.730	0.730
V_k	14.00	12.25	12.32	12.21	12.48

TABLE 8: Computational results for minimizing lightship weight.

Objective	Parsons and Scott [24]	PSO	BPSO	NSPSO	NSBPSO
Transportation cost (T_c)	9.474	9.374	9.341	9.325	9.286
Lightship weight (L_s)	5240.3	5019.9	4946.6	4932.6	4809.2
Annual cargo (A_c)	386500	371960	369257	366855	361372
L	150.73	155.32	154.59	155.7	152.8
B	25.12	24.36	24.42	24.78	22.87
D	13.84	14.43	14.18	13.02	13.92
T	10.39	10.76	10.62	10.38	10.57
C_B	0.750	0.693	0.69	0.705	0.750
V_k	14.00	13.62	13.78	13.29	13.25

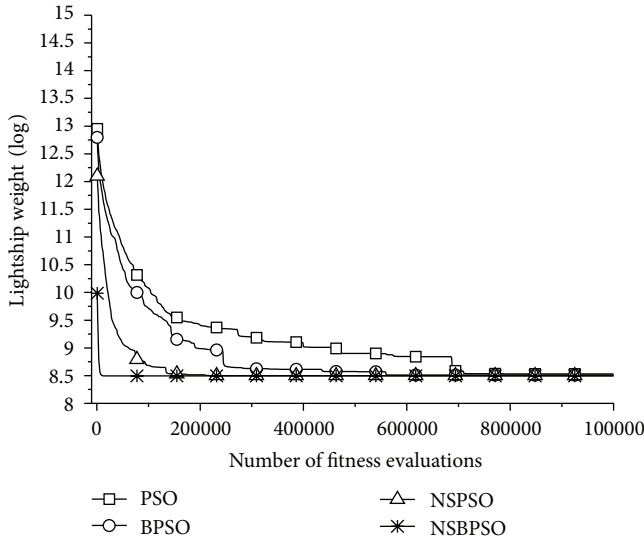


FIGURE 4: The convergence curves of PSO, BPSO, NSPSO, and NSBPSO for minimizing lightship weight.

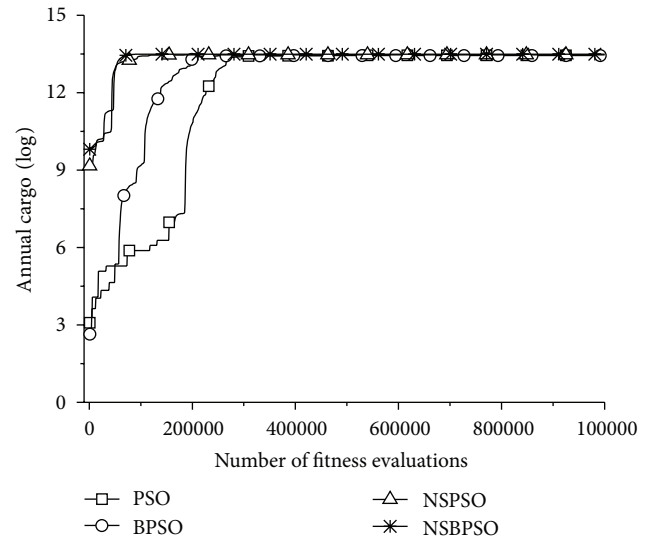


FIGURE 5: The convergence curves of PSO, BPSO, NSPSO, and NSBPSO for maximizing annual cargo.

almost parameter-free algorithm except for the probability of neighborhood search (P_{ns}).

Experimental studies are conducted on twelve well-known benchmark problems, including unimodal, multimodal, and rotated multimodal problems. Computational results show that the parameter P_{ns} does not affect the performance of NSBPSO. When $P_{ns} > 0$, NSBPSO can obtain good performance. Another comparison demonstrates that

NSBPSO performs better than, or at least comparable to, several other state-of-the-art PSO algorithms. Compared to PSO with neighborhood search (NSPSO), our approach NSBPSO not only achieves better results, but also has less control parameters.

For the ship design problem, NSBPSO performs better than other three algorithms when optimizing a single objective. When considering all three objectives, we cannot

TABLE 9: Computational results for maximizing annual cargo.

Objective	Parsons and Scott [24]	PSO	BPSO	NSPSO	NSBPSO
Transportation cost (T_c)	10.294	10.884	11.308	11.59	12.213
Lightship weight (L_s)	12436	12558.2	12694.5	12757.4	12769.2
Annual cargo (A_c)	700533	706405	737952	741928	753137
L	222.49	235.86	231.95	235.27	238.2
B	32.31	32.20	32.32	32.14	32.31
D	15.73	16.60	17.29	16.62	15.73
T	11.71	11.42	11.70	11.70	11.71
C_B	0.750	0.720	0.750	0.750	0.750
V_k	18.00	18.09	18.73	18.68	18.89

determine which algorithm is the best. Because one algorithm only achieves better results than other algorithms on one or two objectives. To tackle this problem, we can use multiobjective optimization algorithms to optimize the three objectives at the same time. This will be investigated in the future work.

Acknowledgment

This work is supported by the National Natural Science Foundation of China (no. 51209057).

References

- [1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.
- [2] J. Mirapeix, P. B. García-Allende, O. M. Conde, J. M. Lopez-Higuera, and A. Cobo, "Welding diagnostics by means of particle swarm optimization and feature selection," *Journal of Sensors*, vol. 2012, Article ID 318038, 11 pages, 2012.
- [3] Y. Hu, Y. Ding, and K. R. Hao, "An immune cooperative particle swarm optimization algorithm for fault-tolerant routing optimization in heterogeneous wireless sensor networks," *Mathematical Problems in Engineering*, vol. 2012, Article ID 743728, 19 pages, 2012.
- [4] S. S. Patnaik and A. K. Panda, "Particle swarm optimization and bacterial foraging optimization techniques for optimal current harmonic mitigation by employing active power filter," *Applied Computational Intelligence and Soft Computing*, vol. 2012, Article ID 897127, 10 pages, 2012.
- [5] C. Knievel and P. A. Hoehner, "On particle swarm optimization for MIMO channel estimation," *Journal of Electrical and Computer Engineering*, vol. 2012, Article ID 614384, 10 pages, 2012.
- [6] Y. Shi and R. Eberhart, "Modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, May 1998.
- [7] K. E. Parsopoulos and M. N. Vrahatis, "UPSO-A unified particle swarm optimization scheme," in *Proceedings of International Conference on Computational Methods in Sciences and Engineering*, pp. 868–873, 2004.
- [8] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 204–210, 2004.
- [9] F. van den Bergh and A. P. Engelbrecht, "A cooperative approach to particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, 2004.
- [10] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.
- [11] H. Wang, Z. Wu, S. Rahnamayan, C. Li, S. Zeng, and D. Jiang, "Particle swarm optimisation with simple and efficient neighbourhood search strategies," *International Journal of Innovative Computing and Applications*, vol. 3, no. 2, pp. 97–104, 2011.
- [12] 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.
- [13] Z. H. Zhan, J. Zhang, Y. Li, and H. S. H. Chung, "Adaptive particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 39, no. 6, pp. 1362–1381, 2009.
- [14] C. Li, S. Yang, and T. T. Nguyen, "A self-learning particle swarm optimizer for global optimization problems," *IEEE Transactions on Systems, Man, and Cybernetics Part B*, vol. 42, no. 3, pp. 627–646, 2012.
- [15] J. Kennedy, "Bare bones particle swarms," in *Proceedings of the IEEE Swarm Intelligence Symposium*, pp. 80–87, 2003.
- [16] M. G. H. Omran, A. P. Engelbrecht, and A. Salman, "Bare bones differential evolution," *European Journal of Operational Research*, vol. 196, no. 1, pp. 128–139, 2009.
- [17] R. A. Krohling and E. Mendel, "Bare bones particle swarm optimization with Gaussian or cauchy jumps," in *Proceedings IEEE Congress on Evolutionary Computation (CEC '09)*, pp. 3285–3291, May 2009.
- [18] T. Blackwell, "A study of collapse in bare bones particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 3, pp. 354–372, 2012.
- [19] H. Wang, "Opposition-based barebones particle swarm for constrained nonlinear optimization problems," *Mathematical Problems in Engineering*, vol. 2012, Article ID 761708, 12 pages, 2012.
- [20] F. Van Den Bergh and A. P. Engelbrecht, "A study of particle swarm optimization particle trajectories," *Information Sciences*, vol. 176, no. 8, pp. 937–971, 2006.
- [21] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 526–553, 2009.
- [22] 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.

- [23] P. Sen and J. B. Yang, *Multiple Criteria Decision Support in Engineering Design*, Springer, London, UK, 1988.
- [24] M. G. Parsons and R. L. Scott, "Formulation of multicriterion design optimization problems for solution with scalar numerical optimization methods," *Journal of Ship Research*, vol. 48, no. 1, pp. 61–76, 2004.
- [25] C. G. Hart and N. Vlahopoulos, "An integrated multidisciplinary particle swarm optimization approach to conceptual ship design," *Structural and Multidisciplinary Optimization*, vol. 41, no. 3, pp. 481–494, 2010.

Research Article

A Simple and Efficient Artificial Bee Colony Algorithm

Yunfeng Xu,^{1,2} Ping Fan,³ and Ling Yuan¹

¹ School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074, China

² Department of Information Security Engineering, Chinese People's Public Security University, Beijing 100038, China

³ School of Computer Science, Hubei University of Science and Technology, Xianning 437100, China

Correspondence should be addressed to Ping Fan; bjfan208@126.com

Received 7 September 2012; Revised 30 November 2012; Accepted 30 December 2012

Academic Editor: Rui Mu

Copyright © 2013 Yunfeng Xu 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.

Artificial bee colony (ABC) is a new population-based stochastic algorithm which has shown good search abilities on many optimization problems. However, the original ABC shows slow convergence speed during the search process. In order to enhance the performance of ABC, this paper proposes a new artificial bee colony (NABC) algorithm, which modifies the search pattern of both employed and onlooker bees. A solution pool is constructed by storing some best solutions of the current swarm. New candidate solutions are generated by searching the neighborhood of solutions randomly chosen from the solution pool. Experiments are conducted on a set of twelve benchmark functions. Simulation results show that our approach is significantly better or at least comparable to the original ABC and seven other stochastic algorithms.

1. Introduction

Optimization problems arise in many application areas such as engineering, economy, and management. Effective and efficient optimization algorithms are always required to tackle increasingly complex real-world optimization problems. In the past several years, some swarm intelligence algorithms, inspired by the social behaviors of birds, fish, or insects, have been proposed to solve optimization problems, such as particle swarm optimization (PSO) [1], ant colony optimization (ACO) [2], artificial bee colony (ABC) [3], and firefly algorithm (FA) [4]. A recent study has shown that ABC performs significantly better or at least comparable to other swarm intelligence algorithms [5].

ABC is a new swarm intelligence algorithm proposed by Karaboga in 2005, which is inspired by the behavior of honey bees [3]. Since the development of ABC, it has been applied to solve different kinds of problems [6]. Similar to other stochastic algorithms, ABC also faces up some challenging problems. For example, ABC shows slow convergence speed during the search process. Due to the special search pattern of bees, a new candidate solution is generated by updating a random dimension vector of its parent solution.

Therefore, the offspring (new candidate solution) is similar to its parent, and the convergence speed becomes slow. Moreover, ABC easily falls into local minima when handling complex multimodal problems. The search pattern of bees is good at exploration but poor at exploitation [7]. However, a good optimization algorithm should balance exploration and exploitation during the search process.

To improve the performance of ABC, this paper proposes a new search pattern for both employed and onlooker bees. In the new approach, some best solutions are utilized to accelerate the convergence speed. In addition, a solution pool is constructed by storing the best 100

p

% solutions in the current swarm with $p \in (0, 1]$. The best solution used in the search pattern is randomly selected from the solution pool. This is helpful to balance the exploration and exploitation. Experiments are conducted on twelve benchmark functions. Simulation results show that our approach outperforms the original ABC and several other stochastic algorithms.

The rest of the paper is organized as follows. In Section 2, the original ABC algorithm is presented. Section 3 gives a brief overview of related work. Section 4 describes the proposed approach. In Section 5, experimental studies are presented. Finally, the work is concluded in Section 6.

2. Artificial Bee Colony

Artificial bee colony (ABC) algorithm is a recently proposed optimization technique which simulates the intelligent foraging behavior of honey bees. A set of honey bees is called swarm which can successfully accomplish tasks through social cooperation. In the ABC algorithm, there are three types of bees: employed bees, onlooker bees, and scout bees. The employed bees search food around the food source in their memory; meanwhile they share the information of these food sources to the onlooker bees. The onlooker bees tend to select good food sources from those found by the employed bees. The food source that has higher quality (fitness) will have a large chance to be selected by the onlooker bees than the one of lower quality. The scout bees are translated from a few employed bees, which abandon their food sources and search new ones [8].

In the ABC algorithm, the first half of the swarm consists of employed bees, and the second half constitutes the onlooker bees. The number of employed bees or the onlooker bees is equal to the number of solutions in the swarm [3].

The ABC generates a randomly distributed initial population of SN solutions (food sources), where SN denotes the swarm size. Let $X_i = \{x_{i,1}, x_{i,2}, \dots, x_{i,D}\}$ represent the i th solution in the swarm, where D is the dimension size. Each employed bee X_i generates a new candidate solution V_i in the neighborhood of its present position as follows:

$$v_{i,j} = x_{i,j} + \phi_{i,j} \cdot (x_{i,j} - x_{k,j}), \quad (1)$$

where X_k is a randomly selected candidate solution ($i \neq k$), j is a random dimension index selected from the set $\{1, 2, \dots, D\}$, and $\phi_{i,j}$ is a random number within $[-1, 1]$. Once the new candidate solution V_i is generated, a greedy selection is used. If the fitness value of V_i is better than that of its parent X_i , then update X_i with V_i ; otherwise keep X_i unchangeable.

After all employed bees complete the search process, they share the information of their food sources with the onlooker bees through waggle dances. An onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount. This probabilistic selection is really a roulette wheel selection mechanism which is described as follows:

$$p_i = \frac{\text{fit}_i}{\sum_{j=1}^{\text{SN}} \text{fit}_j}, \quad (2)$$

where fit_i is the fitness value of the i th solution in the swarm. As seen, the better the solution i , the higher the probability of the i th food source selected.

If a position cannot be improved over a predefined number (called *limit*) of cycles, then the food source is abandoned. Assume that the abandoned source is X_i , then the scout bee discovers a new food source to be replaced with X_i as follows:

$$x_{i,j} = lb_j + \text{rand}(0, 1) \cdot (ub_j - lb_j), \quad (3)$$

where $\text{rand}(0, 1)$ is a random number within $[0, 1]$ based on a normal distribution and lb , ub are lower and upper boundaries of the j th dimension, respectively.

3. Related Work

Since the development of ABC, it has attracted much attention for its excellent characteristics. In the last decade, different versions of ABCs have been applied to various problems. In this section, we present a brief review of these ABC algorithms.

Karaboga and Akay [5] presented a comparative study of ABC. A large set of benchmark functions are tested in the experiments. Results show that the ABC is better than or similar to those of other population-based algorithms with the advantage of employing fewer control parameters. Inspired by differential evolution (DE) algorithm, Gao and Liu [7, 9] proposed two improved versions of ABC. In [7], a new search pattern called ABC/best/1 is utilized to accelerate the convergence speed. In [9], ABC/best/1 and another search pattern called ABC/rand/1 are employed. Moreover, a parameter q is intruded to control the frequency of these two patterns. Zhu and Kwong [8] utilized the search information of the global best solution (g_{best}) to guide the search of ABC. Reported results show that the new approach achieves better results than the original ABC algorithm. Akay and Karaboga [10] proposed a modified ABC algorithm, in which two new search patterns, frequency and magnitude of the perturbation, are employed to improve the convergence rate. Results show that the original ABC algorithm can efficiently solve basic and simple functions, while the modified ABC algorithm obtains promising results on hybrid and complex functions when compared to some state-of-the-art algorithms. Banharnsakun et al. [11] modified the search pattern of the onlooker bees, in which the best feasible solutions found so far are shared globally among the entire swarm. Therefore, the new candidate solutions are similar to the current best solution. Kang et al. [12] proposed a Rosenbrock ABC (RABC) algorithm which combines Rosenbrock's rotational direction method with the original ABC. There are two alternative phases of RABC: the exploration phase realized by ABC and the exploitation phase completed by the Rosenbrock method. Wu et al. [13] combined harmony search (HS) and the ABC algorithm to construct a hybrid algorithm. Comparison results show that the hybrid algorithm outperforms ABC, HS, and other heuristic algorithms. Li et al. [14] proposed an improved ABC algorithm called I-ABC, in which the best-so-far solution, inertia weight, and acceleration coefficients are introduced to modify the search process. Moreover, a hybrid ABC algorithm (PS-ABC) based on g_{best} -guided ABC (GABC) [8] and I-ABC is proposed. Results show that PS-ABC converges faster than I-ABC and ABC.

Karaboga and Ozturk [15] used ABC algorithm for data clustering. Experiments are conducted on thirteen typical test data sets from UCL Machine Learning Repository. The performance of ABC is compared with PSO and other nine classification techniques. Simulation results demonstrate that the ABC algorithm can efficiently solve data clustering. Zhang et al. [16] also used ABC algorithm for clustering. Three data sets are tested. The performance of ABC is compared with genetic algorithm, simulated annealing, tabu search, ACO, and K-NM-PSO. Results demonstrate the

effectiveness of ABC on clustering. Karaboga and Ozturk [17] applied ABC to solve fuzzy clustering. Three data sets including cancer, diabetes, and heart chosen from UCI database are tested. Results indicate that the performance of ABC is successful in fuzzy clustering.

The ABC algorithm is usually used to solve unconstrained optimization problems. In [18], Karaboga and Akay investigated the performance of ABC on constrained optimization problems. In order to handle constraints, Deb's rules consisting of three simple heuristic rules are employed. Mezura-Montes and Velez-Koeppel [19] proposed an elitist ABC algorithm for constrained real-parameter optimization, in which the operators used by different types of bees are modified. Additionally, a dynamic tolerance control mechanism for equality constraints is utilized to facilitate the approach to the feasible region of the search space. Yeh and Hsieh [20] proposed a penalty-guided ABC algorithm to solve reliability redundancy allocation problems. Sabat et al. [21] presented an application of ABC to extract the small signal equivalent circuit model parameters of GaAs metal-extended semiconductor field effect transistor (MESFT) device. The performance comparison shows that ABC is better than PSO.

It is known that the ABC algorithm is good at solving optimization problems over continuous search space. For discrete optimization problems, it is a big challenge for the ABC algorithm. Li et al. [22] used a hybrid Pareto-based ABC algorithm to solve flexible job shop-scheduling problems. In the new algorithm, each food sources is represented by two vectors, that is, the machine assignment and the operation scheduling. Moreover, an external Pareto archive set is utilized to record nondominated solutions. In [23], Kashan et al. designed a new ABC algorithm called DisABC to optimize binary structured problems. Szeto et al. [24] proposed an enhanced ABC algorithm to solve capacitated vehicle routing problem. The performance of the new approach is tested on two sets of standard benchmark instances. Simulation results show that the new algorithm outperforms the original ABC and several other existing algorithms. Pan et al. [25] presented a discrete ABC algorithm hybridized with a variant of iterated greedy algorithm to solve a permutation flow shop-scheduling problem with the total flow time criterion.

4. Proposed Approach

Differential evolution (DE) has shown excellent search abilities on many optimization problems. Like other population-based stochastic algorithms, DE also starts with an initial population with randomly generated candidate solutions. After initialization, DE repeats three operations: mutation, crossover, and selection. Among these operations, mutation operation is very important. The mutation scheme highly influences the performance of DE. There are several different mutation schemes, such as DE/rand/1, DE/rand/2, DE/best/1, and DE/best2 [26].

The property of a mutation scheme determines the search behavior of individuals in the population. For DE/rand/1, it results in good exploration but slow convergence speed. For DE/best/1, it obtains fast convergence speed but poor

exploration. The DE/rand/1 and DE/best/1 are described as follows:

$$\begin{aligned} v_{i,j} &= x_{r1,j} + F \cdot (x_{r2,j} - x_{r3,j}), \\ v_{i,j} &= x_{\text{best},j} + F \cdot (x_{r1,j} - x_{r2,j}), \end{aligned} \quad (4)$$

where X_{r1} , X_{r2} , and X_{r3} are three randomly selected individuals from the current population, $i \neq r1 \neq r2 \neq r3$, X_{best} is the best individual found so far, and the parameter F is known as the scale factor which is usually set to 0.5.

As seen, the search pattern of employed and onlooker bees is similar to the mutation schemes of DE. It is known that the ABC algorithm is good at exploration, but it shows slow convergence speed. By combining the DE/best/1 and the ABC algorithm, it may accelerate the convergence speed of ABC. However, this hybridization is not a new idea. In [7], Gao and Liu embedded DE/rand/1 and DE/best/1 into the ABC algorithm. To balance the exploration and exploitation, a new parameter q is introduced. Results reported in [7] show that the parameter q is problem oriented, and an empirical value $q = 0.7$ is used.

In this paper, we propose a new ABC (called NABC) algorithm by employing a modified DE/best/1 strategy. NABC differs from other hybrid algorithms [7, 9], which combine ABC and DE. Although the global best individual used in DE/best/1 can accelerate the convergence speed by the attraction, it may result in attracting too fast. It means that new solutions move to the global best solution very quickly. To tackle this problem, a solution pool is constructed by storing the best $100p\%$ solutions in the current swarm with $p \in (0, 1]$. The idea is inspired by an adaptive DE algorithm (JADE) [27]. It shares in common with the concept of belief space of cultural algorithm (CA) [28]. Both of them utilize some successful solutions stored in solution pool or situational knowledge to guide other individuals. But the updating rule of the solution pool or situational knowledge is different. The new ABC/best/1 strategy is described as follows:

$$v_{i,j} = x_{\text{best},j}^p + \phi_{i,j} \cdot (x_{r1,j} - x_{r2,j}), \quad (5)$$

where $x_{\text{best},j}^p$ is randomly chosen from the solution pool, X_{r1} , X_{r2} are two randomly selected candidate solutions from the current swarm, $i \neq r1 \neq r2$, j is a random dimension index selected from the set $\{1, 2, \dots, D\}$, and $\phi_{i,j}$ is a random number within $[-1, 1]$. Empirical studies show that a good choice of the parameter p should be set between 0.08 and 0.15. In this paper, p is set to 0.1 for all experiments.

According to the new search pattern described in (5), new candidate solutions are generated around some best solutions. This is helpful to accelerate the convergence speed. For the existing ABC/best/1 strategy proposed in [7], it only searches the neighborhood of the global best solution. In our approach, bees can search the neighborhood of different best solutions. This can help avoid fast attraction.

The main steps of our new approach NABC algorithm are listed as follows.

Step 1. Randomly initialize the swarm.

Step 2. Update the solution pool.

TABLE 1: Benchmark functions used in the experiments.

Name	Function	Range	Opt
Sphere	$f_1 = \sum_{i=1}^D x_i^2$	$[-100, 100]$	0
Schwefel 2.22	$f_2 = \sum_{i=1}^D x_i + \prod_{i=1}^D x_i$	$[-10, 10]$	0
Schwefel 1.2	$f_3 = \sum_{i=1}^D \left(\sum_{j=1}^i x_j \right)^2$	$[-100, 100]$	0
Schwefel 2.21	$f_4 = \max_i (x_i , 1 \leq i \leq D)$	$[-100, 100]$	0
Rosenbrock	$f_5 = \sum_{i=1}^{D-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$	$[-30, 30]$	0
Step	$f_6 = \sum_{i=1}^D (\lfloor x_i + 0.5 \rfloor)^2$	$[-100, 100]$	0
Quartic with noise	$f_7 = \sum_{i=1}^D ix_i^4 + \text{rand}[0, 1)$	$[-1.28, 1.28]$	0
Schwefel 2.26	$f_8 = \sum_{i=1}^D -x_i \sin(\sqrt{ x_i })$	$[-500, 500]$	-12569.5
Rastrigin	$f_9 = \sum_{i=1}^D [x_i^2 - 10 \cos(2\pi x_i) + 10]$	$[-5.12, 5.12]$	0
Ackley	$f_{10} = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2}\right) - \exp\left(\frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i)\right) + 20 + e$	$[-32, 32]$	0
Griewank	$f_{11} = \frac{1}{4000} \sum_{i=1}^D x_i^2 - \prod_{i=1}^D \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]$	0
Penalized	$f_{12} = \frac{\pi}{D} \{10 \sin^2(3\pi y_i) + \sum_{i=1}^{D-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_D - 1)^2\} + \sum_{i=1}^D u(x_i, 5, 100, 4)$	$[-50, 50]$	0

Step 3. For each employed bee, generate a new candidate solution V_i according to (5). Evaluate the fitness of V_i and use a greed selection to choose a better one between X_i and V_i as the new X_i .

Step 4. Each onlooker bee calculates p_i according to (2).

Step 5. Generate a new V_i according to (5) based on p_i and the current solution X_i (food source). Evaluate the fitness of V_i and use a greed selection to choose a better one between X_i and V_i as the new X_i .

Step 6. The scout bee determines the abandoned X_i , if exists and update it by (3).

Step 7. Update the best solution found so far, and cycle = cycle + 1.

Step 8. If the number of cycles reaches to the maximum value MCN, then stop the algorithm and output the results; otherwise go to Step 2.

Compared to the original ABC algorithm, our approach NABC does not add extra operations except for the construction of the solution pool. However, this operation does not add the computational complexity. Both NABC and the original ABC have the same computational complexity.

5. Experimental Study

5.1. Test Functions. In order to verify the performance of NABC, experiments are conducted on a set of twelve benchmark functions. These functions were early used in [29].

According to their properties, they are divided into two classes: unimodal functions ($f_1 - f_7$) and multimodal functions ($f_8 - f_{12}$). All functions are minimization problems, and their dimensional size is 30. Table 1 presents the descriptions of these functions, where Opt is the global optimum.

The experiments are performed on the same computer with Intel (R) Core (TM)2 Duo CPU T6400 (2.00 GHz) and 2 GB RAM. Our algorithm is implemented using C++ and complied with Microsoft Visual C++ 6.0 under the Windows XP (SP3).

5.2. Comparison of NABC with ABC. In order to investigate the effectiveness of our new search pattern, this section presents a comparison of NABC with the original ABC algorithm. In the experiments, both NABC and ABC use the same parameter settings. The population size SN, *limit*, and maximum number of cycles (MSN) are set to 100, 100, and 1000, respectively. The parameter p is set to 0.1 based on empirical studies. All results reported in this section are averaged over 30 independent runs.

Table 2 presents the computational results of ABC and NABC on the twelve functions, where “Mean” indicates the mean function value and “Std Dev” represents the standard deviation. The best results between ABC and NABC are shown in bold. From the results, it can be seen that NABC achieves better results than ABC on all test functions except for f_6 . On this function, both the two algorithms can find the global optimum. For f_8 and f_9 , NABC can successfully find the global optimum, while ABC converges to near-optimal solutions. It demonstrates that the new search pattern used in NABC is helpful to improve the accuracy of solutions.

In order to compare the convergence speed of NABC and ABC, Figure 1 lists the convergence processes of them on

TABLE 2: Results achieved by the ABC algorithm and NABC.

Functions	ABC		NABC	
	Mean	Std Dev	Mean	Std Dev
f_1	$3.75e - 10$	$2.73e - 10$	$4.75e - 16$	$3.86e - 16$
f_2	$2.29e - 06$	$4.26e - 06$	$1.79e - 15$	$2.53e - 15$
f_3	$1.23e + 04$	$2.39e + 03$	$9.90e + 03$	$1.67e + 03$
f_4	$3.92e + 01$	$1.52e + 01$	$1.45e + 01$	$4.32e + 00$
f_5	$2.83e + 00$	$1.79e + 00$	$4.50e - 02$	$2.38e - 02$
f_6	$0.00e + 00$	$0.00e + 00$	$0.00e + 00$	$0.00e + 00$
f_7	$1.91e - 01$	$2.33e - 01$	$1.56e - 02$	$3.24e - 02$
f_8	-12332.4	$1.57e + 02$	-12569.5	$1.23e - 10$
f_9	$2.90e - 09$	$5.31e - 09$	$0.00e + 00$	$0.00e + 00$
f_{10}	$3.93e - 06$	$2.78e - 06$	$3.97e - 14$	$5.12e - 15$
f_{11}	$4.52e - 09$	$3.81e - 09$	$1.13e - 16$	$3.39e - 16$
f_{12}	$1.04e - 11$	$2.43e - 11$	$3.19e - 16$	$3.26e - 16$

some representative functions. As seen, NABC shows faster convergence speed than ABC. It confirms that the new search pattern can accelerate the convergence speed.

5.3. Comparison of NABC with Other Algorithms. To further verify the performance of NABC, this section compares NABC with other population-based algorithms, including some recently proposed ABC algorithms.

5.3.1. Comparison of NABC with Evolution Strategies. This section focuses on the comparison of the NABC algorithm with Evolution Strategies (ES). The versions of the ES include classical evolution strategies (CES) [30], fast evolution strategies (FES) [30], covariance matrix adaptation evolution strategies (CMA-ES) [31], and evolutionary strategies learned with automatic termination (ESLAT) [32].

The parameter settings of CES, FES, CMA-ES, and ESLAT can be found in [32]. For NABC, the population size and the maximum number of fitness evaluations are set to 20 and 100000 (it means that the MSN is 2500), respectively. The parameter *limit* is set to 600 [5]. The parameter *p* is set to 0.1 based on empirical studies. All algorithms are conducted on 50 runs for each test function.

Table 3 presents the comparison results of CES, FES, CMA-ES, ESLAT, and NABC. Results of CES, FES, CMA-ES, and ESLAT were taken from Table 20 in [5]. Among these algorithms, the best results are shown in bold. The last column of Table 3 reports the statistical significance level of the difference of the means of NABC and the best algorithm among the four evolution strategies. Note that here “+” represents the *t* value of 49 degrees of freedom which is significant at a 0.05 level of significance by two-tailed test, “.” indicates the difference of means which is not statistically significant, and “NA” means not applicable, covering cases for which the two algorithms achieve the same accuracy results [33].

From the results, it can be seen that NABC outperforms CES and FES on eight functions, while CES and FES achieve better results on three. For f_6 , CES, FES, and NABC find

the global optimum, while ESLAT and CMA-ES fail to solve it. NABC performs better than ESLAT on ten functions, while ESLAT outperforms NABC for the rest of the two functions. CMA-ES achieves better results than NABC on three functions, while NABC performs better for the rest of the nine functions. The comparison results show that the evolutionary strategies perform better than NABC on unimodal functions, such as $f_1 - f_4$. NABC outperforms the evolutionary strategies on all multimodal functions ($f_8 - f_{12}$).

5.3.2. Comparison of NABC with Other Improved ABC Algorithms. In this section, we present a comparison of NABC with three recently proposed ABC algorithms. The involved algorithms are listed as follows.

- (1) g_{best} -guided ABC algorithm (GABC) [8].
- (2) Improved ABC algorithm (I-ABC) [14].
- (3) Hybridization of GABC and I-ABC (PS-ABC) [14].
- (4) Our approach NABC.

In the experiments, the population size SN is set to 40, and *limit* equals 200. The maximum number of cycles is set as 1000. Other parameter settings of GABC, I-ABC, and PS-ABC can be found in [14]. The parameter *p* used in NABC is set to 0.1 based on empirical studies. All algorithms are conducted 30 times for each test function, and the mean function values are reported.

Table 4 presents the comparison results of NABC with three other ABC algorithms. Results of GABC, I-ABC and PS-ABC were taken from Tables 4 and 5 in [14]. The best results among the four algorithms are shown in bold. From the results, NABC outperforms GABC on all test functions except for f_2 . On this function, GABC is slightly better than NABC. I-ABC achieves better results than NABC on five functions, while NABC performs better on six functions. For the rest of f_9 , I-ABC, PS-ABC, and NABC can find the global optimum. PS-ABC obtains better results than NABC on six functions, while NABC outperforms PS-ABC on five functions. Both I-ABC and PS-ABC achieve significantly better results on three unimodal functions, such as f_1 , f_2 , and

TABLE 3: Comparison of NABC with evolution strategies.

Functions	CES Mean	FES Mean	ESLAT Mean	CMA-ES Mean	NABC Mean	Significance
f_1	1.70e - 26	$2.50e - 04$	$2.00e - 17$	$9.70e - 23$	$2.88e - 16$.
f_2	8.10e - 20	$6.00e - 02$	$3.80e - 05$	$4.20e - 11$	$1.37e - 15$.
f_3	$3.38e + 02$	$1.40e - 03$	$6.10e - 06$	7.10e - 23	$6.86e + 03$.
f_4	$2.41e + 00$	$5.50e - 03$	$7.80e - 01$	5.40e - 12	$4.30e - 01$.
f_5	$2.77e + 01$	$3.33e + 01$	$1.93e + 00$	$4.00e - 01$	2.62e - 01	+
f_6	0.00e + 00	0.00e + 00	$2.00e - 02$	$1.44e + 00$	0.00e + 00	NA
f_7	$4.70e - 02$	1.20e - 02	$3.90e - 01$	$2.30e - 01$	$1.38e - 02$	+
f_8	-8000	-12556.4	-2300	-7637.1	-12569.5	+
f_9	$1.34e + 01$	$1.60e - 01$	$4.65e + 00$	$5.18e + 01$	0.00e + 00	+
f_{10}	$6.00e - 13$	$1.20e - 02$	$1.80e - 08$	$6.90e - 12$	3.25e - 14	+
f_{11}	$6.00e - 14$	$3.70e - 02$	$1.40e - 03$	$7.40e - 04$	1.11e - 16	+
f_{12}	$1.46e + 00$	$2.80e - 06$	$1.50e - 12$	$1.20e - 04$	2.52e - 16	+

TABLE 4: Comparison of NABC with other ABC algorithms.

Functions	GABC Mean	I-ABC Mean	PS-ABC Mean	NABC Mean	Significance
f_1	$6.26e - 16$	0.00e + 00	0.00e + 00	$5.43e - 16$.
f_2	$9.36e - 16$	0.00e + 00	0.00e + 00	$6.24e - 15$.
f_3	$1.09e + 04$	$1.43e + 04$	6.11e + 03	$8.44e + 03$.
f_4	$1.26e + 01$	$1.27e - 197$	0.00e + 00	$5.79e + 00$.
f_5	$7.48e + 00$	$2.64e + 01$	$1.59e + 00$	1.45e - 01	+
f_6	$2.49e - 09$	$3.84e - 10$	$5.72e - 16$	0.00e + 00	+
f_7	$1.56e - 01$	$1.96e - 02$	$2.15e - 02$	1.72e - 02	+
f_8	-12407.3	-12251.03	-12564.2	-12569.5	+
f_9	$3.31e - 02$	0.00e + 00	0.00e + 00	0.00e + 00	NA
f_{10}	$7.78e - 10$	8.88e - 16	8.88e - 16	$1.07e - 13$.
f_{11}	$6.96e - 04$	0.00e + 00	0.00e + 00	$1.11e - 16$.
f_{12}	$5.85e - 16$	$7.11e - 12$	$5.53e - 16$	4.67e - 16	+

f_4 . On these functions, they can find the global optimum, while GABC and NABC only find near-optimal solutions except for f_4 . For f_4 , both GABC and NABC fall into local minima. I-ABC and PS-ABC successfully find the global optimum on f_{11} , while GABC and NABC fail. For function f_{10} , I-ABC and PS-ABC are slightly better than NABC. For other two multimodal functions f_8 and f_{12} , NABC performs better than other three ABC algorithms. Compared to I-ABC and PS-ABC, our approach NABC is simpler and easier to implement.

6. Conclusions

Artificial bee colony is a new optimization technique which has shown to be competitive to other population-based stochastic algorithms. However, ABC and other stochastic algorithms suffer from the same problems. For example, the convergence speed of ABC is typically slower than PSO and DE. Moreover, the ABC algorithm easily gets stuck when

handling complex multimodal problems. The main reason is that the search pattern of both employed and onlooker bees is good at exploration but poor at exploitation. In order to balance the exploration and exploitation of ABC, this paper proposes a new ABC variant (NABC). It is known that DE/best/1 mutation scheme is good at exploitation. Based on DE/best/1, a new search pattern called ABC/best/1 with solution pool is proposed. Our approach differs from other improved ABC algorithms by hybridization of DE/best/1 and ABC.

To verify the performance of our approach, a set of twelve benchmark functions are used in the experiments. Comparison of NABC with ABC demonstrates that our new search pattern can effectively accelerate the convergence speed and improve the accuracy of solutions. Another comparison demonstrates that NABC is significantly better or at least comparable to other stochastic algorithms. Compared to other improved ABC algorithms, our approach is simpler and easier to implement.

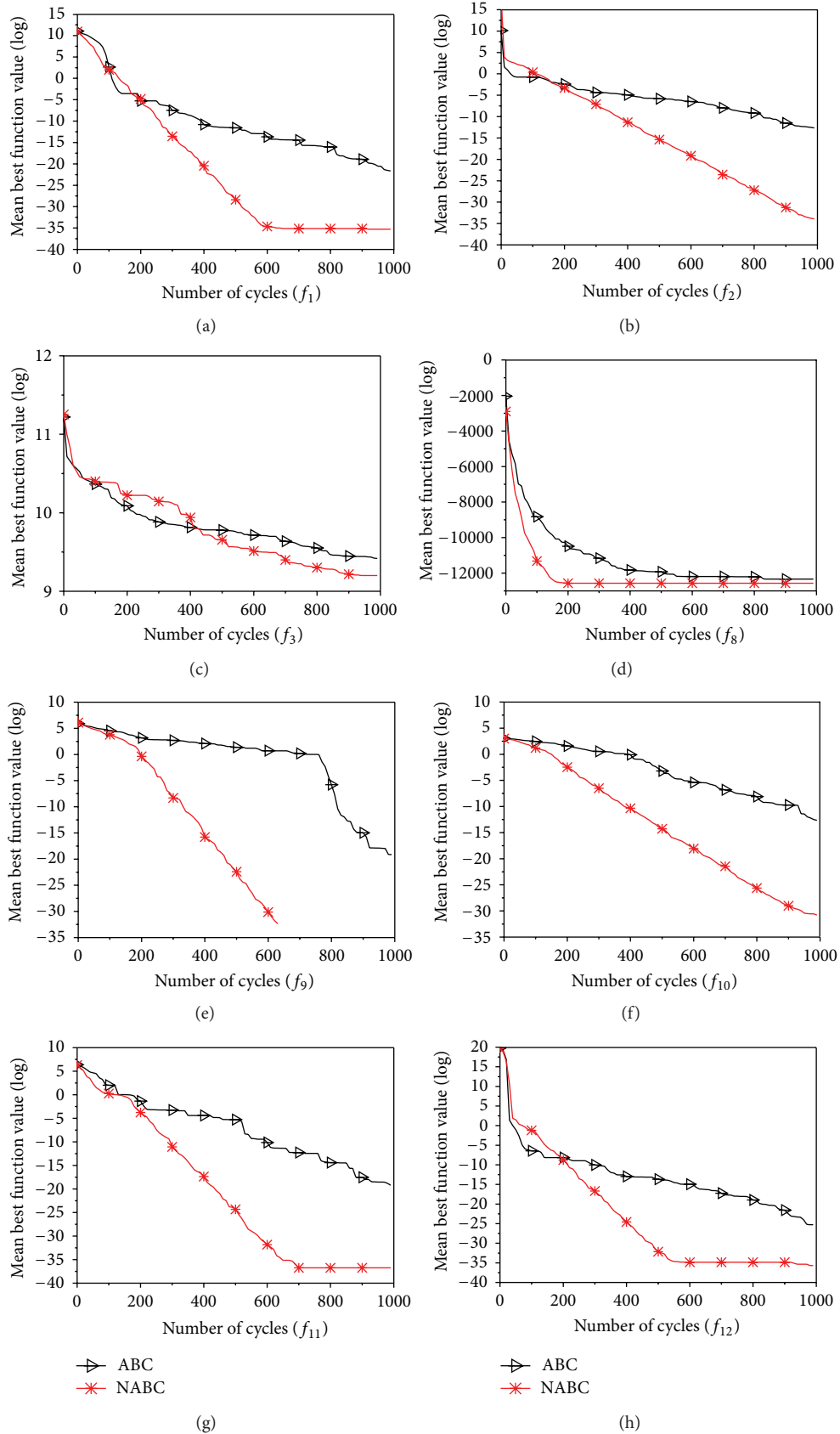


FIGURE 1: The convergence processes of ABC and NABC on some functions.

Acknowledgments

This work was supported by the National Natural Science Fund of Hubei Province under Grant 2012FFB00901, the Science and Technology Research Project of Xianning City under Grant XNKJ-1203, Doctoral Start Fund of Hubei University of Science and Technology under Grant BK1204, the Teaching Research Project of Hubei University of Science and Technology under Grant 2012X016B, Application Innovation Project of the Ministry of Public Security under Grant 2005yyxbhst117, The Key Research Project of Science and Technology of Hubei Province under Grant 2007AA301C33, Key Lab of Information Network Security of Ministry of Public Security under Grant C09602, and Application Innovative Project of Public Security of Hubei Province under Grant hbst2009sjkycx014.

References

- [1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.
- [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*, vol. 26, no. 1, pp. 29–41, 1996.
- [3] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Tech. Rep. TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.
- [4] X. S. Yang, "Firefly algorithm, stochastic test functions and design optimization," *International Journal of Bio-Inspired Computing*, vol. 2, no. 2, pp. 78–84, 2010.
- [5] 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.
- [6] D. Karaboga and B. Akay, "A survey: algorithms simulating bee swarm intelligence," *Artificial Intelligence Review*, vol. 31, no. 1–4, pp. 61–85, 2009.
- [7] W. F. Gao and S. Y. Liu, "A modified artificial bee colony algorithm," *Computers and Operations Research*, vol. 39, no. 3, pp. 687–697, 2012.
- [8] G. Zhu and S. Kwong, "Gbest-guided artificial bee colony algorithm for numerical function optimization," *Applied Mathematics and Computation*, vol. 217, no. 7, pp. 3166–3173, 2010.
- [9] W. Gao and S. Liu, "Improved artificial bee colony algorithm for global optimization," *Information Processing Letters*, vol. 111, no. 17, pp. 871–882, 2011.
- [10] B. Akay and D. Karaboga, "A modified Artificial Bee Colony algorithm for real-parameter optimization," *Information Sciences*, vol. 192, pp. 120–142, 2012.
- [11] A. Banharnsakun, T. Achalakul, and B. Sirinaovakul, "The best-so-far selection in Artificial Bee Colony algorithm," *Applied Soft Computing Journal*, vol. 11, no. 2, pp. 2888–2901, 2011.
- [12] F. Kang, J. Li, and Z. Ma, "Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions," *Information Sciences*, vol. 181, no. 16, pp. 3508–3531, 2011.
- [13] B. Wu, C. Qian, W. Ni, and S. Fan, "Hybrid harmony search and artificial bee colony algorithm for global optimization problems," *Computers & Mathematics with Applications*, vol. 64, no. 8, pp. 2621–2634, 2012.
- [14] G. Li, P. Niu, and X. Xiao, "Development and investigation of efficient artificial bee colony algorithm for numerical function optimization," *Applied Soft Computing*, vol. 12, no. 1, pp. 320–332, 2012.
- [15] D. Karaboga and C. Ozturk, "A novel clustering approach: Artificial Bee Colony (ABC) algorithm," *Applied Soft Computing Journal*, vol. 11, no. 1, pp. 652–657, 2011.
- [16] C. Zhang, D. Ouyang, and J. Ning, "An artificial bee colony approach for clustering," *Expert Systems with Applications*, vol. 37, no. 7, pp. 4761–4767, 2010.
- [17] D. Karaboga and C. Ozturk, "Fuzzy clustering with artificial bee colony algorithm," *Scientific Research and Essays*, vol. 5, no. 14, pp. 1899–1902, 2010.
- [18] 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.
- [19] E. Mezura-Montes and R. E. Velez-Koeppel, "Elitist artificial bee colony for constrained real-parameter optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1–8, 2010.
- [20] W.-C. Yeh and T.-J. Hsieh, "Solving reliability redundancy allocation problems using an artificial bee colony algorithm," *Computers & Operations Research*, vol. 38, no. 11, pp. 1465–1473, 2011.
- [21] S. L. Sabat, S. K. Udgata, and A. Abraham, "Artificial bee colony algorithm for small signal model parameter extraction of MESFET," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 5, pp. 689–694, 2010.
- [22] J. Q. Li, Q. K. Pan, S. X. Xie, and S. Wang, "A hybrid artificial bee colony algorithm for flexible job shop scheduling problems," *International Journal of Computers, Communications and Control*, vol. 6, no. 2, pp. 286–296, 2011.
- [23] M. H. Kashan, N. Nahavandi, and A. H. Kashan, "DisABC: a new artificial bee colony algorithm for binary optimization," *Applied Soft Computing*, vol. 12, no. 1, pp. 342–352, 2012.
- [24] W. Y. Szeto, Y. Wu, and S. C. Ho, "An artificial bee colony algorithm for the capacitated vehicle routing problem," *European Journal of Operational Research*, vol. 215, no. 1, pp. 126–135, 2011.
- [25] Q.-K. Pan, M. F. Tasgetiren, P. N. Suganthan, and T. J. Chua, "A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem," *Information Sciences*, vol. 181, no. 12, pp. 2455–2468, 2011.
- [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] J. Zhang and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.
- [28] R. G. Reynolds, "An introduction to cultural algorithms," in *Proceedings of the 3rd Annual Conference on Evolutionary Programming*, pp. 131–139, 1994.
- [29] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.
- [30] X. Yao and Y. Liu, "Fast evolution strategies," *Control and Cybernetics*, vol. 26, no. 3, pp. 467–496, 1997.
- [31] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '96)*, pp. 312–317, May 1996.

- [32] A. Hedar and M. Fukushima, "Evolution strategies learned with automatic termination criteria," in *Proceedings of the Conference on Soft Computing and Intelligent Systems and the International Symposium on Advanced Intelligent Systems*, pp. 1–9, Tokyo, Japan, 2006.
- [33] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 3, pp. 526–553, 2009.

Research Article

Articulated Human Motion Tracking Using Sequential Immune Genetic Algorithm

Yi Li and Zhengxing Sun

State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China

Correspondence should be addressed to Zhengxing Sun; szx@nju.edu.cn

Received 5 October 2012; Accepted 3 December 2012

Academic Editor: Baozhen Yao

Copyright © 2013 Y. Li and Z. Sun. 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.

We formulate human motion tracking as a high-dimensional constrained optimization problem. A novel generative method is proposed for human motion tracking in the framework of evolutionary computation. The main contribution is that we introduce immune genetic algorithm (IGA) for pose optimization in latent space of human motion. Firstly, we perform human motion analysis in the learnt latent space of human motion. As the latent space is low dimensional and contains the prior knowledge of human motion, it makes pose analysis more efficient and accurate. Then, in the search strategy, we apply IGA for pose optimization. Compared with genetic algorithm and other evolutionary methods, its main advantage is the ability to use the prior knowledge of human motion. We design an IGA-based method to estimate human pose from static images for initialization of motion tracking. And we propose a sequential IGA (S-IGA) algorithm for motion tracking by incorporating the temporal continuity information into the traditional IGA. Experimental results on different videos of different motion types show that our IGA-based pose estimation method can be used for initialization of motion tracking. The S-IGA-based motion tracking method can achieve accurate and stable tracking of 3D human motion.

1. Introduction

Tracking articulated 3D human motion from video is an important problem in computer vision which has many potential applications, such as virtual character animation, human computer interface, intelligent visual surveillance, and biometrics. Despite having been attacked by many researchers, this challenging problem is still long standing because of the difficulties conducted mainly by the complicated nature of 3D human motion, self-occlusions, and high-dimensional search space.

In the previous work, two main classes of motion tracking approaches can be identified: discriminative approaches and generative approaches [1]. Discriminative methods attempt to learn a direct mapping from image features to 3D pose using training data. The mapping is often approximated using nearest neighbor [2], regression models [3] or mixture of regressors [4]. Discriminative approaches are effective and fast. However, they need a large training database and are limited to fixed classes of motion. Moreover, the inherent one-to-many mapping from 2D images to 3D poses is difficult

to learn accurately. In contrast, generative methods exploit the fact that although the mapping from visual features to poses is complex and multimodal, the reverse mapping is often well posed. Therefore, pose recovery is tackled by optimizing an object function that encodes the pose-feature correspondence [5] or by sampling posterior pose probabilities [6]. Compared with discriminative methods, generative methods are usually more accurate. However, generative methods are generally computationally expensive because one has to perform complex search over the high-dimensional pose state space in order to locate the peaks of the observation likelihood. Moreover, prediction model and initialization are also the bottlenecks of the approach in the tracking scenario. In this work, we focus on recovering 3D human pose within the generative framework.

In general, high-dimensional state space and search strategy are two main problems in generative approaches. High-dimensional pose state space makes pose analysis computationally expensive or even infeasible. Despite the high dimensionality of the configuration space, many human motion activities lie intrinsically on low-dimensional latent

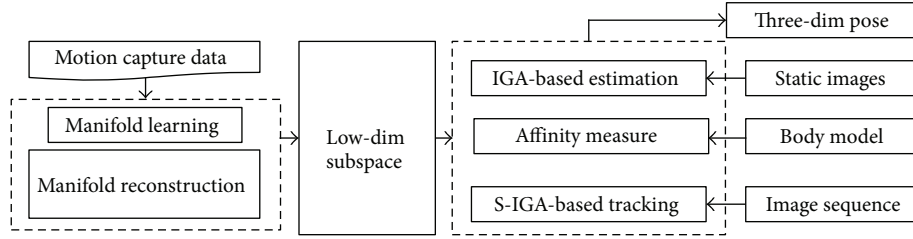


FIGURE 1: The framework of our approach.

space [7, 8]. Motivated by this observation, we use ISOMAP, a nonlinear dimensionality reduction method, to learn the low-dimensional latent space of pose state, by which the aim of both reducing dimensionality and extracting the prior knowledge of human motion are achieved simultaneously. On the other hand, search strategy, in general how to track in the low-dimensional latent space, is another important problem. The search strategy should suit for the characteristics of the subspace and be global, optimal, and convergent. Although considerable work has already been done, a more effective search strategy is still intensively needed for robust visual tracking. In our opinion, motion prior knowledge has great influence on the search strategy, which can aid in performing more stable tracking. Compared with the previous methods, extracting the prior knowledge and introducing it in the designing of search strategy are of particular interests to us.

In this paper, we propose a novel generative approach in the framework of evolutionary computation, by which we try to widen the bottlenecks mentioned above with effective search strategy embedded in the extracted state subspace. The framework of our approach is illustrated in Figure 1. Firstly, we use ISOMAP to learn this latent space. Then we propose a manifold reconstruction method to establish the inverse mapping, which enables pose analysis in this latent space. As the latent space is low dimensional and contents the prior knowledge of human motion, it makes pose analysis more efficient and accurate. In the search strategy we introduce immune genetic algorithm (IGA) for pose optimization. Details of the implementations, such as encoding and initialization, computation of affinity, and genetic and immunity operators, are designed. We propose an IGA-based method for pose estimation, which can be used for initialization of motion tracking. In order to make IGA suitable for human motion tracking, a sequential IGA (S-IGA) framework is proposed by incorporating the temporal continuity information into the traditional IGA. Experimental results on different motion types and different image sequences demonstrate our methods.

The rest of the paper is organized as follows. Section 2 gives an introduction to the related works. Section 3 gives a description of how the latent space is learnt. In Section 4, we give a detailed description of how we apply IGA for pose optimization in the latent space. We then show how to apply IGA-based pose optimization algorithm for pose estimation and tracking in Section 5. Section 6 contains experimental results and comparison with other tracking algorithms. The conclusions and possible extension for future work are given in Section 7.

2. Related Works

There has been a great deal of prior works on human motion analysis from video [1, 8, 9]. Here we focus our survey on the most related research on generative methods. In generative human motion tracking methods, the high-dimensional pose state space is the most significant problem. There are several possible strategies for reducing the dimensionality of the configuration space, including using motion models [10], hierarchical search [11], and dimensionality reduction [5, 12]. Motion models are often derived from training data of a single class of movement. Although they can aid in performing more stable tracking, this comes at the cost of putting a strong restriction on the poses that can be recovered. Another way to constrain the configuration space is to perform a hierarchical search. For example, John et al. [11] proposed a hierarchical particle swarm optimization method to search the best pose hierarchically. An inherent problem with this approach is the need to estimate accurately the position and orientation of the initial body segment (typically the torso), as a wrong pose estimate for the initial segment can distort the pose estimates for subsequent limbs. Nowadays, dimensionality reduction has become the most widely used methods. For example, Urtasun et al. [12] construct a differentiable objective function based on the Principle Component Analysis (PCA) of motion capture data and then find the poses of all frames simultaneous by optimizing a function in low-dimensional space. However, this method needs many example sequences of data to perform PCA, and all of these sequences must keep the same length and same phase by interpolating and aligning. Zhao and Liu [5] use PCA to learn the low-dimensional state space of human pose and perform pose analysis in the latent space. However since the mapping between the original pose space and the latent space is in general nonlinear, linear PCA is inadequate. Nonlinear dimensionality reduction methods have also been used. For example, Sminchisescu and Jepson [13] use spectral embedding to learn the embedding which is modeled as a Gaussian mixture model. Radial Basis Functions (RBFs) are learned for inverse mapping. A linear dynamical model is used for tracking. Elgammal and Lee [14] learn view-based representations of activity manifolds using nonlinear dimensionality reduction method (LLE). Then, the nonlinear mappings from the embedding space into both visual input space and 3D pose space are learnt using the generalized radial basis function. Although nonlinear dimensionality reduction methods can learn this nonlinear mapping, they are not invertible. The smooth inverse mapping is still a not

well-solved problem. In this paper, we use ISOMAP [15], a nonlinear dimensionality reduction method, to learn the low dimensionality subspace of a specific activity. And then, based on the intrinsic executive mechanism of ISOMAP, a manifold reconstruction method is proposed to generate smooth mappings between the subspace and the original space. This enables us to perform human motion tracking in the learned subspace.

Search strategy is another key research problem of pose tracking in the generative framework. They are typically tackled using either deterministic methods or stochastic methods. Deterministic methods usually involve a gradient descent search to minimize a cost function [16]. Although these methods are usually computationally efficient, they easily become trapped in local minima. In contrast, stochastic methods introduce some stochastic factors into the searching process in order to have a higher probability of reaching the global optimum of the cost function. Particle filter [6] is the most widely studied stochastic method which is based on Monte Carlo sampling. Although, in theory, particle filter is very suitable for tracking, it needs a large number of particles to approximate the posterior distributions, and it tends to suffer from sample impoverishment, so that the final particle sets cannot represent the true distributions. Therefore, many improvements on the traditional particle filter have been proposed. For example, Deutscher et al. [17] introduced the annealed particle filter which combines a deterministic annealing approach with stochastic sampling to reduce the number of samples required. At each time step the particle set is refined through a series of annealing cycles with decreasing temperature to approximate the local maxima in the fitness function. In Krzeszowski et al. [18], a particle swarm optimization algorithm is utilized in the particle filter to shift the particles toward more promising configurations of the human model. Compared with the deterministic counterparts, stochastic methods are usually more robust, but they suffer a large computational load, especially in high-dimensional state space. In recent years, evolutionary computing methods, such as genetic algorithm [5] and particle swarm optimization [11, 18, 19], have received increasing attention. For example, Zhao and Liu [5] proposed an annealed genetic algorithm to track human motion in compact base space, where the base space is learned using PCA. John et al. [11] proposed a hierarchical particle swarm optimization (HPSO) algorithm for articulated human tracking. Their comparative experimental results show that HPSO is more accurate than particle filter. However, based on our experimental results, due to the high-dimensional pose state space and imperfect image observations, HPSO may deviate from the pose state space and result in inaccurate tracking. Evolutionary algorithms are all good searching algorithms with an iterative process of generation and test. Two operators, crossover and mutation, give each individual the chance of optimization and ensure the evolutionary tendency with the select mechanism of survival of the fittest. However, the two operators change individuals randomly and indirectly under some conditions. Therefore, they not only give individuals the evolutionary chance but also cause certain degeneracy. Recently, immune algorithms have been another

hotspot succeeding genetic algorithm and particle swarm optimization for its success in solving pattern recognition and optimization problems. Its main advantage, compared with GA and PSO, is it has the ability to use the prior knowledge of problem by vaccination and immune selection [20]. In this paper, we apply immune genetic algorithm (IGA) [20], a novel immune method, for pose optimization. We propose an IGA-based method for pose estimation from monocular images. In order to make IGA suitable for pose tracking, we propose a sequential IGA (S-IGA) algorithm by incorporating the temporal continuity information into the traditional IGA. To the best of our knowledge, the proposed algorithm is new in the human motion tracking literature.

3. Learning the Latent Space of Human Motion

Tracking in a low-dimensional latent space requires three components [8]. First, a mapping between original pose space and low-dimensional subspace must be learned. Second, an inverse mapping must be defined. Third, how tracking within the low-dimensional space occurs must be defined. In this section, we first learn the low-dimensional subspace using ISOMAP [15]. Then, we propose a manifold reconstruction method to establish the mappings between high- and low-dimensional states.

3.1. ISOMAP-Based Latent Space Learning. We describe the human body as a kinematic tree consisting of rigid limbs that are linked by joints. Every joint contains a number of degrees of freedom (DOF), indicating in how many directions the joint can move. All DOF in the body model together form the pose representation. In this paper, the pose is described by a 66D vector $x = \{x_r, x_k\}$, where 3D vector x_r represents the root joint rotations and 63D vector x_k represents the body joints rotations. Apart from the kinematic structure, the human shape is also modeled. Each rigid limb of the body is fleshed out using conic sections with elliptical cross-sections (see Figure 2). Human shape will be used to compute the likelihood function (see Section 4.2).

Since the mapping between the original pose space and latent space is in general nonlinear, linear PCA is inadequate. So we use ISOMAP to learn the nonlinear mapping. We extract the subspace using motion capture data obtained from the CMU database [21].

As for a special activity, such as walking, running, jumping, and so forth, the original pose state space has no relation with the global motion. Different from the previous methods of learning different manifolds for the same activity (such as walking) of different views, we filter out the rotations of root joint (x_r) and represent the pose using the rotations of body joints (x_k) only. Assuming $\{x_i \mid x_i \in X, i = 1, \dots, l\}$ is a given sequence of motion capture data corresponding to one motion type, where $x_i = (x_k)_i$, i is the frame index, l is the number of total frames, and X is the original pose state space, the subspace Y is extracted by ISOMAP as follows.

- (1) *Construct Neighborhood Graph.* Define the graph G over all data points (in our method the data point is one frame in motion sequence) by connecting point

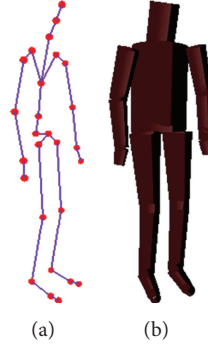


FIGURE 2: (a) The 3D human skeleton model. (b) The shape model.

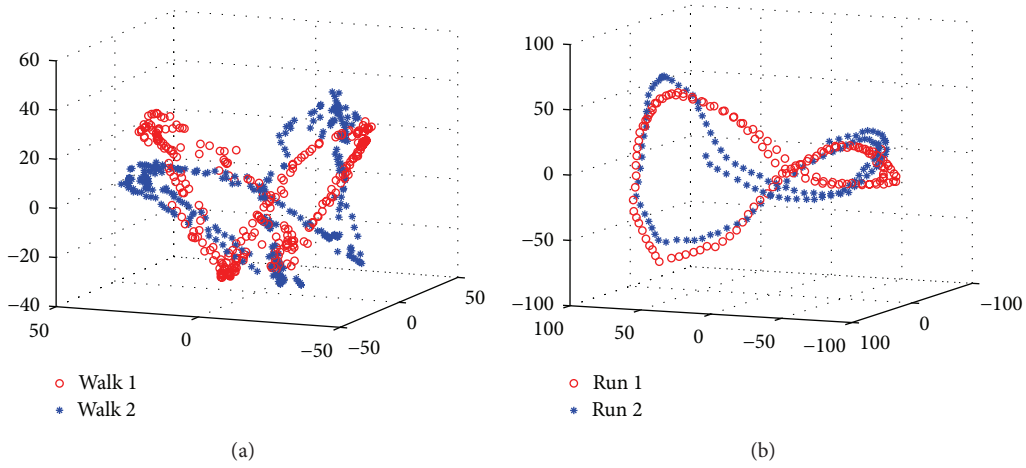


FIGURE 3: ISOMAP-based dimensionality reduction results. (a), (b) are manifolds of two sequences of walking and running in 3D subspace, respectively.

x_i and x_j if $d_x(x_i, x_j) < \epsilon$. Set edge length to be $d_x(x_i, x_j)$. Moreover,

$$d_x(x_i, x_j) = \sum_{m=1}^M \frac{\|(x_i)_m - (x_j)_m\|}{M}, \quad (1)$$

where M is the dimensionality of pose state space; $M = 63$ here.

- (2) *Compute Shortest Paths.* Initialize $d_G(x_i, x_j) = d_x(x_i, x_j)$ if x_i and x_j are linked by an edge; $d_G(x_i, x_j) = \infty$ otherwise. Then for each value of $i = 1, 2, \dots, l$ in turn, replace all entries $d_G(x_i, x_j)$ by $\min\{d_G(x_i, x_j), d_G(x_i, x_t) + d_G(x_t, x_j)\}$. The matrix of final values $D_G = \{d_G(x_i, x_j)\}$ will contain the shortest path distances between all pairs of points in G .
- (3) *Construct d -Dimensional Embedding.* Let λ_p be the p th eigenvalue (in decreasing order) of the matrix $\tau(D_G)$ and v_p^i the i th component of the p th eigenvector. Then set the p th component of the d -dimensional coordinate vector y_i to be equal to $\sqrt{\lambda_p} v_p^i$.

The subspace Y learned by ISOMAP is shown in Figure 3. Actually, similar low-dimensional subspace can be

extracted from the training sequences that belong to the same type of motions but performed by different subjects. And the training sequences corresponding to different types of motions produce different subspaces. For example, experiments demonstrate that different walking sequences generate similar manifolds in the 3D subspace, which is different from that of running motion.

ISOMAP cannot only reduce the dimensionality of high-dimensional input space, but also find meaningful low-dim structures hidden behind their high-dim observations. In doing so, infeasible solutions, namely, the absurd poses, can be avoided naturally during optimization, which will make pose tracking in this subspace more efficient and accurate.

3.2. Mapping between High- and Low-Dimensional States. Traditional ISOMAP can only learn the mapping from the original pose space to the latent space but not the inverse mapping. However, in order to track human motion in the low-dimensional manifold, the inverse mapping is required. Based on the intrinsic executive mechanism of ISOMAP, we proposed an ISOMAP-based manifold reconstruction method to establish the mapping between high- and low-dimensional states.

Input: The training data set $\{x_i \mid x_i \in X, i = 1, \dots, l\}$.

Output: The mappings $g : X \rightarrow Y, y = g(x)$ and $f : Y \rightarrow X, x = f(y)$.

Step 1: (preparing)

(1) Using the ISOMAP algorithm to compute the low-dim vector $\{y_i \mid y_i \in Y, i = 1, \dots, l\}$ for the original input vector $\{x_i \mid x_i \in X, i = 1, \dots, l\}$;

(2) Construct the following matrixes:

$X_i = (x_{i_1} - x_i, \dots, x_{i_{l_i}} - x_i) \in R^{n \times l_i}, Y_i = (y_{i_1} - y_i, \dots, y_{i_{l_i}} - y_i) \in R^{d \times l_i}$, where $\{x_{i_j} \mid j = 1, \dots, l_i\}$ is the ε -neighbor of x_i .

(3) Compute $Q_i = X_i Y_i^T (Y_i Y_i^T)^G \in R^{n \times d}$, where $(Y_i Y_i^T)^G$ is the generalized inverse matrix of $Y_i Y_i^T$.

Step 2: (manifold reconstruction)

(1) Mapping from original space to latent space: $g : X \rightarrow Y, y = g(x)$.

Given a high-dim pose vector x_0 , the corresponding low-dim vector y can be computed as:

(1.1) Find the nearest neighbor of x_0 in $\{x_i \mid i = 1, \dots, l\}$, set it to be x_s ;

(1.2) The low-dim vector correspondence to x_0 can be formulated as:

$$y = \tilde{g}(x_0) = y_s + Q_s^T(x_0 - x_s);$$

(2) Mapping from latent space to original space: $f : Y \rightarrow X, x = f(y)$.

Given a low-dim pose vector y_0 , the corresponding high-dim vector x can be computed as:

(2.1) Find the nearest neighbor of y_0 in $\{y_i \mid i = 1, \dots, l\}$, represented as y_s ;

(2.2) The high-dim vector correspondence to y_0 can be formulated as:

$$x = \tilde{f}(y_0) = x_s + Q_s(y_0 - y_s).$$

ALGORITHM 1: ISOMAP-based manifold reconstruction.

Suppose the pose state space to be $X \subset R^n$ and the low-dim state space to be $Y \subset R^d$. Denote the mapping as $f : Y \rightarrow X, x = f(y)$ and $g : X \rightarrow Y, y = g(x)$, where x, y are the high- and low-dimensional vectors, respectively. The set of input instances is $\{x_i \mid x_i \in X, i = 1, \dots, l\}$, and their corresponding points in the embedding space learned by ISOMAP are $\{y_i \mid y_i \in Y, i = 1, \dots, l\}$. Assume $\{x_{i_j} \mid j = 1, \dots, l_i\}$ are the ε -neighbors of point x_i , where l_i is the number of neighbors. And their corresponding points in the embedding space are $\{y_{i_j} \mid j = 1, \dots, l_i\}$. Then our ISOMAP-based manifold reconstruction method can be described as in Algorithm 1.

Using the ISOMAP-based manifold reconstruction method, we can generate smooth mapping between the original pose space and the latent space, which enables us to track human pose in the latent space. In the following section, we will show how tracking within the latent space occurs.

4. Immune Genetic Algorithm for Pose Optimization

We formulate pose estimation as a constrained optimization problem and solve it using immune genetic algorithm. In this section, we first give a brief introduction to IGA. Then, we design an IGA-based method for pose optimization.

4.1. Immune Genetic Algorithm. In IGA, the idea of immunity is mainly realized through two steps based on reasonably selecting vaccines, that is, a vaccination and an immune selection, of which the former is used for raising fitness and the latter is for preventing the deterioration. A very clear overview of IGA, from immunology and engineering points of view, is presented in [20].

4.1.1. Immunological Terms. In order to describe the IGA clearly, some immunological terms will be given first [22].

Antigen. In immunology, an antigen is any substance that causes immune system to produce antibodies against it. In this paper, IGA is used for optimization:

$$\text{Minimize } f(x), \quad (2)$$

where $x = [x_1, x_2, \dots, x_D] \in F$, F is the feasible region, D is the number of problematic parameter, and the antigen is defined as the objective function $f(x)$.

Antibody and Antibody Population. In this paper, an antibody is a representation of a candidate solution of an antigen. The antibody $\vec{a} = [a_1, a_2, \dots, a_D]$ is the coding of variable x , denoted by $a = e(x)$, and x is called the decoding of antibody a , expressed as $x = e^{-1}(a)$. The representation of antibody a varies with antigen and can be binary string, real number sequence, symbolic sequence, and characteristic sequence. In this study, we adopt real-coded representation, that is, $a = e(x) = x$.

An antibody population,

$$A = \{a_1, a_2, \dots, a_N\}, \quad a \in F, \quad 1 \leq i \leq N, \quad (3)$$

is an N -dimensional group of antibody a , where the positive integer N is the size of antibody population A .

Affinity. In immunology, affinity is the fitness measurement for an antibody. For the optimization problem, the affinity, $\text{Affinity}(a) \geq 0$, is a mapping of the objective function $f(x)$ for a given antibody a .

4.1.2. Description of IGA. In this paper, we use the IGA for optimization task. The flow chart of IGA is shown in Figure 4.

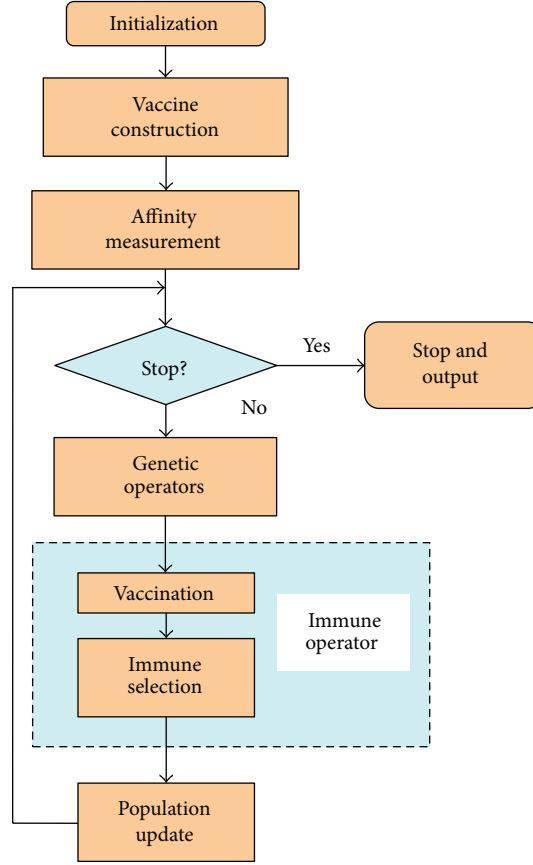


FIGURE 4: Flow chart of IGA for pose optimization.

The main steps of our modified IGA can be summarized as follows.

Step 1. Initialization: randomly generate the initial antibody population $A(k)$; set $k = 0$.

Step 2. Vaccine construction: abstract vaccines according to the prior knowledge.

Step 3. Evaluation: calculate the affinities of all antibodies in $A(k)$.

Step 4. Termination test: if termination criterion is satisfied, export the antibody having the highest affinity in $A(k)$ as the output of the algorithm and stop the algorithm; otherwise, continues.

Step 5. Genetic operators: perform genetic operators on the k th parent $A(k)$ and obtain the results $B(k)$.

Step 6. Vaccination: perform vaccination on $B(k)$ and obtain the results $C(k)$.

Step 7. Immune selection: perform immune selection on $C(k)$ and obtain the next parent $A(k + 1)$, then go to Step 3.

In general, the IGA algorithm is to be implemented as the following evolution process:

$$A(k) \xrightarrow{T_g^p} B(k) \xrightarrow{T_v^p} C(k) \xrightarrow{T_s^p} A(k+1), \quad (4)$$

where $A(k)$, $B(k)$, and $C(k)$ are the antibody populations during different periods in a single evolution generation, k is the iterative step. T_g^p , T_v^p , and T_s^p are the genetic, vaccination, and immune selection operators, respectively.

4.2. Apply IGA for Pose Optimization. In this section, we apply IGA for human pose optimization. Some details of our implementations are discussed below.

4.2.1. Encoding and Initialization. In IGA, each antibody represents a potential solution in the search space. For our problem, we perform human motion analysis in the latent space. So an antibody is corresponding to a pose vector in the latent space. In this paper, we represent the full 3D pose vector as $x = \{x_r, y\}$, where 3D vector $x_r = (r_x, r_y, r_z)$ represents the root joint rotations, $y = (y_1, \dots, y_d)$ corresponds to the pose vector in latent space; we set $d = 6$ here. So x is a 9-dimensional vector. We use real encodings. We represent the

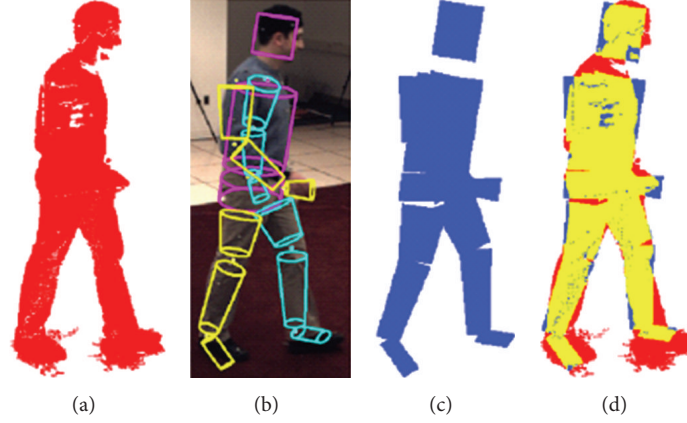


FIGURE 5: Silhouette-based affinity measurement, a bidirectional likelihood version [23].

TABLE 1: The genetic operators in IGA.

Operator	Example
Exchange	$y = (y_1, y_2, y_3, y_4, y_5, y_6) \rightarrow y' = (y_1, y_6, y_3, y_4, y_5, y_2)$
Segment reversion	$y = (y_1, y_2, y_3, y_4, y_5, y_6) \rightarrow y' = (y_1, y_6, y_5, y_4, y_3, y_2)$
Segment shift	$y = (y_1, y_2, y_3, y_4, y_5, y_6) \rightarrow y' = (y_1, y_6, y_2, y_3, y_4, y_5)$
Point mutation	$y = (y_1, y_2, y_3, y_4, y_5, y_6) \rightarrow y' = (y_1, y_2, y'_3, y_4, y_5, y_6)$
Segment mutation	$y = (y_1, y_2, y_3, y_4, y_5, y_6) \rightarrow y' = (y_1, y'_2, y'_3, y'_4, y'_5, y'_6)$

antibody population as $A = \{x_1, x_2, \dots, x_N\}$, where N is the size of population.

4.2.2. Computation of Affinity. For each antibody, an affinity measure needs to be computed to estimate how well a given antibody (pose) matches the observed images. Here we use the bidirectional likelihood proposed by [23]. Let M^b represent the binary silhouette map for the body model and M^f the image foreground. We seek to minimize the non-overlapping regions, red and blue, therefore maximizing the Yellow region (see Figure 5). The size of each region can be computed by summing over all image pixels p using

$$\begin{aligned}
 R &= \sum_p (M^f(p) (1 - M^b(p))), \\
 B &= \sum_p (M^b(p) (1 - M^f(p))), \\
 Y &= \sum_p (M^f(p) M^b(p)).
 \end{aligned} \tag{5}$$

Then the objective function of candidate pose x with regard to image z can be calculated as

$$f(x) = (1 - \gamma) \frac{B}{B + Y} + \gamma \frac{R}{R + Y}, \tag{6}$$

where γ is the weight. We set $\gamma = 0.5$ in this paper.

Affinity is the fitness measurement for an antibody. As defined above, the affinity, $\text{Affinity}(x)$, for a given antibody x

is a linear mapping of the objective function $f(x)$. Therefore, we define the affinity of antibody x as

$$\begin{aligned}
 \text{Affinity}(x) &= S^* \exp(-f(x)) \\
 &= S^* \exp\left(-\left((1 - \gamma) \frac{B}{B + Y} + \gamma \frac{R}{R + Y}\right)\right),
 \end{aligned} \tag{7}$$

where S is a positive constant; we set $S = 100$ in this paper.

4.2.3. Genetic Operators. We design five genetic operators, which are executed orderly in IGA. We introduce the operators by evolving an example antibody $x = (x_r, y) = (x_r, y_1, y_2, y_3, y_4, y_5, y_6)$. The new antibody generated by the operators is denoted as $x' = (x'_r, y')$. Assuming the positions generated randomly are numbers 2 and 6 or 3 (for point mutation operator) of $y = (y_1, y_2, y_3, y_4, y_5, y_6)$, for example, the five operators are illustrated in Table 1. The application order of the genetic operators in the algorithm is just as that listed in Table 1.

The genetic operators were represented as T_g^p . We perform genetic operator on the k th parent $A(k)$ and obtain the results $B(k)$.

4.2.4. Vaccine Construction and Immune Operators. Genetic operators give each antibody the chance of optimization and ensure the evolutionary tendency with the select mechanism of survival of the fittest. However, it changes individuals randomly and indirectly under some conditions. Therefore, they not only give individuals the evolutionary chance but also cause certain degeneracy. In IGA, the idea of immunity is mainly realized through two steps based on reasonably

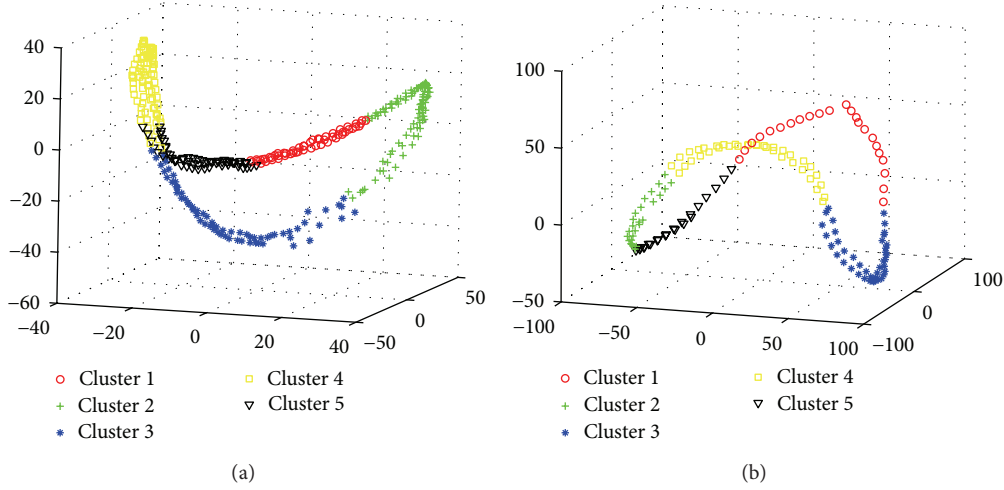


FIGURE 6: Clustering of human pose in 3D subspace, where (a), (b) represent the results of walking and running data, respectively.

selecting vaccines, that is, a vaccination and an immune selection, of which the former is used for raising fitness and the latter is for preventing the deterioration.

In this section, we extract the prior knowledge of human motion and construct two vaccines. Then we design the vaccination and immune selection operators.

(1) *Vaccine Construction*. A vaccine is abstracted from the prior knowledge of the pending problem. Human pose in subspace is located on a manifold structure but not the whole subspace. Actually, pose subspace is a compact space. We constrained the subspace of human motion and construct two vaccines for our human pose estimation problem.

- (i) *Vaccine 1*. Every dimensionality y_i of subspace pose $y = (y_1, y_2, \dots, y_d)$ should be distributed in a scope as $\min(y_i) < y_i < \max(y_i)$, where the bound $\max(y_i)$ and $\min(y_i)$ are learned from the motion training data.
- (ii) *Vaccine 2*. Vaccine 2 is motivated by the fact that every generated pose should locate on the manifold. Based on the consistency of human motion, we partition the manifolds into different subparts with k -means clustering, where the number of class c is 5 in this paper (see Figure 6). For each class c_i , $i = 1, \dots, 5$, we assume the poses in it is of Gaussian distribution, described as follows:

$$p_i(Y) = \frac{1}{(2\pi)^{d/2} |\Sigma_i|^{1/2}} e^{(-1/2)(Y-\mu_i)^T \Sigma_i^{-1} (Y-\mu_i)}, \quad (8)$$

$$i = 1, \dots, 5,$$

where, μ_i is the mean vector, Σ_i is the covariance matrix, $d = 6$ is the dimensionality of the pose subspace. Then the vaccine 2 can be described as for all $y \in Y$, $\exists i$, such that $p_i(y) > \chi$, where $i = 1, \dots, 5$.

- (2) *Vaccination*. A vaccination means the course of modifying the genes of an individual $x = (x_r, y)$ on some bits in accordance with prior knowledge so as to gain higher fitness with greater probability. For an antibody $x' = (x'_r, y')$, $x'_r \in B(k)$ generated using genetic operators, we perform vaccination operator on x' to generate a new antibody $x'' = (x''_r, y'')$.

Inoculation of Vaccine 1. Vaccine 1 indicates that every dimensionality y'_i of subspace pose $y' = (y'_1, y'_2, \dots, y'_6)$ should be distributed in a scope. When it moves out of this scope, we set it to be the border. The process can be formulated as

$$\begin{aligned} & \text{if } (y'_i < \min(y_i)), \\ & y'_i = \min(y_i); \quad \text{if } (y'_i > \max(y_i)), \\ & y'_i = \max(y_i), \quad \text{where } i = 1, \dots, 6. \end{aligned} \quad (9)$$

Inoculation of Vaccine 2. Vaccine 2 indicates that every pose $y' = (y'_1, y'_2, \dots, y'_d)$ should locate on the manifold. If a pose y' does not locate on the manifold, that is, $p_i(y') < \chi$, for $i = 1, \dots, 5$, we first calculate to which class it is most likely to belong, suppose it to be c_i . Then, we set y' to be a random antibody in this class.

The vaccination operator was represented as T_v^p . We perform vaccination operator on the k th parent $B(k)$ and obtain the results $C(k)$.

- (3) *Immune Selection*. This operation is accomplished by the following three components. The first one is the immune test, that is, testing the antibodies. If the affinity is better than that of the parent, we add it to a temporal population $D(k)$, $D(k) = \{x'' \mid \text{Affinity}(x'') > \text{Affinity}(x)\}$. The second one is the annealing selection, that is, if the affinity is worse than

Input: total number of antibodies N , maximum number of generations K .

Steps:

(1) **Initialization:** generate the initial population with N antibodies. The initial population can be represented as $A(0) = \{x_1(0), x_2(0), \dots, x_N(0)\}$.

(2) **Repeat:**

While ($k < K$) **do**

(2.1) Genetic operator: Perform genetic operators T_g^p on the k th parent $A(k)$ and obtain the results $B(k)$;

(2.2) Vaccination: Perform vaccination T_v^p on $B(k)$ and obtain $C(k)$;

(2.3) Immune selection: Perform immune selection T_s^p on $C(k)$ and obtain the next generation population $A(k+1)$;

(2.4) $k = k + 1$.

End while

(3) **Output:** output the population $A(K) = \{x_1(K), x_2(K), \dots, x_N(K)\}$.

ALGORITHM 2: IGA-based pose optimization algorithm.

that of the parent, select an individual in the present population $C(k)$ to join in the temporal population $D(k)$ with the probability as follows:

$$P(x'') = \frac{\text{Affinity}(x'') * e^{\text{Affinity}(x'')/T_k}}{\sum_{i=1}^N \text{Affinity}(x'') * e^{\text{Affinity}(x'')/T_k}}, \quad (10)$$

where $\text{Affinity}(x'')$ is the affinity of the individual x'' and T_k is the temperature controlled series tending towards 0. The third one is the next population generation. We design a hybrid $(\mu + \lambda)$ evolutionary strategy to generate the new generation $A(k+1)$. $(\mu + \lambda)$ evolutionary strategy means selecting the first N individuals from the current population $A(k)$ (with the size of N) and temporal population $D(k)$ (with the size of M) to compose a new parent population $A(k+1)$.

4.3. IGA-Based Human Pose Optimization. Based on the description above, the IGA-based pose optimization algorithm can be described as in Algorithm 2.

We will show how to apply IGA-based pose optimization algorithm for pose estimation and tracking in the next section.

5. Sequential Immune Genetic Algorithm for Pose Tracking

In tracking applications, the data is typically a time sequence, and hence the task is essentially a dynamic optimization problem which distinguishes it from traditional optimization problems. In tracking situation, the previous estimation results can be used to cut the current search space. From the Bayes' view, we can formulate the pose tracking problem as

$$p(x_t | z_t) \propto p(z_t | x_t) p(x_t | x_{t-1}), \quad (11)$$

where $\{x_t | t = 1, 2, \dots, T\}$ and $\{z_t | t = 1, 2, \dots, T\}$ represent temporal states and observations, respectively. How to

determine the conditional distribution $p(x_t | x_{t-1})$ effectively is the core problem for 3D human pose tracking.

In this paper, we proposed a sequential IGA- (S-IGA)-based framework for human motion tracking. The flowchart of the S-IGA framework is shown in Figure 7. First, we perform human pose estimation on the first frame of the video as initialization for tracking. Then, the previous converged antibodies at time t are randomly propagated as initial antibodies for the next time (frame) $t+1$. Finally, we perform IGA-based pose optimization on current antibodies. The individual with best affinity is used to approximate the tracking result of time $t+1$, and the converged antibodies are used to initial the next frame. There are three major stages in the S-IGA framework: automatically initialization, next frame propagation, and IGA-based optimization.

5.1. Pose Estimation for Initialization of Motion Tracking. Initialization is an important problem of human motion tracking. How to begin the tracking process from a good starting point sometimes is an intractable problem. We achieve the automatic initialization by determining the pose of the first frame using the IGA-based human pose estimation algorithm, which can be described as follows.

Pose estimation is the process to estimate articulated human pose from a single image which can be formulated as an optimization process. We apply IGA for pose estimation. For clarity, we redefine the full 3D pose vector as $x = \{x_r, y\}$, where x_r is the global motion of human body with respect to the camera and y is the pose vector in state subspace. We perform the state posterior inference by optimizing the affinity function. The optimal pose can be represented as

$$x = \arg \max_x (\text{Affinity}(x)). \quad (12)$$

We maximize the search efficiency by embedding the global search capability of IGA into the local conditions of state subspace.

The global motion of human body is very important for its visual appearance in an image and is also critical in disambiguating the left-right confusion. Determining this motion

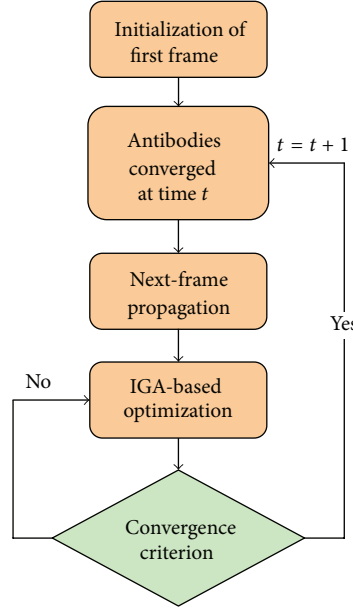


FIGURE 7: Overview of the sequential IGA.

accurately makes our method viewpoint invariant. With the aim of both cutting the search space and determining the motion direction roughly, we incorporated the global motion process step [5] into the framework of IGA. The global motion process can be summarized as follows. (1) In state vector $x = \{x_r, y\}$, the global motion $x_r = (r_x, r_y, r_z)$ include the rotation of the full body about the coordinate axes X , Y , and Z , respectively. In the first round of state evolution ($k = 1$), we only actually search the optimal solutions of global motion. Other state components (y) are taken as one of the clustering centers y_{ci} , $i = 1, \dots, 5$, randomly. The variance domain $\min(x_r)$ and $\max(x_r)$ of x_r is computed by storing the N' best antibodies. N' is determined empirically according to the threshold value of affinity. (2) In the rest rounds of state evolution, the antibody is evolved normally as described in Section 4. In doing so, we can get the coarse scopes of global motion in the first round of state evolution, and the fine tuning of these parameters can be achieved in the followed evolution rounds.

Based on the proposed IGA pose optimization algorithm, the antibody with the highest affinity in population $A(K) = \{x_1(K), x_2(K), \dots, x_N(K)\}$ will be selected to be the optimal pose. Figure 8 is the process of pose estimation, where (a) is one frame of input video, (b) is the initialized poses, (c), and (d) are results with 10, 40 times of iteration, respectively. We can see that the poses generated by our initialization method can cover the whole walking pose state space, and the poses become convergent with times of iteration increase.

5.2. Next-Frame Propagation. Next-frame propagation is the key stage in the S-IGA framework which aims to find the dynamic model $p(x_t | x_{t-1})$. In this paper, we design a randomly propagation method. The randomly propagation method is actually a first-order Gauss-Markov dynamical model. Given the converged antibodies $x_{i,t-1}(K)$ at frame $t-1$,

the antibodies in the next frame t are initialized by sampling a Gaussian distribution centered in the current best antibodies. Consider,

$$x_{i,t}(0) \sim p(x_t | x_{t-1}) = N(x_{i,t-1}(K), \Sigma), \quad (13)$$

where $x_{i,t}(0)$ are the initial antibodies at time t , $x_{i,t-1}(K)$ are the converged antibodies at time $t-1$, $i = 1, \dots, N$, and Σ is the covariance matrix of Gaussian distribution. Low value Σ will promote temporal consistency but is likely to lose the diversity. We set it empirically according to the motion type and speed. S-IGA propagates only a minimal amount of information between frames and does not incorporate any motion model. Although randomly propagation is simple, it is sufficient because it is only used to produce an initial value for a subsequent search for the optimal state.

We do not incorporate any learnt constant motion model here, which is motivated by two considerations.

- (1) Generality: many prior motion models are derived from training data. A possible weakness of these motion models is that the ability to accurately represent the space of realizable human movements generally depends significantly on the amount of available training data. This comes as a cost of putting a strong restriction on the poses that can be recovered. Therefore, we do not use any constant learnt motion models here.
- (2) The effectiveness of our IGA pose optimization algorithm, which can explore efficiently large portions of the search space starting from the initial distribution of antibodies.

Actually, the S-IGA framework is a “sample-and-refine” search strategy. Firstly, the initial antibodies are sampled for the transition distribution as $x_{i,t}(0) \sim N(x_{i,t-1}(K), \Sigma)$. Then

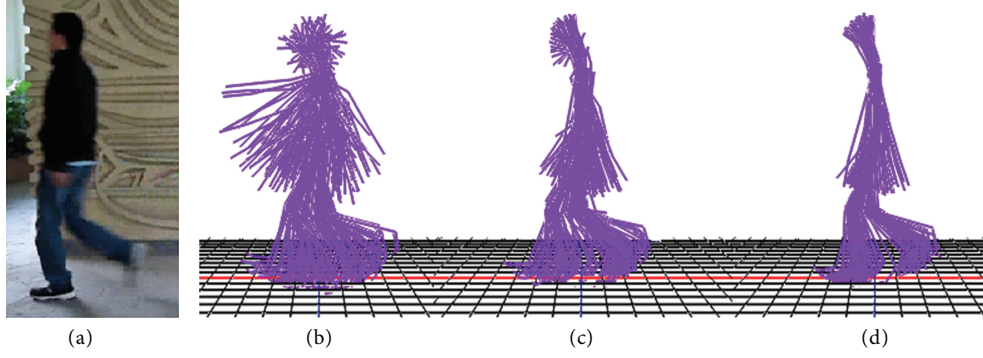


FIGURE 8: The process of human pose estimation, where (a) is a frame of input video, (b) is the initialized poses, and (c), (d) are results with different times of iteration, respectively.

- (1) **Initialization:** perform human pose estimation on the first frame of the video, output the converged antibodies $\{x_{1,t}(K), x_{2,t}(K), \dots, x_{N,t}(K)\}$, where $t = 0$;
- (2) **for** $t = 1 : T$ **do**
- (3) Next-frame propagation: randomly propagate the antibodies to enhance their diversities according to the following transition model:

$$x_{i,t}(0) \sim N(x_{i,t-1}(K), \Sigma), i = 1, \dots, N;$$
- (4) IGA-based pose optimization: using Algorithm 2 to optimize the initial antibodies:

$$\{x_{1,t}(0), x_{2,t}(0), \dots, x_{N,t}(0)\};$$
- (5) Check the convergence criterion: if satisfied, the converged antibodies are used to initial the next frame;
- (6) The individual with best affinity in population $\{x_{1,t}(K), x_{2,t}(K), \dots, x_{N,t}(K)\}$ is used to approximate the tracking result of time t ;
- (7) **end for**.

ALGORITHM 3: S-IGA-based motion tracking algorithm.

the antibodies are updated according to the newest observations in each IGA iteration. Through the IGA iteration, the antibodies are moved towards the region where the likelihood of observation has larger values and are finally relocated to the dominant modes of the likelihood. And in a Bayesian inference view, the IGA iterations are essentially a multi-layer importance sampling strategy which incorporates the new observations into a sampling stage and thus avoids the sample impoverishment problem suffered by the particle filter [6].

5.3. Sequential Immune Genetic Algorithm-Based Pose Tracking. Based on the designing above, we can formulate our sequential IGA for pose tracking as in Algorithm 3.

6. Experimental Results

6.1. Experimental Data and Evaluation Measures

Experimental Data. The data for latent space training is from CMU Database [21]. We quantitatively evaluate our method on synthesized image sequences as in [3]. We also give experimental results on real image sequences from [24], CMU Database [21], and HumanEva [23].

Evaluation Measures. In this paper, we use the evaluation measures proposed in [23]. The average error over all joint angles (in degrees) is defined as

$$D(x, \hat{x}) = \sum_{m=1}^M \frac{\|x_m - \hat{x}_m\|}{M}, \quad (14)$$

where $x = (x_1, x_2, \dots, x_M)$ and $\hat{x} = (\hat{x}_1, \hat{x}_2, \dots, \hat{x}_M)$ are the ground truth pose and the estimated pose date, respectively. For the sequence of T frames, the average performance and the standard deviation of the performance are computed using the following:

$$\mu_{\text{seq}} = \frac{1}{T} \sum_{t=1}^T D(x_t, \hat{x}_t), \quad (15)$$

$$\sigma_{\text{seq}} = \sqrt{\frac{1}{T} \sum_{t=1}^T [D(x_t, \hat{x}_t) - \mu_{\text{seq}}]^2}.$$

6.2. The Convergence of IGA. It is understood that the number of antibodies N and iteration times K will affect the

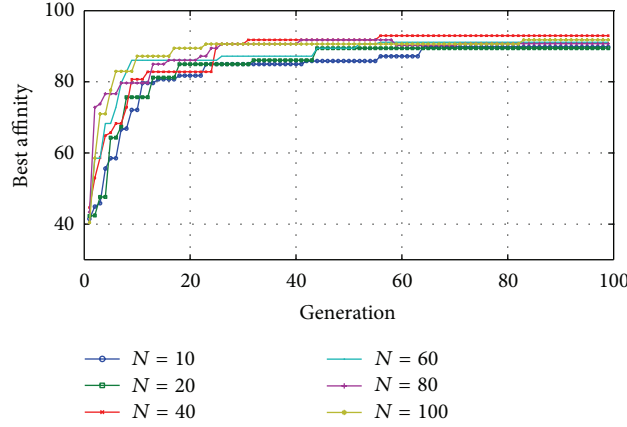


FIGURE 9: The convergence process.

convergence. We take pose estimation experiment on a single image and report the affinities of the best antibody during iteration. Figure 9 demonstrates the convergence process. Different lines represent different numbers of antibodies used. The x -axis is the times of iteration while the y -axis is the affinity value. As shown in Figure 9, the affinities will converge as the times of iteration increase. The experimental results demonstrate that our IGA-based pose optimization is convergent.

We have ascertained experimentally that higher numbers of N and K will achieve better results. However, in order to deal with the tradeoff of computational time and accuracy, we set $N = 40$, $K = 60$.

6.3. IGA-Based Pose Estimation Results. We test our IGA-based pose estimation method on three image sequences, including one straight walk sequence [24], one turning walk sequence [23], and one run sequence [21]. The purpose is to test the capability of the method to cope with limb occlusion, left-right ambiguity, and view-point problems, which are the main challenges that a pose estimation method has to deal with. As mentioned in Section 3, we first learn the subspace of walking and running. To extract the motion subspace of walking, a data set consisting of motion capture data of a single subject was used. The total number of frame is 316. For running subspace learning, a data set with 186 frames was used. It was found that the different subjects and different frame numbers can produce generally identical subspace. So the learned subspaces are also used in the tracking experiments.

For pose estimation on a single image, the parameters of IGA are set as $N = 40$ and $K = 60$ to deal with the tradeoff of computational time and accuracy. We test our IGA-based pose estimation method on 100 frames of images for all three types of motions, and the mean errors of joint angle are reported, which are shown in Figure 10. From Figure 10 we can see that, except for some particular joints, the mean errors of most joints for three sequences are less than 5 degrees. The mean errors of some joint angles are larger than

others because they have wider range of variation or less observability related to 2D image features. Our results are competitive with others reported in the related literatures.

Table 2 shows the ground truth and estimated values of some joint angles in an example frame. Three values in each cell are the rotation angles of the joints around X , Y , and Z axes, respectively. The values come from a frame on the level of average error. Actually, other frames show generally the similar comparison results. From Table 2 we can see that estimated joint angles are close to the ground truth data. The experiment results demonstrate that our IGA-based pose estimation method is effective to analyze articulated human pose from a single image.

The results on real images are shown in Figure 11. From the above experiment results, we can see that, on most of the frames, the occlusion and left-right confusion problems are tackled by searching the optimal pose in the extracted subspace because the prior knowledge about motions is contained in this subspace. And the pose estimator is view invariant, mainly because of the viewpoint-independent manifold learning and special step for searching the global motion. In addition, the experiment results on walking and running sequence demonstrate that our algorithm is efficient for different types of motions. Actually, our method can be generalized to any other types of motions as long as the corresponding subspace can be properly extracted from training data.

6.4. S-IGA-Based Pose Tracking Results. We demonstrate our tracking algorithm on walking and running image sequences. And then we compare S-IGA quantitatively with other tracking methods and include particle filter (PF) method [6], particle swarm optimization (PSO) [11], and pose tracking in linear subspace using annealing genetic algorithm (PCA + GA) [5].

As suggested in [6], for a human model with DOF between 6 and 12, PF needs about 1000 particle to run. And in [17], PF used 4000 particles for a 29 DOF human model. While in [11], 7200 particles are used for a 31 DOF human

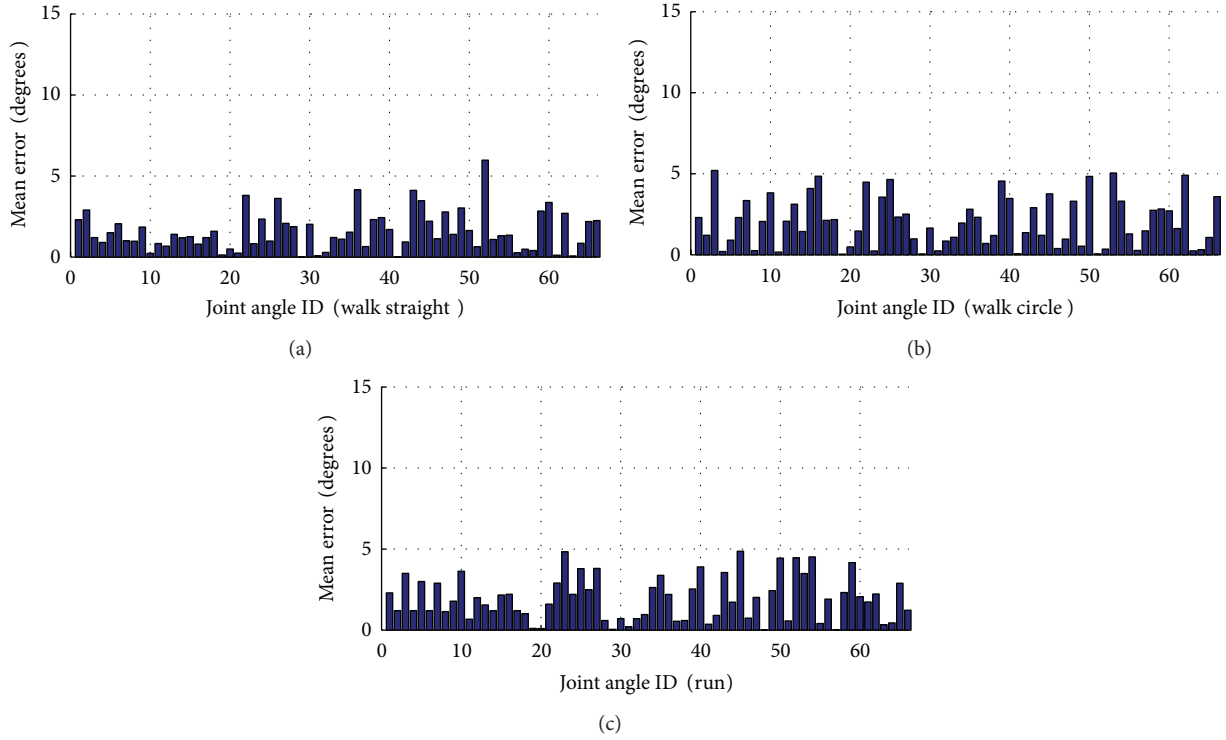


FIGURE 10: The mean errors of individual joint angle for different sequences.

TABLE 2: Ground truth (T) and estimated (E) results of some joint angles for different motions.

		Joint angles			
		L Femur	R Femur	L Knee	R Knee
Walk straight	T	(-23.235, 47.366, 13.754)	(-1.237, 6.456, 25.356)	(-3.245, 50.782, 4.567)	(-1.982, 30.425, 3.904)
	E	(-20.967, 43.459, 8.351)	(-0.923, 4.535, 26.429)	(-3.024, 4.368, 8.546)	(0.673, 30.456, 5.336)
Walk in circle	T	(-15.324, 50.339, 8.479)	(-0.923, 3.546, 20.764)	(-4.234, 59.436, 7.451)	(-1.590, 28.904, 2.405)
	E	(-16.847, 48.837, 5.435)	(-0.456, -0.345, 25.763)	(-3.458, 60.348, 5.345)	(0.890, 34.941, -1.234)
Run	T	(-10.213, 43.225, 10.863)	(0.456, 6.433, 24.567)	(-0.932, 49.687, 8.891)	(-0.379, 34.227, 7.904)
	E	(-10.763, 46.678, 15.304)	(1.023, 5.645, 31.566)	(-0.983, 42.684, 6.894)	(0.374, 36.679, 2.570)

model. In this paper, the human model in the original space is with 66 DOF; we set the particles size to be 12000 for PF. While in IGA, the quantitative results of experiments show that IGA with 40 antibodies yields results, under similar testing conditions, more accurate than PF available to us. For motion tracking, the iteration time is set to be $K = 20$. Thus, the number of likelihood evaluations for a single image would be 800 at most, which is much less than 4000 for GA [5] (size of population is 100, iteration time is 40), 7200 for PSO [13], and 12000 for PF.

We first use IGA-based pose estimation method to analyze human pose on the first image of the video for initialization, where the parameters are set as $N = 40$, $K = 60$ for careful search of the state space in initialization. While on the following frames, we set the iteration times to be $K = 20$. It is mainly because our next-frame propagation strategy can produce a compact antibodies population for optimization.

And in our experiment, we set $\Sigma = 0.01$ for straight walking sequences and $\Sigma = 0.02$ for running sequences.

The mean errors of different methods over all joint angles of the test sequences are shown in Figure 12. And Table 3 is the statistics of the average errors and the standard deviations. From Figure 12 and Table 3, we can see that our method achieve better results. The average errors and the standard deviations over all frames are near 3° and 1° , respectively, in general. It also can be found that the change of mean error of our method in whole sequence is small, which indicates that our method can achieve stable tracking of 3D human pose.

Figure 13 is the tracking results on walking and running image sequences, respectively. From the above experimental results we can see that our IGA-based pose estimation method can successfully be used for initialization of tracking. Acutely, our IGA-based pose estimation method is also used for initialization of PF in our experiments. Experimental



FIGURE 11: Pose estimation results on different image sequences.

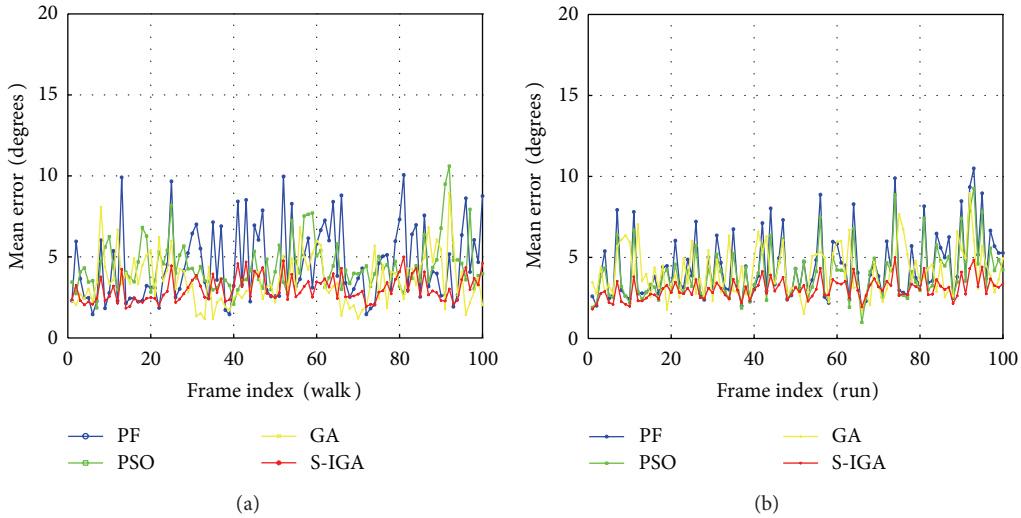


FIGURE 12: Comparison of different tracking methods.

results on different types of motion sequence show that S-IGA has good performance even without any learnt constant motion models, which demonstrate our next-frame propagation strategy is effective to generate initial distribution of antibodies for the next frame.

Experimental results demonstrate that our S-IGA-based tracking method can achieve accurate and stable tracking of 3D human motion. However, our method has some drawbacks as discussed below. Firstly, though pose optimization in the latent space makes our method more effective and accurate, it makes our method not suitable for more complicated motion analysis. So in our future work, we will extend our algorithm to cover a wider class of human motions

and explore switch mechanism between different subspaces. Secondly, in generative tracking approaches, the time taken by an algorithm depends mostly on the number of likelihood evaluations. In our IGA pose optimization method, the time complexity would be $O(NK)$, which makes it cannot work for real time applications. In addition, our method is dependent on the silhouette detection from video. But human silhouette detection from video is difficult especially in uncontrolled environment. More robust human silhouette detection method and more sophisticated image likelihood function will be considered in our future work.

Recently, Gaussian Process Latent Variable Models (GPLVM [25]) has been another widely studied latent space

TABLE 3: Results of different tracking methods.

	Walking		Running	
	Mean error	Standard deviations	Mean error	Standard deviations
PF	4.5113	2.3217	4.4669	2.0188
PSO	4.4369	1.5181	4.3949	0.9821
GA	3.5705	1.5651	4.1494	1.4779
S-IGA	3.0626	0.8345	3.0455	0.6370

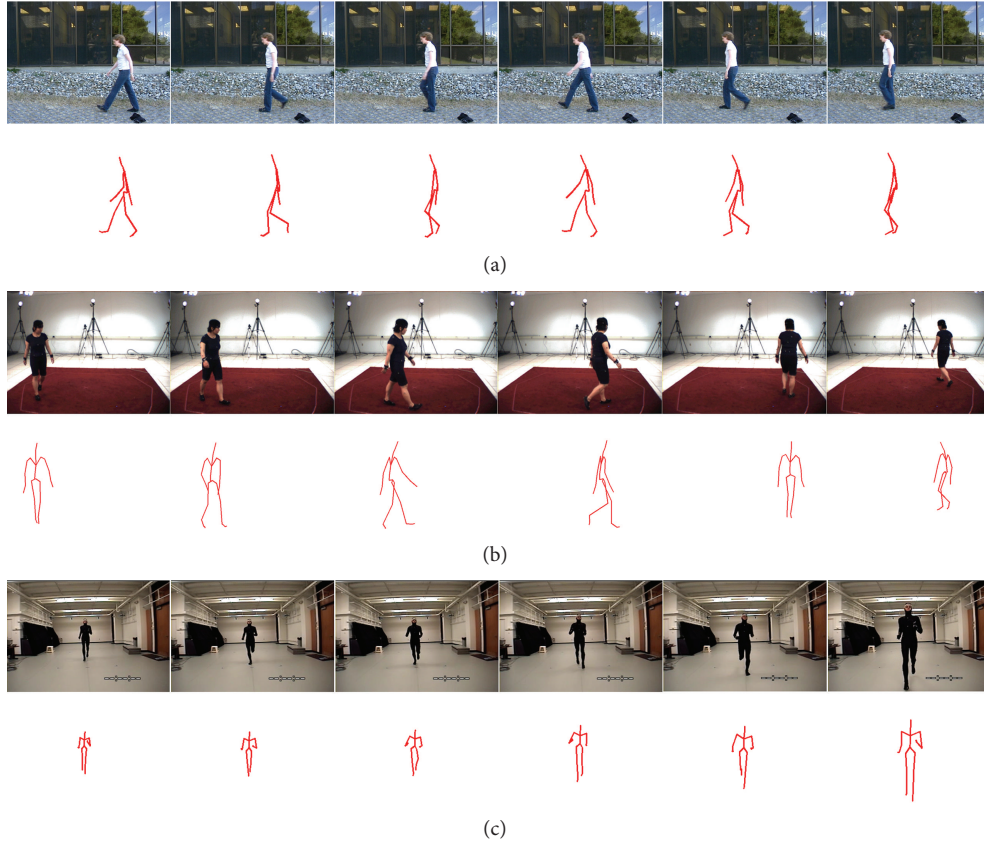


FIGURE 13: Human tracking results on real image sequences, where (a) is results on a subject walking straight (the data is from [24]) (b) is results on a subject walking in circle (the data is from HumanEva [21]), and (c) is results on a subject running (the data is from CMU Mocap database [23]).

learning method for human motion tracking. Compared with manifold learning method (ISOMAP), GPLVM could building the inverse mapping easily. However, GPLVM cannot work well on small training dataset and high-dimensional data. So in our future work, we will study how to apply GPLVM for motion tracking effectively. And more, studies on motion tracking using evolutionary computing methods are still limited. In our future work, we will consider to apply other evolutionary computing methods for motion tracking.

7. Conclusions

We presented a novel generative approach to reconstruct 3D human pose from a single monocular image as well as from monocular image sequences. The main contribution is to optimize human pose in learnt latent human motion

space. Pose analysis in the latent space learnt using ISOMAP happens to be more efficient and accurate. In the search strategy, we apply the immune genetic algorithm for pose estimation. A sequential IGA framework is proposed for pose tracking by incorporating the temporal continuity information into the traditional IGA. Compared with GA and PSO, IGA has the ability to use the prior knowledge of human motion. Experiment results on different motion types and image sequences demonstrated that our IGA-based method for pose estimation is effective to deal with occlusion, left-right ambiguity, and the viewpoint problem. The sequential IGA method can achieve stable and accurate pose tracking. Quantitative experiments compared with other state-of-art methods show that our methods achieve better results.

In the future work, we will extend our algorithm to cover a wider class of human motions and explore switch mechanism

between different subspaces. In addition, we will also consider more sophisticated image likelihood and how to reduce the computation time. How to apply other evolutionary computation methods for human motion tracking will also be considered in our future work.

Acknowledgments

This work is supported by the National High Technology Research and Development Program of China (2007AA01Z334), National Natural Science Foundation of China (61272219, 61021062, and 61100110), Program for New Century Excellent Talents in University of China (NCET-04-04605), Natural Science Foundation of Jiangsu Province (BK2010375), and Key Technology R&D Program of Jiangsu Province (BY2012190, BE2010072, and BE2011058).

References

- [1] C. Sminchisescu, "3D human motion analysis in monocular video, techniques and challenges," in *Human Motion Understanding, Modeling, Capture and Animation*, R. Kleete, D. Metaxas, and B. Rosenhahn, Eds., Springer, New York, NY, USA, 2007.
- [2] G. Mori and J. Malik, "Recovering 3D human body configurations using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 7, pp. 1052–1062, 2006.
- [3] A. Agarwal and B. Triggs, "Recovering 3D human pose from monocular images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 1, pp. 44–58, 2006.
- [4] A. Agarwal and B. Triggs, "Monocular human motion capture with a mixture of regressors," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPRW '05)*, p. 72, San Diego, Calif, USA, June 2005.
- [5] X. Zhao and Y. Liu, "Generative tracking of 3D human motion by hierarchical annealed genetic algorithm," *Pattern Recognition*, vol. 41, no. 8, pp. 2470–2483, 2008.
- [6] M. Isard and A. Blake, "CONDENSATION—conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.
- [7] L. Raskin, M. Rudzsky, and E. Rivlin, "Dimensionality reduction using a Gaussian process annealed particle filter for tracking and classification of articulated body motions," *Computer Vision and Image Understanding*, vol. 115, no. 4, pp. 503–519, 2011.
- [8] R. Poppe, "Vision-based human motion analysis: an overview," *Computer Vision and Image Understanding*, vol. 108, no. 1-2, pp. 4–18, 2007.
- [9] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer Vision and Image Understanding*, vol. 104, no. 2-3, pp. 90–126, 2006.
- [10] C. S. Lee and A. Elgammal, "Coupled visual and kinematic manifold models for tracking," *International Journal of Computer Vision*, vol. 87, no. 1-2, pp. 118–139, 2010.
- [11] V. John, E. Trucco, and S. Ivekovic, "Markerless human articulated tracking using hierarchical particle swarm optimisation," *Image and Vision Computing*, vol. 28, no. 11, pp. 1530–1547, 2010.
- [12] R. Urtasun, D. J. Fleet, and P. Fua, "Monocular 3-D tracking of the golf swing," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, pp. 932–938, San Diego, Calif, USA, June 2005.
- [13] C. Sminchisescu and A. Jepson, "Generative modeling for continuous non-linearly embedded visual inference," in *Proceedings of the 21st International Conference on Machine Learning (ICML '04)*, pp. 759–766, Alberta, Canada, July 2004.
- [14] A. Elgammal and C. S. Lee, "Inferring 3D body pose from silhouettes using activity manifold learning," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '04)*, pp. 681–688, Washington, DC, USA, July 2004.
- [15] 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.
- [16] N. R. Howe, M. E. Leventon, and W. T. Freeman, "Bayesian Reconstruction of 3D human motion from single-camera video," in *Proceedings of Neural Information Processing Systems (NIPS '00)*, pp. 820–826, Denver, Colo, USA, 2000.
- [17] J. Deutscher, A. Blake, and I. Reid, "Articulated body motion capture by annealed particle filtering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '00)*, pp. 126–133, Hilton Head, SC, USA, June 2000.
- [18] T. Krzeszowski, B. Kwolek, and K. Wojciechowski, "Articulated body motion tracking by combined particles swarm optimization and particle filtering," in *Proceedings of the International Conference on Computer Vision and Graphics (ICCVG '10), Part I*, vol. 6374 of *Lecture Notes in Computer Science*, pp. 147–154, 2010.
- [19] X. Zhang, W. Hu, S. Maybank, and L. Xi, "Sequential particle swarm optimization for visual tracking," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '08)*, pp. 23–28, Anchorage, Alaska, USA, 2008.
- [20] W. Lei and J. Licheng, "Immune evolutionary algorithms," in *Proceedings of the International Conference of Signal Process (ICSP '00)*, pp. 1655–1662, Beijing, China, August 2000.
- [21] CMU Database, <http://mocap.cs.cmu.edu/>.
- [22] M. Gong, L. Jiao, and L. Zhang, "Baldwinian learning in clonal selection algorithm for optimization," *Information Sciences*, vol. 180, no. 8, pp. 1218–1236, 2010.
- [23] L. Sigal, A. O. Balan, and M. J. Black, "HumanEva: synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion," *International Journal of Computer Vision*, vol. 87, no. 1-2, pp. 4–27, 2010.
- [24] D. Ormoneit, H. Sidenbladh, M. J. Black, and T. Hastie, "Learning and tracking cyclic human motion," in *Proceedings of Neural Information Processing Systems (NIPS '01)*, pp. 894–900, Vancouver, Canada, December 2001.
- [25] R. Urtasun, D. J. Fleet, A. Hertzmann, and P. Fua, "Priors for people tracking from small training sets," in *Proceedings of the 10th IEEE International Conference on Computer Vision (ICCV '05)*, pp. 403–410, Beijing, China, October 2005.

Research Article

The Optimized Transport Scheme of Empty and Heavy Containers with Novel Genetic Algorithm

Lianwei Qu,¹ Kang Chen,¹ Bao Hongli,¹ Jingshu Ma,¹ and Wan Tao²

¹ Transportation Management College, Dalian Maritime University, Dalian 116023, China

² Tianjin Urban Planning and Design Institute, Tianjin 300201, China

Correspondence should be addressed to Wan Tao; wantao428@163.com

Received 5 October 2012; Accepted 30 December 2012

Academic Editor: Bin Yu

Copyright © 2013 Lianwei Qu 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.

To design the transport scheme of empty and heavy containers reasonably, a model with objective maximizing the route benefits is proposed. The model considered two factors: (1) the fluctuation of cargo transport demand and the switching of different voyages; (2) the optimal transport scheme of empty and heavy containers in slack and brisk seasons and the handover process of these two seasons. In order to solve this model, a novel GA is developed. With this model and algorithm, the optimal transport scheme of empty and heavy containers is put forward, and the optimization allocation of resources can be realized. The case study about China-Europe route proves that this model can improve the liner company's benefits effectively.

1. Introduction

Currently, there are many studies about optimization scheme of empty and heavy containers. Some studies [1–3] developed an optimization method studying the transporting scheme of empty containers based on simulation and two-stage random network model, and the model took short-term container leasing into consideration. Reference [4] studied the transporting plan of empty containers from a new perspective, which determined the number of empty containers to be transported by using inventory management theory and Markov process theory. With the constantly rising proportion of heavy containers to be transported, scholars paid more attention to the heavy container optimization problem among ports. Chen and Zeng [4] attempted to deal with the programming problem of empty and heavy containers simultaneously by utilizing decision supporting system. When studying container route optimization, they did an in-depth analysis about the optimization transporting problem of empty and heavy containers. Concerning the cyclical fluctuations of cargo demand, this paper proposed an optimization model of empty and heavy containers. However, this model does not consider two problems: the first one is the effect of network change has on transporting scheme of

empty and heavy containers; the second one is the adjustment problem of different empty and heavy containers. Therefore, in this paper, we established an integer programming model aiming at maximizing the total voyage profits. Considering the variation of transport demand and alteration of routes, the optimal transporting scheme of empty and heavy containers is proposed. Finally taking the data of China-Europe route as research data, the practicality of model is tested.

Before 2009 most researchers only fixed their attention on the slot allocation for empty containers, often called empty container relocation problem as we know. For example, Crainic et al. [2] raised two dynamic deterministic formulations to deal with the single and multicommodity cases; for empty container relocation they also put forward an ordinary model. Cheung and Chen [3] adopted a two-stage stochastic network model to optimize the empty container relocation problem and ascertained the minimum quantity of leased containers. Imai and Rivera [5] analyzed the accurate number of container fleet size and the relocation of the empty containers by utilizing simulation model. Li et al. [6] disposed the problem of empty container relocation by offering a new method. They applied methods of inventory management and Markov processes to analyze the quantity of empty containers conveyed among ports and compute the

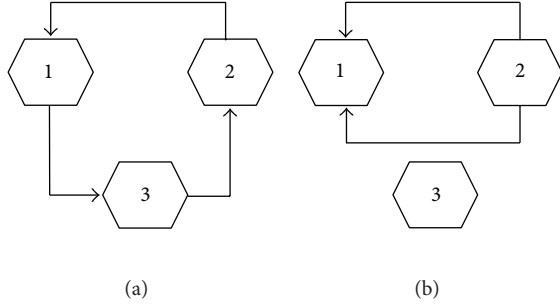


FIGURE 1: The diagram of segments.

floor level and maximum quantity of the empty containers stored in ports. Feng and Chang [7] used a two-stage model considering both inventory management and the nature of the shipping network to deal with the empty container relocation problem. These studies above-cited are based on some assumption that the shipping routes are given and all transport demands are ought to be served. The latter assumption means that from one port to another, the number of full containers is fixed, so is the quantity of slot allocation for full containers.

Lindstrom Bandeira et al. [1] wanted to use a decision support system to optimize the slot allocation both for full and empty containers simultaneously. But the system cannot serve all transport demand. Lu et al. [8] also optimized slot allocation for full and empty containers by using a new integer programming model. In this paper, we assume that shipping lines could give up some transport demand to obtain maximum profit.

2. Problem Description

2.1. The Transporting Process of Empty and Heavy Containers. The transporting process of containers includes two parts: the transporting process at sea and the circulation at ports.

2.1.1. The Transport Process at Sea. The liner transport process at sea is relatively simple. Liner companies accomplish container transport by sailing among ports. During this process, route structure has a big influence on transport plan of empty and heavy containers. As shown in Figure 1, shipping companies should consider the import/export demand of port 3 under the route structure of route (a), which is unnecessary in route (b). Apart from the direct influence route structure has on transport demand, its another influence is it limits the transport capacity of empty and heavy containers. For the case in route (a), shipping company should consider whether the total number of empty and heavy containers exceed the maximum vessel capacity. There are three segments on route (a), and each segment is loaded with containers between different ports. On segment 1, the total number of empty and heavy containers transported from port 3 to port 1 should not exceed the ship's transport capacity.

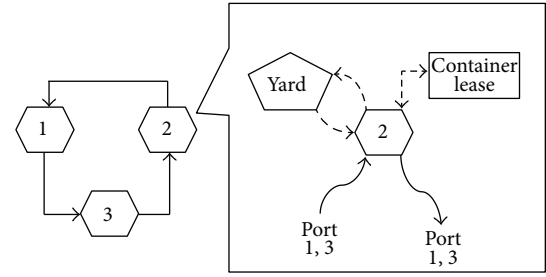


FIGURE 2: The circulation of containers at port.

2.1.2. The Container Circulation at Port. The container circulation process at port includes transport cargoes, stock empty containers, empty heavy containers, and return empty containers. For instance, take port 2 in Figure 2 as example. As shown in Figure 2, port 2 will import empty and heavy containers from port 1 and port 3. According to the attribution of containers, these containers should be divided into three categories: (1) empty containers owned by liner company, (2) heavy containers owned by liner company, and (3) short-term leasing heavy container. The circulation process of these three containers is different. After unloaded, the empty containers owned by liner companies will either be stored in container yard at port, or be loaded with cargoes and transported to the port of discharge. The heavy containers will be delivered to consignees after being unloaded from a ship. After being emptied and returned, heavy containers will either be stored at container yard, or shipped to the ports where empty containers are needed. Different from the two above mentioned containers, after unloading the short-term leasing containers, these containers will be returned to the port of discharge due to the relatively high leasing costs.

It can be learned that from the container circulation process, shipping company should formulate a series of inter-actational plans, as shown below: the transport plan of empty & heavy container owned by liner company, transport plan of short-term leasing heavy containers among ports, and stock capacity of empty containers at each port. These schemes are not only influenced by determined factors like route structure, freight rate, handling costs at port, stockpiling costs, but cyclical fluctuation of transport demand.

2.2. Cyclical Fluctuation of Transport Demand. The fluctuation of transport demand among different ports has the following two characteristics: (1) the fluctuation of transport demand is obviously seasonal, which can be divided into the slack season with less transport demand and brisk season with more demand; and (2) there are slight fluctuations during both slack and brisk seasons, and the overall demand is relatively stable.

According to those characteristics, we use the following method to deal with fluctuations of transport demand. One year is divided into four phases: slack season, the transition phase from slack season to brisk season, brisk season, and

the transition phase from brisk season to slack season. In slack season and brisk season, the transport plan of heavy containers is not changed, while the transport scheme of empty containers can be adjusted anytime. In the other two transition phases, to accomplish the transition process between slack season and brisk season, both the transport scheme of empty and heavy containers can be adjusted. It should be noted that the transition here not only includes the transport scheme of different empty and heavy containers, but the transition of empty container stockpiling plan.

3. The Optimization Model of Empty and Heavy Containers

3.1. Model Hypotheses. This model is established according to the following assumptions: (1) the density of liner schedule is on weekly basis and only container of 20 ft is considered; (2) shipping companies only adjust route structure once a time in a year; (3) every port can lease enough empty containers and have enough space to stock them, if they want; (4) to simplify calculation, the decision variables are not set as integer, besides, K in the model means the total working weeks within a year (52 weeks); (5) in nontransition phases, the number of heavy containers transported does not change; (6) it is assumed that $1 - N_1$ week is set as slack season, $N_1 + 1 - N_2$ is set as the transition phase of slack season and brisk season, $N_2 + 1 - N_3$ is set as brisk season, and $N_3 + 1 - N_4$ is set as the transition phase between brisk season and slack season.

3.2. Model Establishment. Based on the above analyses, a linear integer programming model is established. The specific formulation of model is shown as follows:

$$\begin{aligned} \text{Max } Z = & \sum_{k \in K} \left\{ \sum_{i \in P} \sum_{j \in P} [(R_{ij}^k - CF_{ij}^k) \times ox_{ij}^k \right. \\ & + (R_{ij}^k - CR_{ij}^k) \\ & \left. \times rx_{ij}^k - CE_{ij}^k \times oy_{ij}^k] \right\} \\ & - \sum_{k \in K} \sum_{i \in P} CS_i^k \times sy_i^k, \end{aligned} \quad (1)$$

S.t

$$\sum_{(i,j) \in \text{SEG}_1} (ox_{ij}^k + rx_{ij}^k + oy_{ij}^k) \leq U_1, \quad \forall k \in K, \quad (2)$$

$$\sum_{(i,j) \in \text{SEG}_2} (ox_{ij}^k + rx_{ij}^k + oy_{ij}^k) \leq U_2, \quad \forall k \in K, \quad (3)$$

$$ox_{ij}^k + rx_{ij}^k \leq D_{ij}^k, \quad \forall k \in K, \quad \forall i, j \in P, \quad (4)$$

$$sy_i^k = \sum_j (ox_{ji}^{k-1} + oy_{ji}^{k-1}) + sy_i^{k-1} - \sum_j (ox_{ij}^k + oy_{ij}^k), \quad (5)$$

$\forall k \in K, j \in P,$

$$ox_{ji}^k = ox_{ji}^N, \quad \forall k \in [1, N_1], \quad \forall i, j \in P, \quad (6)$$

$$ox_{ji}^k = ox_{ji}^{N_3}, \quad \forall k \in [N_2, N_3], \quad \forall i, j \in P, \quad (7)$$

$$ox_{ji}^1 = ox_{ji}^{N_4}, \quad \forall i, j \in P, \quad (8)$$

$$oy_{ji}^1 = oy_{ji}^{N_4}, \quad \forall i, j \in P, \quad (9)$$

$$\sum_j (ox_{ij}^k + oy_{ij}^k) + sy_i^k - \sum_j (ox_{ji}^k + oy_{ji}^k) \geq 0, \quad (10)$$

$\forall i \in P.$

3.3. Model Explanation. The model aims at maximizing the total profits Z of voyage, its decision variables are as follows: ox_{ij}^k —the number of heavy containers which are owned by liner companies transported from port i to port j ; oy_{ij}^k —the number of empty containers which are owned by liner companies transported from port i to port j ; rx_{ij}^k —the number of heavy containers which are leased in short-term by liner companies from port i to port j ; sy_i^k —the number of empty containers stocked in port i of week k .

Equation (1) is the objective function, where i and j represent port, P is the set of all ports in all voyages, K is the total working week within a year, R_{ij}^k means the freight rate from port i to port j at week k , CF_{ij}^k and CE_{ij}^k represents from port i to port j , the total handling costs generated by one heavy container and empty container, respectively, CR_{ij}^k represents the total costs of short-term container leasing costs and handling cost from port i to port j , and CS_i^k is the stock cost of one TEU in port i . Equations (2) and (3) ensure that in the case with different route structures, the containers loaded on ship do not exceed shipping capacity on each segment, where SEG_1 and SEG_2 are, respectively, the segment set of route structure 1 and 2. Equation (4) means the heavy containers transported are not bigger than actual transport demand, where D_{ij}^k is the transport demand from port i to port j at week k . Equation (5) means the change of stockpile at ports. Equations (6)–(9) assure that in slack season and brisk season, the number of heavy container transported between ports is constant. The model is liner programming model, which can be solved by using classic algorithm.

4. Solution of the Model

Because the integer programming model considers some additional constraints such as transport of heavy and empty containers, short-term container lease, and stock capacity of ports, it is hard to be solved. Genetic algorithms (GA) have been applied to a very wide range of practical problems, for example, transit dispatching [9, 10], route network optimization [11], and route headway design [12]. To find an

approximate solution, we design a novel GA, named NGA hereafter. The running steps are as follows.

Step 1 (Initialization). Set $n = 0$, and then generate an initial population $P_n = \{\text{pop}_v \mid v = 1, \dots, \lambda\}$ where $\text{pop}_v = \{ox_{ij}^k, oy_{ij}^k\}$; λ is the population size.

Step 2 (Calculation of the Fitness Value). Computing the fitness value of individual v as fit_v with (1).

Step 3 (Convergence Judgment). Checking whether the gap between the maximum fitness and the average fitness in current generation is less than the preinstalled threshold μ , if yes, go to Step 4; otherwise, terminate the calculation.

Step 4 (Implementation of Selection, Crossover, and Mutation Operations). Let $n = n + 1$ and do selection, crossover, and mutation operations to get a new generation of individuals P_n . Then return to Step 1.

5. Case Study

In this paper, we take the China-Europe voyage as our study object. During slack season, the Asia-Europe voyage is Qingdao-Hong Kong-Le Havre-Felix Stowe-Hamburger-Qingdao; during brisk season, the voyage is Dalian (DL)-Qingdao (QD)-Shanghai (SH)-Hong Kong (HK)-Le Havre-Felix Stowe-Hamburger-Dalian. The total amount of containers from Asia to Europe in slack season is about 10500 TEU. The freight in slack season is set as the average Maersk and COSCO's freight rate promulgated between March and April. During brisk season, the transport demand is 50% higher than brisk season, and the freight rate is 100% higher. The stockpile cost of this liner company is 1 USD/TEU at ports in Europe, and 0.5 USD/TEU at ports in Asia.

For calculation, we set the maximum calculating iterations as 200, the crossover probability as 0.92 and the mutation probability as 0.01. We will terminate the calculation when the gap between the maximum fitness and the average one is less than 0.002.

After calculation, the result of model is obtained: the fluctuation of empty and heavy container's transport demand at each port is shown in Figures 3 and 4. It can be learned from Figure 3 that the transport scheme of heavy containers remains stable in majority of time. This result is in accordance with the hypotheses proposed in model.

To stabilize the transport plan of heavy containers, transport scheme of empty containers should be adjusted constantly, and that is the reason why the number of empty container from the first week to the twenty-sixth week of each port changes regularly. The effects of cyclical fluctuation of transport demand and route adjustment have on the allocation of empty containers are shown in Figure 5. It can be seen that during slack season (from the 1th week to the 13th week), the amount of empty containers stocked in Asia ports and Europe ports fluctuates around 6000 TEU and 3000 TEU; while during the handover phase of slack season and brisk season (from the 4th week to the 26th week), the number of containers stocked at Europe port declines, and

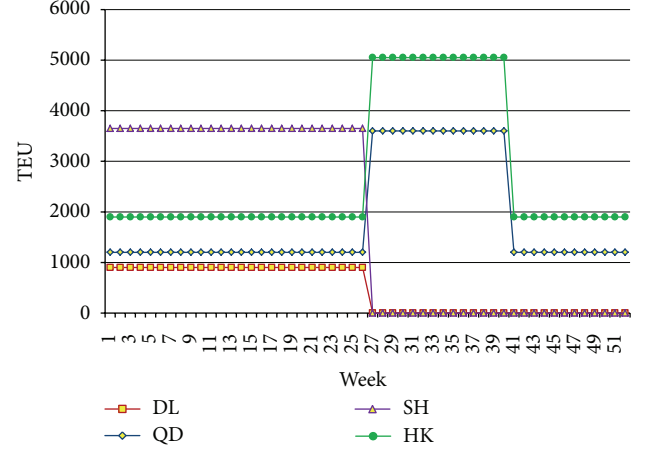


FIGURE 3: Heavy containers fluctuation of Asia ports.

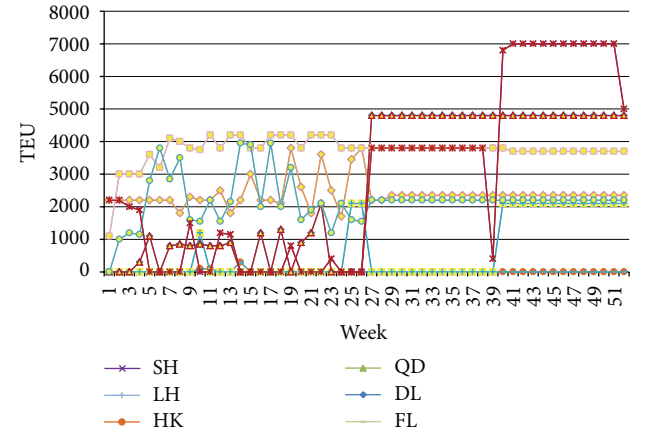


FIGURE 4: Empty container fluctuation of each port.

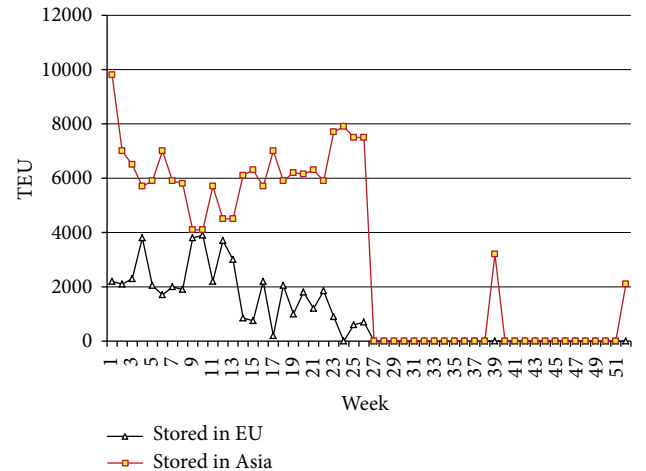


FIGURE 5: Empty container fluctuation of Asia and Europe ports.

the empty containers stocked at Asia ports gradually increase. The reason why this outcome is generated is liner companies should store enough empty containers for the upcoming brisk season in Asia ports. In brisk season (from 27th week to 39th week), the number of empty containers in both Asia and Europe ports is 0. In this case, all the empty containers owned by liner company have been fully utilized, and meanwhile, the use efficiency is improved greatly.

6. Conclusion

In this paper, a model with objective maximizing total voyage profits is proposed. In this model, cyclical fluctuation of voyage, alteration of routes and the handover of transition scheme are taken into consideration. The case study verified the practicability of this model.

References

- [1] D. Lindstrom Bandeira, J. Luiz Becker, and D. Borenstein, "A DSS for integrated distribution of empty and full containers," *Decision Support Systems*, vol. 47, no. 4, pp. 383–397, 2009.
- [2] T. G. Crainic, M. Gendreau, and P. Dejax, "Dynamic and stochastic models for the allocation of empty containers," *Operations Research*, vol. 41, no. 1, pp. 102–126, 1993.
- [3] R. K. Cheung and C. Y. Chen, "A two-stage stochastic network model and solution methods for the dynamic empty container allocation problem," *Transportation Science*, vol. 32, no. 2, pp. 142–162, 1998.
- [4] C. Chen and Q. Zeng, "Designing container shipping network under changing demand and freight rates," *Transport*, vol. 25, no. 1, pp. 46–57, 2010.
- [5] A. Imai and F. Rivera, "Strategic fleet size planning for maritime refrigerated containers," *Maritime Policy and Management*, vol. 28, no. 4, pp. 361–374, 2001.
- [6] J.-A. Li, K. Liu, S. C. H. Leung, and K. K. Lai, "Empty container management in a port with long-run average criterion," *Mathematical and Computer Modelling*, vol. 40, no. 1-2, pp. 85–100, 2004.
- [7] C. M. Feng and C. H. Chang, "Empty container reposition planning for intra-Asia liner shipping," *Maritime Policy and Management*, vol. 35, no. 5, pp. 469–489, 2008.
- [8] H. A. Lu, C. W. Chu, and P. Y. Che, "Seasonal slot allocation planning for a container liner shipping service," *Journal of Marine Science and Technology*, vol. 18, no. 1, pp. 85–92, 2010.
- [9] B. Yu, S. Wu, Z. Yang, and B. Yao, "Dynamic vehicle dispatching at a transfer station in public transportation system," *Journal of Transportation Engineering*, vol. 138, no. 2, pp. 191–201, 2012.
- [10] B. Yu, Z.-Z. Yang, and S. Li, "Real-time partway deadheading strategy based on transit service reliability assessment," *Transportation Research A*, vol. 46, no. 8, pp. 1265–1279, 2012.
- [11] B. Yu, Z.-Z. Yang, P.-H. Jin, S.-H. Wu, and B.-Z. Yao, "Transit route network design-maximizing direct and transfer demand density," *Transportation Research C*, vol. 22, pp. 58–75, 2012.
- [12] B. Yu, Z. Yang, X. Sun, B. Yao, Q. Zeng, and E. Jeppesen, "Parallel genetic algorithm in bus route headway optimization," *Applied Soft Computing*, vol. 11, no. 8, pp. 5081–5091, 2011.

Research Article

Daily Commute Time Prediction Based on Genetic Algorithm

Fang Zong,¹ Haiyun Lin,² Bo Yu,³ and Xiang Pan⁴

¹ College of Transportation, Jilin University, RenMin Street 5988, Changchun 130022, China

² Department of Civil Engineering, City College of New York, 160 Convent Avenue, New York, NY 10031, USA

³ Transportation College, Dalian Maritime University, Dalian 116026, China

⁴ College of Computer Science, Zhejiang University of Technology, 288 Liuhe Road, Hangzhou 310023, China

Correspondence should be addressed to Fang Zong, zongfang@jlu.edu.cn

Received 20 September 2012; Accepted 30 October 2012

Academic Editor: Baozhen Yao

Copyright © 2012 Fang Zong 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 joint discrete-continuous model for activity-travel time allocation by employing the ordered probit model for departure time choice and the hazard model for travel time prediction. Genetic algorithm (GA) is employed for optimizing the parameters in the hazard model. The joint model is estimated using data collected in Beijing, 2005. With the developed model, departure and travel times for the daily commute trips are predicted and the influence of socio-demographic variables on activity-travel timing decisions is analyzed. Then the whole time allocation for the typical daily commute activities and trips is derived. The results indicate that the discrete choice model and the continuous model match well in the calculation of activity-travel schedule. The results also show that the genetic algorithm contributes to the optimization and thus the high accuracy of the hazard model. The developed joint discrete-continuous model can be used to predict the agenda of a simple daily activity-travel pattern containing only work, and it provides potential for transportation demand management policy analysis.

1. Introduction

The time allocation of individuals for trip making is an important determinant of the temporal pattern of traffic demand on a transportation network. An analysis of individual time allocation decision is, therefore, important for the practical work of transportation planning and management as well as the theoretical work about travel behavior analysis and modeling. Practically, understanding individuals' time allocation decisions is essential for (a) planning the development and construction of new transportation infrastructure by providing

predicted temporal travel demands, (b) examining the potential responses to improved operational measures (such as real-time information), (c) assessing the effectiveness of time-specific transportation demand management policies, such as compressed working week [1], staggered shift [2], road tolling [3], and other similar strategies [4, 5]. Theoretically, understanding time allocation decisions will not only facilitate the efforts toward developing a comprehensive full-scale model of daily activity patterns but also provide useful insights into the nature of the impact of sociodemographic variables and time-space constraints on individual dimensions of activity behavior. Therefore, time allocation has been a focused issue for regional and transportation science since the 1970s [6].

In the time dimension, a daily activity-travel pattern includes all the timing and duration of all the activities and trips in a day. The need to analyze the daily activity pattern makes it essential to consider time of day and activity-travel durations together. However, to date, timing and duration have been largely treated separately in the literature [7], and many existing studies only considered part of daily activity agenda. For example, Vovsha and Bradley [8] focused on the departure-from-home and arrival-back-home time decisions. Fujii and Kitamura [9] and Hamed and Mannering [10] examined the time allocation of postwork activities. Bhat and Singh [11] and Habib et al. [12] modeled the departure time of daily commute trips, without involving the commute travel time. This study is aimed at addressing this issue by developing a joint daily time allocation model to predict a typical daily activity-travel schedule.

One important objective of transportation planning is to relief congestion by improving the level of service during the peak hours on the transportation network. Peak periods' traffic demands, or the source of congestion, are largely contributed by commute trips. For example, based on one survey conducted in Beijing, China, in 2005, over 32% of all the trips in both morning and evening peak periods are commute trips. Therefore, commute trips are at the core of many recent studies, such as Habib et al. [12], Zhang et al. [13], and Bhat and Singh [11].

As stressed above, the study on commute trips and daily time allocation is of great importance for learning the travel behavior during the peak hours as well as obtaining the daily activity pattern, both of which are essential component of for transportation planning and management. However, to the authors' knowledge, there is no study that considered the overall daily time allocation of commute activity-travel pattern as a whole and developed a model system to predict it. Therefore, this paper focuses on predicting timing and duration of daily commute trips, expecting that commuter's typical travel schedule can be obtained based on the developed model. Using data from a 2.5% sample household survey in Beijing, China, a joint discrete-continuous model system for prediction of daily commute time allocation was developed and estimated, including ordered probit models for departure time analyzing and hazard models for travel time forecasting.

The choice of parameters is of great importance for the estimation efficiency and prediction accuracy of the models. As there are a lot of potential factors affecting traveler's decision about time allocation, genetic algorithms (GAs) will be employed in parameters optimization. Being one of the heuristic algorithms, GA has been successfully applied in various optimization problems [14, 15].

The remainder of this paper is organized as follows: Section 2 presents the literature review on activity-travel time allocation in general. In Section 3, the joint discrete-continuous time allocation model is built, in which the ordered probit method, AFT model, and GA are employed. Section 4 predicts the commute activity-travel agenda by using the developed

model. The paper closes with some overall conclusions and a discussion of future research directions.

2. Relevant Literature

With respect to the two dimensions of time allocation, timing and duration, existing studies can be classified into three categories: those only looked at departure time, those only looked at travel time, and those looked at both.

For the first direction, timing, the major method employed is discrete choice model. For example, Bowman and Ben-Akiva [16] modeled the departure and arrival time of daily trips using the multinomial logit (MNL) model. Small [17] built an MNL model for home-work morning departure time choice. Vovsha and Bradley [8] and Ettema et al. [7] also employed the MNL model but specified a continuous time variable in its utility function. Bhat [18] and Small [19] derived the ordered generalized extreme value (OGEV) model for departure time choice and time of day analysis, respectively. Habib et al. [12] introduced OGEV model into analysis of work start time and work duration because it allows for the accommodation of correlation among time period alternatives. In this way, it resolves the IIA (independence of irrelevant alternatives) problem of MNL model, which assumes zero correlation among different time periods.

For the second aspect (duration analysis), the most widely used model is the hazard model. The hazard model recognizes the dynamics of travel or activity durations by considering the conditional probability of event termination, usually as a function of covariates (explanatory variables) [12]. Bhat [20, 21] applied hazard modeling framework to analyze after-work activity duration. Juan and Xianyu [22] considered daily travel time using hazard-based duration model. Bhat [20, 21] analyzed the duration of shopping activity by employing hazard-based duration model. As a matter of fact, a few studies also used hazard model in timing analysis. Examples include research by Habib et al. [12] on investigation of trip timing and by Bhat and Steed [23] on departure time choice for shopping trip.

Comparing with the first two categories of studies, those using the joint analysis of timing and duration are more helpful for the modeling of daily activity schedule, by contributing to an insight into the influence and connection among duration and time of day choice as well as activities and trips. The examples include the study done by Janssens et al. [24] on time allocation of daily activity-travel patterns, by Fujii and Kitamura [9] on timing and duration of commuters' daily activity patterns after work hours, by Habib [25] on work start time and work duration, by Raux et al. [6] on daily time allocation of travel and out-of-home activity, by Schwanen and Dijst [26] on relationship between commuting time and work duration, and by Vovsha and Bradley [8] on departure-time and duration of home-based trips. Similar studies also include Ettema et al. [7], Pendyala and Bhat [27], and Habib et al. [28]. The models involved in these studies include reinforcement machine learning technique [24], structural equations model [9], hazard model [6, 25, 28], and MNL model with a continuous time variable in the utility of its function [7, 8].

There is also a kind of conjunct model that was used in the analysis of timing and duration, which is the discrete-continuous choice model. Pendyala and Bhat [27] examined the relationship between time of day choice and activity episode duration using discrete-continuous simultaneous equations, but it was neither on daily time allocation simulation nor the commute trips. Similar discrete-continuous models were employed by Habib et al. [12], Habib [25], and Hamed and Mannering [10]. Both of the first two studies modeled the joint

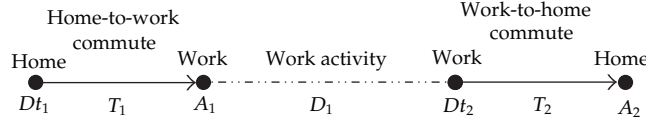


Figure 1: Basic time allocation of daily commute activity-travel pattern.

mode and commute timing choice with discrete choice model and continuous model, but not focusing on daily activity-travel schedule particularly. Concerning the research by Hamed and Mannering [10], the discrete-continuous models are used to predict the travelers activity-type preference (discrete choice model), the travel time to the activity and back home (continuous model), and activity duration (continuous model). For modeling of time allocation, only continuous model was employed. However, the results of these four studies mentioned above all confirmed that the discrete-continuous choice models work well for the activity-travel behavior related study.

Moreover, the study of Pendyala and Bhat [27] suggests that time of day and activity duration is only loosely related for the commuter sample. Therefore, travel departure time and travel duration will be modeled in this paper, instead of activity duration. In this case, the work duration can be calculated according to the arrival time at work location and departure time of the next activity after work.

3. Modeling Time Allocation of Daily Commute Trips

3.1. Analysis of the Commute Activity-Travel Agenda

In this paper, work location refers to the usual work location for a worker and the usual school location for a student. As shown in Figure 1, a typical daily commute activity-travel pattern is home-to-work commute—work activity—work-to-home commute. Key time and duration values in this pattern include home-to-work morning departure time (Dt_1)—home-to-work travel time (T_1)—arrival time at work location (A_1)—duration of work (D_1)—work-to-home evening departure time (Dt_2)—work-to-home travel time (T_2)—arrival time at home (A_2). Within this pattern, the departure times Dt_1 and Dt_2 and the travel times T_1 and T_2 are most important. Known their values, one can easily derive the other three times (A_1 , D_1 , and A_2) using the following equations:

$$\begin{aligned} A_1 &= Dt_1 + T_1, \\ D_1 &= Dt_2 - A_1, \\ A_2 &= Dt_2 + T_2. \end{aligned} \tag{3.1}$$

This study employs the ordered probit model (belonging to the discrete choice models) for departure time forecasting and the hazard model (belonging to the continuous models) for travel time analysis. Four models, home-to-work departure time choice model, home-to-work travel time estimation model, work-to-home departure time choice model, and work-to-home travel time estimation model will be developed and the values of Dt_1 , T_1 , Dt_2 , and T_2 will be predicted. Values of A_1 , D_1 , and A_2 are then calculated based on (3.1). Figure 2 is a schematic representation of the entire modeling process.

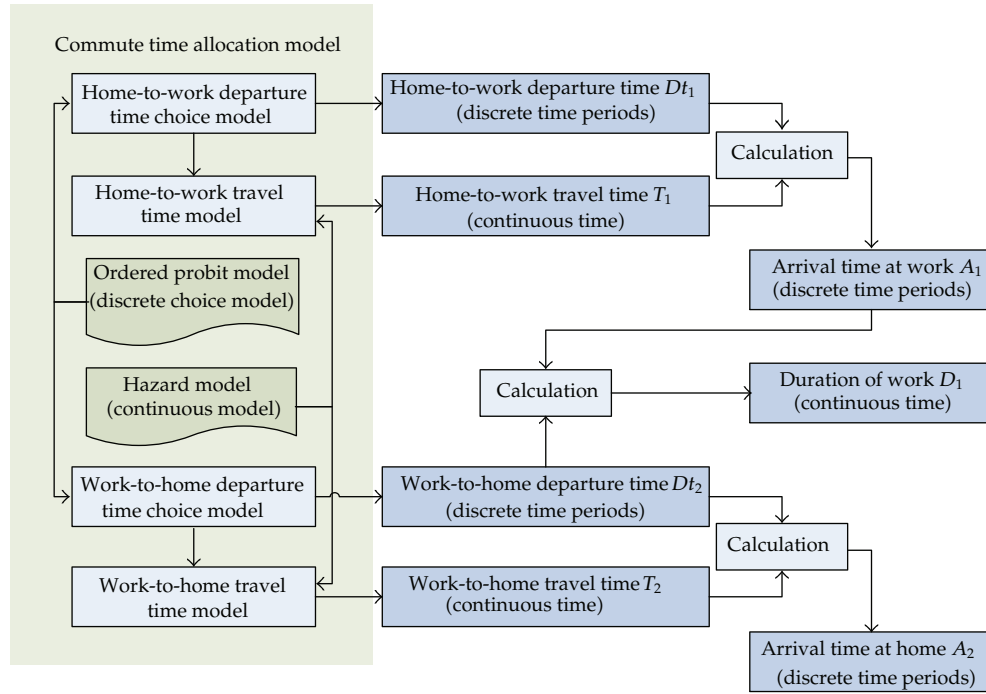


Figure 2: Modeling process of the commute activity-travel time allocation.

3.2. Data

This study uses data from a large-scale daily travel survey conducted in Beijing, China, in 2005. A face-to-face interview was given to a sample of 54,398 households, and activity/travel information of all household members on one particular working day is collected. The study area is 1,368 km² (covering all 18 districts of Beijing) and had more than 30 millions' population in 2005. In addition to weekday OD information, the survey also collected information regarding household (size, car ownership, home location, monthly income, and mobility), people (age, gender, driving license, and occupation), and trips (departure time, arrival time, purpose, mode, transit path, etc.). With records containing missing values eliminated, our final sample consists of 37,842 commute trips of 37,842 individual workers/students from 28,382.

Based on a preliminary correlation test, 15 sociodemographic and trip characteristics variables were selected from the survey, shown in Table 1.

The statistics of the variables based on the sample data are shown in Table 2.

3.3. Departure Time Choice Model

The reported home-to-work morning departure time and the work-to-home evening departure time cover the time period of 4:00 am–12:00 am and 15:00 pm–22:00 pm, respectively, in our sample. Moreover, we observed that the home-to-work morning and the work-to-home evening peak hours in Beijing are 6:00 am–9:00 am and 16:00 pm–19:00 pm using our sample data. In order to reduce the number of alternatives in the models' choice set, we divided

Table 1: Variables in the departure time choice models.

Factors	Variables	Values
Gender	Gender	Male: 1, female: 0
Age	Age	Continuous values Below 1500 RMB: 1 1501–2500 RMB: 2 2501–3500 RMB: 3 3501–5500 RMB: 4 5501–10000 RMB: 5 10001–20000 RMB: 6 20001–30000 RMB: 7 Over 30001 RMB: 8
Month income	Income	
Vocation		
Blue-collar worker (manufacture, construction, maintenance, etc.)	Occu-b	Yes: 1, No: 0
Administration	Occu-a	Yes: 1, No: 0
Education	Occu-e	Yes: 1, No: 0
Services	Occu-s	Yes: 1, No: 0
Health care	Occu-h	Yes: 1, No: 0
Travel mode		
Walk	Mode-w	Yes: 1, No: 0
Bike	Mode-bi	Yes: 1, No: 0
Bus	Mode-bu	Yes: 1, No: 0
Auto	Mode-a	Yes: 1, No: 0
Travel distance	Distance	Continuous value (meter)
Dummy variable of going to work	Work	Yes: 1, no: 0 (going to school)

departure times into one-hour segments in peak periods and segments of two or three hours in off-peak periods. Table 3 shows the discrete alternatives for the home-to-work and work-to-home departure time choices.

As shown in Table 2, the alternatives of the departure time choice models are naturally ordered time periods. MNL model, which is commonly used in departure time modeling, would fail to account for the ordinal nature of the dependent variable and have the problem of IIA. This study will employ the ordered multiple choice model for departure time modeling instead.

The ordered multiple choice model assumes the following relationship:

$$\sum_{j=1}^J P_n(j) = F(\alpha_j - \beta_j X_n, \theta), \quad j = 1, \dots, J-1, \quad (3.2)$$

$$P_n(J) = 1 - \sum_{j=1}^J P_n(j),$$

where $P_n(j)$ is the probability that alternative j is chosen as departure time of trip n ($n = 1, \dots, N$), α_j is an alternative specific constant, X_n is a vector of attributes of trip n , β_j is a vector of

Table 2: Statistics index of the variables.

Variables	Percentage of each value	
	1	0
Gender	50.96%	49.04%
Occu-b	42.44%	57.56%
Occu-a	3.48%	96.52%
Occu-e	30.29%	69.71%
Occu-s	9.19%	90.81%
Occu-h	2.49%	97.51%
Mode-w	42.51%	57.49%
Mode-bi	43.89%	56.11%
Mode-bu	0.31%	99.69%
Mode-a	4.67%	95.33%
Work	72.79%	27.21%
Income	1: 13.46%	5: 11.86%
	2: 24.90%	6: 1.41%
	3: 24.18%	7: 0.16%
	4: 23.92%	8: 0.10%
Continuous variables		
Variables	Mean	Standard deviation
Distance	2806.1	3416.91
Age	32.93	15.32

Table 3: Alternatives in the departure time choice models.

Home-to-work departure time	Work-to-home departure time	Coded value
(4:00, 6:00]	(15:00, 16:00]	1
(6:00, 7:00]	(16:00, 17:00]	2
(7:00, 8:00]	(17:00, 18:00]	3
(8:00, 9:00]	(18:00, 19:00]	4
(9:00, 12:00]	(19:00, 22:00]	5

estimable coefficients, and θ is a parameter that controls the shape of probability distribution F . Therefore, F can have various shapes of distribution based on a different value of θ .

The ordered probit model, which assumes standard normal distribution for F , is the most commonly used. The ordered probit model has the following form:

$$\begin{aligned}
 P_n(1) &= \Phi(\alpha_1 - \beta_j X_n), \\
 P_n(j) &= \Phi(\alpha_j - \beta_j X_n) - \Phi(\alpha_{j-1} - \beta_j X_n), \quad j = 2, \dots, j-1, \\
 P_n(J) &= 1 - \sum_{j=1}^{J-1} P_n(j),
 \end{aligned} \tag{3.3}$$

where $P_n(j)$ is the cumulative standard normal distribution function. For all the probabilities to be positive, we must have $\alpha_1 < \alpha_2 < \dots < \alpha_{J-1}$. The estimation results of the home-to-work and the work-to-home departure time choice models are shown in Table 4.

The estimation results indicate that high-income travelers are more likely to depart from home or work location later than travelers with low income. Older persons tend to have earlier departure times than younger ones. Commuters whose occupations are administration or health care are more likely to depart from home earlier but back to home (from work location) later than those with other occupations. The probability that students or teachers have early departure times both from home and school is high. Workers and servers tend to depart from workplace late. Concerning gender, men are more likely to leave home earlier but leave work later than women. Regarding travel modes, commuters choosing walk or auto have late home-to-work and work-to-home departure times. Bikers tend to leave home earlier, while commuters taking bus depart from workplace later. Long-distance trips tend to be made later from home while earlier from work location. Workers are more likely to leave home later than students.

3.4. Travel Time Estimation Models

3.4.1. AFT Model and KM Estimator

According to the travel survey data in Beijing, the average travel time for the home-to-work commute is 19.36 minutes, with a maximum of 205 minutes and a minimum of 1 minute; the average duration for the work-to-home commute is 18.36 minutes, with a maximum and a minimum of 168 minutes and 1 minute, respectively.

Treating travel times as natural continuous variables, one can use the hazard model to predict both the home-to-work and the work-to-home travel times. Hazard-based duration models are ideally suited to modeling duration data [20, 21], such as travel time and activity duration. The hazard (also called a hazard rate) represents a termination rate of the duration.

Let T be a nonnegative random variable representing the travel time. The hazard at time t on the continuous time-scale $h(t)$ is defined as the instantaneous probability that the travel duration under study will end in an infinitesimal time period Δt after time t , given that the duration has not elapsed until time t . A mathematical definition for the hazard function is as follows:

$$h(t) = \lim_{\Delta \rightarrow 0^+} \frac{P(t \leq T < t + \Delta \mid T > t)}{\Delta}. \quad (3.4)$$

Let $f(\cdot)$ and $F(\cdot)$ be the density and cumulative distribution function for T , respectively. Then the probability of ending in an infinitesimal interval of range Δt , after time t , is $f(t)\Delta t$. And the probability that the process lasts for at least t is given by the survival equation (3.5):

$$S(t) = P(T > t) = 1 - F(t). \quad (3.5)$$

Thus, the hazard function can be further expressed as

$$h(t) = \frac{f(t)}{S(t)} = \frac{dF(t)/dt}{S(t)} = \frac{-dS(t)/dt}{S(t)} = \frac{-d \ln S(t)}{dt}. \quad (3.6)$$

The distribution of the hazard can be assumed to be one of many parametric forms or to be nonparametric. Because the distribution of the travel time is unknown, one of

Table 4: Estimation results of the departure time choice models.

Variables	Home-to-work departure time choice model		Work-to-home departure time choice model	
	Coef.	<i>t</i> -stat.	Coef.	<i>t</i> -stat.
Income	0.09	19.14*	0.06	16.18*
Age	-0.01	-10.49*	-0.01	-29.41*
Occu-w	—	—	0.43	32.00*
Occu-a	-0.14	-4.60*	0.35	11.44*
Occu-e	-0.72	-27.09*	-0.25	-12.96*
Occu-s	—	—	0.27	11.68*
Occu-h	-0.48	-12.96*	0.33	9.24*
Gender	-0.11	-9.34*	0.08	8.36*
Mode-w	0.18	7.89*	0.42	35.49*
Mode-bi	-0.41	-19.30*	—	—
Mode-bu	—	—	0.59	4.76*
Mode-a	0.10	3.12*	0.41	15.29*
Distance	0.000067	-32.89*	-6.26e - 06	-3.11*
Work	0.39	12.92*	—	—
α_1	-2.42	—	-1.30	—
α_2	-0.98	—	0.01	—
α_3	0.58	—	0.97	—
α_4	1.60	—	2.07	—
Hit ratio		64.36%		61.00%
<i>P</i> value		0.0000		0.0000
<i>N</i>			37842	

*Significant at 1% level.

the nonparametric methods, the Kaplan-Meier (KM) product limit estimator, is conducted to explore the covariate effects and the potential distribution.

As a nonparametric method, the KM estimator produces an empirical approximation of survival and hazard but hardly takes any covariate effects into consideration. It is similar to an exploratory data analysis. Denoting the distinct failure times of individuals n as $t_1 < t_2 < \dots < t_m$, the KM estimator of survival at time t_i is computed as the product of the conditional survival proportions:

$$S_{KM}(t_i) = \prod_{k=1}^i \frac{r(t_k) - d(t_k)}{r(t_k)}, \quad (3.7)$$

where $r(t_k)$ is the total trips at risk for ending at t_k and $d(t_k)$ is the number of trips stopping at t_k .

By using the KM estimator, the survival function curves of the home-to-work and the work-to-home travel time are estimated, which are shown in Figures 3 and 4, respectively. The results indicate that the survival probability decreases with travel time, which implies an accelerated failure time (AFT) model with Weibull or Exponential distribution should be employed. Therefore, the AFT model is developed to examine the linkages between travel time and covariates relative to individual and household.

The AFT model is one of the popular parametric forms of hazard model. It permits the covariates to affect the duration dependence. The survival function of AFT model is given as

$$S(t) = S_0[t \cdot \exp(-\beta'X)], \quad (3.8)$$

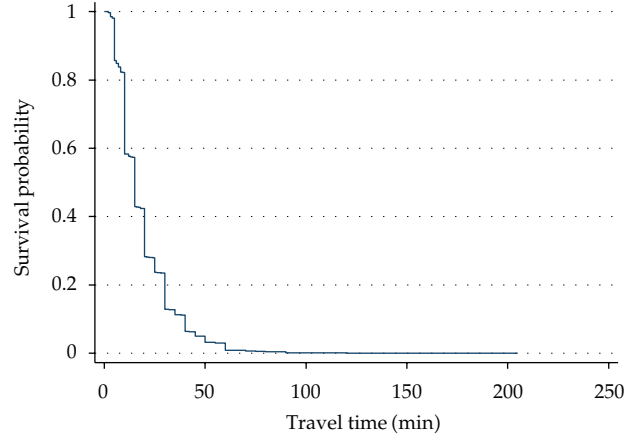


Figure 3: Survival curve of the home-to-work travel time.

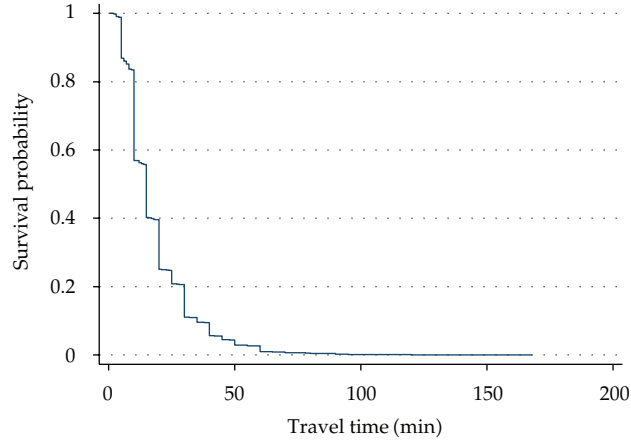


Figure 4: Survival curve of work-to-home travel time.

where $S_0(\cdot)$ is the baseline survival function. The corresponding hazard function is

$$h(t) = \frac{-\partial S(t)/\partial t}{S(t)} = h_0[t \cdot \exp(-\beta'X)] \exp(-\beta'X). \quad (3.9)$$

The AFT model can be expressed as a log-linear model:

$$\ln t = \beta'X + \varepsilon. \quad (3.10)$$

Assuming the random error ε follows either a Weibull distribution or an Exponential distribution, one can get two kinds of AFT models, and both of them are often used in duration analysis.

3.4.2. GA for Parameter Optimization

The parameters in the AFT models will influence the estimation efficiency and prediction accuracy of the models greatly, especially for large-scale or real-time feature practice application. Therefore, this paper attempts to find the appropriate parameters in AFT models by using GA. GA is a part of evolutionary computing, which is a rapidly growing area of artificial intelligence. The process of GA is as follows.

Encoding of Chromosome

GA is started with a set of solutions (represented by chromosomes) called population. The individuals comprising the population are known as chromosomes. In most GA applications, the chromosomes are coded as a series of zeroes and ones, or a binary bit string. For the travel time forecasting models, some parameters are continuous valued (like *distance* and *age*) while some are discrete valued (such as the variables about mode and occupation). Therefore, the real encodings were adopted for continuous-valued parameters, and the binary bit string was adopted for discrete-valued parameters. Thus, each chromosome consists of n "genes", $gen_1^t, gen_2^t, \dots, gen_n^t$, which represents n parameters, respectively.

Crossover

Crossover is a reproduction technique that takes two parent chromosomes and produces two child chromosomes. A commonly used method for crossover called one-point crossover [29, 30] will be employed in this study. In this method, both parent chromosomes are split into left and right subchromosomes, where the left subchromosomes of each parent are the same length, and the right subchromosomes of each parent are the same length. Then each child gets the left subchromosome of one parent and the right subchromosome of the other parent, as shown in Figure 5. The split position (between two successive genes) is called the crossover point. For example, if the parent chromosomes are 011 10010 and 100 11110 and the crossover point is between bits 3 and 4 (where bits are numbered from left to right starting at 1), then the children are 01111110 and 100 10010. We will call crossover applied at the bit level to bit strings binary crossover, and crossover applied at the real parameter level real crossover.

Mutation

Mutation is a common reproduction operator used for finding new points in the search space to evaluate. A genetic mutation operation [31] is used in this paper.

Assume a chromosome is $G = (gen_1^t, gen_2^t, \dots, gen_n^t)$ if the gen_i^t ($i = 1, \dots, n$) is selected for the mutation, the mutation can be shown in

$$G' = (gen_1^{t-1}, gen_2^{t-1}, \dots, gen_n^{t-1}),$$

$$gen_i^t = \begin{cases} gen_i^{t-1} + \Delta(t, gen_{i,\max}^t - gen_i^{t-1}) & \text{if random}(0, 1) = 0, \\ gen_i^{t-1} + \Delta(t, gen_i^{t-1} - gen_{i,\min}^t) & \text{if random}(0, 1) = 1, \end{cases} \quad (3.11)$$

where n is the total number of the parameters.

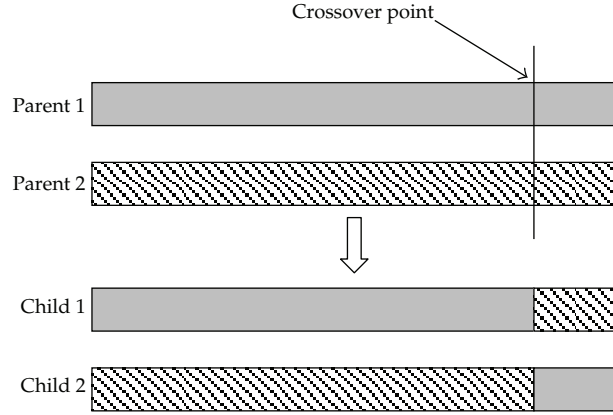


Figure 5: Crossover diagram.

The function $\Delta(t, y)$ returns a value between $[0, y]$ given in

$$\Delta(t, y) = y \times \left(1 - r^{(1-t/T_{\max})^n}\right), \quad (3.12)$$

where r is a random number between $[0, 1]$; T_{\max} is a maximum number of generations. This property causes this operation to make a uniform search in the initial space when t is small and a very local one in later stages.

To deal with the problem that the mutation may violate the parameters constraints, we will assign a relatively high weight to reduce their probability of being selected in the following search [31].

Termination

There are four GA parameters, namely, p_c , p_m , p_{size} , and T_{\max} , that need to be predetermined. Considering the features of this problem and our experiences in GA, the values of four GA parameters are set to be 0.6, 0.05, 80, and 5000, respectively.

3.4.3. Estimation Results

Home-To-Work Travel Time Estimation Model

Figure 6 illustrates the survival curves for the home-to-work travel time influenced by several major variables. It shows that factors of departure time, travel mode, income, gender, and going to work have influences on home-to-work travel time.

Two AFT models are estimated, each of which assumes the random error in (3.10) follows a Weibull distribution, and an Exponential distribution, respectively. The parameters are optimized using GA and then estimated using maximum likelihood estimation (MLE). The estimation results are shown in Table 5.

The mean absolute percentage error (MAPE), which looks at the average percentage difference between predicted values and observed ones, is adopted to examine the accuracy

Table 5: Estimation results of the home-to-work travel time model.

Variables	Weibull distribution		Exponential distribution	
	Coef.	z-stat.	Coef.	z-stat.
Constant	3.11	163.26	2.84	73.09
Income	−0.01	−4.82	−0.01	−2.65
Age	0.01	22.49	0.00	7.00
Occu-e	−0.20	−19.86	−0.10	−4.49
Gender	−0.04	−7.38	−0.04	−3.72
Mode-w	−0.18	−17.28	−0.14	−7.65
Mode-bi	0.07	7.04	0.15	8.93
Mode-bu	0.20	4.28	—	—
Mode-a	−0.04	−2.97	—	—
Distance	0.00013	100.58	0.00012	57.51
Departure time	−0.09	−29.84	−0.09	−13.06
Work	−0.32	−29.82	−0.19	−7.36
γ	0.50	—	1	—
Prob > chi ²	0.0000		0.0000	
N	37842			

Table 6: Goodness of fit index and estimated distribution statistics of the home-to-work travel time model.

Model statistics	Weibull distribution	Exponential distribution
MAPE value	0.4449	0.3882
Mean (min)	18.46	13.66
Maximum (min)	6.02	5.09
Minimum (min)	158.64	114.08

of the developed home-to-work travel time model. MAPE is calculated as

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^n \left| \frac{A_i - P_i}{A_i} \right|, \quad (3.13)$$

where A_i is the observed value and P_i is the predicted value for observation i . The MAPE values of the two AFT models are shown in Table 6.

According to the results shown in Table 5, the MAPE value of the Exponential distribution is less than that of the Weibull distribution, indicating that the values predicted by the AFT model with the Exponential distribution is more close to the actual travel time. Therefore, the Exponential distribution function is chosen. The hazard function and survival function are shown as follows:

$$\begin{aligned}
 h(t) = & \exp (2.84 - 0.01 \text{ Income} + 0.0098 \text{ Age} - 0.1 \text{ Occue} - 0.04 \text{ Gender} - 0.14 \text{ Modew} \\
 & + 0.15 \text{ Modebi} + 0.00012 \text{ Distance} - 0.09 \text{ Departure time} - 0.19 \text{ Work}), \\
 s(t) = & \exp (- \exp (2.84 - 0.01 \text{ Income} + 0.0098 \text{ Age} - 0.1 \text{ Occue} - 0.04 \text{ Gender} - 0.14 \text{ Modew} \\
 & + 0.15 \text{ Modebi} + 0.00012 \text{ Distance} - 0.09 \text{ Departure time} - 0.19 \text{ Work})).
 \end{aligned} \quad (3.14)$$

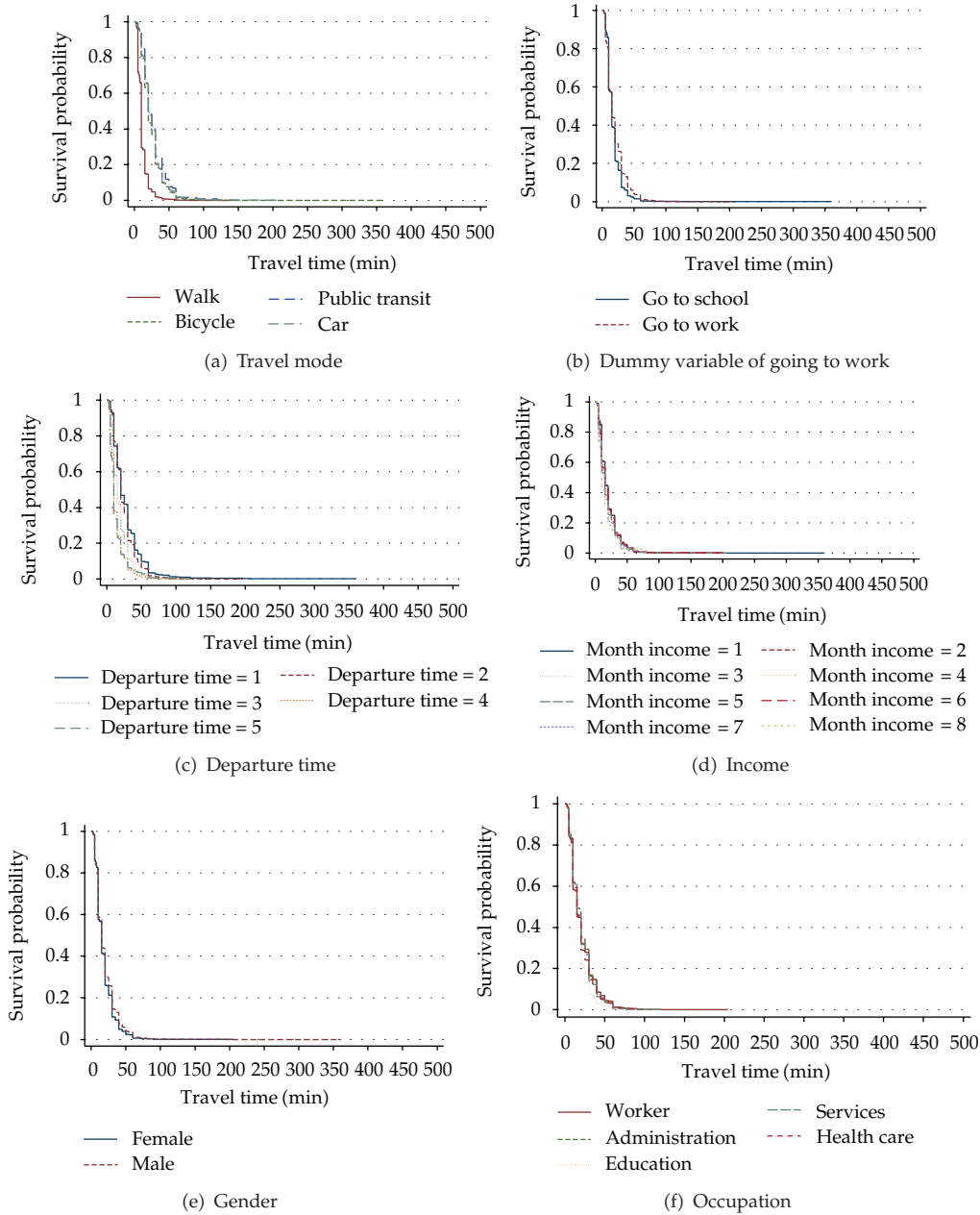


Figure 6: Survival curves of the home-to-work travel time impacted by several factors.

The estimation results indicate that the most essential factor of travel time is distance. The longer the distance from home to work is, the more time it will take. Comparing with other travel modes, the travel times of walk are 14% lower while those of bike are 15% higher. The reason is that, at first short distance encourages short travel time according to the estimation results, and then distance has influence on mode decisions, that is, walking usually belongs to short-distance travel comparing to biking. As for the factor of departure time,

the results show that the later the departure times are, the longer the travel times will be. In the above three parameters, the factors of distance and mode are related to the transportation network, while the real traffic condition is considered by using departure time as a factor, because the traffic condition depends regularly on departure time. For instance, if a traveler departs from home in the morning peak time, the probability that he/she encounters traffic congestion is much larger than that in the nonpeak time. The results also show that the higher travelers' income is, the less their travel time will be. The travel times of students or teachers are about 10% less than those of other travelers. The older the travelers are, the longer the travel time will be. Women have longer travel time than men. The travel times of travelers whose occupation is administration are 19% longer than that of education.

Work-To-Home Travel Time Estimation Model

Figure 7 illustrates the survival curves for the work-to-home travel time influenced by several variables of the interest. The AFT models with both Weibull distribution and Exponential distribution are employed for the work-to-home travel time modeling. The estimation results and the MAPE values of the two AFT models are shown in Tables 7 and 8, respectively.

Same as the home-to-work model, the AFT model of work-to-home travel times with Exponential distribution is better than that with Weibull distribution. Therefore, the former is selected. The hazard function and survival function are as follows:

$$\begin{aligned}
 h(t) &= \exp(2.53 - 0.01 \text{ Income} + 0.0033 \text{ Age} - 0.03 \text{ Occuw} + 0.07 \text{ Occue} - 0.06 \text{ Modew} \\
 &\quad + 0.08 \text{ Modebi} + 0.00015 * \text{Distance} - 0.06 \text{ Departure time}), \\
 s(t) &= \exp(-\exp(2.53 - 0.01 \text{ Income} + 0.0033 \text{ Age} - 0.03 \text{ Occuw} + 0.07 \text{ Occue} \\
 &\quad - 0.06 \text{ Modew} + 0.08 \text{ Modebi} + 0.00015 * \text{Distance} - 0.06 \text{ Departure time})).
 \end{aligned}
 \tag{3.15}$$

The estimation results indicate that the travel time of high-income travelers is 1% lower than that of low-income travelers. Moreover, old persons are likely to spend longer time in work-home trip. Regarding commuter's occupation, blue-collar workers are likely to spend shorter time for evening commute trip, while the travel times for teachers or students are longer. Comparing with other modes, walking trips have shorter time, while cycling trips have longer time. Long-distance trip takes longer travel time. The later the commuters depart from work, the longer the travel times will be.

4. Prediction of the Commute Activity-Travel Agenda

As explained in Section 3.1, the key timings and durations A_1 , D_1 , and A_2 can be calculated once the values of Dt_1 , T_1 , Dt_2 , and T_2 are predicted. Here is an example: the first member in the family with ID number 010104065 in our sample, Mr. Chen, is 45 years old, has an occupation of services, and earns 0–1500 RMB every month. His commute mode is walk, and the distance of one-way commute trip is 1500 meters. He had a typical commute activity-travel pattern on the survey day, which is shown in Figure 8.

Based on the developed models, his daily commute time allocation is predicted and shown in Figure 9.

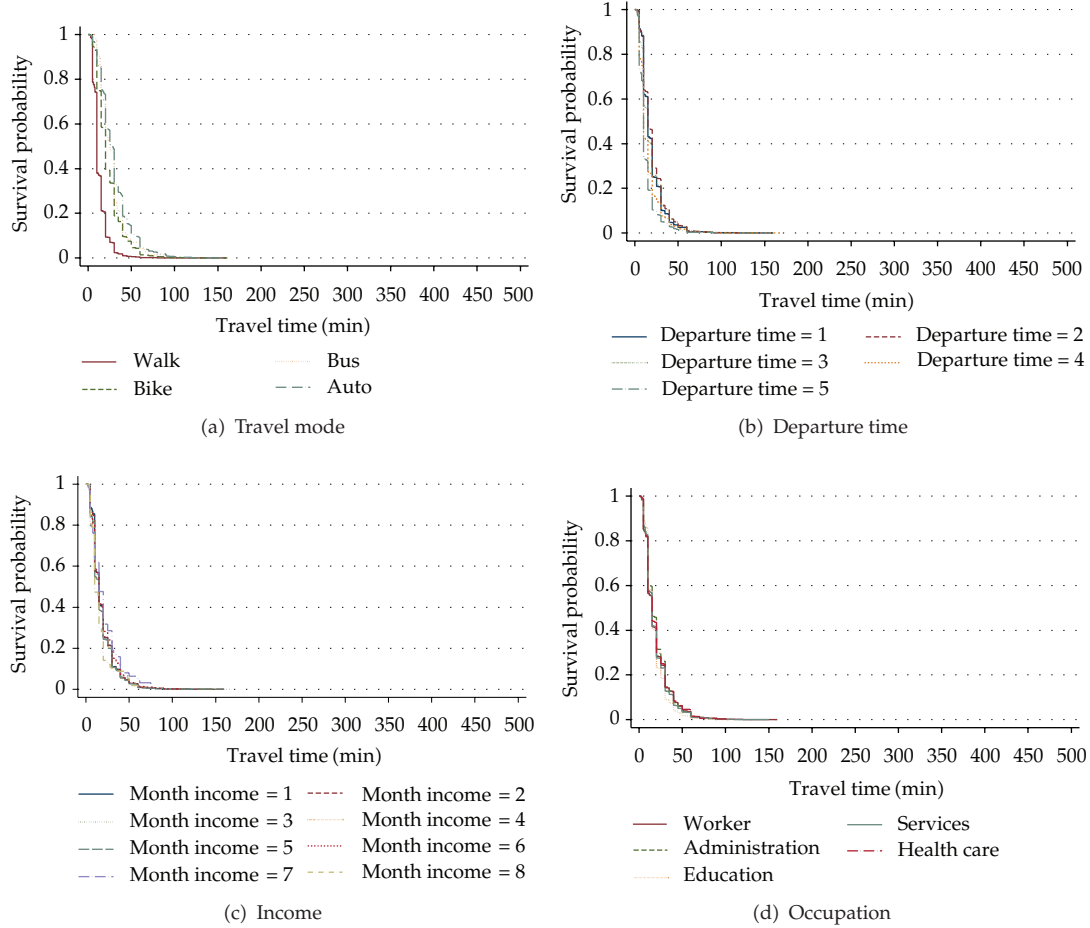


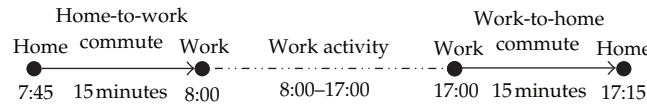
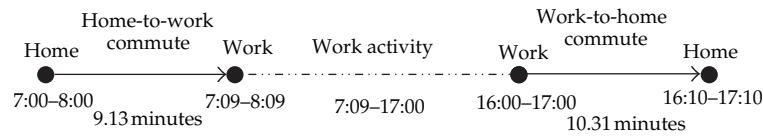
Figure 7: Survival curves of the work-to-home travel time impacted by several factors.

Table 7: Estimation results of the work-to-home travel time model.

Variables	Weibull distribution		Exponential distribution	
	Coef.	z-stat.	Coef.	z-stat.
Constant	2.65	164.15	2.53	79.98
Income	−0.01	−8.13	−0.01	−3.57
Age	0.0034	19.23	0.0033	8.80
Occu-w	−0.03	−6.20	−0.03	−2.22
Occu-e	0.05	6.56	0.07	4.25
Mode-w	−0.06	−5.53	−0.06	−3.51
Mode-bi	0.05	4.38	0.08	−3.51
Mode-a	0.05	3.33	—	—
Distance	0.00015	119.81	0.00015	66.12
Departure time	−0.06	−23.50	−0.06	−11.96
γ	0.4835	—	1	—
Prob > chi ²	0.0000		0.0000	
N	37842			

Table 8: Goodness of fit index and estimated distribution statistics of the work-to-home travel time model.

Model statistics	Weibull distribution	Exponential distribution
MAPE value	0.4541	0.3905
Mean (min)	18.50	13.36
Maximum (min)	8.44	6.17
Minimum (min)	269.11	172.80

**Figure 8:** Observed commute activity-travel pattern and time allocation.**Figure 9:** Predicted commute time allocation.

Comparing with the observed values, the errors of the predicted results can be calculated as follows.

- (i) The errors of home-to-work departure time: the maximum error is 45 minutes, the minimum error is 15 minutes.
- (ii) The error of home-to-work travel time: 5.47 minutes.
- (iii) The errors of arrival time at work location: the maximum error is 51 minutes, and the minimum error is 0 minute.
- (iv) The errors of work-to-home departure time: the maximum error is 60 minutes, and the minimum error is 0 minute.
- (v) The error of work-to-home travel time: 4.29 minutes.
- (vi) The errors of arrival time at home: the maximum error is 65 minutes, and the minimum error is 5 minutes.

By comparing the predicted values of the developed daily time allocation model with the observed values of our sample, the maximum errors for all the departure times and the activity-travel durations are as follows: $\text{error}(Dt_1) = 21.38$ minutes; $\text{error}(T_1) = 7.40$ minutes; $\text{error}(A_1) = 28.78$ minutes; $\text{error}(D_1) = 28.78$ minutes; $\text{error}(Dt_2) = 23.40$ minutes; $\text{error}(T_2) = 7.17$ minutes; $\text{error}(A_2) = 30.57$ minutes. These statistics indicate that the overall goodness of fit of the model is rather satisfying.

Results also show that the errors of departure and arrival time are much larger than that of the travel time. The main reason is that we divided the natural continuous departure time into discrete time interval artificially, which reduces the predictive accuracy of the model. It has been tested that the smaller the interval is, the higher the predictive accuracy will be. However, as there are already five alternatives for both of the departure time choice models, at least half-an-hour interval will make the number of the alternatives double. Then the model will be more complex and the efficiency of the model will be lower.

5. Conclusions

In this paper, we have formulated and estimated a joint model of departure time choice and travel duration for commuters' daily activity-travel time allocation. Two ordered probit models have been employed to forecast the home-to-work and work-to-home commute departure time. By doing so, we were able to recognize the natural temporal ordering among the departure time alternatives and address the IIA limitation of the standard MNL model. Furthermore, two AFT models were built and estimated to predict home-to-work and work-to-home travel times by using GA as parameters optimization. Then timing choice of a simple daily activity-travel pattern has been calculated.

Comparing with the previous studies, this paper developed a joint discrete-continuous model system to predict all the departure times and the activity-travel durations of a typical daily commute activity pattern. Results of this study not only contribute to developing a full-scale daily activity pattern forecasting model but also provide useful insights in the influence of sociodemographic variables on activity-trip timing decisions as well as the time constraint between daily activities and trips. Moreover, GA contributes to the optimization and thus the high accuracy of the travel time prediction model. In addition, this analysis of daily commute time allocation can be applied to a wide range of TDM policies, especially the measures aimed at adjusting the commute times, such as flexible work and compressed working week. For examining the effects of the traffic demand strategies, the developed model cannot only describe the overall change of the daily activity schedule caused by the strategies but also explore the time tradeoff between the connected trips as well as trips and activities. Besides evaluating the effects of the transportation demand management strategies, this study is also essential for planning the development and construction of new transportation infrastructure as well as examining the potential responses to improved traffic operational measures.

The results of this paper confirm that the discrete choice models and the continuous models can match well in the calculation of a whole-day activity-travel schedule, although comparing with the continuous models, the predictive accuracies of the discrete choice models are a little lower, as they divide the naturally continuous time into artificially defined time periods. Similar studies were also found to employ the discrete-continuous methods to model coupled mode and commute timing choice [12, 25]; joint activity-type preference, travel time, and activity duration [10]; as well as other activity-travel behaviors [27]. Therefore, it can be expected that we can further employ the combination of the discrete and continuous models to predict all the dimensions of the entire-day activities and trips. This future study can provide more useful insights into the nature of travelers' daily activity-travel decision making.

It should be pointed out that only the typical commute activity-travel pattern, comprising two commute trips and one work activity, has been considered in this paper. In reality, it is also common to observe other commute activity-travel patterns such as those including work-based subtour or home-based nonwork trips, or those having stops during commute travel. Further study may be done to model such daily activity-travel patterns. It will also be very important to exam one's activity-travel patterns over multiple days if the multiday travel survey data can be obtained.

Acknowledgment

The research is funded by the National Natural Science Foundation of China 50908099 and 51108053 and China Postdoctoral Science special Foundation 201104526.

References

- [1] M. B. Sundo and S. Fujii, "The effects of a compressed working week on commuters' daily activity patterns," *Transportation Research Part A*, vol. 39, no. 10, pp. 835–848, 2005.
- [2] F. Zong, Z. Juan, and H. Jia, "Examination of staggered shifts impacts on travel behavior: a case study of Beijing," *Transport*, STRA-2011-0129.R1, Forthcoming, 2011.
- [3] K. Ozbay and O. Yanmaz-Tuzel, "Valuation of travel time and departure time choice in the presence of time-of-day pricing," *Transportation Research Part A*, vol. 42, no. 4, pp. 577–590, 2008.
- [4] R. Hung, "Using compressed workweeks to reduce work commuting," *Transportation Research Part A*, vol. 30, no. 1, pp. 11–19, 1996.
- [5] G. Giuliano and T. F. Golob, "Staggered work hours for traffic management: a case study," *Transport Research Record*, vol. 1280, pp. 46–58, 1990.
- [6] C. Raux, T. Y. Ma, I. Joly, V. Kaufmann, E. Cornelis, and N. Ovtracht, "Travel and activity time allocation: an empirical comparison between eight cities in Europe," *Transport Policy*, vol. 18, no. 2, pp. 401–412, 2011.
- [7] D. Ettema, F. Bastin, J. Polak, and O. Ashiru, "Modelling the joint choice of activity timing and duration," *Transportation Research Part A*, vol. 41, no. 9, pp. 827–841, 2007.
- [8] P. Vovsha and M. Bradley, "Hybrid discrete choice departure-time and duration model for scheduling travel tours," *Transportation Research Record*, no. 1894, pp. 46–56, 2004.
- [9] S. Fujii and R. Kitamura, "Evaluation of trip-inducing effects of new freeways using a structural equations model system of commuters' time use and travel," *Transportation Research Part B*, vol. 34, no. 5, pp. 339–354, 2000.
- [10] M. M. Hamed and F. L. Mannering, "Modeling travelers' postwork activity involvement: toward a new methodology," *Transportation Science*, vol. 27, no. 4, pp. 381–394, 1993.
- [11] C. R. Bhat and S. K. Singh, "A comprehensive daily activity-travel generation model system for workers," *Transportation Research Part A*, vol. 34, no. 1, pp. 1–22, 2000.
- [12] K. M. N. Habib, N. Day, and E. J. Miller, "An investigation of commuting trip timing and mode choice in the Greater Toronto Area: application of a joint discrete-continuous model," *Transportation Research Part A*, vol. 43, no. 7, pp. 639–653, 2009.
- [13] X. Zhang, H. J. Huang, and H. M. Zhang, "Integrated daily commuting patterns and optimal road tolls and parking fees in a linear city," *Transportation Research Part B*, vol. 42, no. 1, pp. 38–56, 2008.
- [14] B. Dong, C. Cao, and S. E. Lee, "Applying support vector machines to predict building energy consumption in tropical region," *Energy and Buildings*, vol. 37, no. 5, pp. 545–553, 2005.
- [15] P. F. Pai, "System reliability forecasting by support vector machines with genetic algorithms," *Mathematical and Computer Modelling*, vol. 43, no. 3–4, pp. 262–274, 2006.
- [16] J. L. Bowman and M. E. Ben-Akiva, "Activity-based disaggregate travel demand model system with activity schedules," *Transportation Research Part A*, vol. 35, no. 1, pp. 1–28, 2000.
- [17] K. A. Small, "The scheduling of consumer activities: work trips," *American Economic Review*, vol. 72, no. 3, pp. 467–479, 1982.
- [18] C. R. Bhat, "Analysis of travel mode and departure time choice for urban shopping trips," *Transportation Research Part B*, vol. 32B, no. 6, pp. 361–371, 1998.
- [19] K. A. Small, "A discrete choice model for ordered alternatives," *Econometrica*, vol. 55, no. 2, pp. 409–424, 1987.
- [20] C. R. Bhat, "A hazard-based duration model of shopping activity with nonparametric baseline specification and nonparametric control for unobserved heterogeneity," *Transportation Research Part B*, vol. 30, no. 3, pp. 189–207, 1996.
- [21] C. R. Bhat, "A generalized multiple durations proportional hazard model with an application to activity behavior during the evening work-to-home commute," *Transportation Research Part B*, vol. 30, no. 6, pp. 465–480, 1996.
- [22] Z. C. Juan and J. C. Xianyu, "Daily travel time analysis with duration model," *Journal of Transportation Systems Engineering and Information Technology*, vol. 10, no. 4, pp. 62–67, 2010.
- [23] C. R. Bhat and J. L. Steed, "A continuous-time model of departure time choice for urban shopping trips," *Transportation Research Part B*, vol. 36, no. 3, pp. 207–224, 2002.
- [24] D. Janssens, Y. Lan, G. Wets, and G. Chen, "Allocating time and location information to activity-travel patterns through reinforcement learning," *Knowledge-Based Systems*, vol. 20, no. 5, pp. 466–477, 2007.
- [25] K. M. N. Habib, "Modeling commuting mode choice jointly with work start time and work duration," *Transportation Research Part A*, vol. 46, pp. 33–47, 2012.

- [26] T. Schwanen and M. Dijst, "Travel-time ratios for visits to the workplace: the relationship between commuting time and work duration," *Transportation Research Part A*, vol. 36, no. 7, pp. 573–592, 2002.
- [27] R. M. Pendyala and C. R. Bhat, "An exploration of the relationship between timing and duration of maintenance activities," *Transportation*, vol. 31, no. 4, pp. 429–456, 2004.
- [28] K. M. N. Habib, J. A. Carrasco, and E. J. Miller, "Social context of activity scheduling: discrete-continuous model of relationship between "with whom" and episode start time and duration," *Transportation Research Record*, no. 2076, pp. 81–87, 2008.
- [29] A. H. Wright, "Genetic algorithms for real parameter optimization," in *Foundations of Genetic Algorithms*, J. E. Rawlins, Ed., pp. 205–218, Morgan Kaufmann, San Mateo, Calif, USA, 1991.
- [30] N. Ahmad and A. F. M. A. Haque, "Optimization of process planning parameters for rotational components by genetic algorithms," in *Proceedings of the 4th International Conference on Mechanical Engineering*, vol. 7, pp. 227–233, Dhaka, Bangladesh, 2001.
- [31] J. B. . Yao, B. Z. Yao, L. Li, and Y. L. Jiang, "Hybrid model for displacement prediction of tunnel surrounding," *Neural Network World*, vol. 22, no. 3, pp. 263–275, 2012.

Research Article

A Hybrid Approach Using an Artificial Bee Algorithm with Mixed Integer Programming Applied to a Large-Scale Capacitated Facility Location Problem

**Guillermo Cabrera G.,^{1,2} Enrique Cabrera,^{3,4}
Ricardo Soto,^{1,5} L. Jose Miguel Rubio,^{1,6}
Broderick Crawford,¹ and Fernando Paredes⁷**

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

² Department of Engineering Science, University of Auckland, Auckland 1020, New Zealand

³ Instituto de Estadística, Pontificia Universidad Católica de Valparaíso, Valparaíso 2362807, Chile

⁴ CIMFAV Facultad de Ingeniería, Universidad de Valparaíso, Valparaíso 2362735, Chile

⁵ Universidad Autónoma de Chile, Santiago 7500138, Chile

⁶ Departamento de Computación e Informática, Universidad de Playa Ancha, Valparaíso 33449, Chile

⁷ Escuela de Ingeniería Industrial, Universidad Diego Portales, Santiago 8370109, Chile

Correspondence should be addressed to Guillermo Cabrera G., guillermo.cabrera@ucv.cl

Received 6 September 2012; Revised 12 November 2012; Accepted 14 November 2012

Academic Editor: Rui Mu

Copyright © 2012 Guillermo Cabrera G. 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.

We present a hybridization of two different approaches applied to the well-known Capacitated Facility Location Problem (CFLP). The Artificial Bee algorithm (BA) is used to select a promising subset of locations (warehouses) which are solely included in the Mixed Integer Programming (MIP) model. Next, the algorithm solves the subproblem by considering the entire set of customers. The hybrid implementation allows us to bypass certain inherited weaknesses of each algorithm, which means that we are able to find an optimal solution in an acceptable computational time. In this paper we demonstrate that BA can be significantly improved by use of the MIP algorithm. At the same time, our hybrid implementation allows the MIP algorithm to reach the optimal solution in a considerably shorter time than is needed to solve the model using the entire dataset directly within the model. Our hybrid approach outperforms the results obtained by each technique separately. It is able to find the optimal solution in a shorter time than each technique on its own, and the results are highly competitive with the state-of-the-art in large-scale optimization. Furthermore, according to our results, combining the BA with a mathematical programming approach appears to be an interesting research area in combinatorial optimization.

1. Introduction

Heuristics and bioinspired techniques have become efficient and effective alternatives for researchers in solving several complex optimization problems. These types of techniques are able to provide satisfactory solutions for most of the applied problems within acceptable computational times. However, in spite of their effectiveness, these techniques are not able to reach the optimal solution (or ensure its optimality) for large-scale combinatorial optimization problems. In contrast, mathematical programming techniques, particularly the Mixed Integer Programming (MIP), have been studied and developed by scholars over several decades with the main goal of obtaining optimal solutions to difficult problems using as little CPU time as possible. In this case, researchers must face the tradeoff between computational time and the quality of the result. For these reasons, the combination of meta-heuristics and various mathematical approaches has become a well-studied area. Interested readers can find two recent and comprehensive works on the hybridization of stochastic techniques and mathematical programming (MP) approaches in [1, 2].

Swarm Intelligence (SI), as well as other mathematical programming techniques, has been applied successfully to several difficult combinatorial optimization problems (see [3–6] for a range of applications). Additionally, as mentioned above, hybrid strategies have been developed to improve the effectiveness of these techniques. A more complete review is provided in Section 2.

The Artificial Bee algorithm (BA) is a relatively new approach in SI. Originally proposed in 2006 [7], it was mainly inspired by the foraging behavior of honeybees and has been applied to a range of problems in both combinatorial and functional optimizations with highly successful results. The BA has also been applied to large-scale problems [8] and has outperformed other well-known swarm-based algorithms, including Particle Swarm Optimization (PSO) and Differential Evolution (DE). However, as with almost all stochastic approaches, the BA cannot ensure optimality even when the optimal solution is found. For small- and medium-size instances, this limitation is not highly relevant because heuristics techniques have empirically demonstrated their ability to achieve convergence in quite acceptable time; therefore, solutions provided by stochastic approaches are likely to be optimal (or a very good approximation) in large-scale instances. However, when large-scale optimization is considered, it is not possible to know how “good” the solution provided by the heuristic.

In this paper, we propose a hybrid algorithm of the BA and the MIP for solution of several large-scale instances of the well-known Capacitated Facility Location Problem (CFLP). The CFLP is one of the most important problems for companies that distribute products to their customers. The problem consists of selecting specific sites at which to install plants, warehouses, and distribution centers while assigning customers to service facilities and interconnecting facilities using flow assignment decisions. This paper considers a two-level supply chain in which a single plant serves a set of warehouses, which in turn serve a set of end customers or retailers. Figure 1 shows the basic configuration of our supply chain. Therefore, we aim to solve this problem by finding a set of locations that allow us to serve the entire set of customers in an optimal way. As Figure 1 shows, each customer (or cluster) is served only by one warehouse.

Despite its good performance in several optimization problems, the BA is not able to provide an optimal solution for large-scale problems. Furthermore, it is possible to become trapped in a local optimum. To bypass these drawbacks, we propose a hybrid algorithm that selects a subset of promising centers using the BA and subsequently solves the subproblem

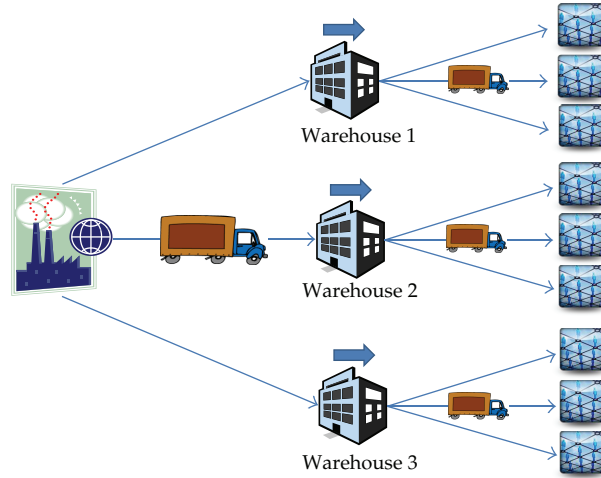


Figure 1: A two-level supply chain network configuration.

using a simple MIP algorithm. The BA guides the process while the MIP provides the optimal values of the simplified problems. One distinctive feature of our algorithm is that it solves the primary problem directly using the MIP algorithm, which is possible due to the reduction of the search space produced by the BA algorithm.

Although several works that have proposed various hybrid approaches to solve the CFLP and its uncapacitated (UFLP) version exist in the literature (e.g., [9, 10]), we are unaware of any prior publications that use the Bees Algorithm (BA) to solve a large-scale CFLP. Moreover, we have found no articles that hybridize the BA with an MP approach in any optimization problem. Therefore, one contribution of this paper is the presentation of a performance analysis for the hybrid BA-MIP algorithm. A second contribution is the application of the BA to a large-scale problem that provides optimal solutions rather than only locally optimal solutions.

The remainder of this paper is organized as follows. Section 2 presents an overview of the CFLP and BA concepts. The hybrid BA-MIP algorithm is covered in Section 3, and a detailed explanation of the algorithm is also presented in this section. Section 4 begins with a brief description of the benchmarks used in this paper, and the experimental results are subsequently presented and discussed. Finally, Section 5 outlines selected conclusions.

2. Literature Review

This section presents a literature review. Section 2.1 discusses the mathematical model for CFLP and provides relevant background for certain approaches presented in the literature. Section 2.2 provides an overview of the BA and highlights its main features.

2.1. Capacitated Facility Location Problem

The CFLP in this work contains a set of warehouses that supply a set of customers that are uniformly distributed in a limited area. The model considers the installation cost (i.e., the cost associated with opening a specific warehouse) and transportation or assignment cost

(i.e., the cost related to transportation of a specific amount of products from a warehouse to a customer). The mathematical model for the CFLP is presented as follows:

$$\sum_{i=1}^N F_i X_i + \sum_{i=1}^N \sum_{j=1}^M C_{ij} Y_{ij}, \quad (2.1)$$

s.t.:

$$\sum_{i=1}^N Y_{ij} = 1, \quad \forall j = 1, \dots, M \quad (2.2)$$

$$Y_{ij} \leq X_i, \quad \forall i = 1, \dots, N, \quad \forall j = 1, \dots, M \quad (2.3)$$

$$\sum_{j=1}^M \mu_j Y_{ij} \leq I_i^{\text{CAP}}, \quad \forall i = 1, \dots, N \quad (2.4)$$

$$X_i \in \{0, 1\}, Y_{ij} \in [0, 1], \quad \forall i = 1, \dots, N; \quad \forall j = 1, \dots, M. \quad (2.5)$$

Equation (2.1) represents the total system cost. The first term denotes the fixed setup and operating cost for opening warehouses, and the second term indicates the daily transport cost between the warehouse and the customers. Equation (2.2) ensures that the customer demands are completely served by the system. Equation (2.3) ensures that the customers are assigned to the installed warehouses ($X_i = 1$). Equation (2.4) states that the summation of the demand μ of each customer served by a particular warehouse i must be less than or equal to a threshold I_i^{CAP} , which can be different for each warehouse. Finally, (2.5) states the integrality (0-1) for the variable X_i and sets the range of the variable Y_{ij} . This model is NP-hard because it is clearly an extension of the UFLP, which is known to be NP-hard. Additionally, this model is one of the most basic examples related to location research, and several comprehensive surveys on location theory can be found in [11, 12].

The CFLP is well known in the operational research literature, and several authors have tackled this problem using different techniques (e.g., Genetic Algorithms are implemented in [13], and the Tabu Search (TS) algorithm is used in [14]). A comparison of the performance of these heuristics is provided in [15]. Additionally, mathematical-based approaches have been extensively developed to solve the CFLP, and algorithms based on Lagrangian relaxation represent the most common math-based approach [4, 8, 16]. In [17], the authors develop a column generation strategy to obtain the exact solution for large-scale instances. Other mathematical approaches are revised in [18, 19]. Moreover, mixed approaches using MP techniques and heuristics have been previously proposed. For instance, in [20], the authors develop a Lagrangian-based heuristic (LH) that provides lower bounds to the problem, and the TS algorithm is subsequently used to find the upper bound of the problem. In this case, the TS is initialized using the primary information provided by LH. Additionally, in [21], the authors combine Lagrangian relaxation with Ant Colony Optimization (ACO). Although the CFLP is one of the most studied models in combinatorial optimization, to the best of our knowledge, no BA study has tackled this problem.

2.2. Bees Algorithm

The Bees Algorithm (BA) is a nature-inspired approach that was originally proposed by Pham et al. [7] to solve complex optimization problems. Subsequently, several authors have applied different versions of the BA to tackle a wide range of combinatorial and continuous optimization problems. The algorithm has been demonstrated to be highly competitive, especially when compared with other swarm/population-based approaches such as the PSO [22] or Genetic Algorithms [23]. Karaboga and Akay [24] provide a comprehensive literature review and a comparison of the most important swarm/population-based approaches. A complete survey of various BA applications is also provided by Karaboga and Akay in [25]. In the following section, we present a brief description of the general structures of our BA. This description is mainly based on the descriptions provided by [7, 24–26].

One of the most important characteristics of swarm intelligence is the ability to exchange relevant information among individuals. This feature allows the swarm to generate and develop collective knowledge. In the case of the BA, this information addresses the recognition of promising sources of food found by any individual insect of the swarm. Another relevant feature of the bee swarm is the ability to intensify the search in certain patches that have been identified as promising. The two main attributes of most heuristics used in optimization are exploration and exploitation. The first provides a fast and wide search throughout the search space (which is usually too large). The second allows an intensive search of certain reduced search spaces (neighborhoods) that have been identified as “good-quality patches” during the exploration phase. In the BA case, the scout bees provide the exploration characteristics. The scout bees seek the search space in a (usually) random manner. Each scout bee visits a patch and evaluates it, and the scout bees have the ability to communicate the quality of the patch (fitness) to the unemployed bees. Depending on the attractiveness of each patch, the unemployed foragers will follow the scout bees to exploit the most promising patches. Once the patch is no longer attractive, a subset of bees will continue to search while the others wait for another promising patch. As in other heuristics, the balance between exploitation and exploration is a notably important issue for the BA. If we prioritize the exploration phase of the BA, it is likely to suffer from rather slow convergence. However, if we prioritize the exploitation phase of the BA, it is likely to become trapped in local minima. In most of the swarm optimization algorithms as well as other heuristics (e.g., Tabu Search), the BA uses memory structures to influence the next population (swarm). In this case, two main strategies provide information from former population to the new population. The first strategy uses experienced foragers, that is, bees with notably good fitness are included in the next population. The second strategy includes the best bee from a subset of the high-quality patches. In this manner, the use of various elements will determine the balance between exploration and exploitation.

3. Hybrid BA and LP Algorithm

In this paper, we present a BA that is hybridized with a MIP algorithm provided by GUROBI. Our algorithm contains the following distinctive features.

3.1. Variable Neighborhood Sizes

We use three different neighborhood sizes. It is important to note that we only refer to the “size” and not the “structure” of the neighborhood. More specifically, the neighborhood

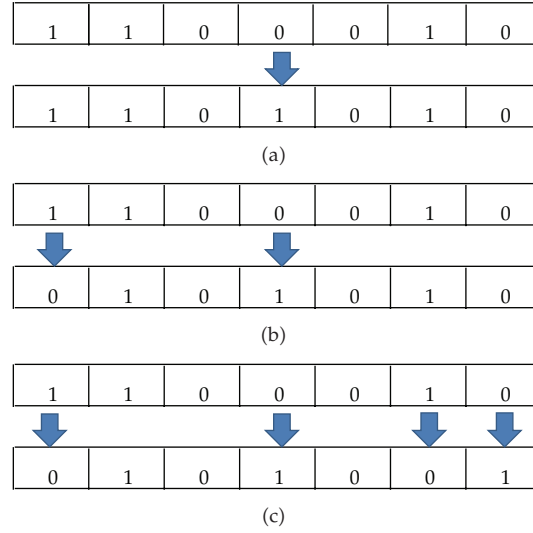


Figure 2: Three implemented neighborhood sizes: (a) *small-size* movement, (b) *medium-size* movement, and (c) *large-size* movement.

structure does not change during the algorithm execution. Figure 2 shows the three sizes and their function in the experiments. Figure 2(a) shows a *small-size* move in which only one location is modified (in this case, warehouse 4 is opened), and Figure 2(b) presents an example of a medium-size move. In this case, two locations are modified (warehouse 1 is closed, and warehouse 4 is opened). Figure 2(c) shows a large-size move.

Using notation from [27], we describe our variable neighborhoods as follow: let S denote any subset of open warehouses ($S \subseteq M$); the solution space \mathfrak{S} may be subsequently defined as all such possible subsets. The total number of solutions in \mathfrak{S} is $2^m - 1$. To define the neighborhoods, [27] defines a distance function as follows: let S_1, S_2 be any two solutions in \mathfrak{S} ; the distance between them is defined by $\rho(S_1, S_2) = |(S_1 \setminus S_2) \cup (S_2 \setminus S_1)|$. Therefore, the distance between one solution and another will increase when allocation of a specific customer in S_1 is different from the allocation of the same customer made in S_2 . An intuitive consequence of the above is that the distance between solutions S_1, S_2 is zero if and only if $S_1 = S_2$. Figure 1 shows an example of the distance between the different solutions. For instance, in Figure 2(a), $\rho(S_1, S_2) = 1$; in Figure 2(b), $\rho(S_1, S_2) = 2$; and in Figure 2(c), $\rho(S_1, S_2) = 4$. The values of $\rho()$ used in our algorithm are shown at the end of Section 3. It is important to note that we do not implement a variable neighborhood strategy as in [27]. Instead, we define different neighborhood structures that are only applied for quite specific tasks during the execution of our algorithm.

- (i) *Neighborhood-based start strategy*: the set of scout bees is initialized with a neighborhood-based strategy that uses the “large” movement to move from one solution to the other. This strategy allows us to use the LP solver more efficiently without compromising the algorithm performance.
- (ii) *Intensification procedure*: we implement an intensification procedure to exploit the promising patches found by the bees. When a promising solution is found, the intensification procedure is triggered, and a local search using a rather “small” neighborhood movement is carried out. The intent behind this procedure is to find

a local (and hopefully global) minimum as fast as possible, and this minimum is likely to be located near the promising solutions.

- (iii) *Roulette-wheel selection procedure*: we implement a procedure based on the well-known roulette wheel to select the elite bees and to assign the follower bees to particular elite (or selected nonelite) bees. This procedure allows for the inclusion of selected diversification mechanisms during the algorithm execution.
- (iv) *Use of experienced foragers*: we include the information of the best solution in the form of “*experienced foragers*” in each iteration of the algorithm, which allows us to include certain historical information to improve the convergence of the algorithm.

The algorithm begins with a set of ns scout bees, which are initialized using the neighborhood-based start strategy. Other techniques have found that certain “warm” start solutions could be implemented as well. However, in our practical experience, our approach is an easier and faster method of initializing the set of scout bees mainly due to the efficiency mechanisms of the solver. Once the ns scout bees are generated, they are sorted according to their fitness. The fitness is calculated using the objective function (2.1), which corresponds to an attractiveness measure in relation to the other scout bees. Note that the cost of each bee corresponds to the optimal solution for the subset of warehouses. This optimal value is provided by the MIP solver in less than two seconds on average. Therefore, this approach allows us to carry out a “global” optimization in different subspaces of the problem. Equations (3.1) and (3.2) state this attractiveness measure:

$$\sum_{b \in S} f(b) = tc, \quad (3.1)$$

$$\text{fitness}_b = \left(\frac{f(b)}{tc} \right), \quad (3.2)$$

where S is the set of scout bees and $f(x)$ is our objective function (2.1). The total cost of the set of scout bees is calculated n (3.1) and in (3.2), and the fitness is calculated based on the fitness of a specific bee and the total cost of the swarm. Because we are aiming to optimize the fitness, our results must be normalized such that the lowest cost becomes the most attractive. Once the scout bees have been sorted, the e bees (elite bees) are selected using the roulette-wheel selection procedure. We note that, if there are important differences among the costs of the ns scout bees, then the algorithm will likely select the e most attractive ones. In the same way, the ne (nonelite) bees are selected from the remaining scout bees. Subsets e and ne are sorted, and the attractiveness of each bee is calculated as explained above. Next, a set F (followers) is generated and assigned to the both elite and nonelite bees. Because the roulette-wheel selection procedure is used again, the elite bees are likely to recruit more followers than the less attractive nonelite bees. A local search is carried out at this stage using medium-size movements. Once the local search is completed, the algorithm selects the best bee from each patch. If a new best solution is reached, the intensification procedure is triggered. Following that procedure, the entire swarm is sorted based on fitness. Next, the elite and nonelite classification is executed again. In this step, a few random bees are added to the swarm to diversify the search. Additionally, the best solution is included via an experienced forager. The pseudocode for our hybrid BA-MIP algorithm is presented as follows Algorithm 1.

```

(1) Init
(2) Generate initial swarm  $S$ 
(3) Solve MIP of each  $S_b$ 
(4) Sort  $S$  based on fitness
(2.5) Select elite and non-elite bees
(3.1) While not end-criterion
(3.2) Assign followers
(8) For each elite and non-elite
(9) Search  $\eta(E) \cup \eta(NE)$ 
(10) End For
(11) Select Best Bee (bBEST)
(12) If bBEST < BestSol
(13) BestSol  $\leftarrow$  bBEST
(14) Intensification
(15) End If
(16) Joint and Sort  $F, E, NE$ 
(17) Sort  $S$  based on fitness
(18) Select elite and non-elite bees
(19) Add Random Bees
(20) Add BestSol Bee
(21) End While
(22) End

```

Algorithm 1: Hybrid BA-MIP algorithm.

Table 1: Results of the parameter tuning process.

Item	Test	Best
ns	10; 20	10
e	10%; 20%; 50% of ns	20%
ne	10%; 40% of ns	10%
Followers	20; 100	20
Neigh small	1; 2	2
Neigh med	4; 6	4
Intensification size	10; 20	20
Elite followers	60%; 90%	60%

3.2. Parameter Tuning

To obtain a parameter set for the BA algorithm, we performed several tests using different values for each parameter of the algorithm, and these values are presented in Table 1. The tests were applied on one of the medium-size instances. The selected values are shown in bold.

4. Computational Experiments and Discussion

This section explains the experiments, certain benchmarks from the literature included to validate our algorithm, and the generation of a set of large-scale instances. The results obtained from our BA-MIP algorithm are compared with those from a simple local search

Table 2: Optimal DND obtained for both the ILM-PR and ILM-CR models.

Instance	Opt	BA	BA + MIP		LS + MIP		
		Time	GAP	Time (sec)	GAP	Time (sec)	GAP
capa	19,240,822.449	Max	>300%	426.022	0%	367.735	0%
capb	13,656,379.578	Max	>300%	345.501	0%	317.468	0%
capc	11,646,596.974	Max	>300%	553.624	0%	129.336	0%

strategy (which uses the MIP to solve the allocation problem) as well as those from both the MIP algorithm and the BA applied independently. A comprehensive analysis and discussion is outlined at the end of this section.

4.1. Experiments and Computational Results

In this subsection, we present the benchmarks applied for performance comparison and the computational results obtained for the hybrid algorithm. Finally, we show a summary of the principal results obtained. The computational experiments were performed on an Intel Core Duo processor CPU T2700, 2.33 GHz with 2 GB of RAM and Windows XP operating system. The BA algorithm was implemented in the JAVA programming language using NetBeans IDE, and the MIP algorithm was modeled with the GUROBI solver.

To validate the algorithm and measure its convergence, we chose a set of medium-size instances that have known optimal solutions, namely, capa, capb, and capc. These instances were obtained from Beasley's ORLibrary (<http://people.brunel.ac.uk/~mastijb/jeb/info.html>). Additionally, a set of large instances (300 warehouses and 1000 clients, 500 warehouses and 1000 clients, and 1000 warehouses and 1000 clients) was created using the strategy provided in [8]. The set of customers and the set of warehouses are uniformly distributed over a plane of 10×10 distance units. The Euclidean distance between a customer i and a warehouse j corresponds to the transportation cost Y_{ij} . The demand d_j is calculated using a uniform distribution U [5, 35]. The I_i^{CAP} is calculated using U [100, 1600], and we amplify the capacity of the warehouse to obtain harder instances. Finally, the fixed cost of warehouse i is calculated by $F_i = U[0, 90] + U[100, 110] \sqrt{I_i^{\text{CAP}}/10}$. This expression takes into account the economies of scale [8]. As proposed in [8], we generate three different classes of problems: 300×1000 , 500×1000 , and 1000×1000 (warehouses \times customers). To avoid any instance-dependent effects, we generated 10 different instances for each class. We also executed the algorithm 30 times for each instance to assess and avoid outlier performance. The results presented in this section correspond to the average values from these experiments. The MIP algorithm was executed only once per instance due to its deterministic behavior, and the results obtained from the MIP algorithm are presented in Table 2. The stop criterion used in the three first instances (capa, capb, and capc) was $\text{GAP} \approx 0\%$, that is, we forced the algorithm to find the (known) optimal solution. Because the optimal solution value is not known for our generated large-scale instances, the algorithm was aborted after 2000 seconds or $\text{GAP} \leq 1\%$, whichever occurred first.

Figures 3 and 4 show the convergence of two algorithms (MIP and BA + MIP) for the instances 500.3 and 1000.4, respectively.

As shown in Figure 3 for the 500.3 instance, the two algorithms are both able to find notably good solutions within the time allotted. As expected, the MIP requires more CPU time than the BA-MIP hybrid algorithm to find the solution, and the solid line shows the LB

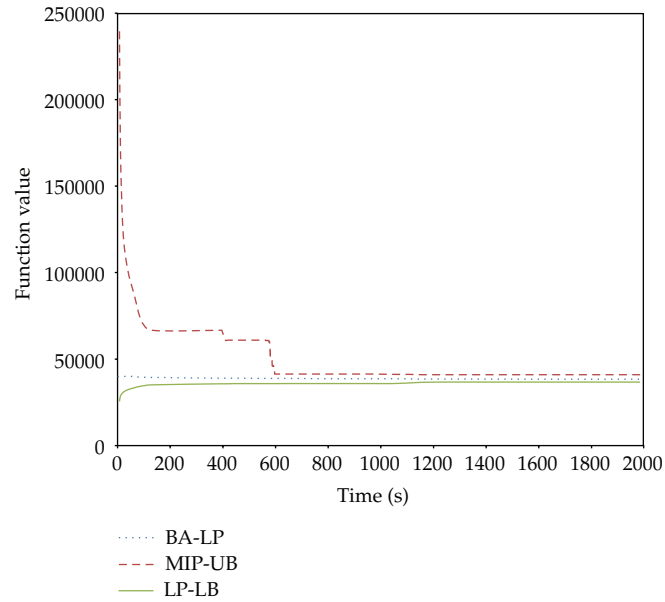


Figure 3: Comparison of the convergence between the MIP solver and the BA-MIP algorithm (instance 500.3).

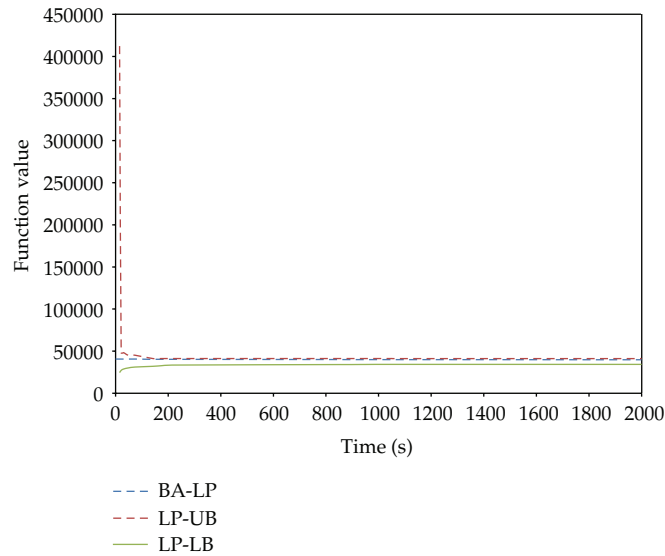


Figure 4: Comparison of the convergence between the MIP solver and the BA-MIP algorithm (instance 1000.4).

of this instance. A remarkable feature of our hybrid approach is its “warm” initial solution. As shown in Figure 3, the first iterations produce very good solutions that are quite close to the best solution reached by the MIP algorithm after the 2,000 seconds. We assume that this observation is due to our *neighborhood-based start strategy*.

Figure 4 shows a similar situation for the 1000_4 instance. The main difference in this case is the rapid convergence of the MIP algorithm. Despite this observation, the MIP is not able to find a better solution than that obtained by our BA-MIP algorithm.

Table 2 shows the average results obtained by the four algorithms for the three medium-size instances. The *Opt* column shows the optimal value produced for the respective instance. The *Time* column reports the CPU time required by the respective algorithm to reach the optimal value. The GAP column shows the difference between the average value and the optimal value as a percentage. A value of $\text{GAP} = 0\%$ indicates that the optimal value was reached, and $\text{GAP} > 0\%$ otherwise. The maximum time available for each instance corresponds to the time in which the MIP algorithm was able to find the optimal value for that instance.

As shown in Table 2, the BA + MIP and the LS + MIP were able to find the optimal solution more quickly than the MIP solver.

These results demonstrate that the robustness of our algorithm highlights the speed of our hybrid approach, which could be a determining factor for notably large-scale instances in which the MIP is not able to find the optimal value within a reasonable CPU time.

The *Time* column in Table 3 shows the time (in seconds) in which the best value was found by the respective algorithm. The GAP column shows the difference between the average value and the lower bound value as a percentage. As stated previously, the maximum time available for each instance was 2000 seconds. The excellent performance of our hybrid approach is quite obvious; it outperforms the MIP approach in almost all cases, especially those instances in which the number of warehouses is greater than or equal to 500. This situation demonstrates the robustness of our algorithm as well as its reliability. It is also important to note that, in most of the cases, the best value obtained by the MIP algorithm was improved upon by our BA-MIP algorithm before 1500 seconds had elapsed, which implies a time saving of 25% compared with the mathematical approach. Moreover, in most cases, the best value provided by our hybrid algorithm was found near the time limit, which could be considered as a quite promising feature because it means that our algorithm is able to escape from local optima. Additionally, we note that, in the few cases in which the MIP reached a better value than our hybrid approach did, the difference is not greater than 1% of the GAP. The results obtained from the simple BA algorithm are surprisingly poor. We believe this situation may be partially due to the size of the search space as well as to a lack of precision in the parameter tuning. However, even when a careful parameter tuning process was conducted, the improvement is only marginal if we consider the current GAP between the LB for each instance and the UB obtained by BA. However, the LS-MIP obtains very good results for all instances. These results confirm that even rather simple heuristic strategies can be significantly improved when combined with mathematical programming approaches. However, we note that most of the best values were obtained early in the time elapsed, which may mean that the LS-MIP algorithm is not able to explore large spaces and quickly converges to local minima. Despite this weakness, the good performance shown by the LS-MIP algorithm encourages further exploration with different hybrid approaches.

Table 4 summarizes the results (GAP, as a percentage) obtained by our three algorithms (the BA is not considered) independent of any instance-dependent effects. It can be clearly observed that despite the effects provoked by particular instances, our algorithm presents the most desirable features, that is, a reduced mean, which suggests that the result is closer to the LB on average, and the small variance can be interpreted as a measure of robustness and reliability.

Table 3: Obtained results per algorithm for all (very) large instances.

Instance	GAP MIP	BA	BA + MIP	LS + MIP			
		Time (sec)	GAP (%)	Time (sec)	GAP (%)	Time (sec)	GAP (%)
300_1	2.54	1,083.566	319.94	1,191.68	2.52	823,748	3.48
300_2	4.32	521.891	326.39	1,114.84	1.22	1,859.55	1.23
300_3	5.38	418.646	344.74	1,183.13	5.67	67.170	5.47
300_4	0.25	576.624	325.64	1,354.32	0.53	291.096	0.92
300_5	7.69	499.160	340.61	1,181.11	3.60	248,047	5.03
300_6	0.10	0.440	311.72	1,779.96	0.76	1,068.721	0.87
300_7	7.24	1,532.097	333.29	1,942.37	4.16	810.303	4.82
300_8	3.39	247.004	322.63	1,716.07	0.91	122.122	2.13
300_9	6.24	1,401.028	331.60	842.08	4.55	446.381	4.89
300_10	3.82	1,786.086	331.14	1111016	2.14	82.730	4.68
500_1	4.30	1,515.316	425.51	1,382.22	4.41	656.431	3.43
500_2	8.34	29.623	432.50	1,960.32	4.43	400.010	5.73
500_3	11.61	741.206	428.80	1,636.51	4.51	83.878	7.23
500_4	8.54	1,740.442	441.34	1,476.66	4.62	519.690	3.78
500_5	7.27	364.557	437.55	1,742.31	4.83	816.807	4.98
500_6	8.09	151.569	430.50	1,306.528	5.20	328.605	6.00
500_7	6.63	1,493.073	425.32	1,892.530	4.48	1,093.399	5.07
500_8	11.18	1,357.436	420.11	1,177.792	4.98	828,960	4.14
500_9	9.64	853.731	427.62	1,647.911	5.16	860.271	4.69
500_10	12.88	1,826.382	423.29	1,333.395	5.99	231,932	6.64
1000_1	18.75	89.000	331.06	1,514.020	15.62	246.482	17.13
1000_2	22.97	92.000	336.27	1,641.200	15.65	1,210.875	14.30
1000_3	20.33	85.000	337.88	1,979.140	16.45	1,165.605	15.22
1000_4	20.06	1,842.591	339.48	1,954.162	16.10	1,997.213	17.97
1000_5	17.91	88.000	332.41	1,974.203	16.64	305,483	17.60
1000_6	16.84	90.000	340.57	1,993.361	14.72	741838	14.41
1000_7	18.80	88.000	339.95	1,898.845	17.30	104.692	17.80
1000_8	19.20	89.000	322.16	1,948.752	14.31	1,206.733	14.76
1000_9	17.62	1,807.162	340.03	1,737.584	15.87	378.913	14.99
1000_10	18.93	317.728	340.53	1,074.975	17.25	609,155	16.98

Table 4: Summary of the results obtained by the MIP, LS-MIP, and BA-MIP.

	MIP	LS-MIP	BA-MIP			
Instances						
300x	4.14	6.47	3.35	3.14	2.61	2.93
500x	8.85	5.89	5.17	1.37	4.86	0.22
1000x	19.14	2.64	16.12	2.04	15.99	0.87

5. Conclusions and Future Work

In this paper, we have presented a hybrid algorithm based on the BA and the MIP and used it to solve the CFLP. The BA is applied primarily for the purpose of solving the location problem, that is, finding a subset from the available set of locations that satisfy the entire demand of the system. In contrast, the MIP is applied for the purpose of finding the optimal

solution by considering a specific subset of locations, that is, solving the allocation problem. Certain considerations are important and must be highlighted in this approach.

First, because the algorithm is able to find the optimal solution for each subset of selected warehouses, we are able to fairly compare those subsets. At the same time, our hybrid algorithm is able to find the optimal solution to medium-large problems, which is not possible with the use of common local search strategies. Additionally, obtaining the optimal solution allows us to compare the different strategies for warehouse selection and local search.

Second, compared with the widely used local search approaches that implement a swap in the assignment vector as a neighborhood move, our approach is able to visit more solutions because the entire tree corresponding to the feasible allocations is considered for each subset of warehouses in the search for the optimal solution of the subproblem by the MIP algorithm.

Third, our BA does not require extensive computational resources because most of the calculations are handled by the MIP solver, which is by far more efficient than many common heuristics implementations.

Our hybrid BA-MIP algorithm was able to find the optimal solution for a set of well-known large-scale instances found in the literature. Moreover, it outperformed both the BA and MIP approaches as applied separately. Compared with the state-of-the-art algorithms for location problems, our BA-MIP algorithm is highly competitive and reaches an optimal solution using less CPU time than both exact and stochastic approaches. Moreover, when the instances are notably large, the algorithm is able to find a better solution than that of the MIP approach in a fraction of the time. As widely reported in the literature over the last three decades, the combination of mathematical programming with heuristics and/or metaheuristics (Matheuristics [1]) provides robust and straightforward approaches to solving these types of problems.

In this paper, we have developed a hybrid algorithm using BA and LP using an approach that has not yet been reported. Due to the hybrid nature of the algorithm, certain changes were made in the structure of the BA heuristic. Use of the variable neighborhood “size,” the neighborhood-based initialization procedure, the intensification procedure, and the experienced foragers are the most important features of our implementation. These distinctive components allow us to improve the performance of the simple BA when combined with MIP.

The hybridization of BA with other such mathematical programming approaches as interior point methods, column generation, and gradient-based algorithms shows promise as a potentially valuable research area. Additionally, the application of similar hybrid approaches to more complex large-scale problem will be an interesting future research line.

References

- [1] V. Maniezzo, T. Stützle, and S. Voß, Eds., *Matheuristics: Hybridizing Metaheuristics and Mathematical Programming*, vol. 10, Annals of Information Systems, 2009.
- [2] C. Blum, J. Puchinger, G. Raidl, and A. Roli, “Hybrid metaheuristics in combinatorial optimization: a survey,” *Applied Soft Computing Journal*, vol. 11, no. 6, pp. 4135–4151, 2011.
- [3] G. Cabrera, S. Roncagliolo, J. P. Riquelme, C. Cubillos, and R. Soto, “Hybrid particle swarm optimization—simulated annealing algorithm for the probabilistic traveling salesman problem,” *Studies in Informatics and Control (SIC)*, vol. 21, no. 1, pp. 49–58, 2012.
- [4] J. E. Beasley, “Lagrangian heuristics for location problems,” *European Journal of Operational Research*, vol. 65, pp. 383–399, 1993.

- [5] M. H. Kashan, N. Nahavandi, and A. H. Kashan, "DisABC: a new artificial bee colony algorithm for binary optimization," *Applied Soft Computing*, vol. 12, no. 1, pp. 342–352, 2012.
- [6] C. Canel and B. M. Khumawala, "A mixed-integer programming approach for the international facilities location problem," *International Journal of Operations & Production Management*, vol. 16, no. 4, pp. 49–68, 1996.
- [7] D. T. Pham, A. Ghanbarzadeh, E. Koç, S. Otri, S. Rahim, and M. Zaidi, "The Bees Algorithm—a novel tool for complex optimization problems," in *Proceedings of the 2nd International Virtual Conference on Intelligent Production Machines and Systems (IPROMS '06)*, pp. 454–459, 2006.
- [8] F. Barahona and F. A. Chudak, "Near-optimal solutions to large-scale facility location problems," *Discrete Optimization*, vol. 2, no. 1, pp. 35–50, 2005.
- [9] M. Caserta and E. Quiñonez Rico, "A cross entropy-based metaheuristic algorithm for large-scale capacitated facility location problem," *Journal of the Operational Research Society*, vol. 60, pp. 1439–1448, 2009.
- [10] K. S. Al-Sultan and M. A. Al-Fawzan, "A tabu search approach to the uncapacitated facility location problem," *Annals of Operations Research*, vol. 86, pp. 91–103, 1999.
- [11] M. S. Daskin, *Network and Discrete Location*, John Wiley & Sons, New York, NY, USA, 1995.
- [12] Z. Drezner and H. W. Hamacher, *Facility Location: Applications and Theory*, Springer, New York, NY, USA, 2004.
- [13] M. J. Cortinhal and M. E. Captivo, "Genetic algorithms for the single source capacitated location problem," in *Metaheuristics: Computer Decision-Making*, M. G. Resende and J.P. de Sousa, Eds., pp. 187–216, Kluwer Academic Publisher, 2004.
- [14] M. H. Sun, "A tabu search heuristic procedure for the capacitated facility location problem," *Journal of Heuristics*, vol. 18, no. 1, pp. 91–118, 2012.
- [15] M. A. J. ArosteGUI, S. N. Kadipasaoglu, and B. M. Khumawala, "An empirical comparison of tabu search, simulated annealing, and genetic algorithms for facilities location problems," *International Journal of Production Economics*, vol. 103, pp. 742–754, 2006.
- [16] G. Cornuejols, R. Sridharan, and J. M. A. Thizy, "comparison of heuristics and relaxation for the capacitated plant location problem," *European Journal of Operational Research*, vol. 50, pp. 280–297, 1991.
- [17] A. Klose and A. Drexl, "Lower bounds for the capacitated facility location problem based on column generation," *Management Science*, vol. 51, no. 11, pp. 1689–1705, 2005.
- [18] P. Sinha, "Observations on some heuristic methods for the capacitated facility location problem," *Opsearch*, vol. 49, no. 1, pp. 86–93, 2012.
- [19] J. K. Sankaran, "On solving large instances of the capacitated facility location problem," *European Journal of Operational Research*, vol. 178, no. 3, pp. 663–676, 2007.
- [20] M. J. Cortinhal and M. E. Captivo, "Upper and lower bounds for the single source capacitated location problem," *European Journal of Operational Research*, vol. 151, no. 2, pp. 333–351, 2003.
- [21] C. Chen and C. Ting, "Combining lagrangian heuristic and ant colony system to solve the single source capacitated facility location problem," *Transportation Research E*, vol. 44, no. 6, pp. 1099–1122, 2008.
- [22] R. C. Eberhart, Y. Shi, and J. Kennedy, *Swarm Intelligence*, Morgan Kaufman, 2001.
- [23] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.
- [24] 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.
- [25] D. Karaboga and B. A. Akay, "survey: algorithms simulating bee swarm intelligence," *Artificial Intelligence Review*, vol. 31, pp. 61–85, 2009.
- [26] L. Özbakir, A. Baykasoglu, and P. Tapkan, "Bees algorithm for generalized assignment problem," *Applied Mathematics and Computation*, vol. 215, no. 11, pp. 3782–3795, 2010.
- [27] P. Hansen, J. Brimberg, D. Urošević, and N. Mladenović, "Primal-dual variable neighborhood search for the simple plant-location problem," *INFORMS Journal on Computing*, vol. 19, no. 4, pp. 552–564, 2007.

Research Article

Economic Analysis on Value Chain of Taxi Fleet with Battery-Swapping Mode Using Multiobjective Genetic Algorithm

**Guobao Ning,¹ Zijian Zhen,² Peng Wang,³
Yang Li,⁴ and Huaixian Yin⁵**

¹ School of Automotive Studies, Tongji University, No. 4800 Cao'an Road, Shanghai 201804, China

² The High Technology Research and Development Center, No. 1 Sanlihe Road, Beijing 100044, China

³ China Automotive Engineering Research Institute Co., Ltd., Xinghuo Mansion, Fengtai District, Beijing 100071, China

⁴ Potevio New Energy Co., Ltd., No. 6 Haidian North Second Street, Haidian District, Beijing 100080, China

⁵ Qingdao University, No. 308 Ningxia Road, Qingdao 266071, China

Correspondence should be addressed to Guobao Ning, guobao-tj@163.com

Received 4 October 2012; Revised 27 November 2012; Accepted 27 November 2012

Academic Editor: Baozhen Yao

Copyright © 2012 Guobao Ning 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 economic analysis model on value chain of taxi fleet with battery-swapping mode in a pilot city. In the model, economic benefits of charging-swapping station group, taxi company, and taxi driver in the region have been taken into consideration. Thus, the model is a multiobjective function and multiobjective genetic algorithm is used to solve this problem. According to the real data collected from the pilot city, the multiobjective genetic algorithm is tested as an effective method to solve this problem. Furthermore, the effects of price of electricity, price of battery package, life cycle of battery package, cost of battery-swapping devices and infrastructure, and driving mileage per day on the benefits of value holders are analyzed, which provide theoretical and practical reference for the deployment of electric vehicles, for the national subsidy criteria adjustment, technological innovation instruction, commercial mode selection, and infrastructure construction.

1. Introduction

Economic analysis on value chain of taxi fleet with battery-swapping mode in a pilot city is a basic condition for the sustainable commercialization of electric vehicle operations. Some factors, such as vehicle cost, life cycle of battery package, and reliability, are used to affect the benefits of the value holders. In addition, electricity price, cost of battery-swapping devices

and infrastructure, and national subsidy criteria also have impacts on the benefits of all value holders as well as the sustainable commercial operation and popularization of electric vehicles. There is much more analysis on current status, policies, regional development, business models, and value chain management model of China electric vehicle demonstration operations [1–5]. However, these traditional methods have qualitatively analyzed current status, policies, regional development, business models, and value chain management model of China's electric vehicle demonstration operations. If an analyzed model can be proposed to provide the economic effect by quantitative analysis, it is very important to judge the feasibility of this project.

Many modeling approaches have been applied to evaluate the economic analysis. Hamilton [6] presented a method on the economic analysis to model the changes in regime which is based on the nonstationary time series and the business cycle. Aidt [7] provided an economic analysis which considered the notion of a benevolent principal as a normative theory of corruption.

To represent the economic analysis on value, the competitive nature about these factors has occurred as a result of parallel development in the application of taxi fleet with battery-swapping. In this paper, the model consists of three parts. One is empirical model of monthly operating benefits of charging-swapping station group in the region, another is empirical model of monthly operating benefits of taxi fleet company, and the third is empirical model of monthly operating benefits of taxi fleet drivers. To reflect the effect of these factors which will directly determine the effectiveness of the new project, the electricity price, battery package price, battery package, battery-swapping devices, infrastructure, and driving mileage per day need to be discussed. Thus, this paper took some pilot cities as an example and established a multiobjective model which is attempted to optimize the economic analysis on value chain of taxi fleet with battery-swapping mode in some pilot cities with a multiobjective model which can describe the economic analysis from different aims.

Taxi fleet with battery-swapping mode in real world, which is related to large of factors, is inefficient to be solved by classical optimization techniques and belongs to a class of problems referred to as nondeterministic polynomial-time hard (NP-hard) [8]. Generally, heuristics are considered as a first choice to solving these combinatorial optimization problems [9–11]. Genetic algorithm [12], which is a multipurpose optimization tool, has successfully been applied in a wide range of optimization problems [13, 14]. Furthermore, GA has been used widely in transportation fields [15–17]. Altıparmak et al. [18] proposed a genetic algorithm to find the solutions for multi-objective supply chain networks. Due to many successful results of combination optimization applications with GA, GA is also used to optimize this taxi fleet with battery-swapping economic mode in this paper.

Some studies have attempted for designing multi-objective genetic algorithms since Schaffer [19]. There are also much literature about multi-objective genetic algorithm for solving lots of complicated problems. Mansouri [20] proposed a multi-objective genetic algorithm to solve a just-in-time sequencing problem, in which the variation of production rates and the number of setups has been optimized simultaneously. Cochran et al. [21] introduced a multi-objective genetic algorithm with a two-stage for solving parallel machine scheduling problems. Ponnambalam et al. [22] proposed a multi-objective genetic algorithm for solving assembly line balancing problems, where the number of workstations, the line efficiency, and the smoothness index are considered as the performance criteria.

Since economic analysis on value chain of taxi fleet with battery-swapping mode is a multi-objective mode, a multi-objective genetic algorithm is also attempted for solving the mode. Thus, this paper is organized in the following way: Section 2 is about the problem of

the basic notations and formulations; Section 3 contains a multi-objective genetic algorithm for the problem; numerical analysis is carried out in Section 4; and, lastly, the conclusions are drawn in Section 5.

2. Economic Benefits Analysis on Taxi Fleet Demonstration with Battery-Swapping Mode

2.1. Empirical Model of Monthly Operating Benefits of Charging-Swapping Station Group in the Region

Our main contribution is to design empirical model of monthly operating benefits of charging-swapping station group in the region. Therefore, before the model is established, we assume the following:

- (1) in order to meet the normal operation of the taxi fleet, a charging-swapping station group should be constructed in some demonstration area after comprehensive consideration of various conditions, which is operated by n_s charging-swapping stations in a network and ensures the normal operation of the taxi fleet in the region;
- (2) charging-swapping station group is responsible for the purchase of battery pack assembled on taxi vehicles; each vehicle can only be fitted with one battery pack at the same time, which is constituted by a number of subpacks;
- (3) all the other costs are ignored except the expenditure and income listed by the empirical model of operating benefits of charging-swapping station group in the region;
- (4) the cost of charge is charged according to vehicles mileage of the taxi fleet by the charging-swapping station group in the region.

The monthly operating benefits of charging-swapping station group in the region B_0 equals the monthly total income of charging-swapping station group in the region B_{st} minus the monthly total expenditure of charging-swapping station group in the region C_s , that is, $B_o = B_{st} - C_s$.

The monthly total income of charging-swapping station group in the region is calculated as

$$B_{st} = \sum_{i=1}^{n_s} B_{si}, \quad (2.1)$$

in which B_{st} : total income of charging-swapping station group, unit: yuan; B_{si} : monthly income of the i th charging-swapping station in the group, unit: yuan; N_s : the total number of the charging-swapping station in the group; i : the number of the charging-swapping station. Consider

$$B_{si} = \sum_{v=1}^{n_v} C \times R_{vm}, \quad (2.2)$$

C : the charges per kilometer which is agreed by the charging-swapping station group and the owner of the vehicle, unit: yuan/kilometer; R_{vm} : the monthly mileage of a vehicle;

$B_{vm} = \sum_{d=1}^{30} R_{vmd}$, unit: kilometer, R_{vmd} : the daily mileage of a vehicle, n_v : the total number of the vehicles which is served by the charging-swapping station group.

The monthly total expenditure of charging-swapping station group in the region is calculated as

$$C_s = C_d + C_p + C_b + C_e, \quad (2.3)$$

in which C_d : total cost of the initial construction of the charging-swapping station group in the region converted to monthly cost C_{dc} and monthly costs for facilities and equipment maintenance C_{dv} , unit: yuan/month,

$$C_d = C_{dc} + C_{dv}, \quad C_{dc} = \frac{C_{bc}/Y_c}{12}, \quad C_{dv} = C_{bc} \times D_{de}, \quad (2.4)$$

C_{bc} : the total cost of the initial construction of the charging-swapping station group in the region; D_{de} : the depreciation rate of the facilities and equipment of the charging-swapping station group in the region, Y_c : the useful lives or the depreciation period of the facilities and equipment of the charging-swapping station group in the region, C_p : the total monthly salaries of the professional services staffs of the charging-swapping station group, $C_p = C_{pp} \times N_{ps}$, unit: yuan; C_{pp} : the personal average monthly salary of the professional services staffs of the charging-swapping station group, unit: yuan; N_{ps} : the number of the professional services staffs of the charging-swapping station group. C_b : The monthly depreciation cost of the charging-swapping station group. Consider

$$C_b = \frac{(R_{bn} \times n_v \times P_{bp} - n_v \times P_{gb})}{Y_{bp}/12}, \quad (2.5)$$

Y_{bp} : the useful lives or the depreciation period of battery package and R_{bn} : the number of battery package of each vehicle, $R_{bn} = (R_{vm} \times 12 \times Y_v) / ((C_{bpc}/C_{per \text{ km}}) \times N_{cy})$, which is influenced by the charging time of the battery. In order to ensure the normal operation of vehicles, $R_{bn} \geq 3$, Y_v : the useful lives or the depreciation period of vehicles; according to the relevant laws and regulations, Y_v should meet the conditions $Y_v \leq 600000 / (R_{vm} \times 12)$ and $Y_v \leq 8$ at the same time. (According to mandatory retirement standard of motor vehicle, if the taxi's total mileage is more than 600000 km, or it has been used for more than 8 years, then it must be scrapped.) C_{bpc} : available power for driving of each battery package, unit: kWh; $C_{per \text{ km}}$: the average consumption level of driving, unit: kWh/km; N_{cy} : of battery package; unit: time; P_{bp} : the price of each battery package, unit: yuan; P_{gb} : state subsidies standards of each battery package, unit: yuan; C_e : monthly charges for electricity paid to the electric power provider by the charging-swapping station group, P_{el} : actual price of AC, unit: yuan/kWh; η : the charging efficiency of the battery package.

Thus, the objective function of the monthly operating benefits of charging-swapping station group in the region can be described as the following:

$$\text{Max } B_0 = B_{st} - C_s. \quad (2.6)$$

2.2. Empirical Model of Monthly Operating Benefits of Taxi Fleet Company

Our main contribution is to design empirical model of monthly operating benefits of taxi fleet company. Therefore, before the model is established, we assume the following:

- (1) the income of taxi fleet company only come from the fee paid by all vehicles in the fleet;
- (2) every taxi has one driver;
- (3) all the other costs are ignored except the expenditure and income listed in the empirical model of the operating benefits of the taxi fleet company.

Then, the operating benefits of the taxi fleet company equals the difference between the monthly fee paid to the taxi fleet by all vehicles B_{st} and the monthly total operating expenditures of all the vehicles in the company C_{cs} , that is, $B_{co} = B_{cst} - C_{cs}$,

$$C_{cs} = C_{vs} + C_{es} + C_{ps} + C_{ma}, \quad (2.7)$$

in which C_{cs} : monthly total operating expenditures of all the vehicles in the company, unit: yuan; B_{cst} : monthly fee paid to the taxi fleet company by all vehicles, namely, $B_{cst} = C_c \times n_v$, unit: yuan; C_c : monthly fee paid to the taxi fleet company per driver, $C_c = \sum_{d=1}^{30} C_{dds}$, unit: yuan; C_{dds} : daily fee paid to the taxi fleet company per driver, unit: yuan; C_{vs} : total car cost of all vehicles in the fleet bare (excluding battery pack) converted to monthly operational expenses, $C_{vs} = P_v \times n_v / Y_v / 12$, unit: yuan; P_v : initial acquisition cost per vehicle (excluding the battery package), unit: yuan; C_{es} : insurance cost of all the vehicles in the fleet converted to monthly operational expenses, $C_{es} = \sum_{n=1}^{n_v} (C_{en} / 12)$; C_{en} : insurance cost standards per year per vehicle, unit: yuan; C_{ps} : fixed remuneration expenditure for all taxi drivers in the fleet, $C_{ps} = \sum_{n=1}^{n_v} S_{pp}$; S_{pp} : fixed remuneration expenditure per driver per month, unit: yuan; C_{ma} : monthly maintenance fee for all the vehicles in the fleet, $C_{ma} = \sum_{n=1}^{n_v} C_{ma \text{ per}}$, unit: yuan; $C_{ma \text{ per}}$: Monthly maintenance fee per vehicle in the fleet, unit: ypuan.

Thus, the objective function of monthly operating benefits of taxi fleet company can be described as the following:

$$\text{Max } B_{co} = B_{cst} - C_{cs}. \quad (2.8)$$

2.3. Empirical Model of Monthly Operating Benefits of Taxi Fleet Drivers

The monthly average operating benefits of new energy taxi fleet drivers B_{do} , in which the income equals the monthly average earnings per vehicle B_{ds} and the fixed salary paid by taxi fleet company S_{pp} and its expenses include the monthly fee paid to the taxi fleet company C_c and the average monthly charge fee paid to the charging-swapping station group in the region C_{de} , is calculated as

$$B_{do} = B_{ds} + S_{pp} - C_c - C_{de}, \quad (2.9)$$

in which B_{ds} : monthly earnings per vehicle, $B_{ds} = \sum_{d=1}^{30} B_{pt} \times T_{pv}$, unit: yuan; B_{pt} : the charges paid by passengers per time, unit: yuan; T_{pv} : the number of service provided by a vehicle per

day; C_{de} : the monthly charge fee paid to the charging-swapping station group in the region, namely, $C_{de} = C \times R_{vm}$, unit: yuan.

Thus, the objective function of monthly operating benefits of taxi fleet company can be described as the following:

$$\text{Max } B_{do} = B_{ds} + S_{pp} - C_c - C_{de}. \quad (2.10)$$

2.4. Economic Efficiency Model of Taxi Fleet with Battery-Swapping

The region, company, and taxi driver are three different aspects which are directly affected by the new taxi fleet with battery-swapping. To meet the benefits of all the three aspects, the three objective functions should be considered simultaneously. Thus, the economic efficiency model of taxi fleet with battery-swapping is constructed to achieve the following three objectives:

$$\begin{aligned} \text{Max } B_{st} \\ \text{Max } B_{co} \\ \text{Max } B_{do}, \end{aligned} \quad (2.11)$$

3. Applying Multiobjective GA for the Economic Efficiency Model of Taxi Fleet with Battery-Swapping Mode

3.1. Initialization

The initialization of the population is generated randomly according to some certain constraints to ensure the feasibility of each population.

3.2. Fitness

The difference between traditional genetic algorithm and multi-objective genetic algorithm is the number of objective functions. The economic efficiency model of taxi fleet with battery-swapping has three objective functions. For each target i , there is an evolution based on the merits of the objective function value and a feasible solution is attained. Then, based on all objectives, the total fitness of each solution can be also attained. The total fitness can be acquired according to the following:

$$\begin{aligned} f_i(x_j) &= \begin{cases} (N + 1 - y_i(x_j))^2 & y_i(x_j) > 1, \\ kN^2 & y_i(x_j) = 1, \end{cases} \\ f(x_j) &= \sum_{i=1}^n f_i(x_j) \quad j = 1, 2, \dots, n, \end{aligned} \quad (3.1)$$

where N is the sum of all the individual, x_j denotes the j th individual, y_i denotes the number of all the individuals on the objective i . $f_i(x_j)$ is the fitness of x_j based on the objective i .

$\sum_{i=1}^n f_i(x_j)$ is the synthesized fitness of x_j based on all the objectives. k is the random number in $(1, 2)$, which is used to adjust the weight of the fitness.

If there are four chromosomes (g_1, g_2, g_3, g_4) , their objective values based on three objectives are $(1.6, 0, 1.4)$, $(1.8, 0.6, 2.6)$, $(2, 1, 2)$, and $(2.1, 2.5, 3.2)$. To prevent the increase of an individual to reduce the opportunity of other individual, $k = 1.01$. Then, to the objective 1, the fitness is $(1, 4, 9, 16.16)$ and the order is $(g_1 < g_2 < g_3 < g_4)$. To the objective 2, the fitness is $(16.16, 9, 4, 1)$ and the order is $(g_1 > g_2 > g_3 > g_4)$. To the objective 3, the order is $(g_1 > g_2 > g_3 > g_4)$. To the objective 4, the order is $(g_1 > g_3 > g_4 > g_2)$. According to (3.1), g_1 has the maximum fitness value.

3.3. Crossover Operation

Crossover is a reproduction operation in GA, which is exchanging genetic information between two parent chromosomes to produce two new children. In the paper, an arithmetic crossover [23] is used to create new offsprings.

$$\begin{aligned} g_{k,I}^t &= \alpha_k g_{k,I}^{t-1} + (1 - \alpha_k) g_{k,II}^{t-1} \\ g_{k,II}^t &= \alpha_k g_{k,II}^{t-1} + (1 - \alpha_k) g_{k,I}^{t-1}, \end{aligned} \quad (3.2)$$

where $g_{k,I}^{t-1}, g_{k,II}^{t-1}$ is a pair of "parent" chromosomes; $g_{k,I}^t, g_{k,II}^t$ is a pair of "children" chromosomes; α_k is a random number between $(0, 1)$; $k \in [1, 2, 3]$ (k is the total genes for the crossover operation).

3.4. Mutation Operation

Mutation is also a reproduction operation in GA, which is applied with a mutation rate to avoid being trapped in local optimal during evolution. Assume that a chromosome is $G = (g_1^t, g_2^t, g_3^t)$, if the g_2^t was selected for the mutation, the mutation can be shown in (3.3):

$$\begin{aligned} G' &= (g_1^{t-1}, g_2^t, g_3^{t-1}) \\ g_2^t &= \begin{cases} g_2^{t-1} + \Delta(t, g_{2\max}^t - g_2^{t-1}) & \text{if random } (0, 1) = 0 \\ g_2^{t-1} + \Delta(t, g_2^{t-1} - g_{2\min}^t) & \text{if random } (0, 1) = 1. \end{cases} \end{aligned} \quad (3.3)$$

The function $\Delta(t, y)$ retains a value between $[0, y]$ given in the following:

$$\Delta(t, y) = y \times \left(1 - r^{(1-t/T_{\max})^\lambda}\right), \quad (3.4)$$

where r is a random number between $[0, 1]$; T_{\max} is maximum number of generations; here $\lambda = 3$. This property causes this operation to make a uniform search in the initial space when t is small and a very local one in later stages.

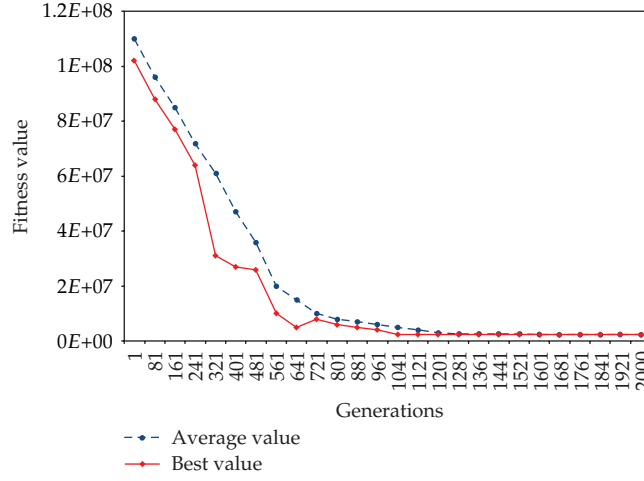


Figure 1: The computing process of GA.

3.5. Termination

In this paper, the search continues until $RMSE_n - RMSE_{n-1} < 0.0001$ or the number of generation reaches the maximum number of generations T_{max} .

4. Numerical Test

4.1. Model Calibration

GA is used to optimize the multi-objective mode of this paper. Before the implementation of GA, there are four parameters for GA, namely, p_c , p_m , p_{size} , and T_{max} , that need to be predetermined. Generally, p_c varies from 0.3 to 0.9. p_m varies from 0.01 to 0.1, p_{size} is the population size which is set according to the size of the samples. T_{max} is the maximum number of generation which can be determined according to a good convergence of the calculation [24]. According the characteristic of this problem, the values of the four parameters (p_c , p_m , p_{size} and T_{max}) are set as 0.65, 0.1, 80, and 2000, respectively.

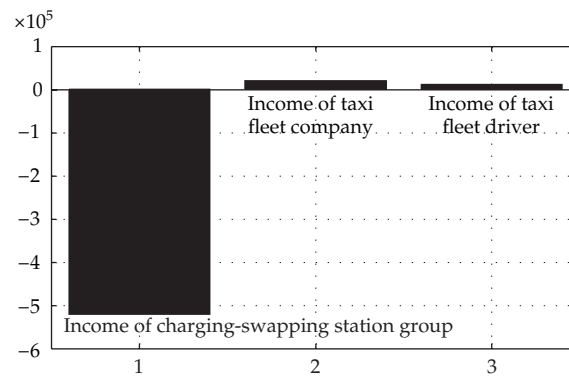
The actual value of the problem in the model is collected in Table 1 based on the model operational data of the battery-swapping mode fleet and infrastructure collected from the pilot cities. Take 200 taxis in a region as an example, the GA reached convergence on the 1361 generation (Figure 1), and the results are shown as the following.

The profit of the taxi fleet company, operators of the infrastructure, and drivers of the taxi fleet company is shown in Figure 2. Through analysis the following conclusion is made: the early-stage construction cost of the charging-swapping stations distributed to individual vehicles is excessively high due to the relatively large investment in local charging-swapping stations and small number of vehicles served by the charging-swapping stations. Consequently, the groups involved in the value chain of fleet demonstration of taxi fleet are suffering serious financial loss, especially for the charging-swapping electricity stations.

Table 1: The information of the instances.

Parameter	Value
n_v	30
C	0.75
R_{vmd}	428
C_{bc}	14,400,000
D_{de}	0.03
Y_c	10
N_{pa}	21
C_{pp}	3000
Y_{bp}	5
Y_v	3.8
C_{bpc}	12.5
$C_{per\ km}$	0.157
N_{cy}	1500
P_{bp}	75000
P_{gb}	60000
P_{el}	0.538
η	0.8
C_{dds}	280
P_v	150000
C_{en}	10000
S_{pp}	2800
C_{maper}	800
B_{pt}	16
T_{pv}	55

Note: T_{pv} Approximately increases with R_{vmd} in proportion.

**Figure 2:** The profit of groups involved in the value chain of fleet demonstration of new energy taxi.

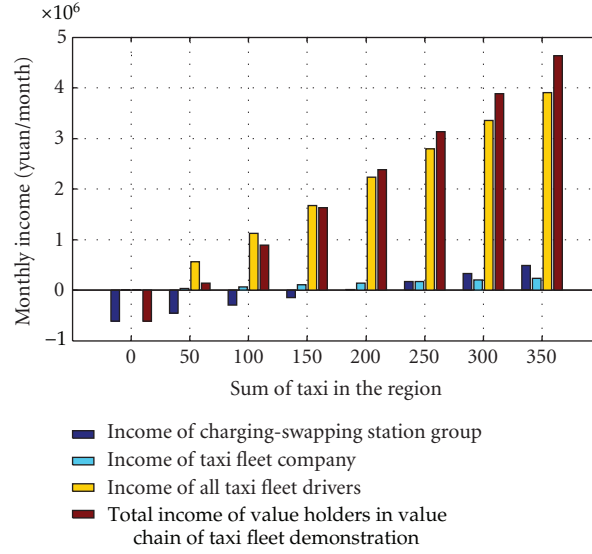


Figure 3: Influence of the total number of vehicles on the profit situation.

4.2. Analysis on the Factors Influencing the Operation Profit

4.2.1. The Operational Load Rate of the Local Charging/Swapping Stations

The service frequency is improved with the operating efficiency of charging-swapping station, which leads to the increase in the total number of taxis sharing the service. Figure 3 illustrates the profit variation of the value holders in the taxi fleet operation value chain in the case of a rising operation load (i.e., the total number of the taxi fleet vehicles that enjoy the service) of the local charging/swapping stations. Analysis manifests that the profit of taxi fleet company and the local charging/swapping stations grows as the operation load rises and the total profit increment of taxi drivers is proportional to that of the number of taxis. The total gain of the value holders in the value chain is positive when the scale of the fleet is close to 50 vehicles.

4.2.2. Early-Stage Construction Cost of the Local Charging/Swapping Stations

Refer to Figure 4 for the influence of early-stage construction cost of the local charging-swapping electricity stations on the total profit of value holders involved in the value chain of new energy taxi fleet demonstration. It is clear that the profit of the local charging-swapping electricity stations diminishes drastically as the early-stage investment cost increases; the total profit of value holders involved in the value chain appear negative when the investment scale exceeds certain extent.

4.2.3. Lifespan of Battery Packages

The curve in Figure 5 illustrates the individual and total profit of value holders involved in the value chain of fleet demonstration. It can be seen that lengthened lifespan of battery

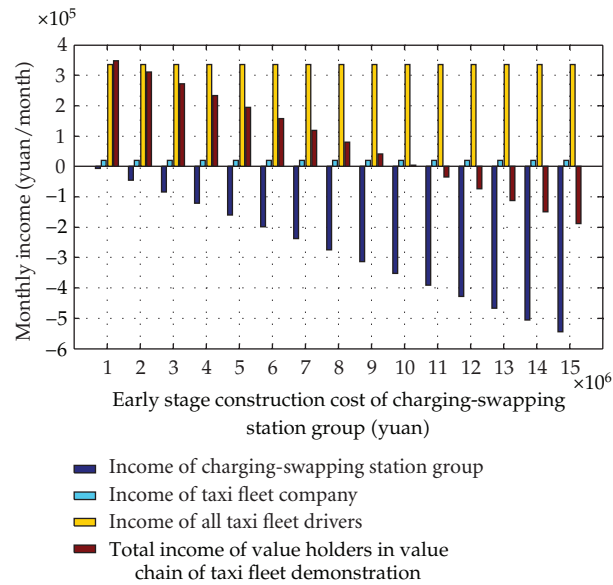


Figure 4: Influence of early stage investment cost of local charging-swapping electricity stations on profit.

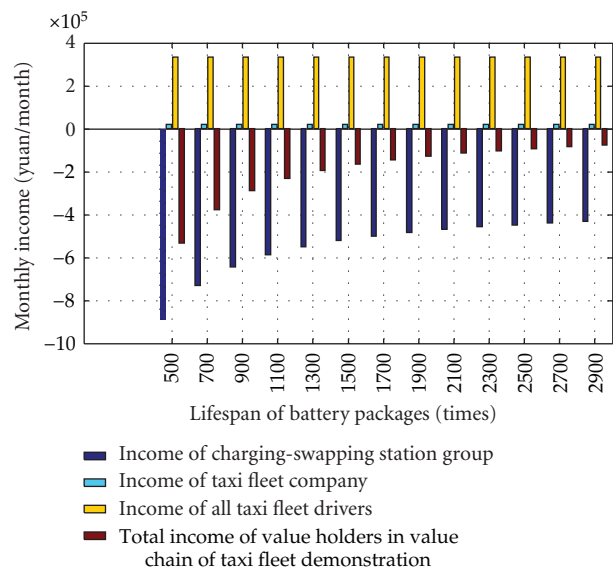


Figure 5: Influence of lifespan of battery packages on profit.

package leads to increment in the overall effectiveness of the charging/swapping stations in the value chain of new energy taxi fleet demonstration and thus the total profit of all the value holders.

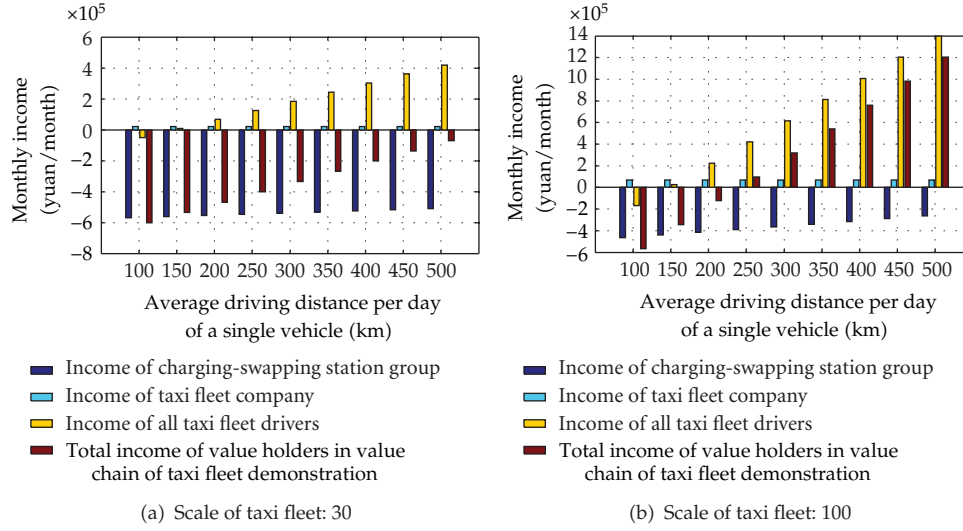


Figure 6: The variation of profit of the value holders involved in the value chain of fleet demonstration with the average driving distance per day of a single vehicle.

4.2.4. The Average Driving Distance per Day of a Single Vehicle

To acquire the effect of the average driving distance per day of a single vehicle, the variation of profit of the value holders involved in the value chain of fleet demonstration with the average driving distance per day of a single vehicle can be shown in Figure 6.

It can be found in Figure 6 that the influence of average driving distance per day of a single vehicle on the profit of the value holders involved in the value chain of fleet demonstration. Analysis suggests that when the fleet is of a small-scale, the profit of taxi fleet drivers grows significantly as the daily driving distance increases whilst the earnings of the local charging/swapping stations remain negative but the amount of losses decrease gradually. Comparing Figures 6(a) and 6(b) yields the following: the total profit of the value holders involved in the value chain of fleet demonstration increases with average driving distance per day of a single vehicle; as the scale of the fleet is enlarged, it is possible for the value holders to make profit at an earlier stage and the total profit relies less on the daily driving distance.

4.2.5. Price of Battery Packages

The influence of price of battery package on profit can be attained in Figure 7. It can be seen that Figure 7 illustrates the influence of the price of battery package on the profit of the value holders involved in the value chain of fleet demonstration. Analysis indicates that increment in the price of battery leads to diminished profit of the local charging/swapping stations and the total profit of the value holders involved in the model value chain.

4.2.6. Electricity Price

Figure 8 shows the influence of electricity price on the value holders in the value chain of fleet demonstration. Analysis indicates that the total profit of the local charging/swapping

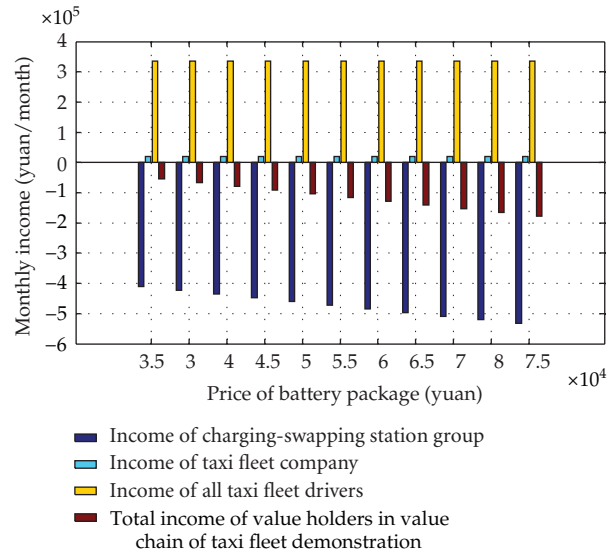


Figure 7: Influence of price of battery package on profit.

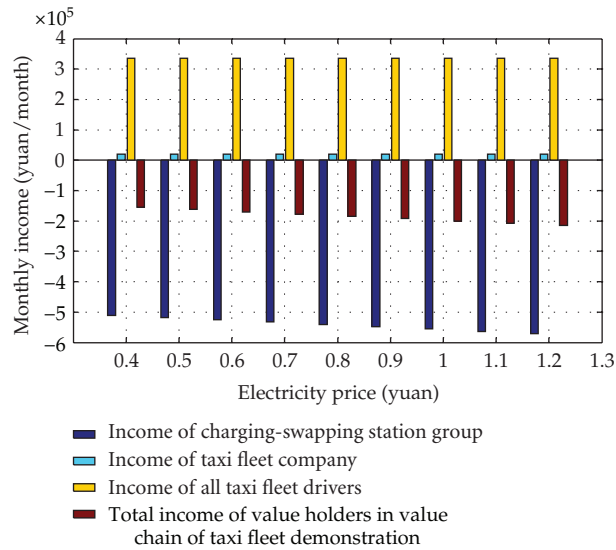


Figure 8: Influence of electricity price on profit.

stations as well as the total profit of the value holders in the model operation value is reduced as the electricity price given by the electricity supplier rises.

5. Conclusions

The verified demonstration operation economic benefit models of changing/swapping taxi fleet are solved by a multi-objective genetic algorithm. The influence of various factors on the individual profit and total profit of the value holders involved in the value chain of fleet

demonstration of taxi fleet, such as electricity price, lifespan and price of battery package, construction cost of infrastructure, and driving mileage per day are summarized as the following.

The total profit of the value holders involved in the value chain of taxi fleet demonstration based on battery-swapping mode decreases as the electricity price rises. As the price of battery packages at the early stage rises, the profit of the local battery charging-swapping electricity groups decreases and thus the total profit of the value holders. With the advances in battery technologies and lengthened lifespan of battery packages, the benefits of the local charging/swapping stations increase significantly, which results in the enlarged total profit of value holders in value chain of the taxi fleet demonstration.

In the case that the taxi fleet is relatively small, as the driving mileage per day (usage rate) increases, the profit of taxi fleet drivers grows whilst the local charging/swapping stations remain losing money but with decreasing financial loss. The total profit of the value holders in the value chain of fleet demonstration grows with driving mileage of individual vehicles per day; as the scale of fleet is enlarged, the total profit of the value holders in the value chain of fleet demonstration is able to make profit at an earlier stage with total profit relying less on driving mileage per day. As operation load rate of the local charging-swapping electricity stations rises, the profit of the new taxi fleet company and suppliers of infrastructure grows and the profit of taxi fleet drivers increases proportionally with the number of taxis.

All value holders are able to make profit when the fleet scale is sufficiently large. The total profit of value holders in value chain of the new energy taxi fleet demonstration can be promoted effectively by adapting low-cost charging-swapping infrastructure, lowering battery price, lengthening the lifespan of battery, and raising the operation efficiency of charging-swapping stations.

References

- [1] K. Chen, Z. P. Wang, and C. Lin, "Development and demonstration of pure electric vehicle in Beijing," *New Materials Industry*, no. 9, pp. 27–30, 2006.
- [2] X. Fu and B. X. Hu, "Wang Y.N. Current Situation of demonstration operations and policy research of the electric vehicle in Wuhan," *Shanghai Automobile*, no. 6, pp. 9–12, 2007.
- [3] F. X. Dai, "Research of demonstration operation's development strategy facing the industrialization of electric vehicle," *Technology Development and Policy*, vol. 26, no. 16, pp. 71–73, 2009.
- [4] F. Q. Zhao and Y. Chen, "Strategy of geographic expansion of electric vehicle demonstration operations based on scale economy," *Modern Management Science*, no. 10, pp. 82–83, 2008.
- [5] Q. Deng, L. M. Zhu, and Y. N. Tang, "Analysis on demonstration operations of parallel hybrid city bus XD6120HEV of XEMC," *Urban Vehicle*, no. 2, pp. 55–58, 2008.
- [6] J. D. Hamilton, "Many modeling approaches have been applied to evaluate the economic analysis," *Econometrica*, vol. 57, no. 2, pp. 357–384, 1989.
- [7] T. S. Aidt, "Economic analysis of corruption: a survey," *Economic Journal*, vol. 113, no. 491, pp. F632–F652, 2003.
- [8] A. Haghani, M. Banishashemi, and K. H. Chiang, "A comparative analysis of bus transit vehicle scheduling models," *Transportation Research B*, vol. 37, no. 4, pp. 301–322, 2003.
- [9] B. Yu and Z. Z. Yang, "An ant colony optimization model: the period vehicle routing problem with time windows," *Transportation Research E*, vol. 47, no. 2, pp. 166–181, 2011.
- [10] B. Yu, Z. Z. Yang, and B. Yao, "An improved ant colony optimization for vehicle routing problem," *European Journal of Operational Research*, vol. 196, no. 1, pp. 171–176, 2009.
- [11] B. Yu, Z.-Z. Yang, and S. Li, "Real-time partway deadheading strategy based on transit service reliability assessment," *Transportation Research A*, vol. 46, no. 8, pp. 1265–1279, 2012.
- [12] J. H. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor, Mich, USA, 1975.

- [13] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.
- [14] L. D. Chambers, *Practical Handbook of Genetic Algorithms: New Frontiers*, CRC Press, Boca Raton Fla USA, 1995.
- [15] A. Wren and D. O. Wren, "A genetic algorithm for public transport driver scheduling," *Computers and Operations Research*, vol. 22, no. 1, pp. 101–110, 1995.
- [16] P. Chakroborty, K. Deb, and P. S. Subrahmanyam, "Optimal scheduling of urban transit systems using genetic algorithms," *Journal of Transportation Engineering*, vol. 121, no. 6, pp. 544–553, 1995.
- [17] P. Chakroborty, "Genetic algorithms for optimal urban transit network design," *Computer-Aided Civil and Infrastructure Engineering*, vol. 18, no. 3, pp. 184–200, 2003.
- [18] F. Altıparmak, M. Gen, L. Lin, and T. Paksoy, "A genetic algorithm approach for multi-objective optimization of supply chain networks," *Computers and Industrial Engineering*, vol. 51, no. 1, pp. 196–215, 2006.
- [19] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *Proceedings of the 1st International Conference on Genetic Algorithms (ICGA '85)*, pp. 93–100, 1985.
- [20] S. A. Mansouri, "A multi-objective genetic algorithm for mixed-model sequencing on JIT assembly lines," *European Journal of Operational Research*, vol. 167, no. 3, pp. 696–716, 2005.
- [21] J. K. Cochran, S.-M. Horng, and J. W. Fowler, "A multi-population genetic algorithm to solve multi-objective scheduling problems for parallel machines," *Computers & Operations Research*, vol. 30, no. 7, pp. 1087–1102, 2003.
- [22] S. G. Ponnambalam, P. Aravindan, and G. Mogileeswar Naidu, "Multi-objective genetic algorithm for solving assembly line balancing problem," *International Journal of Advanced Manufacturing Technology*, vol. 16, no. 5, pp. 341–352, 2000.
- [23] B. Yu, Z. Yang, and C. Cheng, "Optimizing the distribution of shopping centers with parallel genetic algorithm," *Engineering Applications of Artificial Intelligence*, vol. 20, no. 2, pp. 215–223, 2007.
- [24] B. Yu, Z. Z. Yang, P. H. Jin, S. H. Wu, and B. Z. Yao, "Transit route network design-maximizing direct and transfer demand density," *Transportation Research C*, vol. 22, pp. 58–75, 2012.

Review Article

Bus Dispatching Interval Optimization Based on Adaptive Bacteria Foraging Algorithm

Zhong-hua Wei, Xia Zhao, Ke-wen Wang, and Yan Xiong

Beijing Key Laboratory of Traffic Engineering, Beijing University of Technology, Beijing 100124, China

Correspondence should be addressed to Zhong-hua Wei, weizhonghua@bjut.edu.cn

Received 4 September 2012; Accepted 14 November 2012

Academic Editor: Rui Mu

Copyright © 2012 Zhong-hua Wei 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 improved bacterial foraging algorithm was applied in this paper to schedule the bus departing interval. Optimal interval can decrease the total operation cost and passengers' mean waiting time. The principles of colony sensing, chemotactic action, and improved foraging strategy made this algorithm adaptive. Based on adaptive bacteria foraging algorithm (ABFA), a model on one bus line in Hohhot city in China was established and simulated. Two other algorithms, original bacteria foraging algorithm (BFA) and genetic algorithm (GA), were also used in this model to decide which one could greatly accelerate convergence speed, improve searching precision, and strengthen robustness. The final result showed that ABFA was most feasible in optimizing variables.

1. Introduction

Traffic demand becomes increasingly higher with the great development of social economy and urbanization. With the large amount of private cars and limited road facilities, severe traffic congestion occurs inevitably. Municipal governments and road transport authority have strongly recommended public transportation for its higher passenger capacity and smaller coverage area. So it is necessary to improve the quality of bus dispatching management, and a wholesome and intelligent bus scheduling scheme is needed.

Bus interval scheduling is a complex optimization problem for its nonlinear and multiobjective characteristics. It requires traffic planners to take round consideration of multi-interests, such as bus operating enterprises and passengers. The relationship between the two is contradictory. Obviously, bus operating enterprises always try to dispatch buses as few as possible with the longest intervals and the highest profit, whereas passengers are the opposite. According to the two contradictory characteristics, an optimal bus dispatching interval model is urgent to be established to benefit both sides.

As to the optimal algorithms, lots of intelligent algorithms have been adopted to solve transportation optimal problems [1]. Intelligent algorithms enlightened by bacterium have become fashionable recently. Back to the history of bacteria foraging algorithm (BFA), it was initially proposed in 2002 by Passino [2] and applied to dealing with several engineering problems [3–5] successfully. However, it was limited to solo modal function optimization for its poor convergence behavior. So effort to pursue more adaptive algorithms goes on. Muñoz et al. [6] proposed some methods to simplify the algorithm while maintaining its core elements. These included the simplification of the algorithm architecture, the elimination of the bacteria colony scale, a clear adaptation rule for the step size, the use of a uniform distribution the position initialization, and the removal of the cell-to-cell communication. In 2008, Dasgupta et al. [7] mathematically analyzed the chemotactic step of a one-dimensional BFA and proposed the adaptive step size for BFA. In the same year, Chen et al. [8] analyzed how the run length unit parameter controlled the exploration and exploitation process of BFA and then employed the adaptive search strategy to significantly improve the performance of the original algorithm.

In this paper, the adaptive bacteria foraging algorithm was adopted to solve the complex bus interval dispatching problem. Many variables were taken into consideration such as dispatching interval, waiting time, load factor, economic efficiency, and trip service level. Among them, dispatching interval was chosen as the control variable. To compare which one was the best, genetic algorithm [9] (GA) and nonadaptive bacteria foraging algorithm (BFA) were employed in optimizing this model. The simulation results, focusing on minimizing two specific objective functions, depicted the advantages of the mentioned algorithm.

The rest of the paper is organized as follows. Section 2 gives brief views of basic and adaptive bacteria foraging algorithm. The theoretical optimal model for bus dispatching interval is built in Section 3. Based on this model, the simulations on ABFA, GA, and BFA are given in Section 4. Finally, Section 5 gives the conclusions.

2. Adaptive Bacteria Foraging Algorithm

2.1. Basic Bacteria Foraging Algorithm

To forage food, a bacterium needs to communicate with the group with its sensing abilities. By generating common knowledge, developing group identity and recognizing the identity of other colonies, bacterium engages in group decision-making [10]. A sort of collective intelligence subsequently occurs. And this wit consists of four steps: chemotaxis, swarming, reproduction, and elimination and dispersal [11–13].

(1) Chemotaxis

Basically, chemotaxis is a foraging strategy that implements a type of local optimization where the bacteria try to climb up the nutrient concentration [7]. Swimming and tumbling are two aspects of this step. The bacterium's flagella rotation determines its foraging direction in an anticipated route (swimming) or an unexpected one (tumbling). $\varphi(j)$ stands for the unit random length in the direction of a tumble, that is,

$$\theta^i(j+1, k, l) = \theta^i(j, k, l) + C(i)\varphi(j), \quad (2.1)$$

where $\theta^i(j, k, l)$ means the i th bacteria's j th chemotactic behavior at the k th reproductive and l th elimination step. $C(i)$ represents the step size adopted in the random route in a tumble. And C is labeled as run length unit.

(2) *Swarming*

It is assumed that bacterium tends to inform others of its optimal food path, so group can locate food area more swiftly. Swarming gathers bacterium into high-density group and then forage food in a type of concentric circle, that is,

$$J_{cc} = \sum_{i=1}^S J_{cc}^i(\theta, \theta^i(j, k, l)) = \sum_{i=1}^S \left[-d_{\text{attract}} \exp \left(-w_{\text{attract}} \sum_{m=1}^P (\theta_m - \theta_m^i)^2 \right) \right] + \sum_{i=1}^S \left[h_{\text{repellent}} \exp \left(-w_{\text{repellent}} \sum_{m=1}^P (\theta_m - \theta_m^i)^2 \right) \right], \quad (2.2)$$

where $J_{cc}^i(\theta, \theta^i(j, k, l))$ is the cost function value to be minimized; S is the bacterium scale; P is the number of parameters to be optimized. d_{attract} , w_{attract} , $h_{\text{repellent}}$, and $w_{\text{repellent}}$ are different coefficients that are to be chosen carefully.

(3) *Reproduction*

The least healthy bacterium vanishes, and the other healthiest bacterium splits into two and are released in the same position. This makes the population of bacteria constant.

(4) *Elimination and Dispersal*

Possibly in the current surroundings, bacterium's life span changes slightly either by nutrient supplements or other unexpected influence. Accidents can kill some bacteria colonies in one second. But this has the effect of assisting in chemotaxis.

2.2. *Adaptive Bacterial Foraging Algorithm*

BFA is adaptive by adjusting the run length step parameter dynamically during its execution process to balance the exploration or exploitation search [14]. Each bacterium has two different foraging states in ABFA model.

- (1) Exploration: imprecisely explore the regions never gone before quickly in the search space in a large run length unit.
- (2) Exploitation: exploit the potential regions slowly in its immediate vicinity in a small run length unit.

The self-adaptive search is given in pseudocode in Table 1 below.

2.3. *Algorithm Flowchart of ABFA*

The flowchart of ABFA is listed below.

Table 1: Pseudocode for dynamic self-adaptive strategy [17, 18].

```

(1) FOR (each bacterium  $i$ ) IN PARALLEL
(2) IF (Criterion-1) then
(3)  $C^i(t+1) = C^i(t) / \alpha;$       // exploitation
(4)  $\varepsilon^i(t+1) = \varepsilon^i / \beta;$ 
(5) ELSE IF (Criterion-2) then
(6)  $C^i(t+1) = C_{\text{initial}};$       // exploration
(7)  $\varepsilon^i(t+1) = \varepsilon_{\text{initial}};$ 
(8) ELSE
(9)  $C^i(t+1) = C^i(t);$ 
(10)  $\varepsilon^i(t+1) = \varepsilon^i;$ 
(11) END IF
(12) END FOR IN PARALLEL

```

Where Criterion-2: exploration state; Criterion-1: exploitation state;
 t : the current iterations; α , β : default constants; $C^i(t)$: current run-length unit of the i th bacteria; $\varepsilon^i(t)$: the desired precision in the current iteration of the i th bacteria; C_{initial} : the original run length unit; $\varepsilon_{\text{initial}}$: the original precision.

Step 1. Initialize parameters like bacterium scale S , custom constants n , α , and β , run length step C_{initial} and precision $\varepsilon_{\text{initial}}$, and position of bacterium colony.

Step 2 (chemotaxis and swarming). In this process, the signaling concentration (fitness value) released on every path will be calculated. Bacterium will choose paths that own the highest fitness value.

Step 3 (reproduction). Sort by the bacteria colony's fitness values. The best half of the population undergoes reproduction. And the rest are eliminated to accelerate convergence speed.

Step 4 (elimination and dispersal). The wholesome adaptive bacteria colony migrate to other spaces randomly to expand the diversity of the colony.

Step 5 (judgment). If the current iteration is lower than the max iterations, then go to Step 2; if not, the algorithm loop ends.

The algorithm flowchart is given below in Figure 1, where, S —the colony scale; t —the current iterations; N_s —the max iterations in the current nutrient gradient; X_i —the current location in searching space of the i th bacteria; flag^i —the number of times that bacteria's fitness value has no continuous improvement.

3. Theoretical Optimal Model Building

Bus scheduling is a complex optimal problem influenced by varied external environment. According to the specific and available data on bus dispatching, the following hypothesis was made to establish a simulated model [15].

- (1) The type of buses was identical. They were well operated and dispatched strictly according to bus schedule, with no accidents on the road.

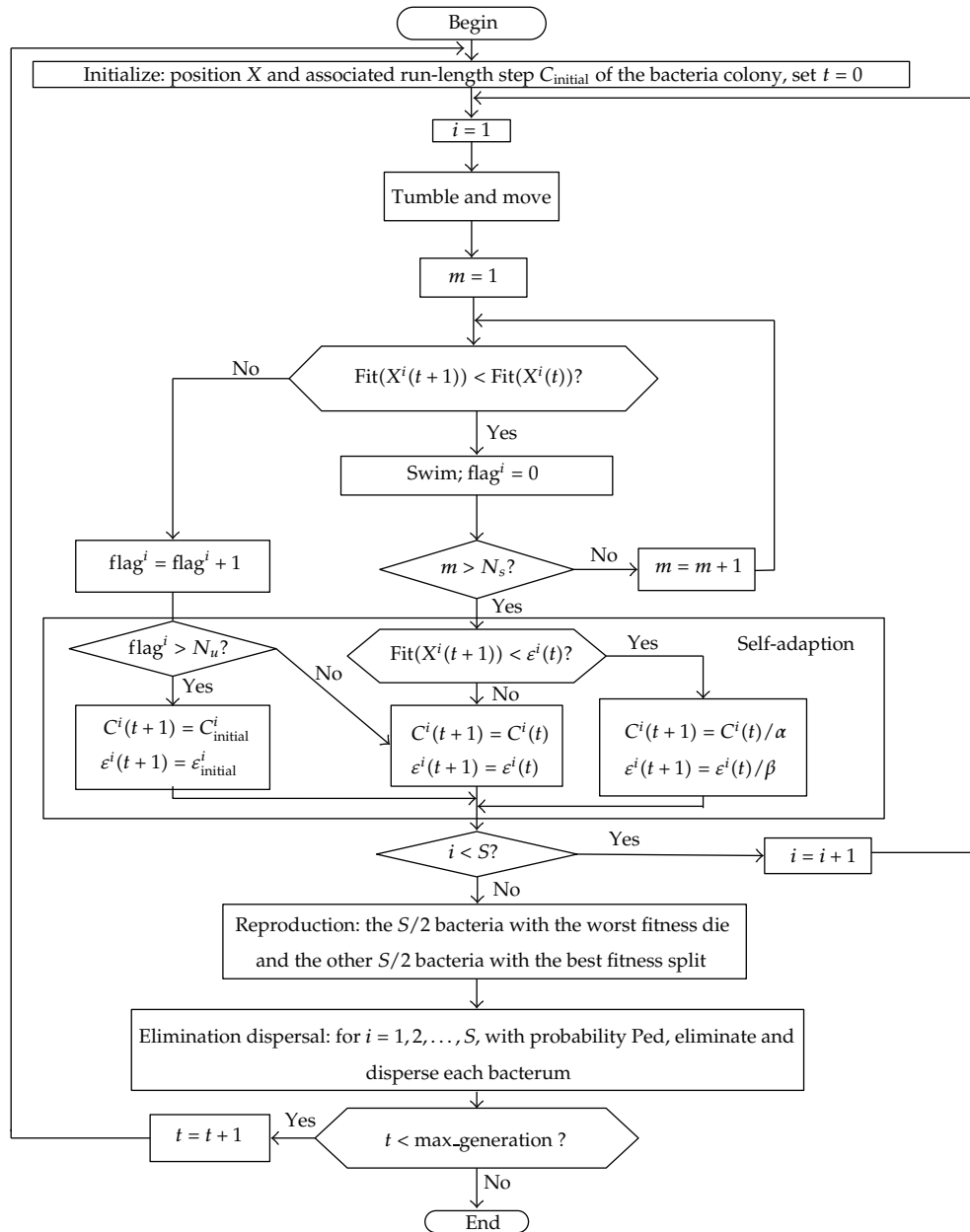


Figure 1: Flowchart of ABFA [15].

- (2) Passengers' traffic flow volume at a bus line was independent of each other; their arrival distribution followed a uniform distribution.
- (3) The dispatching interval of every two successive buses was identical in a given period.
- (4) The trip cost unit for passengers and operation cost unit for bus operating enterprises were fixed.

A certain bus line with its upstream travel direction was selected as the scheduling object. The whole operating time was divided into several one-hour intervals. And the theoretical optimization model was given below.

(1) Control Variable

Bus dispatching interval was chosen as control variable.

(2) Objective Function

Minimum bus operating cost and minimum passengers' waiting cost were two optimal objectives. In formula (3.1), f_1 represented the operation cost bus enterprises invested one day; in (3.2), f_2 represented the trip cost passengers invested one day; in (3.3), f represented the final minimum function with weighting coefficients given:

$$f_1 = \frac{\sum_{k=1}^K (t_k / \Delta t_k)}{t_s / ((t_{\max} + t_{\min}) / 2)}, \quad (3.1)$$

$$f_2 = \frac{\sum_{k=1}^K \sum_{j=1}^J m_j \times \rho_{kj} \Delta t_k^2 / \sum_{k=1}^K \sum_{j=1}^J U_{kj}}{(t_{\max} + t_{\min}) / 2}, \quad (3.2)$$

$$f = \min(\eta \times f_1 + \tau \times f_2), \quad (3.3)$$

where t_k is time duration at k th interval; $k = \max/\min$ means the upper or lower bound of bus interval; $k = s$ means the whole operating time span one day; Δt_k is the k th dispatching interval; m_j is passenger volume at the j th site; ρ_{kj} , u_{kj} is the density of passenger flow or passenger volume arriving at the j th site in the k th interval; U_{kj} is the getting off at the j th site in the k th interval; η , τ is weighting coefficient.

(3) Standard Constraint Condition

Formula (3.4) represented the mean bus load factor to be more than 75%; formula (3.5) represented dispatching interval to be between the lower and upper bounds; (3.6) represented condition to ensure profitability for bus enterprises.

Standard constraint condition

$$\frac{\sum_{k=1}^K \sum_{j=1}^J U_{kj}}{Q \times \sum_{k=1}^K (t_k / \Delta t_k)} \geq 75\%, \quad (3.4)$$

$$t_{\min} \leq \Delta t_k \leq t_{\max}, \quad (3.5)$$

$$\frac{\sum_{k=1}^K \sum_{j=1}^J U_{kj}}{\sum_{k=1}^K (t_k / \Delta t_k)} \geq 2.5 \times L, \quad (3.6)$$

where Q is the rated passenger capacity of a bus; K is time set $K = \{1, \dots, k, \dots, K\}$, K is the total time intervals; J is site set $J = \{1, \dots, j, \dots, J\}$, J is the amount of bus sites of a line; L is the length of the total bus line.

Table 2: Passenger volume in each site.

Time period	Passengers get in	Passengers get off	Time period	Passengers get in	Passengers get off
06:01–07:00	98	52	13:01–14:00	69	78
07:01–08:00	74	105	14:01–15:00	85	90
08:01–09:00	83	97	15:01–16:00	49	45
09:01–10:00	54	52	16:01–17:00	63	81
10:01–11:00	85	69	17:01–18:00	78	93
11:01–12:00	52	44	18:01–19:00	89	96
12:01–13:00	48	45	19:01–20:00	79	59

Table 3: Adjacent site's distance and average running time between sites.

Site no.	Adjacent site's distance (m)	Running time (min)	Site no.	Adjacent site's distance (m)	Running time (min)
1	0	0	18	397	1.87
2	406	2.375	19	562	3.6
3	308	1.28	20	491	2.85
4	562	3.7	21	374	1.73
5	461	2.75	22	259	1.25
6	364	1.54	23	376	1.65
7	347	1.37	24	549	3.2
8	570	3.3	25	523	3.0
9	440	2.5	26	1100	6.7
10	389	2.1	27	563	3.25
11	607	3.7	28	395	1.95
12	709	4	29	397	1.94
13	533	3.1	30	426	2.75
14	604	3.4	31	407	2.25
15	552	3.5	32	714	5.1
16	385	1.75	33	199	0.8
17	546	3.2	34	424	2.5

4. Practical Simulation Based on Optimal Model

A bus line in its upstream travel direction in Hohhot city in China was selected as the scheduling object. Several relevant data had been investigated and listed as follows: the number of total sites was $J = 34$ sites; the operation period was 6:00–20:00 and divided into $K = 14$ hours; the lower and upper bounds of bus dispatching interval were $t_{\min} = 2$ min and $t_{\max} = 15$ min, with the rated passenger capacity $Q = 100$ persons.

4.1. Passenger Traffic Flow and Basic Data on Bus Site Facilities

Data on passenger volume in each site of the upstream line were shown in Table 2 and adjacent sites distance and average running time between them were displayed in Table 3.

Table 4: Bus dispatching interval schedule based on ABFA.

Interval	Bus departure schedule									Departure times
1	6:00	:04	:08	:12	:16	:20	:24	:28	:32	15
	:36	:40	:44	:48	:52	:56				
2	7:00	:03	:06	:09	:12	:15	:18	:21	:24	20
	:27	:30	:33	:36	:39	:42	:45	:48	:51	
	:54	:57								
3	8:00	:04	:08	:12	:16	:20	:24	:28	:32	15
	:36	:40	:44	:48	:52	:56				
4	9:00	:05	:10	:15	:20	:25	:30	:35	:40	12
	:45	:50	:55							
5	10:00	:05	:10	:15	:20	:25	:30	:35	:40	12
	:45	:50	:55							
6	11:00	:05	:10	:15	:20	:25	:30	:35	:40	12
	:45	:50	:55							
7	12:00	:05	:10	:15	:20	:25	:30	:35	:40	12
	:45	:50	:55							
8	13:00	:06	:12	:18	:24	:30	:36	:42	:48	10
	:54									
9	14:00	:06	:12	:18	:24	:30	:36	:42	:48	10
	:54									
10	15:00	:05	:10	:15	:20	:25	:30	:35	:40	12
	:45	:50	:55							
11	16:00	:04	:08	:12	:16	:20	:24	:28	:32	15
	:36	:40	:44	:48	:52	:56				
12	17:00	:03	:06	:09	:12	:15	:18	:21	:24	20
	:27	:30	:33	:36	:39	:42	:45	:48	:51	
	:54	:57								
13	18:00	:04	:08	:12	:16	:20	:24	:28	:32	15
	:36	:40	:44	:48	:52	:56				
14	19:00	:06	:12	:18	:24	:30	:36	:42	:48	10
	:54									
Total bus departure times										190

4.2. Simulation Results and Analysis

The bus dispatching interval optimal model was built and simulated on ABFA comparison with GA and BFA. First, basic settings were initialized: the population scale was 40. The maximum iterations was 1000. $\alpha = \beta = 10$. $C_{\text{initial}} = 0.1$. $\varepsilon_{\text{initial}} = 100$. The parameters of BFA and GA were similarly set. Three graphs showing convergence trend were given in Figure 2.

It could be drawn from Figure 2 that ABFA was the best one to accelerate convergence speed, improve searching precision, and strengthen robustness. So with the desired algorithm, latest bus interval was calculated in Table 4. Each bus departure interval was

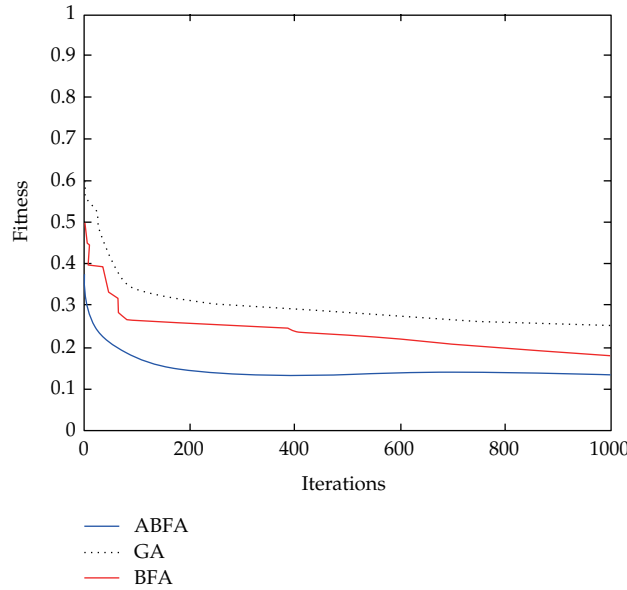


Figure 2: The convergence trend based on 3 algorithms [16].

$\Delta t_1 = 4; \Delta t_2 = 3; \Delta t_3 = 4; \Delta t_4 = 5; \Delta t_5 = 5; \Delta t_6 = 5; \Delta t_7 = 5; \Delta t_8 = 6; \Delta t_9 = 6; \Delta t_{10} = 5;$
 $\Delta t_{11} = 4; \Delta t_{12} = 3; \Delta t_{13} = 4; \Delta t_{14} = 6.$

5. Conclusions

In this paper, an interval optimal model was applied into one bus line in Hohhot city in China. The bus interval dispatching issue was considered as a nonlinear and multiobjective optimization. The enterprise profits, load factor, and dispatching interval were particularly chosen as the primary parameters in this optimization. Two relevant objective functions were defined. The proposed optimal algorithm, ABFA, combined colony sensing, chemotactic action, and improved foraging strategy to solve the distributed bacterial optimization. For comparison purpose, two other algorithms, GA and BFA, were employed to decide which was the best one. Furthermore, extensive sight would be thrown years ahead to evaluate the practical merits of ABFA in traffic and transportation optimization problems.

Acknowledgment

This work is supported by funded projects: National Key Basic Research Program of China (SN: 2012CB723303).

References

- [1] I. A. Farhat and M. E. El-Hawary, "Dynamic adaptive bacterial foraging algorithm for optimum economic dispatch with valve-point effects and wind power," *IET Generation, Transmission and Distribution*, vol. 4, no. 9, pp. 989–999, 2010.
- [2] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control," *IEEE Control Systems Magazine*, vol. 22, no. 3, pp. 52–67, 2002.

- [3] E. Daryabeigi, M. Moazzami, A. Khodabakhshian, and M. H. Mazidi, "A new power system stabilizer design by using smart bacteria foraging algorithm," *Canadian Conference on Electrical and Computer Engineering*, Article ID 6030547, pp. 713–716, 2011.
- [4] M. Tripathy, S. Mishra, L. L. Lai, and Q. P. Zhang, "Transmission loss reduction based on FACTS and bacteria foraging algorithm," in *Proceedings of the 9th International Conference on Parallel Problem Solving from Nature*, pp. 222–231, 2006.
- [5] D. H. Kim and J. H. Cho, "Adaptive tuning of PID controller for multivariable system using bacterial foraging based optimization," in *Proceedings of the 3rd International Atlantic Web Intelligence Conference on Advances in Web Intelligence*, pp. 231–235, 2005.
- [6] M. A. Muñoz, S. K. Halgamuge, W. Alfonso, and E. F. Caicedo, "Simplifying the bacteria foraging optimization algorithm," in *Proceedings of the 6th IEEE World Congress on Computational Intelligence*, pp. 1–7, 2010.
- [7] S. Dasgupta, A. Biswas, A. Abraham, and S. Das, "Adaptive computational chemotaxis in bacterial foraging algorithm," *IEEE Computer Society*, vol. 6, pp. 64–72, 2008.
- [8] H. Chen, Y. Zhu, and K. Hu, "Self-adaptation in bacterial foraging optimization algorithm," in *Proceedings of 3rd International Conference on Intelligent System and Knowledge Engineering (ISKE '08)*, pp. 1026–1031, November 2008.
- [9] F. A. Kidwai, "A genetic algorithm based bus scheduling model for transit network," *Proceedings of the Eastern Asia Society For Transportation Studies*, vol. 5, pp. 477–489, 2005.
- [10] E. B. Jacob, I. Becker, Y. Shapira, and H. Levine, "Bacterial linguistic communication and social intelligence," *Trends in Microbiology*, vol. 12, no. 8, pp. 366–372, 2004.
- [11] S. Mishra, "A hybrid least square-fuzzy bacterial foraging strategy for harmonic estimation," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 1, pp. 61–73, 2005.
- [12] W. J. Tang, Q. H. Wu, and J. R. Saunders, "Bacterial foraging algorithm for dynamic environments," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1324–1330, 2006.
- [13] S. D. Müller, J. Marchetto, S. Airaghi, and P. Koumoutsakos, "Optimization based on bacterial chemotaxis," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 1, pp. 16–29, 2002.
- [14] R. P. Gendron and J. E. R. Staddon, "Searching for cryptic prey: the effect of search rate," *American Naturalist*, vol. 121, no. 2, pp. 172–186, 1983.
- [15] G. Zhengwei, H. L. Pang, and D. W. Wang, "Adaptive bacterial foraging optimization and its application for bus scheduling," *Journal of System Simulation*, vol. 23, no. 6, pp. 1151–1160, 2011.
- [16] S. Sumathi, T. Hamsapriya, and P. Surekha, *Evolutionary Intelligence: An Introduction to Theory and Applications with Matlab*, 2008.
- [17] H. Chen, Y. Zhu, and K. Hu, "Adaptive bacterial foraging optimization," *Abstract and Applied Analysis*, vol. 2011, Article ID 108269, 7 pages, 2011.
- [18] S. Dasgupta, S. Das, A. Abraham, and A. Biswas, "Adaptive computational chemotaxis in bacterial foraging optimization: an analysis," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 919–941, 2009.

Research Article

Direct Index Method of Beam Damage Location Detection Based on Difference Theory of Strain Modal Shapes and the Genetic Algorithms Application

**Bao Zhenming,¹ Zhao Deyou,¹ Lin Zhe,¹
Ma Jun,¹ and Zhen Yaobao²**

¹ Department of Naval Architecture and Ocean Engineering, Dalian University of Technology,
Dalian 116024, China

² School of Mechanical Engineering, Dalian University of Technology, Dalian 116024, China

Correspondence should be addressed to Zhen Yaobao, xu_xia@yahoo.cn

Received 5 October 2012; Accepted 14 November 2012

Academic Editor: Rui Mu

Copyright © 2012 Bao Zhenming 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.

Structural damage identification is to determine the structure health status and analyze the test results. The three key problems to be solved are as follows: the existence of damage in structure, to detect the damage location, and to confirm the damage degree or damage form. Damage generally changes the structure physical properties (i.e., stiffness, mass, and damping) corresponding with the modal characteristics of the structure (i.e., natural frequencies, modal shapes, and modal damping). The research results show that strain mode can be more sensitive and effective for local damage. The direct index method of damage location detection I_{MSD} is based on difference theory, without the modal parameter of the original structure. FEM numerical simulation to partial crack with different degree is done. The criteria of damage location detection can be obtained by strain mode difference curve through cubic spline interpolation. Also the genetic algorithm box in Matlab is used. It has been possible to identify the damage to a reasonable level of accuracy.

1. Introduction

Recently the development of new methods of evaluating the integrity of structures has been attached great importance. The vibrational characteristics of structures can be easily alternated once the damage occurs. In order to analyze the state of the structure integrity and the location of structural damage, the modal analysis with vibrational test data is widely employed [1], though the modal vectors were effectively ignored earlier [2, 3]. The reason is that the parameters of modal analysis depend only on the mechanical characteristics of

the structure. Considering this, the characteristics of modal vibration, that is, the natural frequencies, mode shapes, and so on are considered to represent the state of structure [4].

Although damage index based on strain modal shape is much more sensitive [5–8], in most cases, this index can only indicate the damage location both in undamaged and damaged state with modal data. In fact, these kinds of methods are difficult to employ, because the baseline under suddenness and geological disaster is difficult to master. The problem is solved by Gu et al. [9], who put forward the direct index method of damage location detection I_{MSD} based on difference theory without baseline modal parameters. The index I_{MSD} contains three elements: the effective distance ratio between two adjacent effective extreme points, the absolute difference value of the two effective extremums, and the maximum absolute value of the effective extremum.

Then, the genetic algorithm was gradually applied to solve the problem of damage detection [10]. The method is demonstrated on a simulated beam example and an experimental plate example. Based on the genetic algorithm, Yi and Liu estimated the structural damage with measured dynamic data [11]. Some improved strategies such as multiparent crossover and adjustment of variables are introduced to detect the damage of fixed-end beams and continuous beams. And some satisfied results are achieved. Based on conventional modal analysis theory, Mares and Surace [12] coded a function of binary, and a case of truss-type structure is used to identify the structural damage. And the accurate identification of both the location and the degree of the damage had become possible to be achieved.

Genetic algorithm appears to provide a robust search procedure for solving difficult problems. Due to the way the genetic algorithm explores, the region of interest avoids getting stuck at a particular local minimum and locates the global optimum. To formulate an objective function, with the parameters related to the physical properties and state of the structure, is the aim of the detection of damage in structures. When evaluated with the true parameters, the maximum value of the objective function is obtained. Genetic search algorithm is an optimization procedure which can be employed to determine the values of these parameters by following an iteration process, selecting parameters to maximize the objective function. When the optimization procedure is over, the state of the structure, that is, where and how it is damaged is known.

The objectives of this paper are to develop and apply models to diagnose the damage location and damage degree. This paper is organized as follows: the first section provides a brief introduction to GA, the next section contains results and analyses including performance evaluation of the methodology, and finally the conclusions are presented.

2. Model Development

Modal test of continuum structure uses a series displacements of discrete measuring points to describe all order modes of mode functions. The mode results of all order are the ratio of normalization displacements of discrete measuring points. The strain distributing can be calculated by the difference of deformable displacements.

The vibrational differential equation of straight Euler beam in transverse section is

$$\frac{\partial^2}{\partial x^2} \left(EI \frac{\partial^2 u}{\partial x^2} \right) + \rho A \frac{\partial^2 u}{\partial t^2} = 0. \quad (2.1)$$

The displacement containing all order modes of displacement $\phi_r(x)$ ($r = 1, 2, \dots$) is

$$u(x, t) = U(x)e^{j\omega t} = \sum_{r=1}^{\infty} \phi_r(x) Q_r e^{j\omega t}. \quad (2.2)$$

Curvature mode is the difference of displacement mode.

The discrete curvature mode can be approximately calculated by center difference method as

$$\phi''_{rk} = \frac{\phi_{r(k-1)} - 2\phi_{rk} + \phi_{r(k+1)}}{dx^2} \approx \frac{\phi_{r(k-1)} - 2\phi_{rk} + \phi_{r(k+1)}}{h^2} \quad (2.3)$$

ϕ_{rk} is the r th order of displacement amplitude; k is the calculation point; h is the distance between two adjacent calculation points.

Suppose $z(x)$ is the distance from the middle bending surface to any point, then the strain of this point in x direction is

$$\varepsilon_{z,x} = \frac{\partial u}{\partial x} \approx \frac{\partial^2 \phi_{rk}}{\partial x^2} z(x) = \phi''_{rk} z(x). \quad (2.4)$$

The relationship between strain mode and curvature mode can be transmitted through formula (2.4). Strain mode shape is more sensitive than displacement mode shape at damage detection.

The curve of strain mode shape has extremum and sharp variation at the damage location for simply supported beam, which is the common structural style. The difference curve of strain mode shape must be smoothed by cubic spline interpolation. Central difference method is a numerical method. The advantages of this method are symmetry, a certain accuracy, and easy to use. The format at the nonboundary nodes are central difference. The format at boundary nodes is eccentric difference.

The beam is separated into $n - 1$ units, namely, n nodes. The equidistance is h . $\Phi_r^\varepsilon(x)$ is the r th order strain mode function. The basic equidistance difference format is

$$\left(\frac{d\Phi_r^\varepsilon}{dx} \right)_i = \frac{\Phi_r^\varepsilon(x_{i+1}) - \Phi_r^\varepsilon(x_{i-1}))}{2h}. \quad (2.5)$$

The difference formulation at boundary node 1 is

$$\left(\frac{d\Phi_r^\varepsilon}{dx} \right)_1 = \frac{\Phi_r^\varepsilon(x_2) - \Phi_r^\varepsilon(x_1)}{h}. \quad (2.6)$$

The difference formulation at boundary node n is

$$\left(\frac{d\Phi_r^\varepsilon}{dx} \right)_n = \frac{\Phi_r^\varepsilon(x_n) - \Phi_r^\varepsilon(x_{n-1}))}{h}. \quad (2.7)$$

When difference curve fits, the 2nd order difference values of two extreme points are

$$\begin{aligned} \left(\frac{d^2 \Phi_r^\varepsilon}{dx^2} \right)_1 &= \frac{\Phi_r^\varepsilon(x_1) + \Phi_r^\varepsilon(x_3) - 2\Phi_r^\varepsilon(x_2)}{2h^2}, \\ \left(\frac{d^2 \Phi_r^\varepsilon}{dx^2} \right)_n &= \frac{\Phi_r^\varepsilon(x_{n-2}) + \Phi_r^\varepsilon(x_n) - 2\Phi_r^\varepsilon(x_{n-1})}{2h^2}. \end{aligned} \quad (2.8)$$

Nearby the sharp variation section or the peak value of the mode shape, the extreme points at strain mode curve in damage are located. The damage location is at the curve's sharp variation section.

For actual engineering, Extreme point is unnecessarily just zero at the difference of ideal curve. Three reasons are explained as follows: firstly, the space principles of measuring point or unit classification in numerical simulation are not just at the damage location. Secondly, the accuracy problem of both the strain mode test and the computation is to be considered. Thirdly, the accuracy of difference computation is to be considered.

Interpolation method is used to solve the question of the zero point of strain mode difference curve and confirm the extremums. The computational value at nodes is equal to the supposed function. Many function values are calculated and the function curves draw. Cubic spline interpolation is used in this paper, a widely used interpolation method.

3. Genetic Algorithm

Complex adaptive systems are extremely difficult to comprehend, when unexpected and unpredictable results occur. Holland succeeds to solve this problem by presenting genetic algorithm. In recent years, genetic algorithms are very popular procedure of robust research for solving maximizing or minimizing a given objective function often subject to some constraints [13–20].

Genetic algorithm derives from the process of natural selection and evolution. Because of the inherent advantage of being able to process with a large population of designs and facilitating arrival at the globally optimal solution, the philosophy of “survival of the fittest” has been adopted. It is necessary to devise a general coding system for the representation of the design variables, namely, a directly analogy of the DNA structure of chromosomes. Most commonly, the design variables are coded by a bit-string which is a binary representation. It can only be coded as integers. The progress of genetic algorithm is in the same way of the natural evolution of a species: the fundamental concepts of reproduction, chromosomal crossover, and occasional mutation of genes. Fitness function, which determines possible solutions to the problem, is used to estimate the quality of the represented solution (chromosome). Crossover is a reproduction operation in GA, which is used to vary the programming from one generation to the next by exchanging genetic information between parent chromosomes.

In applying the evolution theories to designing optimization, a number of candidate design variables either randomly or heuristically are created. Then, they evolve over generations to produce new designs which are “fitter.” The “fitness” of the designs is

evaluated according to the objective function, a specific optimization problem. The point with no further improvement is the solution. This paper applies genetic algorithm to the problem of damage detection using strain mode.

In the application to the damage detection in structures, the aim is to formulate an objective function in terms of damage degree of the structure. The objective function must be formulated in such a way that the maximum value is obtained when evaluated with the true parameters. Genetic algorithm can be employed to determine the values of these parameters by following an iteration process, selecting parameters to maximize the objective function. When the optimization procedure arrives at the solution, the values of the parameters indicate the state of the structure, that is, if, where and how it is damaged. The detailed steps are shown as follows.

Step 1. Initialize: the initial population is generated with binary coding, and each individual represents a initial solution.

Step 2. Individual evaluation: calculate the fitness value of each individual.

Step 3. Convergence judgment: implement Step 4, if the fitness value is still not convergent or cannot reach the limit of iteration times, otherwise, terminate the calculation.

Step 4. Individual crossover, mutation, and selection: return to Step 2 after executing crossover (one-point crossover), mutation (discrete mutation), and selection (sampling randomly) process.

Thus, the search process of genetic search can be seen in Figure 1.

4. Example

In the genetic search procedure, selection, crossover, and mutation operators, each with a given probability, are applied to each current population to create the new generations. The results published by DeJong, Grefenstette, and Schaffer indicate the most appropriate parameter setting: a population size of 20–30 individuals, a crossover rate of 0.6–0.95, and a mutation rate 0.01–0.02.

Take the “damage rate 6.25%” as an example to apply the calculation. Figure 2 shows that during the 100 evolutionary generations, the result began to converge after the 80th iteration. Therefore, the final result is obtained: in the first order, the best value of $I_{\text{MSD}}(2)$ is convergent to 0.0048.

The mathematical model has solved the maximum of formula (4.5), in which $x \in [1, 41]$. Selecting binary code, the swarm number is 10. The length of binary code is 19. Crossover probability is 0.95. The mutation probability is 0.01. After the 100 times of genetic iterative, the results were close to the data of Table 1.

The example was a numerically simulated simply beams (see Figure 3) with a finite element model. The characteristics of the beam were as follows: length $l = 0.4\text{ m}$, cross-sectional area $A = 0.0002\text{ m}^2$, in quarter span damage, a one fifth length of the beam width, modulus of elasticity $E = 211\text{ GPa}$, density $\rho = 7850\text{ kg/m}^3$, Poisson ratio $\mu = 0.33$, damage extent $\alpha_1 = 6.25\%h$, $\alpha_2 = 12.5\%h$, $\alpha_3 = 25\%h$, $\alpha_4 = 31.25\%h$, eight node solid 45 unit in Ansys, 25 grids at transverse section, 16 grids at vertical section, damage width $w = 0.002\text{ m}$.

On the basis of numerical simulation results of the strain modes, the smooth strain mode difference curve can be draw based on central difference and cubic spline interpolation.

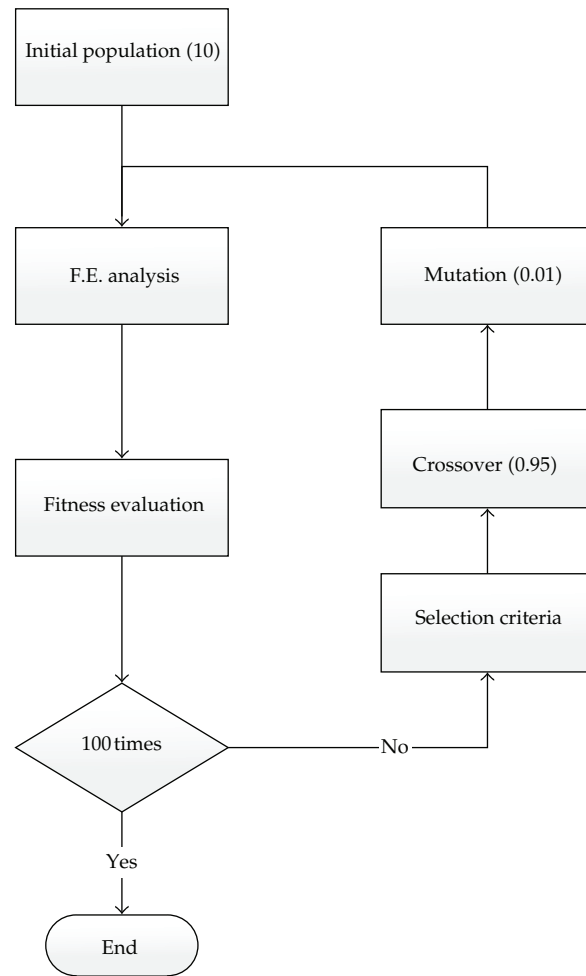


Figure 1: Flowchart of the genetic search.

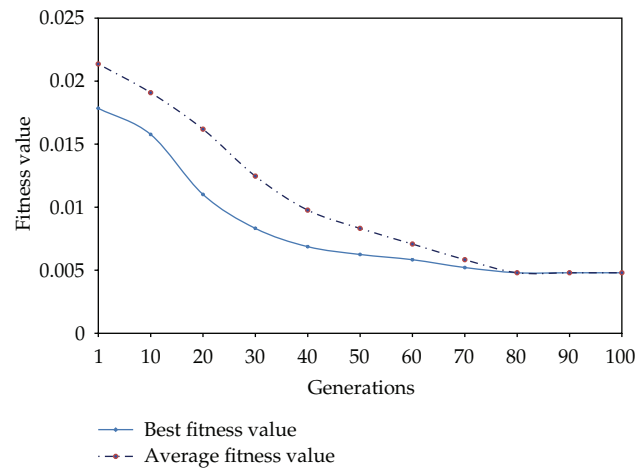


Figure 2: Calculation results.

Table 1: Damage location indices.

Damage location	Order	Damage degree			
		6.25%	12.5%	25%	31.25%
Quarter span	First order				
	$I_{\text{SMSD}}(1)$	0.0018	0.0049	0.0111	0.0159
	$I_{\text{SMSD}}(2)$	0.0048	0.0092	0.0197	0.0238
	$I_{\text{SMSD}}(3)$	0.0011	0.0013	0.0114	0.0167
	$I_{\text{SMSD}}(4)$			0.0016	0.0022
	$I_{\text{SMSD}}(5)$			0.0010	
	Second order				
	$I_{\text{SMSD}}(1)$	0.0162	0.0205	0.0673	0.1003
	$I_{\text{SMSD}}(2)$	0.0202	0.0497	0.1054	0.1647
	$I_{\text{SMSD}}(3)$	0.0137	0.0162	0.0655	0.0997
	$I_{\text{SMSD}}(4)$		0.0127	0.0000078	0.0035
	$I_{\text{SMSD}}(5)$			0.0122	0.0105
	$I_{\text{SMSD}}(6)$				0.0154
	Third order				
	$I_{\text{SMSD}}(1)$	0.0787	0.0877	0.1005	0.1335
	$I_{\text{SMSD}}(2)$	0.0110	0.0661	0.1699	0.2660
	$I_{\text{SMSD}}(3)$	0.0457	0.0754	0.0885	0.1821
	$I_{\text{SMSD}}(4)$	0.0544	0.0539	0.0649	0.0644
	$I_{\text{SMSD}}(5)$		0.0701	0.0707	0.0753

Note: overstriking index and overstriking data correspond to damage location.

**Figure 3:** FEM modal of local damage in simply supported beam.

The figures (from Figures 4, 5, 6, and 7) were the smooth strain mode difference curves after cubic spline interpolation. The damage extents were 6.25%, 12.5%, 25%, and 31.25% of the height.

With the exception of supports, strain mode difference curves have sharp variation at the damage location. For different damage degrees, strain mode curve is slightly different, but the rule has some consistency.

4.1. Mathematical Model of Direct Index Method of Damage Location

4.1.1. Direct Index Method of Damage Location

From the data change at sharp variation zone between two adjacent extremums, damage index was presented. The extents of the curve's variation were reflected.

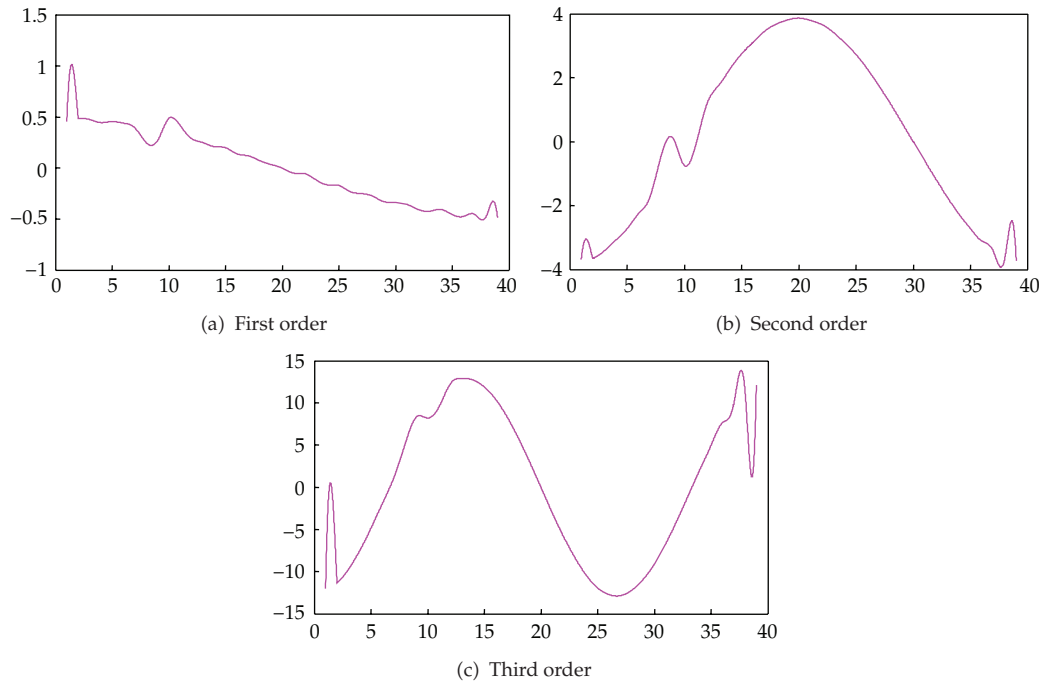


Figure 4: When damage 6.25%, the first three strain modal difference curves.

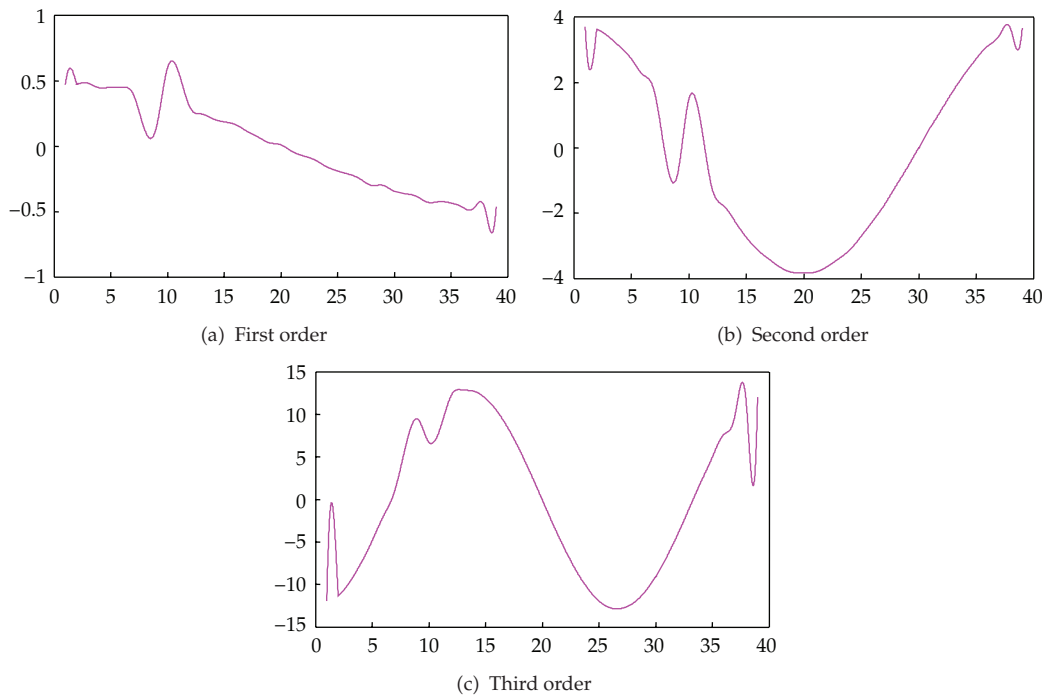


Figure 5: When damage 12.5%, the first three strain modal difference curves.

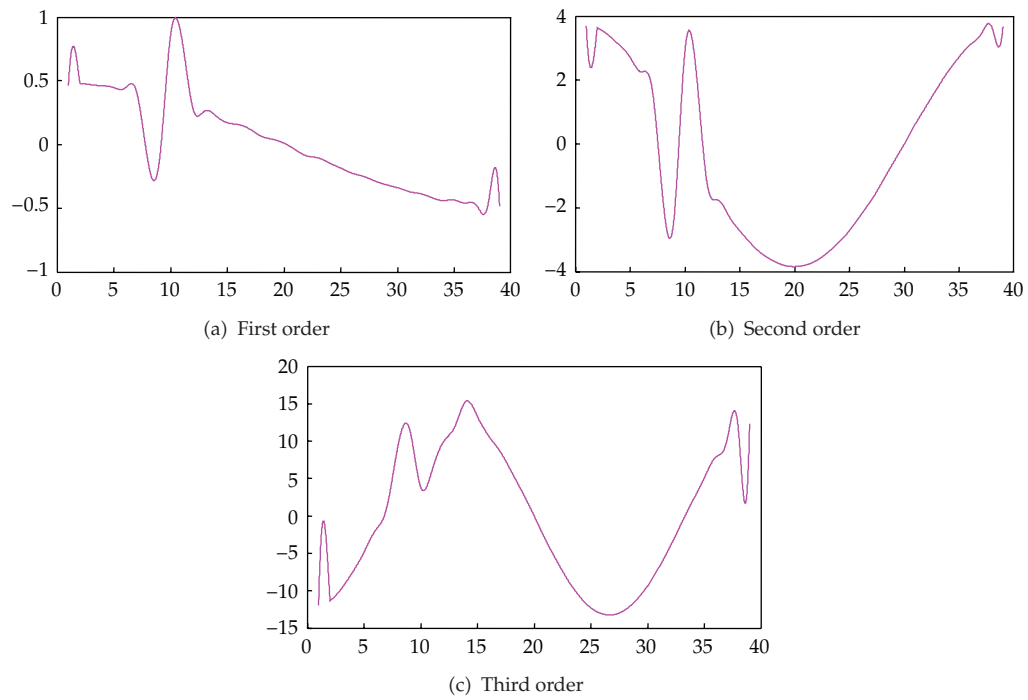


Figure 6: When damage 25%, the first three strain modal difference curves.

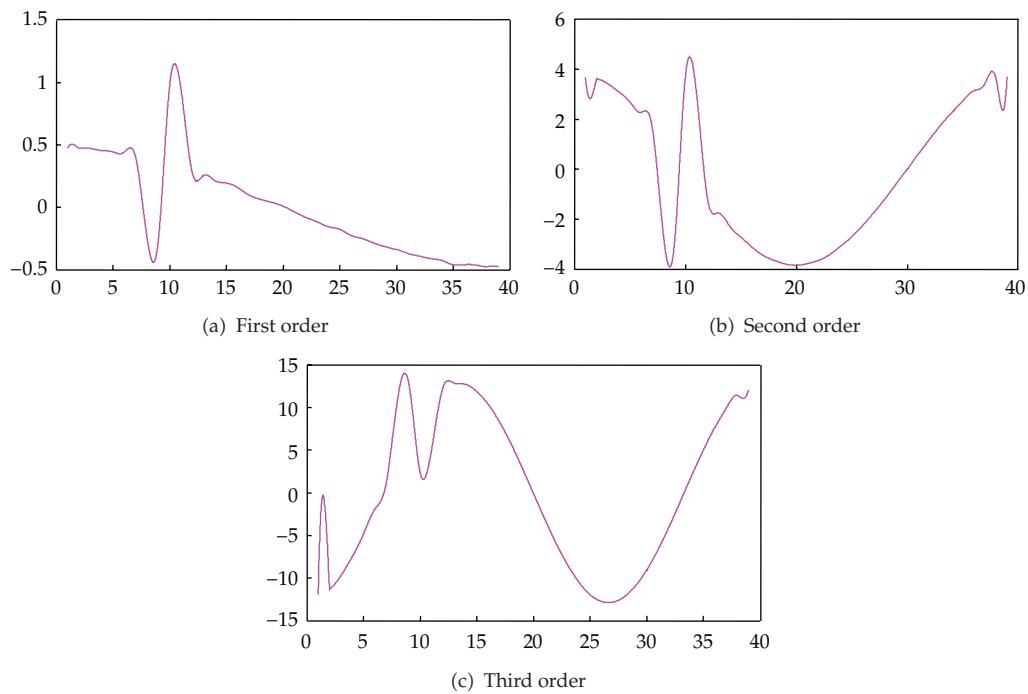


Figure 7: When damage 31.25%, the first three strain modal difference curves.

4.1.2. Effective Distance Ratio

The distance between two adjacent effective extremums is effective distance. Effective distance ratio is the ratio of effective distance to span.

Suppose: span l , effective extremum points: $x_0, x_1, x_2, \dots, x_p$, effective extremums: $y_0, y_1, y_2, \dots, y_p$, then the effective distance ratios are

$$I_{1j} = \frac{(x_j - x_{j-1})}{l}, \quad j = (1, 2, \dots, p). \quad (4.1)$$

The smaller distance ratio of the two adjacent effective extremum is, the greater probability of damage is as follows:

$$I_{2j} = |y_{j2} - y_{j1}|, \quad j = (1, 2, \dots, p). \quad (4.2)$$

Let

$$I_{3j} = \frac{I_{2j}}{I_{1j}}, \quad j = (1, 2, \dots, p). \quad (4.3)$$

4.2. The Absolute Maximum of Effective Extremum: I_{SMSD}

As strain mode difference curve of different orders was nonnormalized, the computational results had no comparison.

In order to solve this problem, the results of formula (4.3), divided the absolute maximum of effective extremums of all orders, the data of different orders can be compared

$$I_{\text{SMSD}}(j) = \frac{I_{2j}}{I_{1j} \max(|y_0|, |y_1|, |y_2|, \dots, |y_q|)}. \quad (4.4)$$

Namely,

$$I_{\text{SMSD}}(j) = l \frac{|y_{j2} - y_{j1}|}{(x_j - x_{j-1}) \max(|y_0|, |y_1|, |y_2|, \dots, |y_q|)}. \quad (4.5)$$

Formula (4.5) is called the direct damage location index. The greater value it is, the more probability of damage occurs. Combined with the rules of difference curve, the damage location can be determined.

Based on formula (4.5), the numerical computation results of damage location indices are listed in Table 1.

From the illustrative examples used, the method using genetic algorithms is very useful to identify both the damage location and the damage degree with a reasonable accuracy. The striking feature of genetic algorithm is that the optimal solution can be determined even with perturbed data having noise. As a parallel search technique, the genetic algorithm makes this approach attractive for problems of large dimensions.

5. Conclusion

The damage indices of the first and second orders are obviously greater than others in the table, and the damage degrees are in accordance with it. The direct index method can accurately detect the damage location and damage degree, especially lower damage. Genetic algorithm is a powerful tool. From the example, accurate identification of both damage location and damage degree had been possible in the structure model. For a protracting beam, there is no macrocrack. The difference curve of strain modes at the damage location has sharp variation, but the numerical value probably is not zero.

Acknowledgment

This work was supported by National Natural Science Foundation of China (51208079).

References

- [1] M. Basseville, A. Benveniste, B. Gach-Devauchelle et al., "In situ damage monitoring in vibration mechanics: diagnostics and predictive maintenance," *Mechanical Systems and Signal Processing*, vol. 7, no. 5, pp. 401–423, 1993.
- [2] J. K. Vandiver, "Detection of the Structural failure of fixed platforms by measurement of dynamic response," in *Proceeding of the Offshore Conference Society of Petroleum Engineering*, pp. 243–252, Richardson, Tex, USA, 1975.
- [3] R. N. Coppolino and S. Rubin, "Detectability of structural failure in off-shore platforms by ambient vibration monitoring," in *Proceedings of the Offshore Conference Society of Petroleum Engineering*, pp. 101–110, Richardson, Tex, USA, May 1975.
- [4] D. J. Ewins, *1984 Modal Testing: Theory and Practice*, Wiley, New York, 1980.
- [5] B. S. Wang, X. B. Liang, Y. Q. Ni et al., "Comparative study of damage indices in application to a long-span suspension bridge," in *Advance in Structural Dynamics*, pp. 1085–1092, Elsevier Science Ltd, Oxford, UK, 2000.
- [6] Y. Y. Lee and K. W. Liew, "Detection of damage location in a beam using the wavelet analysis," *International Journal of Structural Stability and Dynamics*, vol. 1, no. 3, pp. 455–465, 2001.
- [7] J. T. Kim, S. H. Jung, Y. K. Lee, and J. W. Yun, "Damage identification in bridge using vibration-based system identification scheme," in *Proceedings of the 18th International Modal Analysis Conference (IMAC '00)*, pp. 1327–1333, San Antonio, Tex, USA, 2000.
- [8] J. T. Kim and N. Stubbs, "Model-uncertainty impact and damage-detection accuracy in plate girder," *Journal of Structural Engineering*, vol. 121, no. 10, pp. 1409–1417, 1995.
- [9] P. Y. Gu, M. Deng, and F. S. Wu, *Structure Mode Analysis and Damage Diagnosis*, Southeast University Press, Missouri, Mo, USA, 2008.
- [10] M. I. Friswell, J. E. T. Penny, and S. D. Garvey, "A combined genetic and eigensensitivity algorithm for the location of damage in structures," *Computers and Structures*, vol. 69, no. 5, pp. 547–556, 1998.
- [11] W. J. Yi and X. Liu, "Damage diagnosis of structures by genetic algorithms," *Engineering Mechanics*, vol. 18, no. 2, pp. 64–71, 2001.
- [12] C. Mares and C. Surace, "An application of genetic algorithms to identify damage in elastic structures," *Journal of Sound and Vibration*, vol. 195, no. 2, pp. 195–215, 1996.
- [13] S. Rajeev and C. S. Krishnamoorthy, "Discrete optimization of structures using genetic algorithms," *Journal of Structural Engineering*, vol. 118, no. 5, pp. 1233–1250, 1992.
- [14] W. M. Jenkins, "Plane frame optimum design environment based on genetic algorithm," *Journal of Structural Engineering*, vol. 118, no. 11, pp. 3103–3112, 1992.
- [15] W. M. Jenkins, "Towards structural optimization via the genetic algorithm," *Computers and Structures*, vol. 40, no. 5, pp. 1321–1327, 1991.
- [16] A. J. Keane, "Passive vibration control via unusual geometries: the application of genetic algorithm optimization to structural design," *Journal of Sound and Vibration*, vol. 185, no. 3, pp. 441–453, 1995.
- [17] B. Yu and Z. Z. Yang, "An ant colony optimization model: the period vehicle routing problem with time windows," *Transportation Research E*, vol. 47, no. 2, pp. 166–181, 2011.

- [18] B. Yu, Z. Z. Yang, and S. Li, "Real-time partway deadheading strategy based on transit service reliability assessment," *Transportation Research A*, vol. 46, no. 8, pp. 1265–1279, 2012.
- [19] B. Yu, Z. Z. Yang, P. H. Jin, S.-H. Wu, and B. Z. Yao, "Transit route network design using ant colony optimization," *Transportation Research C*, vol. 22, pp. 58–75, 2012.
- [20] B. Yu, Z. Z. Yang, and B. Yao, "An improved ant colony optimization for vehicle routing problem," *European Journal of Operational Research*, vol. 196, no. 1, pp. 171–176, 2009.

Research Article

Genetic Algorithm for Multiuser Discrete Network Design Problem under Demand Uncertainty

Wu Juan,^{1,2} Lu Huapu,¹ Yu Xinxin,³ and Bian Changzhi⁴

¹ Institute of Transportation Engineering, Tsinghua University, Beijing 100084, China

² Academy of Military Transportation, Tianjin 300161, China

³ Transport Planning and Research Institute, Ministry of Transport, Beijing 100028, China

⁴ China Academy of Urban Planning and Design, Beijing 100044, China

Correspondence should be addressed to Wu Juan, wujuan_qh@yahoo.cn

Received 6 September 2012; Accepted 29 October 2012

Academic Editor: Baozhen Yao

Copyright © 2012 Wu Juan 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.

Discrete network design is an important part of urban transportation planning. The purpose of this paper is to present a bilevel model for discrete network design. The upper-level model aims to minimize the total travel time under a stochastic demand to design a discrete network. In the lower-level model, demands are assigned to the network through a multiuser traffic equilibrium assignment. Generally, discrete network could affect path selections of demands, while the results of the multiuser traffic equilibrium assignment need to reconstruct a new discrete network. An iterative approach including an improved genetic algorithm and Frank-Wolfe algorithm is used to solve the bi-level model. The numerical results on Nguyen Dupuis network show that the model and the related algorithms were effective for discrete network design.

1. Introduction

With the development of cities, travel demand is high and widely spread. The capacity of the current transport system remains limited to accommodate the increasing demand. It has emerged as an important area for progress in handling effective transport planning, in which some new links or roadway segments are added to expanding the current system capacity. The discrete network design problem (DNDP) deals with the selection of link additions to an existing road network, with a given demand from each origin to each destination. The objective of DNDP is often to optimize a given system performance measure such as to minimize total system travel cost, while accounting for the route choice behaviors of network users. Farvaresh and Sepehri [1] presented a single-level mixed integer linear formulation for discrete network design. Miandoabchi and Farahani [2] presented a discrete network design

model, in which the concurrent design of street capacity, street direction, and lane allocations for two-way street are optimized based on the reserve capacity maximization. In traditional transportation network design, the travel demand is often assumed as a constant and users are assumed to belong to a single class. However, the assumptions are inappropriate to real-life applications.

Different forms may be adopted for different aims in those literatures; but uncertainty in decision making and the diversity of the users are not considered. This paper seeks to make two contributions to the literature. Firstly, when the travel demand is uncertainty, uncertain optimization theory will be used. Secondly, when the users are diversity, they will be classified and treated, respectively. Therefore, a bi-level model was proposed for traffic network design. The upper-level model takes the travel time as the main consideration factor to optimize the network design model. Considering the fact that the travel demand changes along with the change of the network, the lower-level model assigns the users again in the network optimized by the upper-level model.

The uncertain transportation network design model based on the stochastic programming theory assumes that travel demand is a stochastic variable submitting to a known probability distribution. At the same time, the optimal network plan will be obtained with stochastic bi-level programming model. Patriksson [3] considered demand uncertainty in stochastic bi-level programming model, in which the upper-level model is to minimize expected value of the objective function, and the lower-level model is the equilibrium conditions of variation inequality.

Ukkusuri and Waller [4] provided the chance constrained programming model and two-phase compensation stochastic programming model for designing the transportation network with a single end and uncertain OD demand. When traffic flow meets the dynamic user equilibrium condition, cell transmission model (CTM) can be used, and the numerical calculation shows that the suboptimal solutions will be obtained without considering the uncertainty of demand. Karoonsoontawong and Waller [5] established a continuous transportation network design model under the demand uncertainty. Assume that every demand situation meets a dynamic user equilibrium to describe the traffic flow based on the CTM model of Daganzo [6]. The model minimizes the weighted average of the expected mean value and the expected risk to improve model robustness against demand uncertainty. Yang [7] analyzed the behavior of equilibrium flows with elastic demand which can be used to measure the demand and performance characteristics of the transportation networks. Li et al. [8] attempted to present road toll design model for congested road networks with uncertain demand. A heuristic algorithm based on the sample average approximation approach and a sensitivity analysis is used to solve the network design model.

There are two types of methods dealing with multiuser problem in transportation network design problem. One method is to classify users according to traffic mode characteristics and vehicle types; then each category of users has different cost functions. Similar study has been done by Smith [9]; and so forth. Another method assumes that vehicle types of travelers are the same and have the same effect on traffic flow; but there are differences in time value. According to the different way of study, time value distribution can be assumed as discrete or continuous, corresponding to limited categories of users or infinite categories of users. Research about multiuser network equilibrium based on discrete time value has been done by Daganzo [6], Yang and Zhang [10]. Research about multiuser network equilibrium based on continuous time value has been done by Dial [11].

In order to correct the inappropriate assumptions in traditional transportation network design, in this paper, the OD trip demand elements are supposed as stochastic variables

submitting to given probability distribution. Travelers are divided into different groups by the value of time and a novel multiuser network design model is established. However, for network design, it is difficult to be solved through classical optimization techniques [12]. Recently many studies have proved that heuristic algorithms are suitable for large-scale transit network optimization problems, such as ant colony algorithm [13–15] and simulated annealing algorithm [16].

Genetic algorithm (GA) is a search heuristic that based on the idea extracting from the process of natural evolution. Recently many studies have proved that genetic algorithm is suitable for solving network design problem. Pattnaik et al. [17] presented a GA-based optimization method to design transit network, in which the total cost of user and operator was to be minimized. Agrawal and Mathew [12] presented an optimization model for transit network. The model was aiming to minimize the total system cost which included the operating cost and the generalized travel cost. Bielli et al. [18] developed a heuristic based on GA to design transit network. In the heuristic, a multicriteria analysis was used to estimate the fitness values. Thus, in this paper, GA is also used to solve our discrete network design problem.

This paper has been organized in the following way. Section 2 describes the travelers time value and multiuser classification; Section 3 is about the optimization model, including the problem formulations and the basic notations of variables; Section 4 describes genetic algorithm and Frank-Wolfe algorithm for discrete network design problem. Numerical analysis is carried out in Section 5, and lastly, the conclusions are drawn in Section 6.

2. Analysis of the Fundamental Factors

2.1. Traveler Time Value

In economics, social activities can be abstracted into production and consumption behavior. The elements to describe different activities are often different, and time consumption is often used to measure the activity efficiency. Time as a resource, its value should be reasonable measured for better and efficient allocation. Time value represents time saving in terms of money. Under a given space-time environment, the factors that affect time value mainly include traveler characteristics, travel purpose, transportation modes, and other aspects.

In different conditions, the influence degree of each factor is different. Evaluation of traveler time value is a comprehensive reflection of these factors. The following is the introduction of the main factors that affect traveler time value.

(1) Traveler Characteristics

Different social and economic characteristics often affect traveler behavior. The income level is the greatest effect among their characteristics. High-income passengers pay more attention to quickness, comfort, safety, and service level than the travel fee while low-income passengers tend to use more time to save money.

(2) Travel Purpose

Travel purpose is the motive of a trip. When travelers are confronted with different travel purposes, there are often different choices for them to select and different choices with different time and cost. For example, a trip for work has time constraint while a trip for

shopping has more free time; so the time value for work is more than the time value for shopping. In some special circumstances, such as the traffic accidents or emergency which need medical assistance, time value is much higher than that of normal work trip.

(3) *Transportation Mode*

Traveler time value is not only related to travel purpose, but also transportation mode. Different transportation modes have different speeds, convenience, and comfort, and those differences affect travelers to select different transportation modes. When travelers choose some transportation modes, they often consider those factors like travel time, travel cost, and the auxiliary or additional travel time. For example, car can provide prompt door to door service, so its time value is high. The bus needs more time not only aboard bus but also out of bus for waiting and walking; so its time value is low.

(4) *Other Factors*

In addition to the influences of the above factors on traveler time value, travel distance, road traffic conditions, vehicles conditions, and the service level [19], to a certain degree, also have influences on the evaluation of travelers time value.

2.2. User Classification

There are two types of methods dealing with multiuser problem in transportation network. One method is to classify users according to transportation modes and vehicle types, in which each category of users has its cost function. In this paper, for the convenience of the study, all the traveler time values are also categorized into two kinds: discrete and continuous. In the same way, the users are considered as two kinds: limited categories of users and infinite categories. This paper assumes that the difference among transportation network users is the traveler time value, and the other characteristic is not considered.

3. Multiuser Discrete Network Design Model under OD Demand Uncertainty

3.1. Basic Notations

The following are the notations used in the model formulation.

- N : Transportation network nodes set
- A : Transportation network links set
- O_r : The trip generation flow from the terminal r
- D_s : The trip attraction flow from the terminal s
- P_{rs} : The routes set from the origin terminal r to the destination terminal s
- x_a : Traffic flow on link a
- $t_a(x)$: Travel time impedance function of link a
- f_k^{rs} : Traffic flow on path k between OD pair r and s

- c_k^{rs} : Traffic cost on path k between OD pair r and s
 $\delta_{a,k}^{rs}$: When a belongs to path k which is between OD pair r and s , $\delta = 1$; otherwise $\delta = 0$
 \bar{A} : Set of new built or expanded links
 y_a : Decision variable of link a , $y_a \in \{0, 1\}$, when link will be new built or expanded, $y_a = 1$; otherwise, $y_a = 0$
 C_a : Transport capacity of link a
 L_a : Length of link a
 $G_a(y_a)$: Cost of new built or expanded link a
 B : Budget of all the new built or expanded links
 Ω : All possible scenarios set of uncertain travel demand
 ω : Any realization of uncertain travel demand
 p^ω : Realization probability of uncertain travel demand scenario ω
 ρ : Mean and variance weight of travel time given by planning decision makers.

3.2. Multiuser Assumption

Each group of travelers has similar social and economic characteristics (such as income level). If travelers can be divided into discrete I groups according to time value, the time value of travelers i from the origin terminal r to the destination terminal s is set to g_{rs}^i when $i \in I$. Thus, the user cost includes two parts. One is the cost of travel time value which is related to route flow; the other is the toll fee τ_a , which is a constant. The sum of the cost from two parts is changed as the variation of the travel time value. Formula (3.1) is the generalized cost of the use of link a by i group users. Formula (3.2) is the time cost on path k between OD pair r and s . Formula (3.3) is the expenses cost on path k between OD pair r and s . Formula (3.4) generalized the cost of the use of path k between OD pair r and s by i group users:

$$g_a^i(x_a) = t_a(x_a) + \frac{1}{\varphi_i} \tau_a, \quad \forall a \in A, i \in I, \quad (3.1)$$

$$c_k^{rs} = \sum t_a(x_a) \delta_{a,k}^{rs}, \quad \forall r \in R, s \in S, k \in K, \quad (3.2)$$

$$\tau_k^{rs} = \sum \tau_a \delta_{a,k}^{rs}, \quad \forall r \in R, s \in S, k \in K, \quad (3.3)$$

$$g_{k,i}^{rs} = c_k^{rs} + \frac{1}{\varphi_i} \tau_k^{rs}, \quad \forall r \in R, s \in S, k \in P_{rs}, i \in I. \quad (3.4)$$

3.3. Multiuser Network Optimization Model under OD Demand Uncertainty

OD trip demand of each group travelers is supposed as a random variable submitting to the given probability distribution. In practical calculation, when Monte-Carlo random sampling is used to form a demand scenario set Ω , any demand scenario realization is ω . OD trip demand is $q_{rs}^{i,\omega}$, where i is the set of groups of travelers. Scenario realization probability is p^ω . Multiuser discrete network design model under OD trip demand uncertainty includes upper-level model (3.5) and lower-level model (3.6), which are correlated by network improved decision variable y and traffic flow x .

The upper-level model (3.5) is to minimize the system total travel time mean and standard deviation with the random demand in all scenarios realization condition, when planners choose new built and rebuilt links under the capital budget constraints. The lower-level model (3.6) is the corresponding multiuser equilibrium of each demand scenario under the improved decisions conditions decided by the upper-level model. On has

$$\begin{aligned} \min \quad Z(\mathbf{x}, \mathbf{y}) = & \rho \sum_{\omega} p^{\omega} \left[\sum_a x_a^{\omega} t_a^{\omega}(x_a^{\omega}, y_a) \right] + (1 - \rho) \\ & \times \left[\sum_{\omega} p^{\omega} \left\{ \sum_a x_a^{\omega} t_a^{\omega}(x_a^{\omega}, y_a) - \sum_{\omega} p^{\omega} \left[\sum_a x_a^{\omega} t_a^{\omega}(x_a^{\omega}, y_a) \right] \right\}^2 \right]^{1/2}, \end{aligned} \quad (3.5)$$

$$\text{s.t.} \quad \sum_{a \in \bar{A}} G_a(y_a) \leq B, \quad (3.5a)$$

$$y_a \in \{0, 1\}, \quad \forall a \in \bar{A}, \quad (3.5b)$$

where $x = x(y)$ is implicit function of y , decided by lower-level model (3.6). On has

$$\min \quad T(\mathbf{x}) = \sum_a \int_0^{x_a^{\omega}} t_a^{\omega}(w) dw + \sum_a \sum_{i \in I} \frac{1}{\psi_i} x_a^{i, \omega} \tau_a \quad (3.6)$$

$$\text{s.t.} \quad \sum_{k \in P_{rs}} f_{k,i}^{rs, \omega} = q_{rs}^{i, \omega}, \quad \forall r \in R, s \in S, \forall i \in I, \omega \in \Omega \quad (3.6a)$$

$$f_{k,i}^{rs, \omega} \geq 0, \quad \forall r \in R, s \in S, k \in P_{rs}, i \in I, \omega \in \Omega \quad (3.6b)$$

$$x_a^{i, \omega} = \sum_{r \in R} \sum_{s \in S} \sum_{k \in P_{rs}} f_{k,i}^{rs, \omega} \delta_{ak}^{rs, \omega}, \quad \forall a, i \in I, \omega \in \Omega \quad (3.6c)$$

$$x_a^{\omega} = \sum_{i \in I} x_a^{i, \omega}, \quad \forall a, \omega \in \Omega. \quad (3.6d)$$

Here, travel time of link a is described as BPR function.

The right balance between the mean and standard deviation is kept by weight factor ρ ($\rho \in [0, 1]$), where ρ shows the prediction of the planners for the average performance of uncertainty and the discrete degree of depart from the average performance. The bigger ρ value is, the less the planners would like to select.

In this paper, the multiuser discrete network design problem under demand uncertainty can be shown in Figure 1.

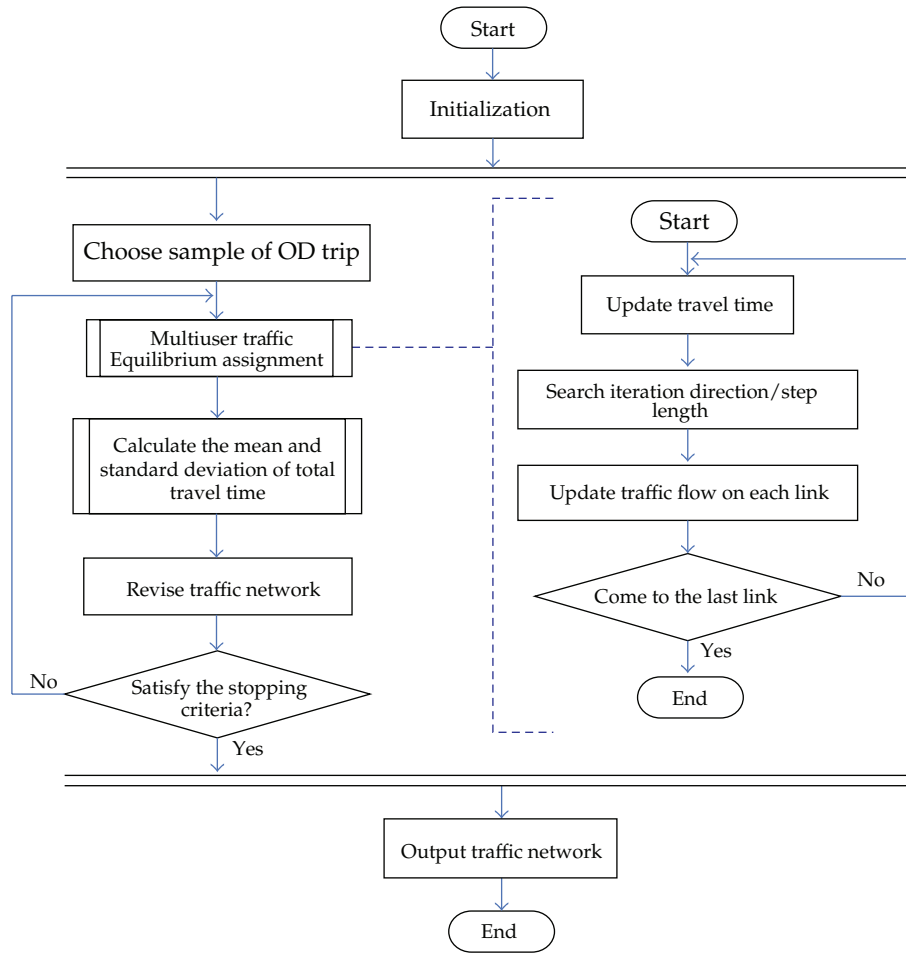


Figure 1: The optimized process of discrete network design problem.

4. Solution Approach

4.1. Multiuser Traffic Equilibrium Assignment

Using Frank-Wolfe method to solve multiuser equilibrium model consists of the following steps.

Step 1. Initialization: according to $\{t_a^0 = t_a(0)\}$ and $\{\tau_a\}$, 0-1 traffic assignment based on generalized cost is conducted to each group of users demand $\{q_{rs}^i\}$ when $n = 1$.

Step 2. Updating travel time on each link: determining generalized cost $g_a^{in}(x_a^n)$ of each group of users on each link when $t_a^n = t_a(x_a^n)$, for all a .

Step 3. Searching for iterative direction: conduct 0-1 traffic assignment according to generalized cost $\{g_a^{in}(x_a^n)\}$ to each group of users and get a set of additional traffic flow $\{y_a^{in}\}$.

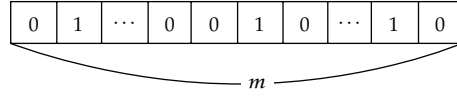


Figure 2

Step 4. Searching for iteration step length λ_n based on $\min_{0 \leq \lambda_n \leq 1} T(\mathbf{x}^{in} + \lambda_n(\mathbf{y}^{in} - \mathbf{x}^{in}))$: to seek λ_n until it meets $\min_{0 \leq \lambda_n \leq 1} T(\mathbf{x}^{in} + \lambda_n(\mathbf{y}^{in} - \mathbf{x}^{in}))$.

Step 5. Updating road traffic flow on each link:

$$x_a^{i(n+1)} = x_a^{in} + \lambda_n(y_a^{in} - x_a^{in}), \quad \forall a, i; \quad x_a^{n+1} = x_a^n + \lambda_n(y_a^n - x_a^n), \quad \forall a. \quad (4.1)$$

Step 6. Convergence test: if $\{x_a^{n+1}\}$ already meets specified convergence criteria, calculation stops and $\{x_a^{n+1}\}$ are the balance solutions. Otherwise update $n = n + 1$, return to Step 2.

4.2. GA for Multiuser Discrete Transportation Network Design

GA is inspired by evolutionary biology like inheritance, selection, crossover, and mutation. Based on a fitness function, GA attempts to retain relatively good genetic information from generation to generation. GA has been used for solving approximately combinatorial optimization problems [20]. In this paper, GA is adopted to solve multiuser discrete transportation network design model under demand uncertainty. The following is the steps of GA.

(1) Encoding

For a discrete network design, the added links or expanded links are to assign to the current network. Thus, an integer-coded scheme is selected to represent the alternative links in this paper, and a chromosome example is as shown in Figure 2, where “0” represents the corresponding link remaining the current situation while “1” represents the links need to add new links or be expanded. For example, the chromosome is 1100110000 which represents that route no. 1, 2, 5, and 6 needs to add new links or be expanded. The other routes remain the current situation.

(2) Fitness Function

Generally, GA is optimal searching method to find the maximum fitness of the individual chromosome. Therefore, a constant Q is introduced to transform our objective function to a maximum fitness function and the chromosomes are evaluated as follows:

$$M(f) = \frac{Q}{Z}, \quad (4.2)$$

where $M(f)$ is the fitness function and Q is a constant.

(3) Selection

The basic part of the selection process is to stochastically select from one generation to create the basis of the next generation. The requirement is that the fittest individuals have a greater chance of survival than weaker ones. That is, the better the chromosomes are, the more chances to be selected they have. Therefore, the Roulette wheel selection method is used for the selection of chromosomes in this paper. In addition, to increase the performance of GA, elitism is used for selection. That is, if the elitism parameter was set to R ; then the top R chromosomes in the population are copied to the next generation.

(4) Crossover

Crossover is a genetic operator that exchanges genetic information between two parents' chromosomes to produce two new children chromosomes. The crossover operator occurs during evolution according to a given crossover rate p_c . In this paper, in crossover operation, the two links are selected, based on a simple arithmetic crossover [20], from the parent chromosomes and exchange the two links, and then generate two new children chromosomes:

$$\begin{aligned} \text{gen}_{k,I}^t &= \alpha_i \text{gen}_{k,I}^{t-1} + (1 - \alpha_k) \text{gen}_{k,II}^{t-1}, \\ \text{gen}_{k,II}^t &= \alpha_i \text{gen}_{k,II}^{t-1} + (1 - \alpha_k) \text{gen}_{k,I}^{t-1}, \end{aligned} \quad (4.3)$$

where $\text{gen}_{k,I}^{t-1}, \text{gen}_{k,II}^{t-1}$ is a pair of "parent" chromosomes; $\text{gen}_{k,I}^t, \text{gen}_{k,II}^t$ is a pair of "children" chromosomes; α_k is a random number between (0,1); $k \in [1, 2, 3]$ (k is the total genes for the crossover operation).

(5) Mutation

Like the crossover, the mutation operator is also associated with a mutation rate (P_m) to determine whether or not the mutation operator is to be applied to the chromosome. An arithmetic mutation like the crossover is designed, and then a new offspring chromosome is acquired by mutation operator.

Assume a chromosome is $G = (\text{gen}_1^t, \dots, \text{gen}_k^t, \dots, \text{gen}_m^t)$, if the gen_2^t was selected for the mutation, the mutation can be shown in (4.4):

$$\begin{aligned} G' &= (\text{gen}_1^{t-1}, \dots, \text{gen}_k^t, \dots, \text{gen}_m^{t-1}), \\ \text{gen}_k^t &= \begin{cases} \text{gen}_k^{t-1} + \Delta(t, \text{gen}_{k_{\max}}^t - \text{gen}_k^{t-1}), & \text{if random}(0, 1) = 0, \\ \text{gen}_k^{t-1} + \Delta(t, \text{gen}_k^{t-1} - \text{gen}_{k_{\min}}^t), & \text{if random}(0, 1) = 1. \end{cases} \end{aligned} \quad (4.4)$$

The function $\Delta(t, y)$ returns a value between $[0, y]$ given in (4.5).

$$\Delta(t, y) = y \times \left(1 - r^{(1-t/T_{\max})^\lambda}\right), \quad (4.5)$$

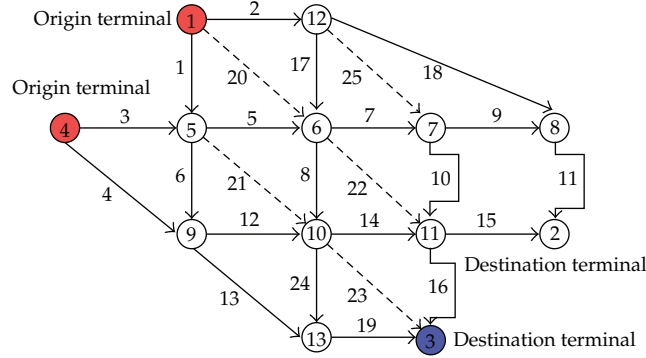


Figure 3: The networks of Nguyen Dupuis.

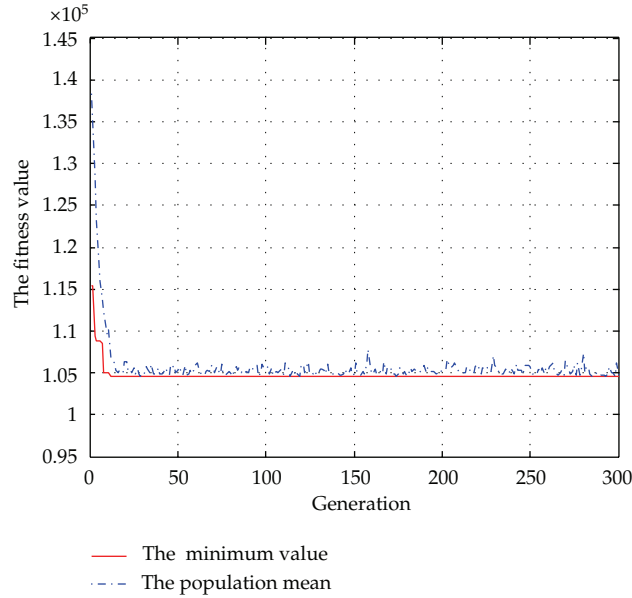


Figure 4: The solution of scenario I.

where r is a random number between $[0,1]$; T_{\max} is the maximum number of generations; here $\lambda = 3$. This property causes this operation to make a uniform search in the initial space when t is small and a very local one in later stages.

5. Case Studies

5.1. Network Structure

In this paper, the test network of Nguyen and Dupuis [21] is used as a case study. This network has 13 nodes, 19 links, and 4 OD pairs. The basic structures of this network is shown in Figure 3, in which a red node is the symbol for a travel demand generation point, a blue node is the symbol for a travel demand attract point, a solid line is the symbol for

Table 1: The attribute of the Nguyen Dupuis network.

Link	Free flow time	Existing capacity	Planning capacity	Construction cost
1	12	250	500	100
2	12	250	500	100
3	12	250	500	100
4	24	150	250	100
5	12	250	500	100
6	12	250	500	100
7	12	250	500	100
8	12	250	500	100
9	12	250	500	100
10	12	250	500	100
11	12	250	500	100
12	12	250	500	100
13	24	150	250	100
14	12	250	500	100
15	12	250	500	100
16	12	250	500	100
17	12	250	500	100
18	36	150	250	100
19	12	250	500	100
20	24	0	250	100
21	24	0	250	100
22	24	0	250	100
23	24	0	250	100
24	12	0	500	100
25	24	0	250	100

Table 2: The data of OD.

Category	Number	Trip generation and attraction	Truncation normal distribution of trip generation and attraction
Trip generation site	1	$\overline{O_1}$	$TN(\overline{O_1}, \mu \overline{O_1})$
	4	$\overline{O_4}$	$TN(\overline{O_4}, \mu \overline{O_4})$
Trip attraction site	2	$\overline{D_2}$	$TN(\overline{D_2}, \mu \overline{D_2})$
	3	$\overline{D_3}$	$TN(\overline{D_3}, \mu \overline{D_3})$

a existing road, and a dotted line is the symbol for a road to be built. Table 1 shows the basic information of the network, including free flow time, traffic capacity under the present situation, traffic capacity under planning situation, construction cost, and so forth. Table 2 is OD trip demand information, including deterministic demand and truncated normal distribution travel demand.

5.2. Calculation Results

This network is assumed to have three types of users, and the OD trip demand of each type of users submits to truncation normal distribution; time value is set to 0.5, 1, and 2, respectively.

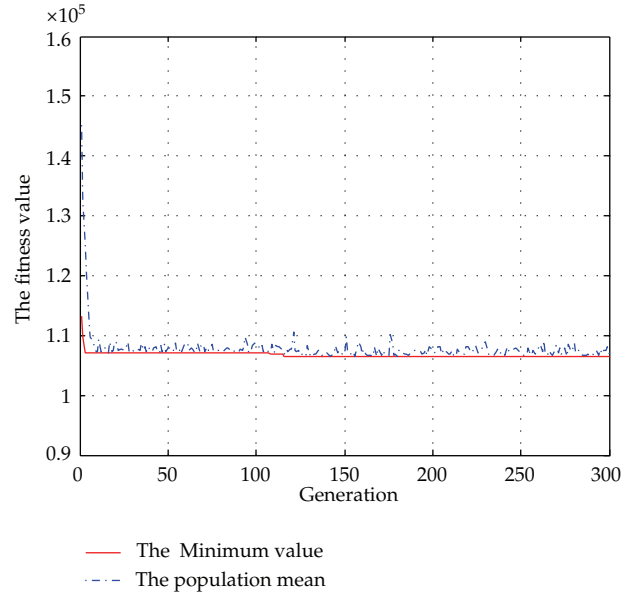


Figure 5: The solution of scenario II.

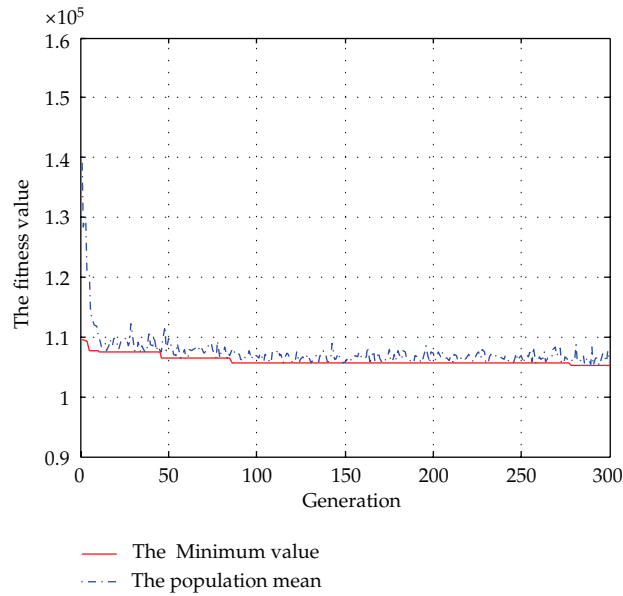


Figure 6: The solution of scenario III.

There are 25 routes for new links and expanding links. So the parameters in genetic algorithm are set as the following. That is population size as 40, evolutionary generation range as 100, chromosome length as 25, crossover probability as 0.8, and mutation probability as 0.01 (Table 3).

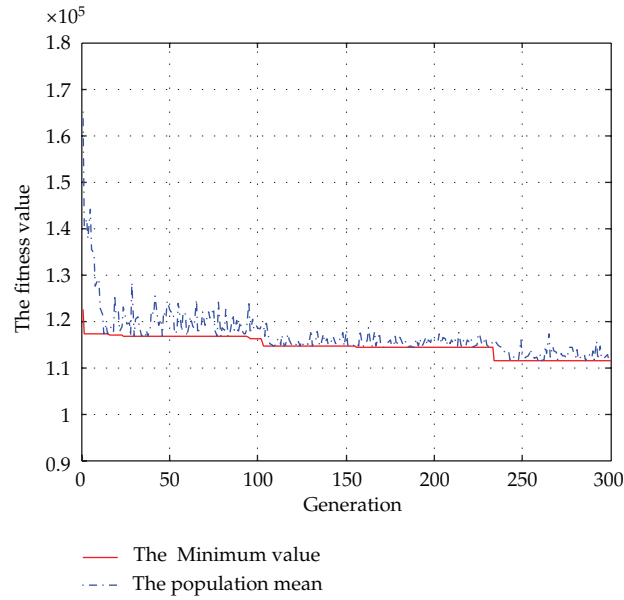


Figure 7: The solution of scenario IV.

Table 3: The parameters of the scenario.

Scenario	User demand	Mutation coefficient μ	Budget level	Risk coefficient ρ	Sample size
I	100/150/100	0	1000	1	1
II	50/250/50	0	1000	1	1
III	250/50/50	0	1000	1	1
IV	50/50/250	0	1000	1	1
V	100/150/100	0.2	1000	1	50
VI	100/150/100	0.5	1000	1	50
VII	100/150/100	0.2	1000	0.5	50
VIII	350	0	1000	1	1

The evolution process under deterministic OD trip demand is shown in Figures 4, 5, 6, and 7. The evolution process under uncertain OD trip demand is shown Figures 8, 9, and 10. The evolution process with only one type of users is shown in Figure 11.

Table 4 shows the calculation results of multiuser discrete transportation network under OD trip demand uncertainty, from which we can obtain the following conclusion.

- (1) Scenario I and scenario VIII have the same total demand of all OD pairs, while scenario VIII has only one type of users and scenario I has three types of users. Results show that network planning scheme based on multiuser equilibrium model is different from that of single user model, and the total travel time of the system based on multiuser equilibrium model is higher.
- (2) Contrasting from scenario I to scenario IV, OD trips of each type of users are different. The calculation results show that the OD trips proportion of different users to the total amount of the network has a significant impact on the planning scheme. In the transportation planning practice, the trip amount of different users

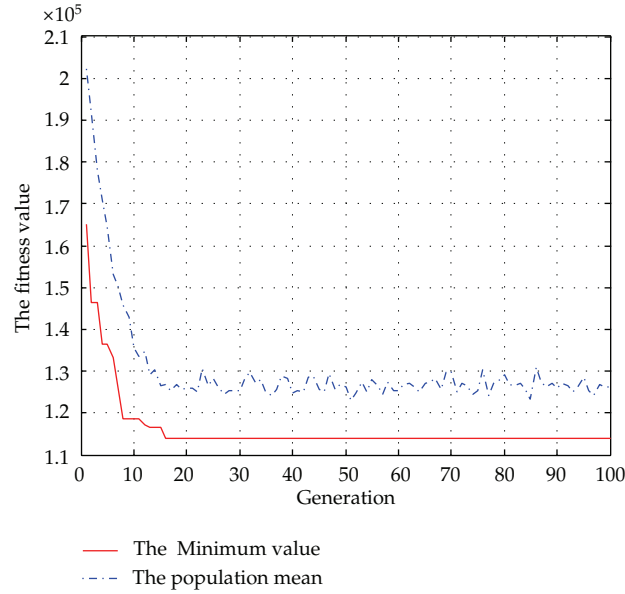


Figure 8: The solution of scenario V.

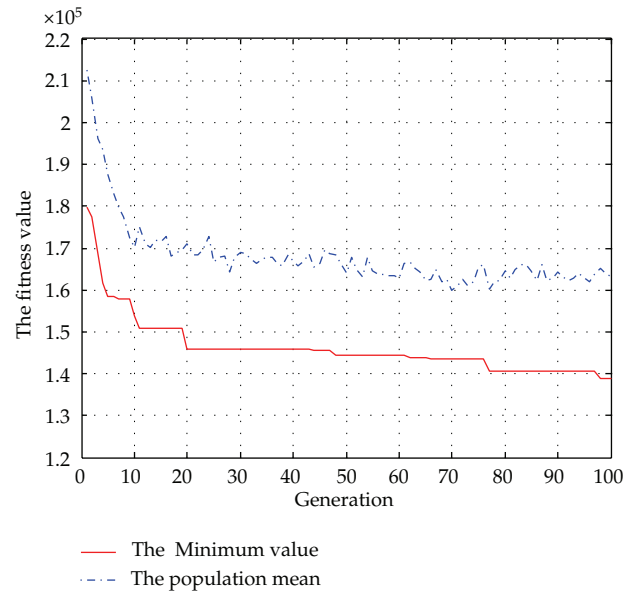


Figure 9: The solution of scenario VI.

should be determined according to social and economic characteristics of the region's inhabitants to provide a more powerful support for transportation network planning decision.

- (3) Scenario V and scenario VI show the network planning results under the target function of system expected total travel time under OD trip uncertainty. It shows

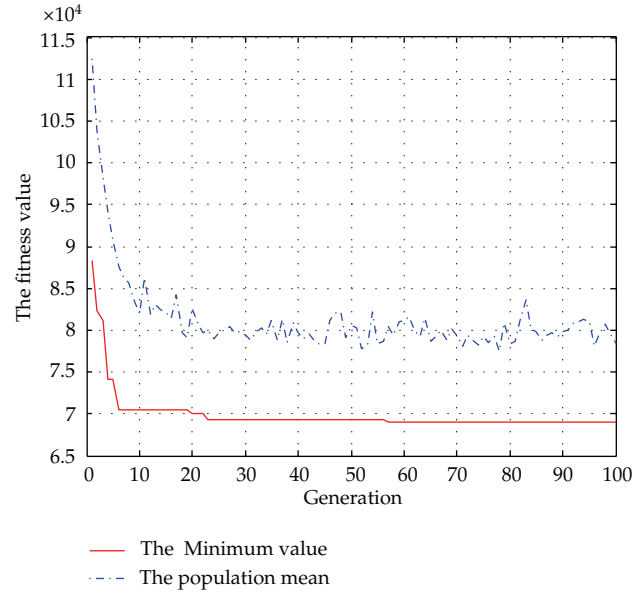


Figure 10: The solution of scenario VII.

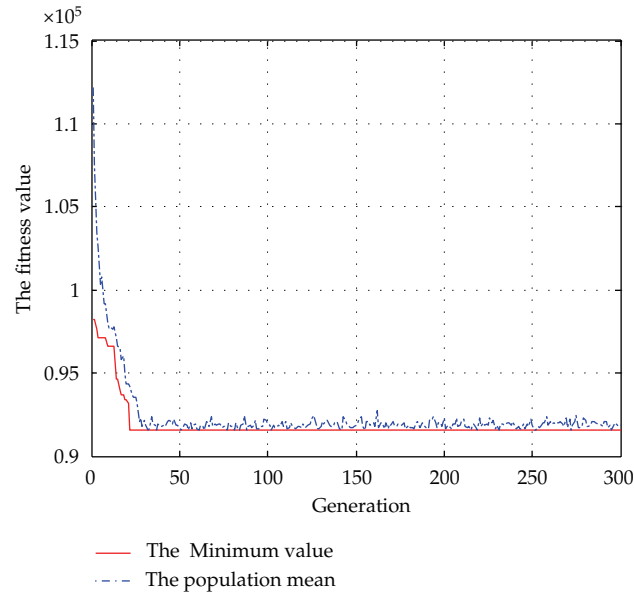


Figure 11: The solution of scenario VIII.

that the greater the OD demand uncertainty degree is, the greater the mean of the system total travel time is.

- (4) Scenario V and scenario VII have the same degree of OD trip uncertainty. Risk preference of decision makers influences the final network planning scheme.

Table 4: The solution of the sample.

Scenario	The fitter chromosome	Newly built roads	Extension roads	Fitness value
I	011101010110 0000101010000	21, 22, 23, 25	2, 4, 5, 11, 13, 15	$1.05 * 10^5$
II	100101010100 0001101010001	20, 23, 25	2, 4, 10, 11, 13, 15, 19	$1.07 * 10^5$
III	110101011100 0000100100010	20, 21, 23, 25	2, 3, 4, 11, 14, 18	$1.05 * 10^5$
IV	001101110100 0000011010100	22, 23, 25	1, 2, 4, 12, 13, 15, 17	$1.11 * 10^5$
V	110000001100 0100001100111	20, 21	3, 4, 8, 13, 14, 17, 18, 19	$1.14 * 10^5$
VI	000101010101 0001101100010	23, 25	2, 4, 6, 10, 11, 13, 14, 18	$1.39 * 10^5$
VII	101101010101 0000001010001	20, 22, 23, 25	2, 4, 6, 13, 15, 19	$6.91 * 10^4$
VIII	111100011100 0000001110000	20, 21 22, 23	2, 3, 4 13, 14, 15	$9.15 * 10^4$

6. Conclusions

In this paper, a discrete transportation network design problem is investigated, in which the trip generation flow and trip attraction flow are supposed as stochastic variables submitting to the given probability distribution. When travelers are divided into different groups by travel time value, a novel multiuser discrete network design model based on demand uncertainty is established. Genetic algorithm and Monte-Carlo simulation algorithm are integrated to solve the bi-level model for discrete network design. Calculation results on Nguyen Dupuis network show that user heterogeneity has a significant impact on network planning outcome under uncertain conditions. Furthermore, it can be found that GA is a potential tool for multiuser discrete transportation network design problem.

Acknowledgments

This work was supported by the Ph.D. Programs Foundation of Ministry of Education of China (20070003065), National High Technology Research and Development Program 863 (2007AA11Z202 and 2007AA11Z233).

References

- [1] H. Farvaresh and M. M. Sepehri, "A single-level mixed integer linear formulation for a bi-level discrete network design problem," *Transportation Research E*, vol. 47, no. 5, pp. 623–640, 2011.
- [2] E. Miandoabchi and R. Z. Farahani, "Optimizing reserve capacity of urban road networks in a discrete network design problem," *Advances in Engineering Software*, vol. 42, no. 12, pp. 1041–1050, 2011.
- [3] M. Patriksson, "Robust bi-level optimization models in transportation science," *Philosophical Transactions of the Royal Society of London A*, vol. 366, no. 1872, pp. 1989–2004, 2008.

- [4] S. V. Ukkusuri and S. T. Waller, "Linear programming models for the user and system optimal dynamic network design problem: formulations, comparisons and extensions," *Networks and Spatial Economics*, vol. 8, no. 4, pp. 383–406, 2008.
- [5] A. Karoonsoontawong and S. T. Waller, "Robust dynamic continuous network design problem," *Transportation Research Record*, vol. 2029, pp. 58–71, 2007.
- [6] C. F. Daganzo, "Stochastic network equilibrium with multiple vehicle types and asymmetric, indefinite link cost jacobians," *Transportation Science*, vol. 17, no. 3, pp. 282–300, 1983.
- [7] H. Yang, "Sensitivity analysis for the elastic-demand network equilibrium problem with applications," *Transportation Research B*, vol. 31, no. 1, pp. 55–70, 1997.
- [8] Z.-C. Li, W. H. K. Lam, S. C. Wong, and A. Sumalee, "Environmentally sustainable toll design for congested road networks with uncertain demand," *International Journal of Sustainable Transportation*, vol. 6, no. 3, pp. 127–155, 2012.
- [9] M. J. Smith, "The marginal cost taxation of a transportation network," *Transportation Research B*, vol. 13, no. 3, pp. 237–242, 1979.
- [10] H. Yang and X. Zhang, "Multiclass network toll design problem with social and spatial equity constraints," *Journal of Transportation Engineering*, vol. 128, no. 5, pp. 420–428, 2002.
- [11] R. B. Dial, "Bicriterion traffic assignment: basic theory and elementary algorithms," *Transportation Science*, vol. 30, no. 2, pp. 93–111, 1996.
- [12] J. Agrawal and T. V. Mathew, "Transit route network design using parallel genetic algorithm," *Journal of Computing in Civil Engineering*, vol. 18, no. 3, pp. 248–256, 2004.
- [13] B. Yu, Z. Z. Yang, and B. Yao, "An improved ant colony optimization for vehicle routing problem," *European Journal of Operational Research*, vol. 196, no. 1, pp. 171–176, 2009.
- [14] B. Yu, Z.-Z. Yang, P.-H. Jin, S.-H. Wu, and B.-Z. Yao, "Transit route network design using ant colony optimization," *Transportation Research C*, vol. 22, pp. 58–75, 2012.
- [15] B. Yu and Z. Z. Yang, "An ant colony optimization model: the period vehicle routing problem with time windows," *Transportation Research E*, vol. 47, no. 2, pp. 166–181, 2011.
- [16] F. Zhao and X. G. Zeng, "Simulated annealing-genetic algorithm for transit network optimization," *Journal of Computing in Civil Engineering*, vol. 20, no. 1, pp. 57–68, 2006.
- [17] S. B. Pattnaik, S. Mohan, and V. M. Tom, "Urban bus transit route network design using genetic algorithm," *Journal of Transportation Engineering*, vol. 124, no. 4, pp. 368–375, 1998.
- [18] M. Bielli, M. Caramia, and P. Carotenuto, "Genetic algorithms in bus network optimization," *Transportation Research C*, vol. 10, no. 1, pp. 19–34, 2002.
- [19] B. Yu, W. H. K. Lam, and M. L. Tam, "Bus arrival time prediction at bus stop with multiple routes," *Transportation Research C*, vol. 19, no. 6, pp. 1157–1170, 2011.
- [20] B. Yu, Z. Z. Yang, and C. T. Cheng, "Optimizing the distribution of shopping centers with parallel genetic algorithm," *Engineering Applications of Artificial Intelligence*, vol. 20, no. 2, pp. 215–223, 2007.
- [21] S. Nguyen and C. Dupuis, "Efficient method for computing traffic equilibria in networks with asymmetric transportation costs," *Transportation Science*, vol. 18, no. 2, pp. 185–202, 1984.

Research Article

Simulated Annealing-Based Ant Colony Algorithm for Tugboat Scheduling Optimization

Qi Xu,¹ Jun Mao,^{1,2} and Zhihong Jin¹

¹ Transportation Management College, Dalian Maritime University, Dalian 116026, China

² Dalian China Railway International Container Ltd., Dalian 116004, China

Correspondence should be addressed to Qi Xu, cogi@163.com

Received 5 September 2012; Accepted 23 September 2012

Academic Editor: Rui Mu

Copyright © 2012 Qi Xu 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.

As the “first service station” for ships in the whole port logistics system, the tugboat operation system is one of the most important systems in port logistics. This paper formulated the tugboat scheduling problem as a multiprocessor task scheduling problem (MTSP) after analyzing the characteristics of tugboat operation. The model considers factors of multianchorage bases, different operation modes, and three stages of operations (berthing/shifting-berth/unberthing). The objective is to minimize the total operation times for all tugboats in a port. A hybrid simulated annealing-based ant colony algorithm is proposed to solve the addressed problem. By the numerical experiments without the shifting-berth operation, the effectiveness was verified, and the fact that more effective sailing may be possible if tugboats return to the anchorage base timely was pointed out; by the experiments with the shifting-berth operation, one can see that the objective is most sensitive to the proportion of the shifting-berth operation, influenced slightly by the tugboat deployment scheme, and not sensitive to the handling operation times.

1. Introduction

Container terminal is an important part in international logistics and plays a significant role in world trade. Recently, more and more people become to recognize the importance of global logistic business via container terminals. As the throughput of containers in container terminal increases and competition between ports becomes fierce, how to improve the efficiency in container terminal has become an important and immediate challenge for port managers. One of the most important performance measures in container terminals is to schedule all kinds of equipment at an optimum level and to reduce the turnaround time of vessels. Tugboat is one such kind of vital equipments in container terminal.

The performance of the tugboat operation scheduling has a direct influence on time when a ship can start its handling operation and when a ship can leave the port. Scheduling

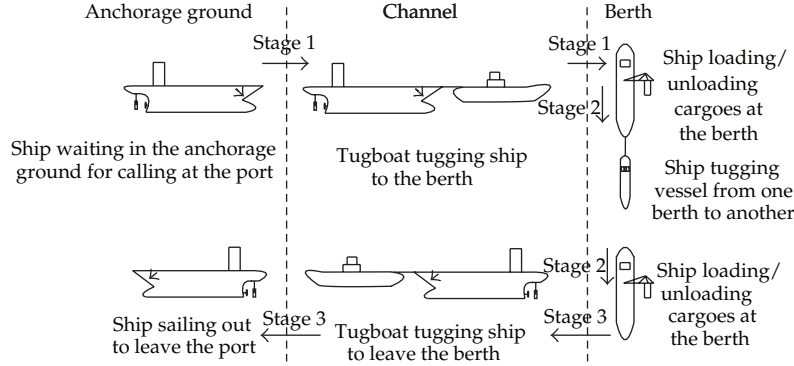


Figure 1: Illustration of a typical tugboat operation process.

on tugboats with good performance may lower the turnaround of ships in a port. Thus the tugboat scheduling problem is an important one to be solved in the field of the port logistics.

When ships arrive at a port, if their target berths are not available immediately, they cannot enter into the berths directly and have to wait in the anchorage ground. Then they have to be tugged by certain amount of tugboats according to some rules. Moreover, the moving between two berths and the department of vessels also need to be tugged by tugboats. To improve the ship operation efficiency, tugboats should be scheduled at an optimum level.

According to the analysis mentioned above, the three types of service that a tugboat can provide are (a) tugging coming ships to the berth (viz., berthing); (b) tugging ships from one berth to another (namely, shifting-berth); (c) tugging ships leaving the berth (viz., unberthing). Not every ship will experience all the three types of services. That is, the shifting berth operation is not necessary, while the berthing and unberthing operations are necessary for all ships.

A typical tugboat operation process is illustrated in Figure 1. As Figure 1 shows, the duration from the time when a tugboat starts tugging a ship to the finishing time of the berthing operation is treated as stage 1, the duration when a tugboat starts tugging the exact ship leaving the first berth to the finishing time when that ship enter into the second target berth is treated as stage 2, and the duration from the starting time of the unberthing operation to the time when the ship leaves the port is looked upon as stage 3.

Practically, tugboat scheduling managers allocate suitable tugboats to ships according to their length. Each ship can have one or more tugboats serving for it simultaneously by the scheduling rules.

The main idea of the scheduling rules is that big ships should be served by big tugboats (as with the horsepower), and small ships should be served by small tugboats; if more than one tugboat with the same horsepower are available, the allocation among the available tugboats is made by some heuristic rules.

For example, there are six types of tugboats in a port according to the horsepower unit, such as 1200PS, 2600PS, 3200PS, 3400PS, 4000PS, and 5000PS. The scheduling rules for allocating are as follows:

(a) S1 (less than 100 meter): 1200PS (or bigger)*1,

(b) S2 (100–200 meter): 2600PS (or bigger)*2,

- (c) S3 (200–250 meter): 3200PS (or bigger)*2,
- (d) S4 (250–300 meter): 3400PS (or bigger)*2,
- (e) S5 (greater than 300 meter): 4000PS (or bigger)*2.

And the heuristics rules concluded from real-world practice include

- (a) TSD rule: choosing the tugboat with the shortest distance from the scheduled ship to serve for it;
- (b) FAT rule: choosing the tugboat which is the first available one for the scheduled ship;
- (c) UWAT rule: from the perspective of balancing all tugboats' working amount, choosing the tugboat with the minimum working amount up till now to serve for the scheduled ship.

According to the hybrid flow shop theory, the tugboat scheduling can be considered as a multiprocessor task scheduling problem (MTSP) with 3 stages. In the scheduling system, tugboats are taken as movable "machines," and ships have to experience the berthing, shift-berth (if there exists this operation), and unberthing operations operated by tugboats sequentially.

On the other hand, compared with a typical MTSP, the tugboat scheduling problem has its own characteristics. Firstly, the exact same tugboat can provide all the three types of service (berthing, shifting berth, and unberthing), which means that the machine set for all the three stages is the same. This is different from a typical MTSP in which the available machine set in each stage is not the same. Besides, not all ships have to experience the shift-berth operation, which makes the problem different from a typical MTSP with the characteristics that all jobs have to experience all the stages.

Anyway, the tugboat scheduling problem is a kind of unconventional scheduling problem, an NP-hard problem which cannot be solved by exact methods. Some scholars have begun to make research on the topic.

Ying and Lin [1] proposed the ant colony approach to solve the MTSP. Xuan and Tang [2] explored the complexity of the MTSP and designed a Lagrange relaxation algorithm combined with heuristic rules to solve the MTSP. Liu [3] established a mathematical model on the tugboat scheduling problem combined with the MTSP theory and adopted the hybrid evolutionary strategy to solve the model. Liu [4] established an tugboat scheduling model considering the minimum operation distance of the tugboats, and compared the performance of hybrid evolutionary strategy with the particle swarm optimization algorithm for solving the addressed problem. Wang and Meng [5] used a hybrid method that combined ant colony optimization and genetic algorithm to resolve the tugboat allocation problem. Wang et al. [6] formulated a mix-integer model for the tugboat assignment problem combined with the existing scheduling rules and analyzed the effects of the number and service capacity of tugboats on the turnaround time of ships. Liu and Wang [7] considered the tugboat operation scheduling problem as a parallel machine scheduling problem with special process constraint and employed a hybrid algorithm based on the evolutionary strategy to solve the problem. Dong et al. [8] adopted the improved particle swarm optimization combined with dynamic genetic operators to solve the formulated tugboat dispatch model. Liu and Wang [9] used the particle swarm optimization algorithm combined with the local search approach to solve the tugboat scheduling model they proposed.

As we can see from the previous research, scholars have begun to use many approaches to solve the tugboat scheduling problem, including the genetic algorithm, ant

colony optimization, and particle swarm optimization. However, most literature only considers the situation of single operation stage and single anchorage base and neglects the influence of the tugboats' and ships' location information on the problem difficulty. That makes the model formulated far from reality. Thus, this paper will make research on tugboat scheduling problem considering multi-anchorage bases, different operation modes, and three operation stages.

The rest of paper is organized as follows. Section 2 formulates a tugboat scheduling model combined with the MTSP theory. Section 3 proposes a simulated annealing-based ant colony algorithm to solve the formulated model, and Section 4 discusses the simulation experiments using ACO in container terminals. Finally, we make conclusions and introduce the future work in Section 5.

2. Model Formulation

2.1. Assumptions

The following assumptions are introduced for the formulation of the problem.

- (a) The planning horizon is one day.
- (b) Three operation stages (i.e., berthing, shifting-berth, and unberthing) are taken into consideration, but not all ships have to experience the shifting-berth operation. For ship which does not have to experience the second operation, assume there is a virtual shifting-berth operation and the operation time for that is zero.
- (c) The ready times for all the tugboats are 0, and all the tugboats are at the anchorage bases at time 0; all the ships to be served have arrived at the anchorage ground at time 0.
- (d) There are three types of locations in a port: berths for ships to load/unload cargoes, meeting locations where ships meet tugboats at the entrance of port, and the anchorage bases.
- (e) Two operation modes (restricted cross-operation mode RCOM and unrestricted cross operation mode UCOM) may be adopted to schedule the tugboats in a port.
- (f) All the ships enjoy the same precedence.
- (g) The scheduling rules for allocating tugboats to ships are what we mentioned in Section 1.
- (h) The sailing speeds of all tugboats whenever sailing are the same.
- (i) The tugboats may return to the anchorage base during the planning horizon according to the scheduling plans.

In assumption (e), the RCOM means that all anchorage bases have their fixed service area in the port, which means that each tugboat in every base can only operate in its corresponding service area, while the UCOM means that all tugboats can operate in the whole area of a port.

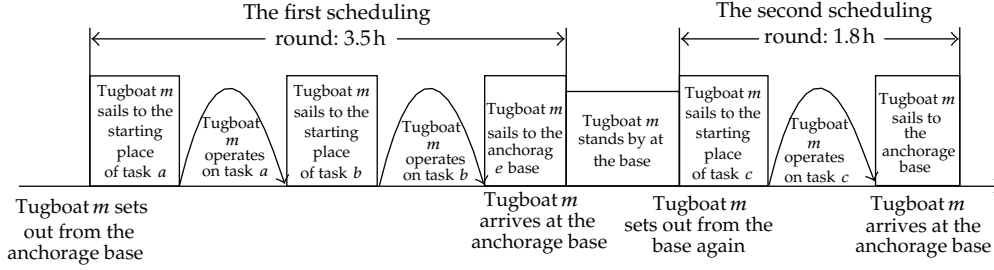


Figure 2: Illustration of the tugboat scheduling rounds.

2.2. Definition of the Scheduling Round

Before the tugboat scheduling model is formulated, a concept named scheduling round should be introduced first.

In practice, a scheduling round is used to define the duration from the time when a tugboat leaves for its target place from the anchorage base to the time when it returns to the base after finishing a certain amount of tasks (may be one task, maybe more than one). As Figure 2 illustrates, tugboat m has to operate on three tasks (i.e., a , b , c) in the planning horizon: after finishing the task a , the tugboat sails directly to the starting place of task b and sails back to the anchorage after finishing the task b . That duration can be defined as the first scheduling round of tugboat which lasts for 3.5 hours. On arriving at the anchorage base, the tugboat stands by until it sails to the starting place of task c . After finishing the task c , m sails back to the base again. And that duration from the time when m leaves the base again to the time when it arrives at the base is the second scheduling round which lasts for 1.8 hours. According to the definition, two scheduling rounds occur as to tugboat m in the planning horizon, and the total duration for the two scheduling rounds for tugboat m is $3.5 \text{ h} + 1.8 \text{ h} = 5.3 \text{ h}$.

2.3. Notations

(a) Parameters

j, l : Stage index, $j, l \in J = \{1, 2, 3\}$, in which 1–3 represent berthing, shifting-berth, and unberthing operations

i, k : Ship index

cy_i : The descriptive binary parameter that illustrates whether ship i will experience the shifting-berth operation (if $cy_i = 1$, it means that ships i will experience the shifting-berth operation, otherwise will not experience the operation)

m : Tugboat index

M : The set of all the tugboats

ta_m : Style of tugboat m (which may be 1–6, representing 1200PS, 2600PS, 3200PS, 3400PS, 4000PS, and 5000PS, resp.)

N : The set of all ships, $N = \{1, 2, \dots, n\}$

S_i : Style of ship i

set_i : Set of tugboat style which can provide the related service for ship i

O_{ij} : Operation of ship i at stage j

CM_b : Set of tugboats in the anchorage base b ($b \in B$, B is the set of all the anchorage bases); thus we can get $\bigcup_{b \in B} CM_b = M$

M_{ijb} : Set of tugboats in base b that can serve for operation O_{ij} based on the scheduling rules; thus the set of tugboats in all the bases that can serve for operation O_{ij} can be expressed as $M_{ij} = \bigcup_{b \in B} M_{ijb} = \{m \mid ta_m = set_i, \forall m \in CM_b\}$

E_{jm} : The set of ships that might be served by tugboat m at stage j

LOS_{ij} : Location where operation O_{ij} starts (if $j = 1$, LOS_{ij} is the meeting place where ship i meets tugboat at the entrance of the port; else if $j = 2$, LOS_{ij} is the first berth where ship i loads/unloads its cargo; else if $j = 3$, LOS_{ij} is the second berth where ship i loads/unloads its cargo, while $LOS_{i3} = LOS_{i2}$ if $cy_i = 0$)

LOF_{ij} : Location where operation O_{ij} finishes (if $j = 1$, LOF_{ij} is the first berth where ship i loads/unloads its cargo; else if $j = 2$, LOF_{ij} is the second berth where ship i loads/unloads its cargo, while $LOF_{i2} = LOF_{i1}$; else if $j = 3$, LOF_{ij} is the meeting place where ship i meets tugboat at the entrance of the port)

$ST(a, b)$: Duration for sailing between locations a and b

p_{ij} : Processing time of operation O_{ij}

tb_i : Sailing time of ship i from the waiting place to the berthing place, and $tb_i = ST(LOS_{i1}, LOF_{i2})$

te_i : Berthing time of ship i at berth

toa_i : Duration of ship i for loading and unloading cargoes at the first target berth

tob_i : Duration of ship i for loading and unloading cargoes at the second target berth (if there exists a shifting-berth operation)

tu_i : Unberthing time of ship i at berth

tl_i : Sailing time of ship i from the unberthing place to the place where ship i leaves the port

s_{ijkl}^m : Set-up time between task O_{ij} and O_{kl} by tugboat m

bp_m : The anchorage base where tugboat m belongs

H : A sufficiently large constant.

(b) *Decision Variables*

$$\begin{aligned}
x_{ijm} &= \begin{cases} 1, & \text{if } O_{ij} \text{ is assigned to tugboat } m \\ 0, & \text{otherwise,} \end{cases} \\
y_{ijkl}^m &= \begin{cases} 1, & \text{if } O_{ij} \text{ and } O_{kl} \text{ are assigned to the same tugboat } m \\ 0, & \text{otherwise,} \end{cases} \\
u_{ijkl}^m &= \begin{cases} 1, & \text{if } O_{ij} \text{ precedes } O_{kl} \text{ (not necessarily immediately) on tugboat } m \\ 0, & \text{otherwise,} \end{cases} \\
z_{ijkl}^m &= \begin{cases} 1, & \text{if } O_{ij} \text{ immediately precedes } O_{kl} \text{ on tugboat } m \\ 0, & \text{otherwise,} \end{cases} \\
w_{ijm} &= \begin{cases} 1, & \text{if tugboat } m \text{ goes back to the anchorage base after completing operation } O_{ij} \\ 0, & \text{otherwise.} \end{cases}
\end{aligned} \tag{2.1}$$

(c) *State Variables Decided by Decision Variables*

TS_{ij}: The starting time of O_{ij}

TF_{ij}: The finishing time of O_{ij}

BT_m: The setting-out time of tugboat m from its anchorage base in the planning horizon

FT_m: The returning time of tugboat m after finishing its last task in the planning horizon

sh_{mh}: The duration of the h th scheduling round for tugboat m in the planning horizon

g_m: Number of the scheduling rounds for tugboat m in the planning horizon.

2.4. Model(a) *Objective*

In this paper, the objective is to minimize the total operation times of tugboats, which can be equal to the total duration for all the scheduling rounds of all tugboats. Thus we have to derive the calculation method for scheduling rounds.

From the definition of the scheduling round, the relation between the decision variable (w_{ijm}) and the quantity of scheduling rounds in the planning horizon (g_m) can be concluded as follow:

$$g_m = |\{w_{ijm} \mid w_{ijm} = 1, \forall i \in N, \forall j \in J\}|. \tag{2.2}$$

Equation (2.2) means that the value of g_m equals the times for which tugboat m returns to the anchorage base.

Define the set of tasks right before which tugboat m returns to the base as OS_m , and all the tasks in OS_m are ordered by the operation sequence. By that definition, we come to know the calculation method for duration of each scheduling round of tugboat m as follows:

$$\begin{aligned}
sh_{m1} &= TF_{OS_m\{1\}} + ST(LOF_{OS_m\{1\}}, bp) - BT_m \\
sh_{m2} &= TF_{OS_m\{2\}} + ST(LOF_{OS_m\{2\}}, bp) - (TS_{ij} - ST(bp, LOS_{ij})) \\
&\quad \left\{ (i, j) \mid z_{ijkl}^m = 1, (k, l) = OS_m\{1\} \right\} \\
&\quad \vdots \\
sh_{mh} &= TF_{OS_m\{h\}} + ST(LOF_{OS_m\{h\}}, bp) - (TS_{ij} - ST(bp, LOS_{ij})) \\
&\quad \left\{ (i, j) \mid z_{ijkl}^m = 1, (k, l) = OS_m\{h-1\} \right\} \\
&\quad \vdots \\
sh_{mg_m} &= TF_{OS_m\{g_m\}} + ST(LOF_{OS_m\{g_m\}}, bp) - (TS_{ij} - ST(bp, LOS_{ij})) \\
&\quad \left\{ (i, j) \mid z_{ijkl}^m = 1, (k, l) = OS_m\{g_m-1\} \right\}.
\end{aligned} \tag{2.3}$$

Equation (2.3) reveals the duration of each scheduling round equals the finishing time when tugboat completes its last task in the scheduling round plus the sailing time from the location where the last task of tugboat is completed to the tugboat's anchorage base minus the time when tugboat begins its first task in the scheduling round minus the sailing time from the base to the location where the first task starts.

As it has been discussed before, the total operation times of tugboats are equal to the total duration for all the scheduling rounds of all tugboats. Thus the objective function can be expressed as follows:

$$\text{Minimize } F = \sum_{m \in M} \sum_{h \in g_m} sh_{mh}. \tag{2.4}$$

(b) Constraints

The constraints in the proposed model include the following equations:

$$TS_{ij} \geq 0, \quad \forall i \in N, \quad \forall j \in J, \tag{2.5}$$

$$TS_{i1} + p_{i1} + toa_i \cdot cy_i \leq TS_{i2}, \quad \forall i \in N, \tag{2.6}$$

$$TS_{i2} + p_{i2} + tob_i \cdot cy_i + toa_i \cdot (1 - cy_i) \leq TS_{i3}, \quad \forall i \in N,$$

$$\begin{aligned}
&\sum_{m \in M} x_{ijm} = 1, \quad S_i = S1, \\
&\sum_{m \in M} x_{ijm} = 2, \quad \text{otherwise}, \quad \forall i \in N, \quad \forall j \in J,
\end{aligned} \tag{2.7}$$

$$\begin{aligned}
\text{set}_i &= \{1, 2, 3, 4, 5, 6\} \quad S_i = S1, \\
\text{set}_i &= \{2, 3, 4, 5, 6\} \quad S_i = S2, \\
\text{set}_i &= \{3, 4, 5, 6\} \quad S_i = S3, \\
\text{set}_i &= \{4, 5, 6\} \quad S_i = S4, \\
\text{set}_i &= \{5, 6\} \quad S_i = S5,
\end{aligned} \tag{2.8}$$

$$y_{ijkl}^m \leq 0.5(x_{ijm} + x_{klm}) \leq y_{ijkl}^m + 0.5, \quad \forall i, k \in E_{jm}, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \quad \forall j, l \in J, \tag{2.9}$$

$$y_{ijkl}^m = y_{klij}^m, \quad \forall i, k \in E_{jm}, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \quad \forall j, l \in J,$$

$$u_{ijkl}^m + u_{klij}^m = y_{ijkl}^m, \quad \forall i, k \in E_{jm}, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \quad \forall j, l \in J, \tag{2.10}$$

$$u_{ijkl}^m - z_{klij}^m \geq 0, \quad \forall i, k \in E_{jm}, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \quad \forall j, l \in J, \tag{2.11}$$

$$\begin{aligned}
\sum_{k \in E_{jm}} z_{ijkl}^m &\leq 1, \quad \forall i \in E_{jm}, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \quad \forall j, l \in J, \\
\sum_{k \in E_{lm}} z_{klij}^m &\leq 1, \quad \forall i \in E_{jm}, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \quad \forall j, l \in J,
\end{aligned} \tag{2.12}$$

$$\text{TS}_{ij} + p_{ij} + s_{ijkl}^m \leq \text{TS}_{kl} + H(1 - z_{ijkl}^m), \quad \forall i, k \in N, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \quad \forall j \in J, \tag{2.13}$$

$$\text{TS}_{kl} + p_{kl} + s_{klij}^m \leq \text{TS}_{ij} + H(1 - z_{klij}^m), \quad \forall i, k \in N, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \quad \forall j \in J,$$

$$p_{ij} = tb_i + te_i, \quad j = 1,$$

$$p_{ij} = (tu_i + \text{ST}(\text{LOS}_{i2}, \text{LOF}_{i2}) + te_i) \cdot cy_i, \quad j = 2 \quad \forall i \in N, \tag{2.14}$$

$$p_{ij} = tu_i + tl_i, \quad j = 3,$$

$$s_{ijkl}^m = \text{ST}(\text{LOF}_{ij}, \text{LOS}_{kl}) \cdot z_{ijkl}^m \quad \forall i, k \in N, \quad \forall j, l \in J, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \tag{2.15}$$

$$w_{ijm} \cdot H \geq z_{ijkl}^m \cdot [(\text{TS}_{kl} - \text{TF}_{ij}) - (\text{ST}(\text{LOF}_{ij}, bp_m) + \text{ST}(bp_m, \text{LOS}_{kl}))],$$

$$\forall i, k \in E_{jm}, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \quad \forall j, l \in J,$$

$$w_{ijm} < \left(z_{ijkl}^m \cdot \frac{\text{TS}_{kl} - \text{TF}_{ij}}{2 \times (\text{ST}(\text{LOF}_{ij}, bp_m) + \text{ST}(bp_m, \text{LOS}_{kl}))} + 0.5 \right), \tag{2.16}$$

$$\forall i, k \in E_{jm}, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \quad \forall j, l \in J,$$

$$x_{ijm}, y_{ijkl}^m, u_{ijkl}^m, z_{ijkl}^m, w_{ijm} = 0 \text{ or } 1, \quad \forall i, k \in N, \quad \forall m \in \bigcup_{b \in B} M_{ijb}, \quad \forall j, l \in J. \tag{2.17}$$

Constraint (2.5) guarantees that each operation begins after time zero. Constraint (2.6) ensures that for every ship, the shifting-berth operation begins only after the berthing and handling operations are completed, and the unberthing operation begins only after the shifting-berth and handling operations are completed. Constraint (2.7) means that if the style of the ship is 1, only one tugboat is needed; otherwise, two tugboats are needed. Constraint (2.8) defines the available set of tugboat style which can serve for ship i according to the scheduling rules. Constraint (2.9) defines $y_{ijkl}^m = y_{klij}^m = 1$ when $x_{ijm} = x_{klm} = 1$. Constraint (2.10) guarantees that every tugboat can only serve for one operation at any time. Constraint (2.11) is set to make sure that $u_{ijkl}^m = 1$ when $z_{ijkl}^m = 1$. Constraint (2.12) guarantees that there are at most one predecessor and successor for operation O_{ij} on tugboat m . Constraint (2.13) simultaneously determines that the starting time of any operation has to be after the time when its immediately preceding task finishes. Constraint (2.14) defines the processing time for each task. Constraint (2.15) defines the set-up time for each operation O_{ij} . Constraint (2.16) simultaneously determines when tugboat m should return to the anchorage base: if the sum of the sailing time from the finishing place of O_{ij} to the base and the sailing time from the base to the starting place of O_{ij} 's successor task on tugboat m (i.e., O_{kl}) is less than the time cost if m directly sails to O_{kl} 's starting place and waits there until the task begins, then tugboat m should return to the base; otherwise, m should sail directly to O_{kl} 's starting place. Constraint (2.17) specifies the binary property of the decision variables.

3. Proposed Hybrid Algorithm (PHA)

3.1. The Basic Idea of the Ant Colony Algorithm

Ant colony metaheuristic is a concurrent algorithm in which a colony of artificial ants cooperates to find optimized solutions of a given problem (see Boveiri [10]). The ant algorithm was first proposed by Dorigo et al. [11] as a multiagent approach to the traveling salesman problem (TSP), and it has been utilized successfully to many difficult discrete optimization n problems such as job shop scheduling, vehicle routing, graph coloring, sequential ordering, and network routing.

The inspiring natural process of ACS is the foraging behavior of ants. A colony of ants can identify the shortest pathway from a food source to their anthill without using visual cues; they communicate through an aromatic substance, called pheromone. While walking, ants secrete pheromone on the ground and follow, in probability, the pheromone previously laid by other ants. Ants are more likely to follow pathways marked by a larger accumulation of pheromone from other ants that have previously walked that route. Since ant searching a food source by shorter pathways will come back to the anthill sooner than ants traveling via longer pathways, the shorter pathways will have a higher traffic density than those of the longer ones. Hence, the pheromone accumulation will build up more rapidly on shorter pathways than on longer ones. Consequently, the fast accumulation of pheromone on the shorter pathways will cause ants to quickly choose the shortest routes. The described foraging behavior of ants can be used to solve scheduling problems by simulation: the objective value (e.g., flow time) corresponds to the quality of the food source (e.g., distance), artificial ants searching for the solution space simulate real ants searching for their environment, and an adaptive memory corresponds to the pheromone trail. In addition, the artificial ants are equipped with a local heuristic function to guide their search through the set of feasible solutions [1].

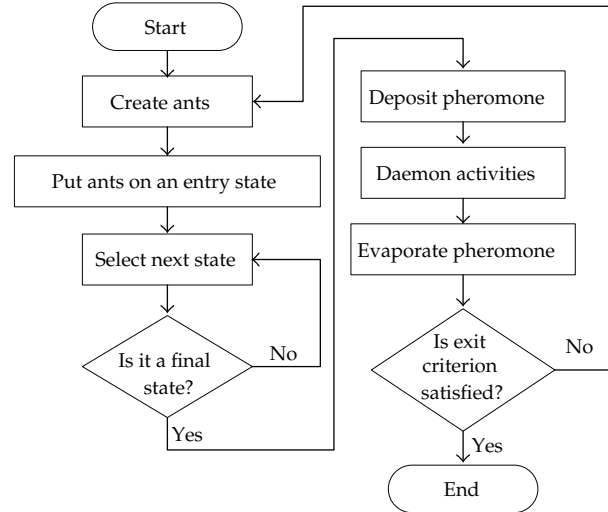


Figure 3: Flowchart of ant colony optimization.

The main procedure of the ant colony algorithm is as follows.

- (a) Generate ant (or ants).
- (b) Loop for each ant (until complete scheduling of tasks).
 - (i) Select the next task with respect to pheromone variables of ready tasks.
- (c) Deposit pheromone on visited states.
- (d) Daemon activities.
- (e) Evaporate pheromone.

The flowchart of ant colony algorithm is illustrated as Figure 3.

However, the solutions were generated by each ant in the basic ant colony algorithm by random, and those solutions may not be the optimal solutions or satisfactory solutions. That makes the updating of the pheromone be done by random too, which may cause a lot of time costs to get the optimal value, and that value may also be the local optima. To avoid that phenomenon, the diversity of the population should be considered.

By that thought, we introduce the simulated annealing into the ant colony algorithm, which can guarantee the quality of the search and avoid the phenomenon of the local optima. Thus, the simulated annealing-based ant colony algorithm is proposed.

3.2. Procedure of the Proposed Algorithm

According to the analysis above, we introduce a simulated annealing-based ant colony algorithm to solve the formulated tugboat scheduling problem. The basic procedure of the algorithm is as Figure 4. In the algorithm, the ACO performs the role of simulation, while the simulated annealing algorithm performs the role of searching for global optimization.

Step 1. Generate the initial tugboat scheduling plans (individuals) which act as representing codes for the simulated annealing algorithm.

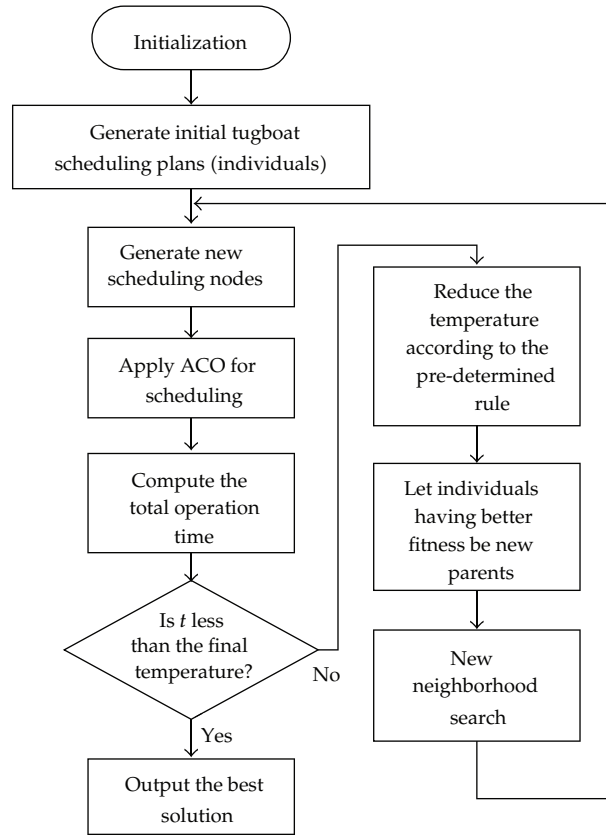


Figure 4: Flowchart of the proposed algorithm.

Step 2. Generate new scheduling nodes used to apply for the ant colony algorithm.

Step 3. Apply the ant colony algorithm for the scheduling process.

Step 4. Compute the total operation time for all tugboats in the planning horizon as the key indicator for the system.

Step 5. If the current temperature is less than the final temperature, then go to *Step 9*; else go to *Step 6*.

Step 6. Reduce the temperature according to the predetermined rule.

Step 7. Let the individuals having better fitness be new parents.

Step 8. Based on the new parents, perform a new neighborhood search to get the new individuals.

Step 9. Output the best solution.

3.3. Key Operations of the Algorithm

3.3.1. Ant Colony Optimization (ACO)

In the ant colony algorithm for solving the proposed problem, jobs are defined as ants and resources are defined as nodes. The main procedure of ant colony optimization has been discussed in Section 3.1, and in this section, two key operations of the ACO (i.e., initialization of ants and updating of the pheromone) will be introduced.

(a) Initialization of Ants

According to the algorithm, a certain amount of ants have to be generated. In order to make the schedules by which ants travel satisfy the requirements of the scheduling system, three arrays were set in the algorithm: *tour* which represents tasks not yet operated; *ournext* which represents the tasks to be operated in the next step; *visited* which represents tasks having been operated. All the ants can only choose the tasks for the next operation from the *ournext* array, so that the feasibility of the schedule traveled by ants can be guaranteed. Then, we just need to judge whether all the tasks have been traveled. Then the schedule generated by ants in the *visited* is the schedule we want.

The selection of nodes during the algorithm is referenced by the roulette wheel. Thus the state transit rule can be concluded as.

$$p_{ij} = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha [1/T_{ij}]^\beta}{\sum_{s \in \text{allowed}} [\tau_{ij}(t)]^\alpha [1/T_{ij}]^\beta} \\ 0. \end{cases} \quad (3.1)$$

In (3.1), T_{ij} means the processing time of ship i by tugboat j , *allowed* = *ournext*.

(b) Updating of the Pheromone

After all ants of a generation have traveled all the tasks, compute the total operation times of the tugboats and update the pheromone according to those values. In our research, we choose the five ants with the minimal operation times to update the pheromone, and the updating rules are as follows:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + V_{\tau_{ij}}, \quad (3.2)$$

$$V_{\tau_{ij}} = \frac{Q}{T_{\min}}. \quad (3.3)$$

In (3.2), ρ means the evaporation coefficient, T_{\min} in (3.3) means the minimal operation times of all ants in the current generation, and Q is the quantity of pheromone in the unit path.

3.3.2. Simulated Annealing (SA)

The key operations in the simulated annealing include individual coding, initial individuals' generation, and the neighborhood search scheme.

Ship index	→ 2	2	3	2	1	3	4	3	1	4	1	4
Tugboat 1	→ 1	0	3	1	2	1	2	1	0	2	2	1
Tugboat 2	→ 0	0	2	0	0	3	3	2	0	3	0	3
Whether tugboat 1 returns to the base after the task	→ 1	0	0	0	1	0	0	1	0	1	1	1
Whether tugboat 2 returns to the base after the task	→ 0	0	1	0	0	1	1	0	0	1	0	1

Figure 5: Illustration of coding for an individual.

(a) Individual Coding

In this paper, the real integer method is adopted to code for an individual. As every ship may experience at most 3 stages of operation, we set the number of columns as three times of the number of ships. Assume that there are 4 ships to be served (ship 1, 2 do not have to shift a berth, while ship 3, 4 will experience a shifting-berth operation) and 3 available tugboats, then the coding expression of the individuals should be a $5 \times (3 \times 6)$ matrix, which can be illustrated as Figure 5.

The first row of the coding representation means the service order for ships, and the next two rows are the indexes of tugboats serving for ships in the first row. Note that each index appears three times in the first row: if it is the first time an index appears, it means that the ship is berthing; for the second time it appears, it may be a virtual or real shifting-berth operation; otherwise, the unberthing operation. The fourth and fifth rows are descriptive parts which tell us whether tugboat 1 and 2 return to the base after finishing the task.

As ship 1 and 2 do not have to shift a berth, the virtual shifting-berth operations are proposed to keep the total operations three times of the number of ships. That can be illustrated as the shadow parts with diagonal lines in Figure 5. Besides, if the ship style is 1, then an index of tugboat is generated from the available tugboat set to fill in the corresponding second row, and the third row is zero (as shown in the shadow parts with grids); otherwise, two indexes of tugboats are generated to fill in the two rows. Thirdly, as all tugboats have to return to the base after finishing their last tasks, the corresponding symbols in the fourth or fifth rows should be 1 (as the shadow parts with dots).

According to that individual coding, the service order for ships in Figure 5 is as follows: ship 2 (berthing)—ship 2 (virtual shifting-berth)—ship 3 (berthing)—ship 2 (unberthing)—ship 1 (berthing)—ship 3 (shifting-berth)—ship 4 (berthing)—ship 3 (unberthing)—ship 1 (virtual shifting-berth)—ship 4 (shifting-berth)—ship 1 (unberthing)—ship 4 (unberthing). The tugboat providing the berthing service for ship 2 is tugboat 1, and after finishing the berthing service for ship 2, tugboat 1 returns to the anchorage base, and so on.

(b) Initial Individuals' Generation

The procedure for generating the initial schedule can be described as Figure 6.

As we can see from Figure 6, the procedure for the initial individuals' generation mainly include three parts: randomly generating the service order for ships; allocating the tugboat serving for ships; deciding whether tugboats should return to the base after completing the operation.

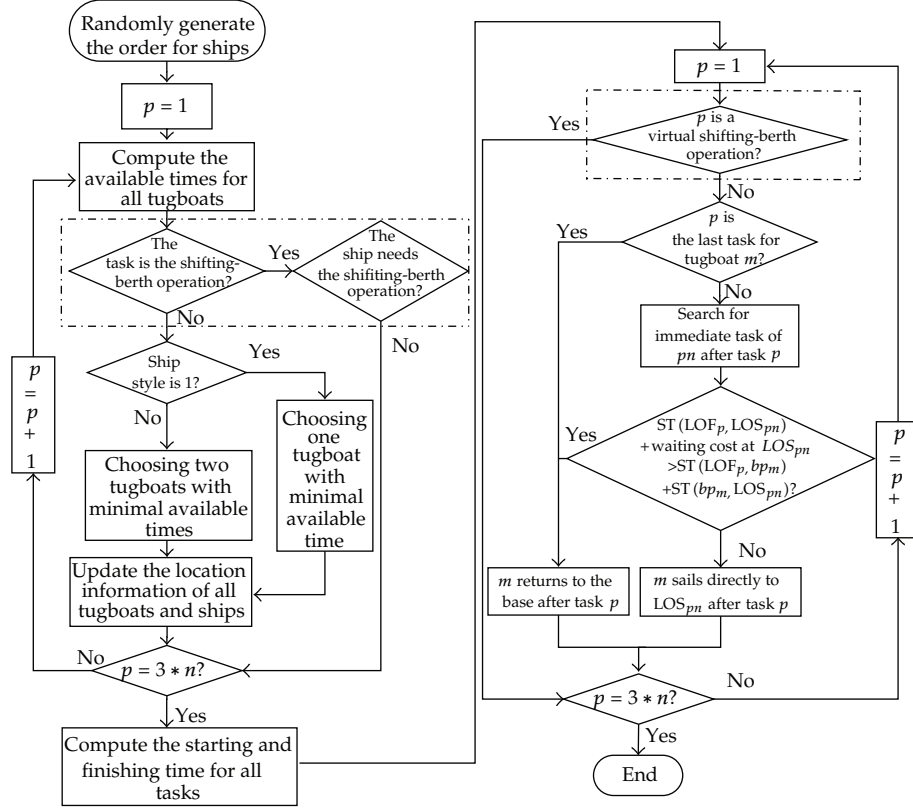


Figure 6: The generation procedure of the initial individuals.

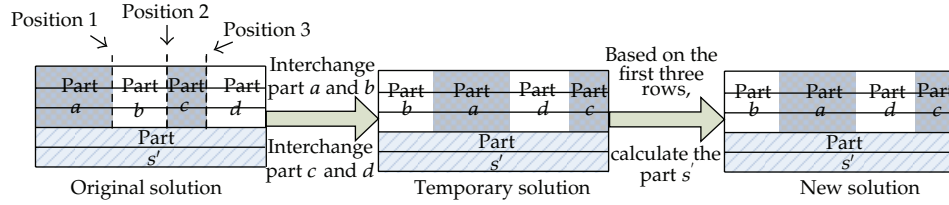


Figure 7: The neighborhood search scheme.

(c) Neighborhood Search Scheme

The procedure for the neighborhood search scheme can be concluded as Figure 7.

Given a solution p , a neighbor of p can be obtained by using the three-point interchanging scheme proposed in this section. The main idea is as follows: randomly generate three positions in the original solution, so that the original solution is divided into five parts; let a, b, c, d, s be the four partial solutions of p ; a temporary solution is obtained by interchanging a and b, c and d ; based on the three rows of the temporary solution, calculate part s' according to the rules expressed by (2.16).

Position 1			Position 2			Position 3					
2	2	3	2	1	3	4	3	1	4	1	4
1	0	3	1	2	1	2	1	0	2	2	1
0	0	2	0	0	3	3	2	0	3	0	3
1	0	0	0	1	0	0	1	0	1	1	1
0	0	1	0	0	1	1	0	0	1	0	0

The task column for ship 2's berthing operation			The task column for ship 2's unberthing operation			The virtual shifting-berth operation is behind the unberthing operation			The temporary solution generated by the three-point interchange		
3	2	1	2	2	4	3	4	3	1	4	1
3	1	2	1	0	1	1	2	1	0	2	2
2	0	0	0	0	3	3	3	2	0	3	0
1	0	0	0	1	0	0	1	0	1	1	1
0	0	1	0	0	1	1	0	0	1	0	0

Figure 8: The infeasible solution generated by the three-point interchange.

However, during the neighborhood search process, the temporary solution may be an infeasible solution. For example, the virtual shifting-berth operation (the shadow parts in Figure 8) is after the unberthing operation, which is infeasible.

Thus it is necessary to modify the temporary solution. Steps for modifying the temporary solution are as follows.

Step 1. Initialize $p = 1$.

Step 2. Judge if the second and third rows of the p th column are both zero.

- (a) If both the values are zero, which means that the task in the p th column is a virtual shifting-berth operation,
 - (i) search for two columns: one for the berthing operation for ship served in the p th column; one for the unberthing operation for the same ship. Define the places of the two columns as $p1$ and $p2$,
 - (ii) if p is less than $p1$, interchange values of the first three rows in the two columns, then go to Step 3,
 - (iii) if p is larger than $p2$, interchange values of the first three rows in the two columns, then go to Step 3.
- (b) If the two values are not both zero, then go to Step 3.

Table 1: Sailing times between each location.

	P1	P2	P3	P4	P5	P6	P7	P8	M1	M2	B1	B2
P1	0	18	14	20	32	34	35	30	19	29	15	32
P2	18	0	23	15	31	33	27	35	21	33	12	35
P3	14	23	0	12	39	34	30	32	15	38	16	31
P4	20	15	12	0	35	38	31	39	12	31	18	34
P5	32	31	39	35	0	18	12	19	31	12	29	11
P6	34	33	34	38	18	0	13	15	34	11	36	15
P7	35	27	30	31	12	13	0	12	29	18	25	19
P8	30	35	32	39	19	15	12	0	33	15	39	12
M1	19	21	15	12	31	34	29	33	0	30	15	25
M2	29	33	38	31	12	11	18	15	30	0	28	16
B1	15	12	16	18	29	36	25	39	15	28	0	26
B2	32	35	31	34	11	15	19	12	25	16	26	0

Step 3. Judge if p is equal to $3 * n$:

- (a) if p is equal to $3 * n$, then the modification is completed;
- (b) else, set $p = p + 1$, and go to Step 2.

After being modified according to the steps introduced above, the temporary solution can be changed to a new solution by deciding whether tugboats should return to the base according to (2.16).

4. Computational Experiments

4.1. Experimental Data

To implement a comparison of the findings from the proposed algorithm, some experimental data were randomly generated, details of which are as follows.

- (a) Location data: the sailing times between each location (P1–P8, M1–M2, B1–B2) are as Table 1. Therein, P1–P8 are locations of 8 berths; M1 is the location where ships whose target berths P1–P4 meet tugboats at the entrance of port; M2 is the location where ships whose target berths P5–P8 meet tugboats at the entrance of port; B1 and B2 are two anchorage bases of tugboats whose service area are P1–P4 and P5–P8, respectively.
- (b) Ship data: styles of ships are generated to S1, S2, S3, S4, and S5 which take up about 10%, 20%, 40%, 20%, and 10% of the total ships, berthing/unberthing times, loading and unloading times of ships are normally distributed in $N(35,25)$, $N(300,1600)$, and the berthing locations of ships are uniformly distributed to P1–P8.
- (c) Tugboat data: quantities of the six kinds of tugboats in the two anchorage bases are all one.

Table 2: Results of the PHA versus existing scheduling rules.

Number of ships	RCOM				UCOM			
	PHA	TSD	FAT	UWAT	PHA	TSD	FAT	UWAT
10	3743	4222	4181	4280	3743	4200	4156	4245
15	4983	5551	5504	5609	4951	5485	5456	5575
20	6800	7436	7357	7483	6705	7353	7242	7451
25	8481	9132	9030	9220	8405	8900	8858	9100
30	10012	11185	10993	11530	9988	11031	10920	11235

Table 3: Results from the proposed algorithm under two different operation modes.

Number of ships	Total operation times of all tugboats			Average operation times of all tugboats		
	RCOM	UCOM	GAP ¹	RCOM	UCOM	GAP ²
10	3743	3743	0	312	312	0
15	4983	4951	32	415	413	3
20	6800	6705	95	567	559	8
25	8481	8405	76	707	700	6
30	10012	9988	24	834	832	2

¹Operation times of all tugboats under the RCOM—values under the UCOM; ²average operation times of all tugboats under the RCOM—values under the UCOM.

4.2. Experiments without the Shifting-Berth Operation

Suppose the basic data are as shown in Section 4.1, no shifting-berth operation exists and all tugboats do not return to the anchorage base during the planning horizon, use the proposed hybrid algorithm to solve the established tugboat scheduling problem, and then we can get the performance comparisons of the hybrid algorithm with three existing scheduling rules with different number of ships, which are shown in Table 2.

As we can see from Table 2, the PHA's solved results are all far less than those from the three existing scheduling rules. All those can fully describe the efficiency of the PHA. Besides, the performance of the three scheduling rules reveals the same rules: FAT is superior to TSD, and TSD is better than UWAT. That is because the FAT rule considers both the TSD and UWAT rules, while the UWAT rule only considers the uniform scheduling of every tugboat but might cause the postponement of the waiting time of ships for tugboats.

Besides, we can see from Table 3 that the difference between the two operation modes increases from zero and then decreases to near zero. The reason for that phenomenon can be concluded as follows: when the number of ships is small and the available tugboats are abundant, the optimal solution is the scheduling scheme under the RCOM (as the solutions under the UCOM include those under the RCOM, so the optimal solution and optimal value of the modes are the same), which means there is no need to transfer tugboats from another anchorage base; as the number of the ships increases and the tugboat resource becomes scarce, the cost for ships to wait for unoccupied tugboats in another anchorage base is less than that of waiting for busy tugboats in its own anchorage base, so the UCOM is better than RCOM; while the number of ships is great and all tugboats in both anchorage bases are busy, there is no point of transferring tugboats from another anchorage base, which means the RCOM is better.

After the basic analysis above, we compare the operation times on whether tugboats return to the anchorage base during the planning horizon, the results of which can be shown as Table 4.

Table 4: Comparisons between the total operation times on whether tugboats return to the base.

Number of ships	RCOM			UCOM		
	Results if tugboats do not return to the base (f_1)	Results if tugboats return to the base (f_2)	GAP*	Results if tugboats do not return to the base (f_1)	Results if tugboats return to the base (f_2)	GAP*
10	3743	2675	39.93%	3743	2675	39.93%
15	4983	4005	24.42%	4951	3952	25.28%
20	6800	5285	28.67%	6705	5206	28.79%
25	8481	6575	28.99%	8405	6503	29.25%
30	10012	8007	25.04%	9988	7951	25.62%

*Percentage that f_1 is larger than that of f_2 , which can be calculated by $(f_1 - f_2)/f_2 \times 100\%$.

Table 5: Results with different proportion of the shifting berth operation.

Number of ships	Results with different proportion of the shifting-berth operation								
	0% ¹	5% ²	GAP1 ^a	10% ³	GAP2 ^b	15% ⁴	GAP3 ^c	20% ⁵	GAP4 ^d
10	2675	2845	6.36%	3076	13.04%	3224	20.52%	3415	27.66%
15	3952	4205	6.40%	4518	12.53%	4703	19.00%	4961	25.53%
20	5206	5485	5.36%	5824	10.61%	6121	17.58%	6452	23.93%
25	6503	6857	5.44%	7331	11.29%	7715	18.64%	8035	23.56%
30	7951	8381	5.41%	9122	12.84%	9423	18.51%	9731	22.39%
Average	/	/	5.79%	/	12.06%	/	18.85%	/	24.61%

^a(value of 2 – value of 1)/value of 1 \times 100%; ^b(value of 3 – value of 1)/value of 1 \times 100%; ^c(value of 4 – value of 1)/value of 1 \times 100%; ^d(value of 5 – value of 1)/value of 1 \times 100%.

Based on Table 4, we can see that the operation times if tugboats do not return to the base are 30% larger than those if tugboats return to the base during the planning horizon. That means if tugboats do not return to the base during the horizon, there exists at least 30% of the total sailing routes which are ineffective sailing routes, which is infeasible and does not coincide with the modern concept of green transportation.

4.3. Experiments with the Shifting-Berth Operation

In this section, sensitivity analysis of the three elements to the objective is to be made, and all the experiments done are under the UCOM mode and based on the assumption that tugboats can return to the anchorage base during the planning horizon.

(a) Sensitivity Analysis of the Proportion of the Shifting-Berth Operation

Assume that there are 0%, 5%, 10%, 15%, 20% of the total ships which have to experience the shifting-berth operation, the minimal total operation times of all tugboats when the number of ships is 10, 15, 20, 25, 30 are summarized in Table 5.

As we can see from Table 5, the GAPs (a, b, c, d) are all larger than the proportion of the shifting-berth operation. That is because a single shifting-berth operation contains an unberthing operation, a shift between the berths, and a berthing operation, thus needs more tugboats' resource than normal berthing and unberthing operations. So it is necessary

Table 6: Results with different distribution characteristics of the handling operation times.

Number of ships	Results with different distribution characteristics of the handling operation times				
	$N(300, 1600)^1$	$N(350, 2500)^2$	GAP1 ^a	$N(400, 3600)^3$	GAP2 ^b
10	2845	2815	-1.05%	2862	0.60%
15	4205	4240	0.83%	4200	-0.12%
20	5485	5522	0.67%	5488	0.05%
25	6857	6870	0.19%	6840	-0.25%
30	8381	8387	0.07%	8428	0.56%
Average	/	/	0.14%	/	0.17%

^a(value of 2 – value of 1)/value of 1 \times 100%; ^b(value of 3 – value of 1)/value of 1 \times 100%.

to reduce the number of shifting-berth operation in practice, so that the full utilization of limited tugboat resources.

(b) Sensitivity Analysis of the Distribution Characteristics of the Handling Operation Times

Assume that the distribution characteristics of handling operation times of ships at berth are $N(300, 1600)$, $N(350, 2500)$, and $N(400, 3600)$, the proportion of the shifting-berth operation is 5%. The results by the PHA are concluded in Table 6.

As we can see from Table 6, there is no obvious trend about the total operation times according to the changing of the handling times of ships at berth. That is to say, the objective function is not sensitive to the change of the handling times. The reason for that phenomenon can be concluded as follows.

Compared with the operation times of tugboats, the handling times are much larger. After completing a certain task, a tugboat can return to the base to have a rest and then sail to its next target location. With the increase of the handling operation times, the wait times in the base may also increase, which are not parts of the total operation times of tugboats. Thus, the objective does not reveal obvious reaction to the change of the handling times.

(c) Sensitivity Analysis of the Tugboat Deployment Scheme

We then assume different deployment schemes of the available tugboats in the port (i.e., Scheme 1: the number of all types are 1; Scheme 2: the number of type 6 are 2, others are 1; Scheme 3: the number of type 5 and 6 are 2, others are 1). The results solved by the PHA are summarized in Table 7. Therein, the proportion of the shifting-berth operation is still 5%.

As Table 7 shows, the total operation times of all tugboats reveal a mild trend of decrease as the number of tugboats deployed increases. That is to say, the total operation times of tugboats can only be slightly reduced by simply increasing the number of tugboats deployed, and the cost of increasing tugboats may well be larger than the time cost saved by that. Under that circumstance, adding extra tugboats is not advised.

By the analysis, we can say that the objective is most sensitive to the proportion of the shifting-berth operation, influenced slightly by the tugboat deployment scheme, and not sensitive to the handling operation times.

Table 7: Results with different tugboat deployment schemes.

Number of ships	Results with different tugboat deployment schemes				
	Scheme 1 ¹	Scheme 2 ²	GAP1 ^a	Scheme 3 ³	GAP2 ^b
10	2845	2833	−0.42%	2808	−1.30%
15	4205	4186	−0.45%	4159	−1.09%
20	5485	5421	−1.17%	5394	−1.66%
25	6857	6807	−0.73%	6785	−1.05%
30	8381	8325	−0.67%	8299	−0.98%
Average	/	/	−0.69%	/	−1.22%

^a(value of 2 – value of 1)/value of 1 × 100%; ^b(value of 3 – value of 1)/value of 1 × 100%.

5. Concluding Remarks

This paper formulated the tugboat scheduling problem as a multiprocessor task scheduling problem (MTSP). The model considers factors of multi-anchorage bases, different operation modes, and three stages of operations (berthing/shifting-berth/unberthing). A hybrid simulated annealing-based ant colony algorithm is proposed to solve the addressed problem. By the numerical experiments without the shifting-berth operation, the effectiveness were verified, and the fact that more effective sailing may be possible if tugboats return to the anchorage base timely was pointed out; by the experiments with the shifting-berth operation, the paper proved that the objective is most sensitive to the proportion of the shifting-berth operation, influenced slightly by the tugboat deployment scheme, and not sensitive to the handling operation times.

Future work about the topic should be to extend the problem from the static situation to a dynamic one, although it may be much more difficult but more meaningful.

References

- [1] K. C. Ying and S. W. Lin, "Multiprocessor task scheduling in multistage hybrid flow-shops: an ant colony system approach," *International Journal of Production Research*, vol. 44, no. 16, pp. 3161–3177, 2006.
- [2] H. Xuan and L. X. Tang, "Dynamic hybrid flowshop scheduling problem with multiprocessor tasks," *Computer Integrated Manufacturing Systems*, vol. 13, no. 11, pp. 2254–2288, 2007.
- [3] Z. Liu, "Hybrid evolutionary strategy optimization for port tugboat operation scheduling," in *Proceedings of the 3rd International Symposium on Intelligent Information Technology Application (IITA '09)*, pp. 511–515, November 2009.
- [4] Z. Liu, "Port tugboat operation scheduling optimization considering the minimum operation distance," *Journal of Southwest Jiaotong University*, vol. 46, no. 5, pp. 875–881, 2011.
- [5] S. Wang and B. Meng, "Resource allocation and scheduling problem based on genetic algorithm and ant colony optimization," in *Proceedings of the 11th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD '07)*, vol. 4426 of *Lecture Notes in Computer Science*, pp. 879–886, Nanjing, China, 2007.
- [6] S. Wang, I. Kaku, G. Chen, and M. Zhu, "Research on the modeling of Tugboat Assignment Problem in container terminal," *Advanced Materials Research*, vol. 433–440, pp. 1957–1961, 2012.
- [7] Z. X. Liu and S. M. Wang, "Research on bi-objectives parallel machines scheduling problem with special process constraint," *Computer Integrated Manufacturing Systems*, vol. 11, no. 11, pp. 1616–1620, 2005.
- [8] L. C. Dong, Z. Q. Xu, and W. J. Mi, "The dynamic tugboat schedule based on particle swarm algorithm combined with genetic operators," *Mathematics in Practice and Theory*, vol. 42, no. 6, pp. 122–133, 2012.
- [9] Z. X. Liu and S. M. Wang, "Research on parallel machines scheduling problem based on particle swarm optimization algorithm," *Computer Integrated Manufacturing Systems*, vol. 12, no. 2, pp. 183–296, 2006.

- [10] H. R. Boveiri, "ACO-MTS: a new approach for multiprocessor task scheduling based on ant colony optimization," in *Proceedings of the International Conference on Intelligent and Advanced Systems (ICIAS '10)*, pp. 175–179, June 2010.
- [11] M. Dorigo, G. DiCaro, and L. Gambardella, "Ant algorithm for discrete optimization," *Artificial Life*, vol. 5, no. 2, pp. 137–172, 1999.

Research Article

Guidance Compliance Behavior on VMS Based on SOAR Cognitive Architecture

**Shiquan Zhong,¹ Hongwei Ma,² Lizhen Zhou,¹ Xuelian Wang,³
Shoufeng Ma,¹ and Ning Jia¹**

¹ College of Management and Economic, Tianjin University, Tianjin 300072, China

² Transportation Planning Center, Tianjin Municipal Engineering Design and Research Institute, Tianjin 300051, China

³ School of Management, Hebei University of Technology, Tianjin 300130, China

Correspondence should be addressed to Hongwei Ma, mhw355@163.com

Received 13 July 2012; Accepted 18 September 2012

Academic Editor: Baozhen Yao

Copyright © 2012 Shiquan Zhong 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.

SOAR is a cognitive architecture named from state, operator and result, which is adopted to portray the drivers' guidance compliance behavior on variable message sign (VMS) in this paper. VMS represents traffic conditions to drivers by three colors: red, yellow, and green. Based on the multiagent platform, SOAR is introduced to design the agent with the detailed description of the working memory, long-term memory, decision cycle, and learning mechanism. With the fixed decision cycle, agent transforms state through four kinds of operators, including choosing route directly, changing the driving goal, changing the temper of driver, and changing the road condition of prediction. The agent learns from the process of state transformation by chunking and reinforcement learning. Finally, computerized simulation program is used to study the guidance compliance behavior. Experiments are simulated many times under given simulation network and conditions. The result, including the comparison between guidance and no guidance, the state transition times, and average chunking times are analyzed to further study the laws of guidance compliance and learning mechanism.

1. Introduction

Reality indicates that the effects are very limited if one just simply relies on constructing new roadway, adding new traffic facilities or adopting traditional management methods to solve the traffic problems [1]. Intelligent traffic systems (ITSs) have been put forward as early as 30 years ago, which is treated as one of the most important measures to solve traffic issues [2]. Among the various means and methods involved in ITS, traffic guidance can help drivers to make more efficient route choices and reduce the anxiety and stress of trip, which is one of the means that can truly improve the performance of systems and ensure traffic safety. Guidance

compliance behavior is a basic problem in traffic guidance as it represents the perception of driver to the guidance information, the trust degree in the information, and the regular pattern of the degree variation. The guidance compliance rate is the external manifestation of driver's guidance compliance behavior, and an effective guidance system is established based on the high compliance rate of its users [3].

In the past, the planning and design of traffic guidance system are normally based on the assumptions that drivers completely obey the guidance or choose route according to the compliance rate set by the system, [4–7], or simply think that drivers regard expected benefit as a goal and they are assigned to the network with the users equilibrium after system implementation [8–12]; however, the assumptions are not true. First, the guidance compliance rate is not static but changes as drivers gain further knowledge of the rate while the guidance system is implemented. Second, the guidance compliance behavior which treats expected benefit as the only goal cannot reflect drivers' real decision-making process. Whether or not the drivers comply with the guidance information is not only related with the expected travel time and trip cost, but also closely refer to their perception degree to the guidance information, the familiarity with the road network, the attitude towards risk, the release modes of guidance information, and the approach of information display. Finally, the decision-making process of drivers in the previous guidance system planning and design is viewed as absolutely rational, it is assumed that drivers have full awareness of guidance information and that they have a clear perception of a good or a bad trip, and they completely understand the current road network condition. However, many studies have shown that the behavior of drivers in practice is limited rational [9]. Thus, traditional approaches of guidance system planning and design are not effectively capable of reflecting the impacts of release modes and display approaches of guidance information, the position of VMS in the complex traffic environment on drivers' thinking and decision-making processes, and the changes of the trip demand and the spatial and temporal distribution of traffic flow following the impacts.

Without the accurate estimations of the compliance rate and its evolutionary process, drivers cannot perform effectively even if the system has provided good guidance suggestions, so the expected guidance effect cannot be achieved. In fact, according to investigation and research many researchers have shown that the guidance compliance rate is far lower than the system expects [13–17]. Thus, Erke et al. [17] have pointed out that it is not advisable to optimize in the process of system planning and design based on the set rate which is close to 100% in order to maximize the utility. Because of the above facts, many scholars have realized the significance of researching into the guidance compliance behavior on planning and design the guidance system. Koutsopoulos and Xu [18] and Ozbay and Bartın [19] have stated that the effects of travelers on both guidance compliance rate and the change process of the rate must be considered if a guidance system wants to be useful. Adler [20] has indicated that the successful implementation of the advanced traveler information system (ATIS) and the in-depth research of the traffic flow model under the guidance information are both very important in studying the traffic guidance compliance behavior.

2. Literature Review

The traditional guidance theoretical methods and the methods of processing compliance rate in guidance system design can be approximately divided into two categories. The first is

based on the presupposition of the guidance compliance rate, such as Thakuriah and Sen [4], Wang et al. [5] assumed that drivers completely obeyed the guidance when various guidance strategies had been simulated, Deflorio [6] set the rate as a constant between 0 and 1, and Yin and Yang [7] set the rate to match the assumption that drivers cost less travel time when they obey the guidance than they did not. The second method avoids guidance compliance rate but converses it to the method which is based on the random utility theory and user equilibrium theory. Based on this method, multinomial probability model (MNP) [8, 9], the theoretical model based on the generalized extreme value (GEV) [10], and the stochastic user equilibrium model (SUE) are proposed [11, 12].

Many surveys show that the real guidance compliance rate deviates from the expected rate severely after the guidance system is implemented [13–17], it challenges the traditional methods of processing the compliance rate. Hence, many scholars realize that researching the guidance compliance rate of the groups demands the study of guidance compliance behavior from the individual point [18–20]. The main three methods of studying the guidance compliance behavior are survey, experiment, and simulation models.

2.1. The Method Based on Survey

The most direct way to study the compliance behavior under a specific guidance system is to survey the service groups and extract the factors affecting the actual compliance rate to be analyzed for guiding the planning and design of the guidance system. The two main survey methods are questionnaire and network monitoring. Bonsall and Joint [13] have investigated 100 drivers with vehicle-mounted guidance systems and found that 30% of the drivers did not comply with the guidance information when they are familiar with the road network, and 90% complied when they are unfamiliar. Cummings [14] discovered that only 4–7% of the drivers obey the guidance in normal conditions and only 13% in special conditions after investigating 20 variable message systems in Europe; in addition, 5% of the drivers did not understand the information shown on the VMS, and over 10% misunderstood the information. Tarry and Graham [15] monitored the VMS near Birmingham through the network and found out that 27–40% of the drivers obeyed the guidance when the VMS showed that an accident happened in front, but only 2–5% when the VMS just displayed traffic jams without reporting the reasons. Swann et al. [16] investigated the VMS near the estuary district in Forth of Scotland and found that 16% of the drivers routed to the recommended path when VMS showed traffic congestions in the front. Erke et al. [17] found that about 20% of the drivers followed the recommendation and almost 100% when the VMS showed that the road ahead was closed through a spot field investigation on two sites of motorways. Zhou and Wu [21] analyzed 497 valid questionnaires in Beijing and found that proportions of drivers who would change the route, maybe change, would not change, and did not know what to do were 16.9%, 65.4%, 11.5%, and 6.2%, respectively. Chen et al. [22] and Mo and Yan [23] studied the parking guidance system in Nanjing and Shanghai and concluded that the behavior of the driver in discovering, understanding, and complying with the guidance information varied very much when drivers' personal properties, travel characteristics, and parking lot selection changed.

The above investigation results show that the compliance rate is normally low whether for the vehicle-mounted guidance system or VMS. The main reasons of low rate have been analyzed as follows: (1) there is no need for the guidance information because drivers are familiar with the roads; (2) drivers do not notice the guidance information; (3) drivers do not

understand the guidance information; (4) drivers do not believe the guidance information; (5) the information is received too late that the drivers have chosen the route; (6) the readable distance of VMS is too short, drivers have no time to read, and thus miss the guidance information [13–17].

2.2. The Method Based on Experiment

The method based on experiment is to find out the laws of behavior occurrence and some measures to affect the behavior by setting different guidance environments in the road network to analyze the response of the drivers. Allen et al. [24] observed drivers experiment with different OD and different roads congestion conditions, and the results showed that their compliance rate is high (more than 70%) when using route suggestion strategy. However, they did not consider that the recommended route would be congested and then influenced the guidance compliance behavior, which actually was very common and practical. Srinivasan and Jovanis [25] adopted the Designer Workbench model of Corypheaus Corporation to create a software to develop experimental environments, in which different display approaches of guidance information were used to test the effectiveness for 10 participants; they found that the display approach with the highest compliance rate was the one using the countdown progress bar to show the distance between the current location and the front crossing. Chen and Paul [26] allowed 99 participants to join a continuous 20-day experiment in computerized simulation road network and found that the factors affecting drivers' guidance compliance rate were characteristics of guidance information (e.g., whether the information had shown the recommended path or not, whether the road connected or closed to highway or not), the characteristics of drivers (e.g., age, gender, and educational level), and whether the information provided the reasons of accidents or congestion or not. Wachinger and Boehm-Davis [27] found that drivers preferred different display modes and seemed to be more willing to comply with the favorite one. Adler [20] divided 80 participants into 4 groups, and two-factor measurement experiments with 20 persons per group, 15 times per person were repeated successfully in a simulated road network. The result showed that the compliance rates of drivers unfamiliar with the road network were higher than the familiar ones because the familiar drivers benefited slightly from the guidance information.

2.3. The Method Based on Simulation Model

Due to the rapid development of computer technology, people have begun using simulation model to study the guidance compliance behavior of drivers and have attained certain achievements. Lu and Tan [28] proposed a complexity model of guidance compliance rate based on Logit traffic assignment model and analyzed the changing properties of the rate based on the simulation. Huang et al. [29] proposed a stochastic user equilibrium model to study the changing process of the compliance rate in ATIS. In recent years, people have paid increasing attention to the multiagent simulation techniques to study the guidance compliance behavior of drivers. Adler and Blue [30] studied drivers' guidance compliance behavior from the perspective of cooperative game by simulating the interaction of various agents in the traffic system. Wahle et al. [31] proposed a two-layer agent framework to study the guidance compliance behavior of drivers; the first layer showed their perception and reaction to guidance information, and the second layer described their decision-making process. Dia [32] added beliefs and capacities of drivers and various rules of behavior into

agents when the reactions of drivers to different traffic information had been studied, and they proposed a agent framework with cognitive function. Based on this architecture, they studied the changing process of the guidance compliance behavior.

3. Research Motivation

The current research results of guidance compliance behavior provide many methods and basis to design, implement, and evaluate the traffic guidance system, the results also promote the development of intelligent traffic system, but certain shortcomings still remain. First, it is obviously unfeasible to presuppose the guidance compliance rate without scientific analysis as there are so many factors influencing the rate. The way which converses the rate leads to problem solving difficulty or deviation from the actual situation because of model complexity and too many assumptions (e.g., travelers are assumed to be fully rational), and there are still much work to do to apply it into practice. Second, it can only roughly reflect drivers' perception of guidance information with the method based on survey; moreover, their guidance compliance behavior will change as time passes, so relying solely on survey to analyze the internal causes of guidance compliance behavior scientifically and accurately is not an easy work. Third, simulation is a good way to study the guidance compliance behavior, but the development of the realistic guidance simulator costs too much. It is also difficult to recruit volunteers to participate into the experiment for as long as several months, so the simulation can only be done in a relatively short period. It is also worth studying whether the memory, thinking, and decision-making reflected by this intensive simulation are consistent with real conditions or not. Fourth, the current simulation models cannot reflect the actual process of guidance compliance behavior because the studies on the process of perceiving, recognizing and remembering information, the decision-making, feedback, and the impact of these activities on the behavior are all insufficient.

As one of the three cognitive architectures, SOAR is based on chunks theory. It uses rule-based memory to access search control knowledge and operators and finally achieves common problem solving. This paper will adopt SOAR to build the cognitive process model of guidance compliance behavior. This is mainly based on the following reasons. First, as mentioned above, guidance compliance behavior is a dynamic learning process, which changes as time passes. SOAR can learn from experience, which means it can remember how it solves a previous problem and then uses the experiment and knowledge for subsequent problem solving tasks. It can dynamically organize the accessible knowledge to decide and also set subgoals dynamically if the knowledge is incomplete or inconsistent with decision [33]. This is very similar with the thinking and decision-making of the guidance compliance behavior of drivers. For example, a driver organizes and analyzes the received knowledge according to his own judgment of the current road conditions and the past accuracy of the guidance information to decide if he complies with the guidance or not under his certain goal. If the driver receives the guidance information, he is not sure if it is correct, and he is unfamiliar with the road network, he cannot make a decision to minimize the travel time with the accessible knowledge, then, he is likely to set a subgoal to test the guidance information. Drivers pay more attention to the accuracy of the guidance information and are often more sensitive to the delay of the recommended path under this condition. Second, as a mature cognitive theory, the content and framework of SOAR have been sufficiently described, including the depth description of perception, memory, decision making process, and learning mechanism; thus, SOAR portrays the guidance compliance behavior with a

more actual architecture. Third, SOAR has been used to simulate different aspects of human behavior successfully, such as team behavior, decision-making of people in virtual games, and decision-making behavior of pilot, empirical supports are provided by many scholars [34, 35]. Thus, SOAR provides a good idea to study traffic guidance compliance behavior.

Today, studying traffic guidance compliance behavior has received more and more attention with the growing development of traffic guidance system. People have gradually become conscious of the fact that more detailed simulations about traffic behavior are needed. It is a hot topic of the emerging field to study guidance compliance behavior based on a multiagent framework with the integration of multidisciplinary as the platform. In this paper, based on the multiagent platform, SOAR architecture is added to describe the driver agent, and the empirical method is combined to the computer simulation to study the guidance compliance of drivers.

4. SOAR Cognitive Architecture

SOAR is a general intelligent architecture developed in 1987 by Laird et al. [36]. It is a cognitive architecture with a wide range of applications, and it mainly focuses on knowledge, thinking, intelligence, and memory. SOAR is constructed with the assumption that all goal-oriented behavior can be likened to choosing an operator from a state. A state is a representation of the current problem-solving situation; an operator transforms a state (makes changes to the representation) and produces a new state. A goal is a desired outcome of the problem-solving activity. As SOAR runs, it is continually trying to apply the current operator and select the next operator (a state can have only one operator at a time), until the goal has been achieved.

As shown in Figure 1 [35], SOAR architecture has the following main parts: input and output interface, long-term memory, and working memory. There are also some underlying mechanisms, such as decision-making and learning.

SOAR interacts with the environment through the perception and action interface. The environment is mapped into working memory through perception, and the inner representations are returned to the exterior, after which actions are generated to act on the environment through action interface. SOAR has two kinds of memories with different forms of representation: a working memory that describes the current problem solving situations and a long-term memory that stores long-term knowledge. In SOAR, the current situation, including data from sensors, results of intermediate inferences, active goals, and active operators, is held in working memory represented as a hierarchical graph of states or goals. Long-term memory contains production memory, semantic memory, and episode memory. SOAR achieves choosing and applying operators through decision-making cycle, which is a fixed processing mechanism. Along with the decision cycle, SOAR has four different types of learning mechanism, namely, reinforcement learning, chunking, episode learning, and semantic learning [37].

5. Agent Design of Traffic Guidance Compliance Behavior

Vehicle and driver are integrated as a whole because their behaviors are inseparable in the study of traffic guidance compliance behavior: the driver operates by observing the external environment, and the vehicle interacts with the external environment by carrying out the operation of the driver. Thus, the driver-vehicle unit is regarded as an agent in our study.

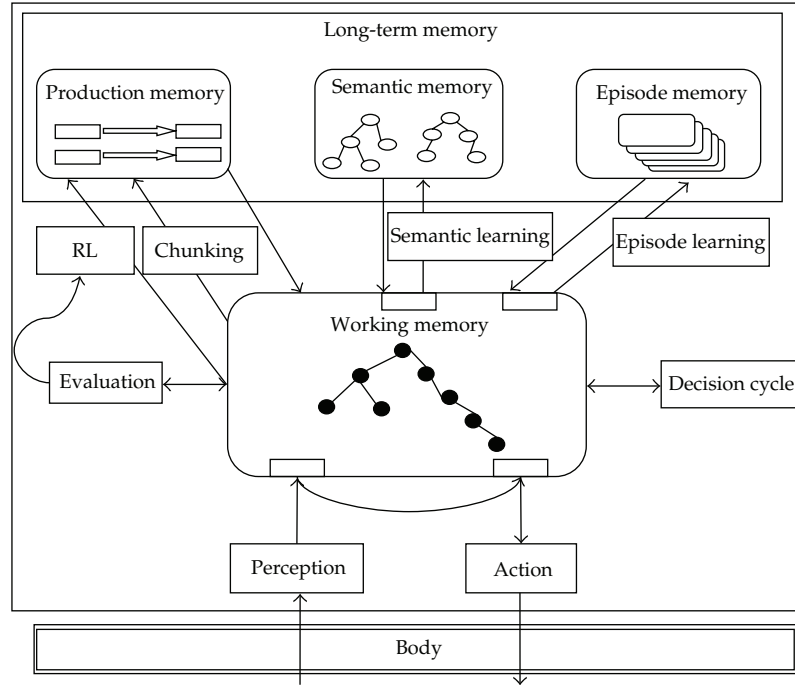


Figure 1: SOAR architecture.

As Figure 2 shows, VMS uses three colors to represent the traffic condition of front roads: red for traffic congestion, yellow for a little crowded, and green for smoothness. Here, we will analyze how SOAR describes the specific guidance compliance behavior under VMS. It is assumed that a driver **A** drives to his destination **B**. On his way, he passes a VMS which shows he can reach **B** through any of the downstream roads (left road, forward road, and right road) of crossing ahead. Driver **A** finally will choose one of the three roads according to the combination of many factors, such as his familiarity with the road network, the current guidance information on VMS, the traffic congestion situation in sight, the external environment, and the former experience about the accuracy of the information shown on VMS. The SOAR cognitive model is adopted to describe the guidance compliance behavior of the driver when he passes the VMS repeatedly, including the process of perception, memory, decision, and learning.

5.1. Problem Space, State, and Operator

Problem space is the internal representation of the problem. It contains three types of states: initial state, intermediate state, and goal state. A state involves all the information of the current situation in the problem solving process, as the results of perception, the description of the current goal, and the problem space. An operator transforms a state to a new state. The problem solving is to find out a sequence of operators to transform the initial state to a goal state. Figure 3 is the problem space comprised by states and operators.

In Figure 3, squares represent states containing features and values, which reflect the internal and external situations. Goal states, states in which features have values that indicate

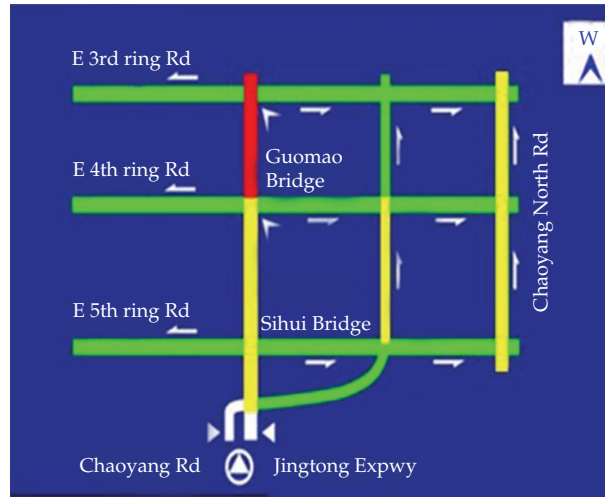


Figure 2: The display approach on VMS.

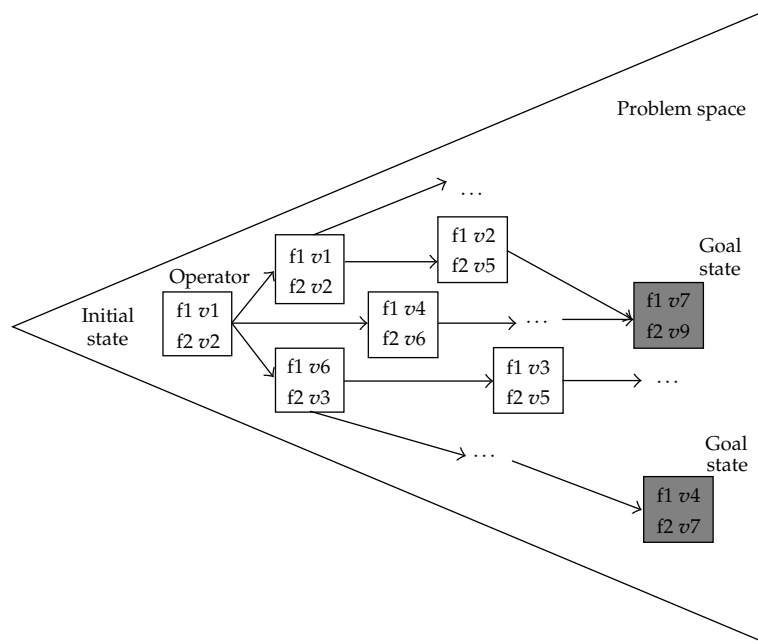


Figure 3: Problem space.

the goal has been achieved, are shaded. Arrows represent operators that change or transform states.

In the guidance compliance scene, the initial state is the information, including guidance information shown on VMS, the perception of the external traffic environment, the judgment of the information accuracy, and the experience about the downstream road conditions, when **A** drives into the visual range of VMS. Goal state is when **A** has chosen the downstream route. The psychological process between initial state and goal state is described

by intermediate states. Operators transform the initial state to immediate state and finally to goal state through several immediate states. Four kinds of operators are considered in this study, which are listed as follows.

- (1) The first operator is choosing route, including the driver choosing to forward, left or right.
- (2) The second is changing road condition, according to the external environment and his travel experience, the driver infers the downstream roads conditions which they think are most consistent with the truth.
- (3) The third is changing the driving goal from saving money (or saving time) to saving time (saving money). Saving money requires the driver to choose the shortest path to the destination, and saving time requires the driver to choose the path which costs the least time to arrive at the destination.
- (4) The fourth is changing mood. The driver may feel easy when the accuracy of matching is high, while he may be impatient when the accuracy is low.

5.2. Working Memory

Dynamic information about the world and internal reasoning, including data from sensors, results of intermediate inferences, hierarchy states, active goals and active operators, is all held in the working memory.

In our guidance compliance scene, an agent is the unit of driver and vehicle, and the agent achieves the goal state by transforming among several states in one decision-making process. The attributes of any state involve the attributes of the driver, vehicle, and the external traffic environment. The attributes of driver include gender, age, character, driving years, monthly income, mood, his/her familiarity with the road network, the understanding of the guidance signal, destination, driving goal, and the current location. The attributes of the vehicle include the usage, size, and the speed, and the attributes of roads and surrounding environment include the congestion situation of the current road in sight, the predicted congestion situation of the downstream roads through perception, the true congestion of the downstream roads by feedback, the current control signal of different directions, and the guidance information shown on VMS. The state has some other attributes, such as the name and the super state of the state. Parts of the attributes, such as the mood, driving goal, and the predicted congestion situation, change with the transformation of states while the others are static. The agents with the same static attributes are classified into one category. The value range of each attribute is shown in Tables 1, 2, and 3.

5.3. Long-Term Memory

Long-term memory is the area where achievements are stored. Although all types of long-term knowledge (procedural, semantic, and episodic) are useful, procedural knowledge is primarily responsible for controlling behavior and maps directly onto operator knowledge. Semantic and episodic knowledge usually come into play only when procedural knowledge is somewhat incomplete or inadequate for the current situation. In our study of the traffic behavior, procedural memory and episodic memory are involved.

The procedural knowledge is represented by production rules. The rules combine the procedural knowledge with the operations to things. When some specific conditions are

Table 1: The value ranges of the attributes of the driver.

Attribute (unit)	Value 1	Value 2	Value 3	Value 4
Character	Risky	Steady	Reserved	/
Mood	Easy	Impatient	/	/
Gender	Male	Female	/	/
Driving years (year)	0-1	1-5	>5	/
Age (year)	<25	25-40	>40	/
Incomings (yuan)	<2000	2000-5000	5000-10000	>10000
Destination	Area 1	Area 2	Area 3	/
Location	0	1	2	3
Driving goal	Saving money	Saving time	/	/
Familiarity	Familiar	Normal	Strange	/
Understanding level	Do not notice	Do not see	Notice but misunderstand	Understand

Table 2: The value ranges of the attributes of the vehicle.

Attribute (unit)	Value 1	Value 2	Value 3
Speed (k/s)	>60	40-60	<40
Size	Small	Normal	Big
Type (according to usage)	Private car	Taxi	Official car

Table 3: The value ranges of the attributes of the traffic environment.

Attribute (unit)	Value 1	Value 2	Value 3
Current density	2	1	0
Predicted density	2	1	0
True density	2	1	0
Control signal	G	Y	R
VMS sign	G	Y	R

satisfied, a set of actions in the matched rule are triggered. When the associations between goals and subgoals are triggered, production system is transformed from one to another, that is, once a production system is triggered and implemented, the control of action or behavior will be transformed to the other production system which meets the conditions. Production is represented by an if-then rule. The “if” portion of each rule contains the conditions, while the “then” portion contains action or behavior. Partial initial rules of an agent in the above guidance compliance behavior are shown in Table 4.

In Table 4, [I] is for “if”, [T] is for “then”, and C-D, V-S, C-S, P-D, Des, Loc, Moo, and M-A are short for current-density, VMS-sign, control-signal, predicted-density, destination, location, mood, and match-accuracy, respectively. Take r8 as an example, which means that if the current road is severely crowded, the VMS shows that all the roads are congested, and the traffic light allows vehicles to turn right, then the driver becomes impatient, and the matchaccuracy decreases by 10%. The meanings of other rules are expressed in the same way. The above rules are just partial initial rules of the driver agent, the others are not given in detail because of the limited space.

The episode knowledge is the specific experience and memories of a agent, and it is the source of episodic learning. Episodic memory records the events and history that are

Table 4: Partial initial rules of a driver agent.

Number	If/then	If/then	If/then	If/then	If/then	If/then	If/then
r1	[I]C-D(2)	[I]V-S(RRG)	[I]C-S(RRG)	[I]P-D(220)	[I]Des(1)	[I]Goal(T-S)	[T]Loc(3)
r2	[I]C-D(2)	[I]V-S(RRG)	[I]C-S(RRG)	[I]P-D(221)	[T]P-D(200)	—	—
r3	[I]C-D(2)	[I]V-S(RRG)	[I]C-S(RRG)	[I]P-D(220)	[I]Des(1)	[I]Goal(M-S)	[T]Goal(T-S)
r4	[I]C-D(2)	[I]V-S(RRG)	[I]C-S(RRG)	[I]P-D(200)	[I]Goal(T-S)	[T]Loc(2)	—
r5	[I]Goal(M-S)	[I]Des(1)	[T]Loc(1)	—	—	—	—
r6	[I]Goal(M-S)	[I]Des(2)	[T]Loc(2)	—	—	—	—
r7	[I]Goal(M-S)	[I]Des(3)	[T]Loc(3)	—	—	—	—
r8	[I]C-D(2)	[I]V-S(RRR)	[I]C-S(RRG)	[T]Moo (I)	[T]M-A(↓0.1)	—	—
r9	[I]C-D(2)	[I]V-S(RRR)	[I]C-S(GGR)	[I]Des(1)	[T]Loc(1)	—	—
r10	[I]C-D(2)	[I]V-S(RRR)	[I]Goal(T-S)	[T]Goal(M-S)	—	—	—

embedded into experience. An episode can be used to answer questions about the past to predict the outcome of possible courses of action or to help keep track of progress on long-term goals. In SOAR, episodes are recorded automatically when a problem is solved [38]. An episode consists of a subset of the working memory elements that exist at the time of recording. SOAR then selects those working memory elements that have been used recently. The episode that best matches the cue is found and recreated in working memory. Once the episode is retrieved, it can trigger rule firings, or even serve as the basis for creating new cues for further searches of episodic memory. In the guidance compliance scene, once a decision cycle is finished, episodic memory records the chosen operator and its preference in the current state which contains the specific guidance information, control signal, current density, predicted density, driving goal, destination and location, and preparing for the next impasse coming.

Long-term memory changes dynamically, more episodes are added to the episodic memory and new production rules are added to the procedural memory through the different learning mechanisms in the process of decision-making.

5.4. Decision Cycle

The decision cycle is the most basic processing mechanism in SOAR. The core functions of SOAR are selecting the operator and then applying it, but only a single operator can be selected for a state at a given time.

The decision cycle starts with input, during which working memory elements are created to reflect changes in perception. In our simulation, the initial state, including the current location, the guidance information shown on VMS and the road conditions in sight, of the guidance compliance problem, is created by the initial operator. The state elaboration is not the knowledge directed toward selecting and applying the operator, but the knowledge to create a new description of the state to affect the operator selection and application, after which a new description can evoke the operator to be selected and applied. The stages of state elaboration, operator proposal, operator selection, and operator application in the decision cycle retrieve the rules in the long-term memory; however, comparison among operations is achieved by preference.

If the preferences can be compared successfully, such as there is only a single candidate operator to be proposed or an operator is obviously better than the others, and then the best selected operator of the current state will be added to the working memory. When the collision among the operators occurs, as two operators have the best preference at the same time, or operator A is better than B and B is better than A are both put forward, an impasse requiring the chunking learning mechanism to solve is created.

In the simulation of the guidance scene, each rule in the long-term memory contains conditions, the operator which matches the conditions, and the numeric preference of the proposed operator. Once a new rule is added to long-term memory, the initial numeric preference of the operator in the rule is judged, and its value can be updated along the decision-making process according to the feedback from the external environment in order to make it closer to reality and provide more precise information to drivers for decision.

5.5. Learning Mechanism

The SOAR agent of the guidance compliance behavior has four different learning mechanisms. They generate the representation forms of knowledge in SOAR together, but each of them has a different source of knowledge. The source of knowledge, learning time, and learning outcome of different learning mechanisms are shown in Table 5. In this paper, chunking and reinforcement learning are mainly introduced and adopted according to the characteristics of the traffic guidance compliance behavior.

5.5.1. Chunking

Chunking occurs to learn when the impasse is solved, and the chunking rules are learned in the process of solving a substate. An impasse is how an architect defines a lack of available operators in the working memory of the system which make movement through the problem space, and a new rule is automatically created to solve the current impasse. The establishment of the chunking rule needs to analyze the production rules in the long-term memory and the episodic cues relating to the results.

Taking the SOAR agent of guidance compliance behavior as an example, it is assumed that the preferences of operator O_1 which will change the predicted road condition and operator O_2 which will change the driving goal cannot be compared successfully (i.e., the two operators are just as good or as bad with each other); thus, the decision cycle cannot decide, and an impasse occurs. At the same time, a substate which is meant to solve the impasse is produced. If S_1 is the state producing an impasse, and S_2 is the substate, we say S_1 is the super state of S_2 . An impasse can be solved by inputting new rules from outside, recalling the episodic knowledge in long-term memory, or randomly selecting an operator from several operators. The steps of solving an impasse in this paper are as follows.

- (1) Conditions which bring impasse: $O(s_i) = \Phi$ or $|O(s_i)| > 1$ and $p[o_{\max}(s_i)] - p[o_{\sec}(s_i)] < \tau(s)$, where s_i is the current state transformed from the initial state by i times state transition, and $O(s_i)$ is the set of candidate operators under the current state, its base number is $|O(s_i)|$. $p[o_{\max}(s)]$ and $p[o_{\sec}(s)]$ are the best and the second best operator of s , respectively, and $\tau(s)$ is the domain range for selecting the operator of state s directly.

Table 5: The comparison of the different learning mechanisms.

Learning mechanism	Source of knowledge	Learning time	Learning outcome
Chunking	The process of substates	When the impasse is solved	Chunking
Reinforcement learning	Feedback or reward	When reward happens	New rule
Episodic learning	Experience	When a problem is solved	Episodic memory

- (2) The approach of solving an impasse: when the conditions of an impasse are matched, the episodic memories containing the current state s_i are retrieved first in order to find out the best operator to solve the impasse. If the episodic memories do not contain the cue of state s_i , then the match accuracy of s_i decreases by step λ , and all the long-term memories are to be searched for the matched operator to move the problem to the goal state.
- (3) The creation of the chunking rule: if $T(s_e) - T_e(s_e) \leq \eta_z$ after agent exits out the downstream road, the operators of solving the impasse are to be updated to create chunking rules. $T(s_e)$ is the true travel time, $T_e(s_e)$ is the expected travel time, and η_z is the domain range of chunking rules. If the same rule is updated for continuous η_g times, the operator in the rule is to be added to the state under which the impasse happens in the decision cycle and chunking is achieved.

5.5.2. Reinforcement Learning

The source of reinforcement learning is the feedback of external environment, or what is often called a “reward.” The reward can come from the “body,” in which a model is embedded (to the model, this can be considered an external environment), or it can be generated internally when the goal is achieved. Based on experience, reinforcement learning adjusts predictions of future rewards which are then used to select actions that maximize future expected rewards. In the SOAR agent of guidance compliance behavior, the driving time is associated with the total feedback of the operator. $p_r(s_e) = [T(s_e) - T_e(s_e)]^\alpha$ is the total feedback under final state s_e , the parameter α in this paper is set as 0.5, $T(s_e)$ is the true travel time the agent costs to get to the destination under state s_e , $T_e(s_e) = T_d(t, l) + T_v[v(l)]$ is the expected travel time, and T_d is the mean driving time on road l at time t , and it indicates driver’s experience. $T_v[v(l)] = \bar{T}(v, l) - T_r(l)$ represents the affection of road condition information shown on VMS on travel time, $\bar{T}(v, l)$ is the mean travel time on road l when the guidance information is v , and $T_r(v, l)$ is the reference travel time on road l . The reference travel time adopted in this paper is figured with the assumption that the percentage occupancy is 0.5.

Given that many states and operators are involved in one decision cycle of guidance compliance behavior, the feedback is allocated to a state according to the distance between the state and the goal state. The feedback of $o(s_k)$, which is the operator corresponding to the k th state of the state transition path on cycle t , is $\lambda[d(s_k, s_e)]p_r(s_e)$. $p_r(s_e) = [T(s_e) - T_e(s_e)]^\alpha$ is the total feedback when the goal state s_e is reached, and α is 0.5. $\lambda[d(s_k, s_e), r(s_k)]$ is the weighting factor of $p_r(s_e)$ which are allocated to $o(s_k)$, and it is the function of $d(s_k, s_e)$ and $r(s_k)$. $d(s_k, s_e)$ is the distance between s_k and s_e , and $r(s_k)$ is the transition path. In this paper, $\lambda[d(s_k, s_e), r(s_k)] = (1/d(s_k, s_e)) / \sum_{i=1}^{|r(s_k)|} (1/d(s_i, s_e))$, where $|r(s_k)|$ is the state number involving in state transition path which contains s_k .

6. Simulation Experiment and Analysis

6.1. Simulation Environment

The adopted simulation road network contains 6 nodes and 7 one-way three-lane (left-turning, forward-going, and right-turning) roads, the structure and the length of roads are shown in Figure 4. VMS is set about 100 meters far away crossing 2 on road s_1 . The guidance information shown on VMS is the real-time traffic flow situation, which is represented by different colors. The corresponding occupancy percentages of green, yellow, and red colors are $[0, 0.5]$, $[0.5, 0.7]$, and $[0.7, 1]$, respectively. Traffic flow is generated in crossing 1 and passes the VMS on s_1 to select one of the three downstream roads, including left (s_2 – s_5), forward (s_3), and right (s_4 – s_6), with the guidance of VMS to get to crossing 5, before finally to crossing 6 in sequence. Crossing 2 and crossing 5 are equipped with two-phase fixed-time controllers, while the others are not. The microscopic traffic flow simulation platform based on the cellular automaton theory is adopted, and the free flow speed of each road is 60 km/h. The departure frequency of crossing 1 is set to simulate for 30 continuous days from 5 am–10 am (i.e., 540,000 seconds). In order to simulate the change law of guidance compliance behavior during a long period after the VMS is newly set on road s_1 , the vehicle is required to pass VMS repeatedly. Thus, a vehicle comes into s_1 again after it exits out from crossing 6 if the departure frequency of crossing 1 is matched.

Based on the basic structure of road network, the lengths of entry and exit roads, the position of VMS, and the traffic flow are adjusted accordingly. We find that the simulation results are basically identical when the adjustments are not too large. Hence, the simulation results and analysis for the basic structure mentioned above are only presented.

6.2. Simulation Results and Analysis

6.2.1. Guidance Effects

Two simulation experiments are conducted with the same road network and conditions which are described in Section 6.1 except no VMS is set on s_1 in one experiment, so that the differences of traffic flow operation between guidance and without guidance can be studied. Road s_1 is divided into 10 sections represented by I1 to I10 from crossing 1 to crossing 2, and the length of each section is 120 meters. Figures 5 and 6 are the spatial and temporal distributions of vehicles with and without guidance in the 30th day.

When there is no guidance information shown on VMS, the average number of vehicles on section I10 in peak period approaches 30 and its lasting time is long, the road occupancy is more than 0.8, and the average amount of vehicles on I9 in peak time is also over 20 (Figure 5). These indicate that the vehicle begins to decelerate at the place which is a bit far from the downstream crossing, resulting in the decrease of traffic capacity on s_1 in morning peak. On the other hand, when there is guidance information shown on VMS, the mean number of vehicles on I10 and I9 decrease by 22.7% and 17.6%, respectively, from 7–8 am of the morning peak (Figure 6). These show that the guidance information alleviate the congestion situation of morning peak on s_1 effectively. In addition, comparing Figure 5 with Figure 6, the vehicle numbers of each section on s_1 in off-peak time are pretty much the same, which means that the guidance information does not bring an obvious effect when traffic flow is small.

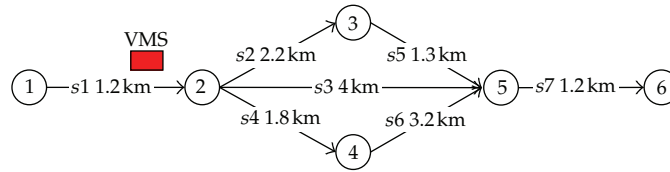


Figure 4: The basic structure of road network in simulation.

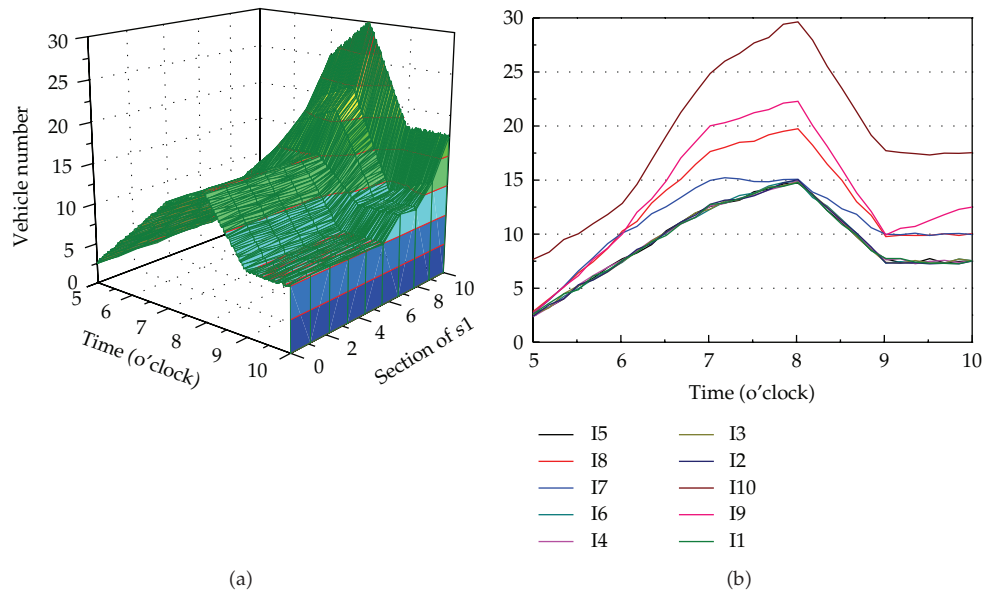


Figure 5: The spatial and temporal distribution of vehicles in each section on s1 without guidance.

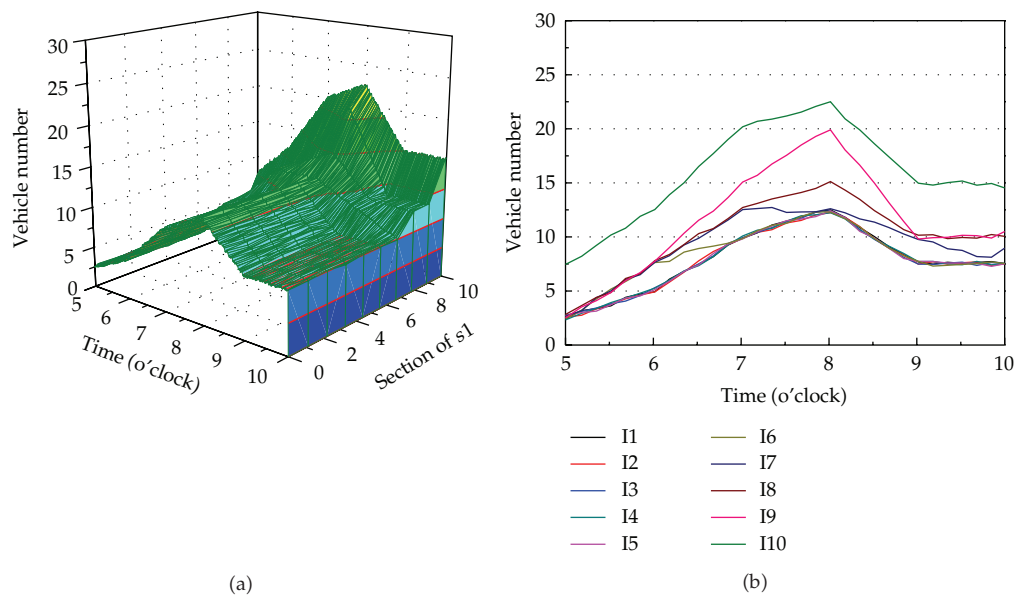


Figure 6: The spatial and temporal distribution of vehicles in each section on s1 with guidance.

Figure 7 shows the effect of the guidance information on traffic flow in peak and off-peak period from the perspective of vehicle speed on downstream roads. Whether there is guidance information or not, vehicles on the downstream roads behind VMS remain at high speed, especially in the morning between 5-6 am when they almost travel by free speed entirely; thus, their travel time is almost the same no matter which route the drivers choose. In this case, whether there is guidance or not has no obvious meaning to drivers. During peak hours, the differences between guidance and without guidance are well represented by speed. When there is no guidance information shown on VMS, s_3 is congested severely at 8 am, the average speed is just 24.87 km/s, but s_2 keeps smooth with its average speed being 36.35 km/s, and the speeds of s_2 and s_3 differ nearly by 50%. When there is guidance information, the speeds of the downstream roads behind VMS are almost the same, and the highest difference is just about 9.25%, which indicate that the capacities of the downstream roads are maximized as VMS can allocate the traffic flow well in peak time. It also represents the effectiveness of VMS in alleviating the traffic congestion.

6.2.2. The Guidance Compliance Regular Pattern and Learning Law

Figure 8 shows the route switching times of drivers in peak and off-peak time when there is guidance information shown on VMS as well as the average switching time in all periods. Simulation is used to investigate the guidance effects and its change regular pattern after VMS is newly added on s_1 . Given that all agents are newly added in the initial stage of the simulation, it takes some time for drivers to be acquainted with the guidance of VMS. The initial route switching rate is close to 0.5, indicating that 50% of the drivers have chosen different downstream roads when the same guidance information is encountered every two times in this period, as the time goes by, the average route switching rate becomes 0.15 in the 10th day (Figure 8). In this process, drivers become more and more familiar with the guidance performance of VMS. Whether the individual driver complies with the guidance or not, the guidance compliance behavior of the driver groups represented by guidance compliance rate has basically reached a stable state after drivers gradually adapt to the guidance function of VMS.

Although the average guidance compliance rate is stable, the guidance compliance behaviors of peak time and off-peak time are inconsistent. The route switching rate of peak period is very low after being steady, which is about 0.07 (Figure 8). Thus, once the drivers adapt to the VMS in the peak time, their guidance compliance habit will rarely change. This also proves that the costs of changing behavior in the peak time are very high. The route switching rate of off-peak period is twice that in peak time, indicating that drivers' guidance compliance behavior in off-peak time is with a certain degree of arbitrariness. This is because the average vehicle speeds of the downstream roads are all high, and the differences of choosing different routes are not obvious. This is also a reason for the ineffectiveness of the guidance system in off-peak time.

The number of state transition is a significant indicator which reflects the degree of difficulty involved in the process of decision-making. Figure 9 shows the average state transition times of 800 agents on cycles 0–300, all the agents are chosen randomly from the drivers whose decision cycles are over 300. At first, the average state transition number of agents is 7, indicating that the route from initial state to goal state is a little complicated in the beginning of the simulation, and several times of state transition and operator selection are needed. With the increase of decision-making number, after several rounds of feedbacks

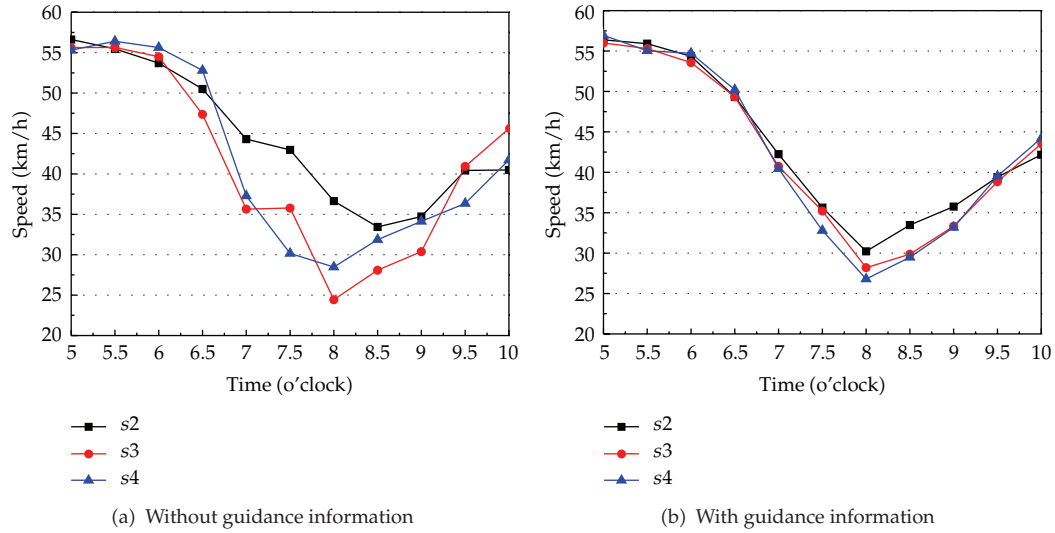


Figure 7: The driving speed of vehicle on s2, s3, and s4 at different times.

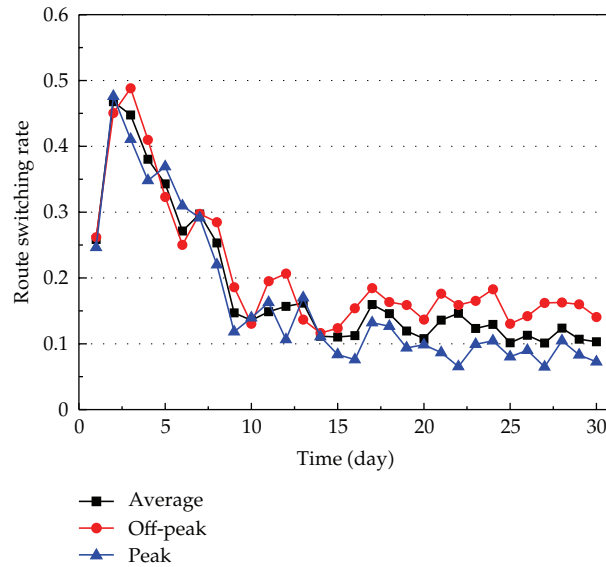


Figure 8: Route switching rate.

and learning, the operator of choosing route is directly reached through an average of a little more than 1 time state transition. In this case, the map from current state to goal state is almost finished directly in the decision-making process.

No matter how perfect the initial rules of SOAR agent are, it is still impossible to contain all the rules to be used in the decision-making process. The initial rules and operators are difficult to be set exactly the same with the actual situation, so rules can be added or revised in the learning process, and chunking is the most important learning mechanism in SOAR. Figure 10 shows the average chunking times in each decision-making process of all

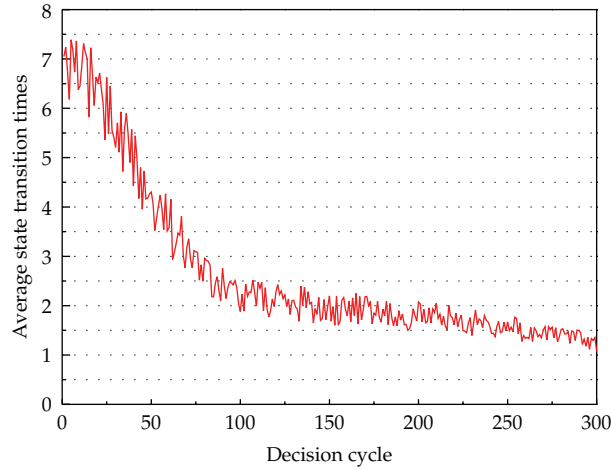


Figure 9: Average state transition times.

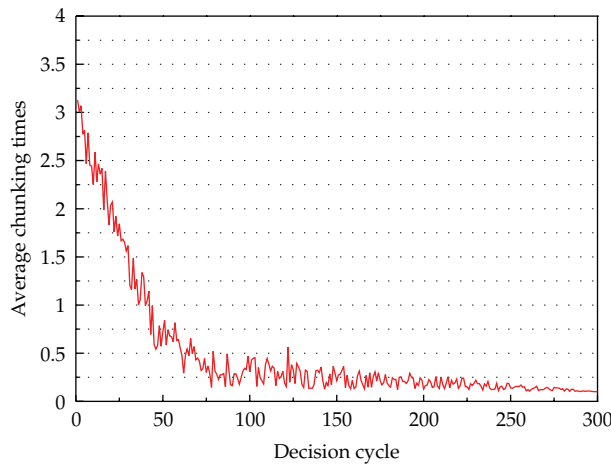


Figure 10: Average chunking times.

the chosen agents. The average chunking times of each decision-making is 3 in the beginning of the simulation, which indicate that the initial rules are insufficient to support choosing downstream route, so impasse may happen with a very high probability. The success of chunking rapidly increases the rules of agents to support decision-making. Therefore, the number of chunking decreases fast and gradually towards 0 along the decision-making process. This is consistent with the decreasing tendency of state transition times in Figure 9. When the goal state can be reached by one state transition, this indicates that the map from the perception of external traffic situation and the guidance information shown on VMS to the operator of choosing the downstream route is finished directly, and the chunking times are 0 in this mapping process.

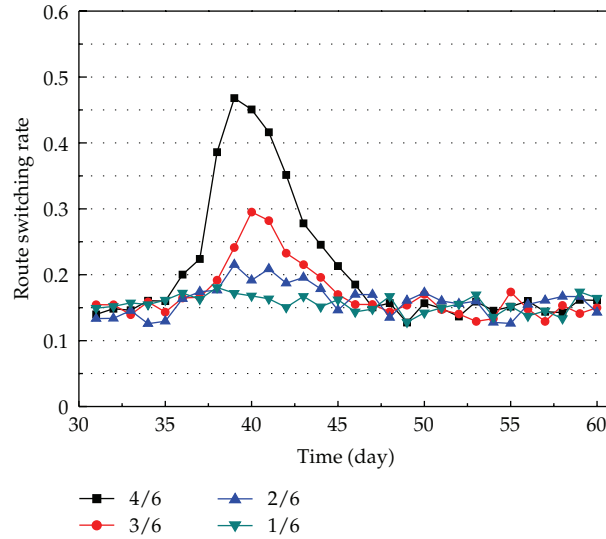


Figure 11: Route switching rate of old drivers when new drivers are added.

6.2.3. The Effect of New Drivers Adding

The above simulation experiments investigate the changing laws of the guidance compliance behaviors of drivers on s_1 after adding VMS. In the beginning of simulation, all the agents are newly added, and they are unaware of the guidance system. After several decision-making processes after passing VMS, a set of cognitive experience about VMS is formed, and agents finish choosing downstream route based on this cognitive experience. In our simulation, in order to finish repeated decision-making process all the agents queue again to enter s_1 after they exit from s_7 . Thus, very few new agents are added under this condition. Based on the simulation situation of the previous 30 days, 4 more simulation experiments are conducted to simulate the traffic guidance situations from 5 am to 10 am of each day from the 30th to 60th days. 1/6, 2/6, 3/6, and 4/6 of new agents are added to the four simulations, respectively from the 35th day to inspect the effect of adding new agents on the guidance compliance behavior of the old ones. Figure 11 shows the variation of the route switching rates of the old agents after different percentages of new ones are added.

Adding 1/6 of new agents has little impact on the guidance compliance behavior of old ones, their route switching rate still keeps stable. When the percentage increases to 3/6, the interferences of new agents to the decision-making process of old ones are obvious, and when 4/6 are added, the old agents almost recomplete one learning process, which is similar to the learning of new agents (Figure 8). Their largest route switching rate approaches 0.5 and it takes 10 simulation days to reach stability. The above results indicate that the guidance compliance behaviors of drivers have a certain anti-interference ability when the guidance system reaches stability; however, in order to maintain stability, the adding proportion of new drivers should be lower than 1/2 in this simulation.

7. Conclusions

In this paper, the formation mechanism and variation law of drivers' guidance compliance behavior are studied based on SOAR cognitive architecture. Traffic guidance system is a very

large and complex system, and the study of guidance compliance behavior is a fundamental problem in achieving the functions and goals of guidance system. This paper begins with the analysis of individual driver, wherein the perception, memory, decision-making, and learning of drivers' guidance compliance behavior are described in detail based on SOAR architecture. The guidance effect, guidance compliance law, learning law, and the anti-interference ability of the behavior are analyzed through simulation. As there are many factors affecting the guidance compliance behavior of drivers, apart from the impact of driver individual described in this paper, the release mode and display approach of guidance information, and the position of VMS all have a significant effect on guidance compliance behavior, and further research on these aspects will be conducted in the future.

Acknowledgments

The research described in this paper was substantially supported by Grants (50908155, 70971094) from the National Natural Science Foundation of China, and the Young Teachers Foundation Project (20090032120032) supported by the Doctorate in Higher Education Institutions of Ministry of Education. Meanwhile, the authors appreciate the help and comments from editors and reviewers.

References

- [1] G. He, *ITS System Engineering Introduction*, Chinese Railway Press, Beijing, China, 2004.
- [2] H. Huang, "Dynamic modeling of urban transportation networks and analysis of its travel behaviors," *Chinese Journal of Management*, vol. 2, pp. 18–22, 2005.
- [3] M. Wardman, P. W. Bonsall, and J. D. Shires, "Driver response to variable message signs: a stated preference investigation," *Transportation Research C*, vol. 5, no. 6, pp. 389–405, 1997.
- [4] P. Thakuriah and A. Sen, "Quality of information given by advanced traveler information systems," *Transportation Research C*, vol. 4, no. 5, pp. 249–266, 1996.
- [5] Y. Wang, M. Papageorgiou, G. Sarros, and W. J. Knibbe, "Real-time route guidance for large-scale express ring-roads," in *Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC '06)*, pp. 224–229, September 2006.
- [6] F. P. Deflorio, "Evaluation of a reactive dynamic route guidance strategy," *Transportation Research C*, vol. 11, no. 5, pp. 375–388, 2003.
- [7] Y. Yin and H. Yang, "Simultaneous determination of the equilibrium market penetration and compliance rate of advanced traveler information systems," *Transportation Research A*, vol. 37, no. 2, pp. 165–181, 2003.
- [8] H. S. Mahmassani and Y. H. Liu, "Dynamics of commuting decision behaviour under advanced traveler information systems," *Transportation Research C*, vol. 7, no. 2-3, pp. 91–107, 1999.
- [9] R. C. Jou, S. H. Lam, Y. H. Liu, and K. H. Chen, "Route switching behavior on freeways with the provision of different types of real-time traffic information," *Transportation Research A*, vol. 39, no. 5, pp. 445–461, 2005.
- [10] S. Bekhor and J. N. Prashker, "GEV-based destination choice models that account for unobserved similarities among alternatives," *Transportation Research B*, vol. 42, no. 3, pp. 243–262, 2008.
- [11] C. F. Daganzo and Y. Sheffi, "On stochastic models of traffic assignment," *Transportation Science*, vol. 11, no. 3, pp. 253–274, 1977.
- [12] D. Watling, "User equilibrium traffic network assignment with stochastic travel times and late arrival penalty," *European Journal of Operational Research*, vol. 175, no. 3, pp. 1539–1556, 2006.
- [13] P. W. Bonsall and M. Joint, "Driver compliance with route guidance advice. The evidence and its implications," in *Proceedings of the Vehicle Navigation & Information Systems Conference*, vol. 2, pp. 47–59, October 1991.
- [14] M. Cummings, "Electronic signs strategies and their benefits," in *Proceedings of the 7th International Conference on Road Traffic Monitoring and Control*, pp. 141–144, London, UK, 1994.
- [15] S. Tarry and A. Graham, "The role of evaluation in ATT development. IV: evaluation of ATT systems," *Traffic Engineering and Control*, vol. 36, no. 12, pp. 688–693, 1995.

- [16] J. Swann, I. W. Routeledge, J. Parker, and S. Tarry, "Result of practical applications of variable message signs(VMS):A64/AI accideng reduction scheme and Forth Estuary Driver Information and Control System(FEDICS)," in *Proceedings of the Seminar G Held at the 23rd PTRC European Transport Forum*, pp. 149–167, 1995.
- [17] A. Erke, F. Sagberg, and R. Hagman, "Effects of route guidance variable message signs (VMS) on driver behaviour," *Transportation Research F*, vol. 10, no. 6, pp. 447–457, 2007.
- [18] H. N. Koutsopoulos and H. Xu, "An information discounting routing strategy for advanced traveler information systems," *Transportation Research C*, vol. 1, no. 3, pp. 249–264, 1993.
- [19] K. Ozbay and B. Bartın, "Estimation of economic impact of VMS route guidance using microsimulation," *Research in Transportation Economics*, vol. 8, pp. 215–241, 2004.
- [20] J. L. Adler, "Investigating the learning effects of route guidance and traffic advisories on route choice behavior," *Transportation Research C*, vol. 9, no. 1, pp. 1–14, 2001.
- [21] Y. Zhou and J. Wu, "The research on drivers' route choice behavior in the presence of dynamic traffic information," in *Proceedings of the IEEE Intelligent Transportation Systems Conference*, pp. 17–20, 2006.
- [22] J. Chen, P. Liu, and W. Wang, "Methods for en route parking guidance and information system survey in Nanjing," *Urban Transport of China*, vol. 4, pp. 79–83, 2006.
- [23] Y. Mo and K. Yan, "Analysis of the performance of the PGIS in metropolitan area," *Road Traffic and Safety*, vol. 5, pp. 33–36, 2007.
- [24] R. W. Allen, A. C. Stein, T. J. Rosenthal, D. Ziedman, J. F. Torres, and A. Halati, "Human factors simulation investigation of driver route diversion and alternate route selection using in-vehicle navigation systems," in *Proceedings of the Vehicle Navigation & Information Systems Conference*, vol. 2, pp. 9–26, October 1991.
- [25] R. Srinivasan and P. P. Jovanis, "Evaluation of the attentional demand of selected visual route guidance systems," in *Proceedings of the 6th 1995 Vehicle Navigation and Information Systems Conference*, pp. 140–146, August 1995.
- [26] W. Chen and P. J. Paul, "Analysis of a driver en-route guidance compliance and driver learning with ATIS using a travel simulation experiment," Research Report UCD-ITS-RR-97-12, Institute of Transportation Studies, University of California, Davis, Calif, USA, 1997.
- [27] K. Wachinger and D. Boehm-Davis, "Navigational preference and driver acceptance of advanced traveler information systems," in *Ergonomics and Safety of Intelligent Devices*, Y. Noy, Ed., pp. 345–362, 1997.
- [28] C. Lu and Y. Tan, "The simulation and analysis of urban transportation system complexity," *System Engineering*, vol. 23, pp. 84–87, 2005.
- [29] H. J. Huang, T. L. Liu, and H. Yang, "Modeling the evolutions of day-to-day route choice and year-to-year ATIS adoption with stochastic user equilibrium," *Journal of Advanced Transportation*, vol. 42, no. 2, pp. 111–127, 2008.
- [30] J. L. Adler and V. J. Blue, "A cooperative multi-agent transportation management and route guidance system," *Transportation Research C*, vol. 10, no. 5-6, pp. 433–454, 2002.
- [31] J. Wahle, A. L. C. Bazzan, F. Klügl, and M. Schreckenberg, *Anticipatory Traffic Forecast Using Multi-Agent Techniques, Traffic and Granular Flow*, Springer, Heidelberg, Germany, 1999.
- [32] H. Dia, "A conceptual framework for modeling dynamic driver behavior using intelligent agents," in *Proceedings of the 6th International Conference on Applications of Advanced Technologies in Transportation Engineering*, pp. 28–30, Singapore, 2000.
- [33] S. Nason and J. E. Laird, "Soar-RL: Integrating reinforcement learning with SOAR," *Cognitive Systems Research*, vol. 6, no. 1, pp. 51–59, 2005.
- [34] R. M. Jones, J. E. Laird, and P. E. Nielsen, "Automated intelligent pilots for combat flight simulation," in *Proceedings of the 10th Conference on Innovative Applications of Artificial Intelligence (IAAI '98)*, pp. 1047–1054, July 1998.
- [35] R. P. Marinier, J. E. Laird, and R. L. Lewis, "A computational unification of cognitive behavior and emotion," *Cognitive Systems Research*, vol. 10, no. 1, pp. 48–69, 2009.
- [36] J. E. Laird, A. Newell, and P. S. Rosenbloom, "SOAR: An architecture for general intelligence," *Artificial Intelligence*, vol. 33, pp. 1–64, 1987.
- [37] P. S. Rosenbloom, J. E. Laird, and A. Newell, "Knowledge-level learning in Soar," in *Proceedings of the 6th National Conference on Artificial Intelligence (AAAI '87)*, pp. 499–504, Morgan Kaufmann, Los Altos, Calif, USA, 1987.
- [38] A. Nuxoll and J. Laird, "A cognitive model of episodic memory integrated with a general cognitive architecture," in *Proceedings of the International Conference on Cognitive Modeling*, 2004.