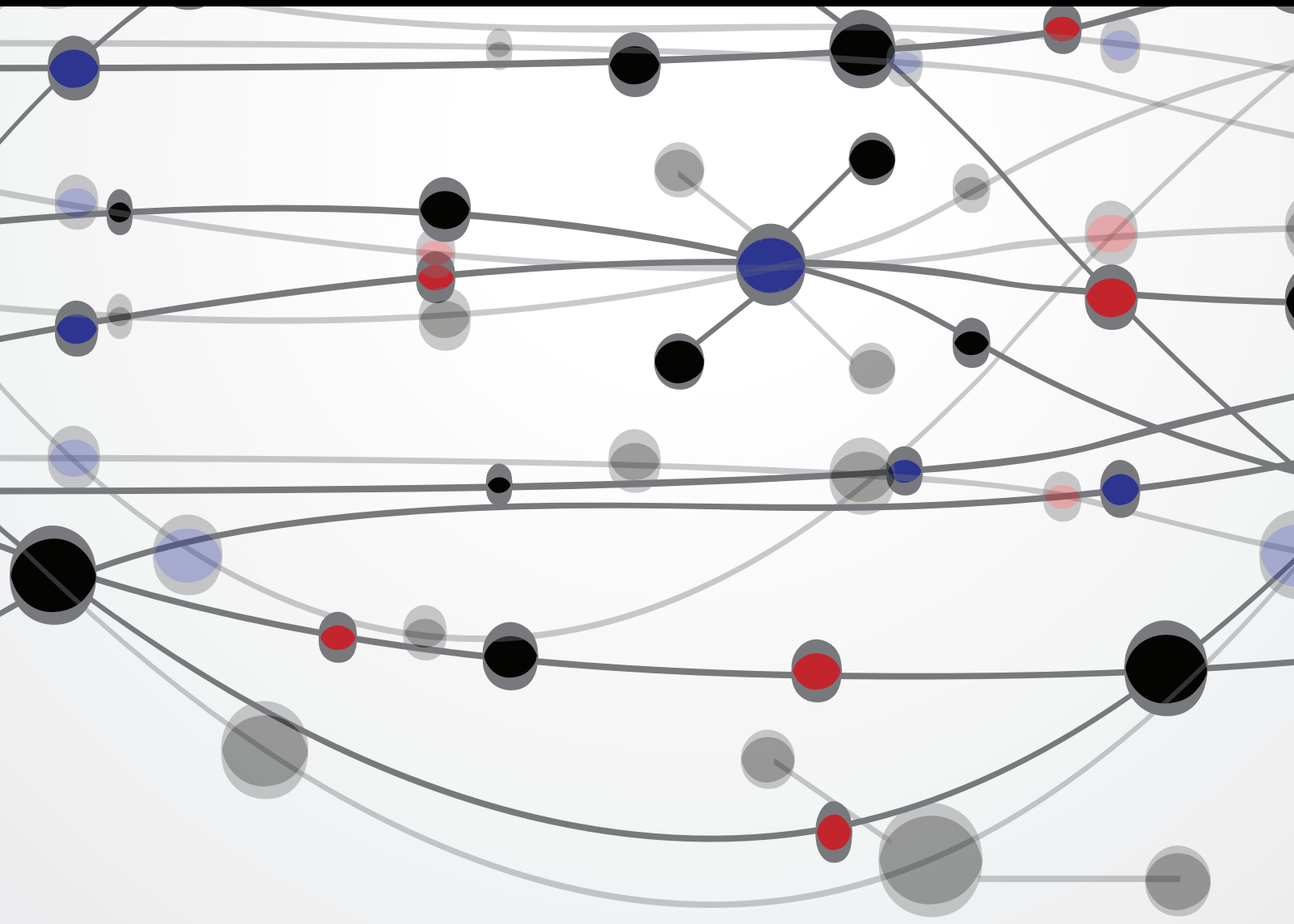
The Scientific World Journal

Swarm Intelligence and Its Applications

Guest Editors: Yudong Zhang, Praveen Agarwal, Vishal Bhatnagar, Saeed Balochian, and Jie Yan



Swarm Intelligence and Its Applications

The Scientific World Journal

Swarm Intelligence and Its Applications

Guest Editors: Yudong Zhang, Praveen Agarwal,
Vishal Bhatnagar, Saeed Balochian, and Jie Yan

Contents

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 528069, 3 pages
http://dx.doi.org/10.1155/2013/528069

Editorial

Swarm Intelligence and Its Applications

Yudong Zhang,[1] Praveen Agarwal,[2] Vishal Bhatnagar,[3] Saeed Balochian,[4] and Jie Yan[5]

[1] *School of Computer Science and Technology, Nanjing Normal University, Nanjing, Jiangsu 210023, China*
[2] *Department of Mathematics, Anand International College of Engineering, Near Kanota, Agra Road, Jaipur 303012, India*
[3] *Ambedkar Institute of Advanced Communication Technologies and Research, New Delhi, India*
[4] *Department of Electrical Engineering, Islamic Azad University, Gonabad Branch, Gonabad, Iran*
[5] *Suzhou University, Suzhou, Jiangsu, China*

Correspondence should be addressed to Yudong Zhang; zhangyudongnuaa@gmail.com

Received 16 September 2013; Accepted 16 September 2013

Swarm intelligence (SI) is the collective behavior of decentralized, self-organized systems, natural or artificial. SI systems are typically made up of a population of simple agents interacting locally with one another and with their environment. The inspiration often comes from nature, especially biological systems. The agents follow very simple rules, and although there is no centralized control structure dictating how individual agents should behave, local, and to a certain degree random, interactions between such agents lead to the emergence of "intelligent" global behavior, unknown to the individual agents. Natural examples of SI include ant colonies, bird flocking, animal herding, bacterial growth, and fish schooling.

Research in SI started in the late 1980s. Besides the applications to conventional optimization problems, SI can be employed in library materials acquisition, communications, medical dataset classification, dynamic control, heating system planning, moving objects tracking, and prediction. Indeed, SI can be applied to a variety of fields in fundamental research, engineering, industries, and social sciences.

The main objective of this special issue is to provide the readers with a collection of high quality research articles that address the broad challenges in application aspects of swarm intelligence and reflect the emerging trends in state-of-the-art algorithms.

The special issue received 42 high quality submissions from different countries all over the world. All submitted papers followed the same standard (peer-reviewed by at least three independent reviewers) as applied to regular submissions to "this journal". Due to the limited space, 15 papers were finally included. The primary guideline was to demonstrate the wide scope of SI algorithms and applications in various aspects. Besides, mathematically oriented papers with promising potential in practical problems were also included.

The paper authored by Y.-L. Wu et al. (National Chiao Tung University and Ming Chuan University) presents an integer programming model of the studied problem by considering how to select materials in order to maximize the average preference and the budget execution rate under some practical restrictions including departmental budget and limitation of the number of materials in each category and each language. They propose a discrete particle swarm optimization (DPSO) with scout particles, design an initialization algorithm and a penalty function to cope with the constraints, and employ the scout particles to enhance the exploration within the solution space.

In the paper by Z. Yin et al. (Harbin Institute of Technology), they propose an efficient multiuser detector based on a suboptimal code mapping multiuser detector and artificial bee colony algorithm (SCM-ABC-MUD) and implement the proposed algorithm in direct-sequence ultrawideband (DS-UWB) systems under the additive white Gaussian noise (AWGN) channel.

M. S. Uzer et al. (Selçuk University) offer a hybrid approach that uses the artificial bee colony (ABC) algorithm for feature selection and support vector machines for classification. For the diagnosis of hepatitis, liver disorders, and diabetes datasets from the UCI database, the proposed system reached classification accuracies of 94.92%, 74.81%, and 79.29%, respectively.

Another paper is by M. Karakose (Fırat University) and U. Cigdem (Gaziosmanpaşa University). It proposes a new approach for improvement of DNA computing with adaptive parameters towards the desired goal using quantum-behaved particle swarm optimization (QPSO). Experimental results obtained with MATLAB and FPGA demonstrate ability to provide effective optimization, considerable convergence speed, and high accuracy according to DNA computing algorithm.

In the paper by Y. Celik (Karamanoglu Mehmetbey University) and E. Ulker (Selcuk University), their research proposes an improved marriage in honey bees optimization (IMBO) by adding Levy flight algorithm for queen mating flight and neighboring for worker drone improving. The IMBO algorithm's performance and its success are tested on the well-known six unconstrained test functions and compared with other metaheuristic optimization algorithms.

M. Baygin (Ardahan University) and M. Karakose (Fırat University) study a new approach of immune system-based optimal estimate for dynamic control of group elevator systems. The method is mainly based on estimation of optimal way by optimizing all calls with genetic, immune system and DNA computing algorithms, and it is evaluated with a fuzzy system. With dynamic and adaptive control approach in this study, a significant progress on group elevator control systems has been achieved in terms of time and energy efficiency according to traditional methods.

The paper by M. Karakose (Fırat University) proposes a reinforcement-learning based artificial immune classifier. The proposed new approach has many contributions according to other methods in the literature such as effectiveness, less memory cell, high accuracy, speed, and data adaptability. Some benchmark data and remote image data are used for experimental results. The comparative results with supervised/unsupervised based artificial immune system, negative selection classifier, and resource limited artificial immune classifier are given to demonstrate the effectiveness of the proposed new method.

In their paper, T. J. Choi et al. (Sungkyunkwan University) and (Daegu Gyeongbuk Institute of Science and Technology) present an adaptive parameter control DE algorithm. The control parameters of each individual are adapted based on the average of successfully evolved individuals' parameter values using the Cauchy distribution. The experimental results show that their proposed algorithm is more robust than the standard DE algorithm and several state-of-the-art adaptive DE algorithms in solving various unimodal and multimodal problems.

In the paper by R.-J. Ma et al. (Southwest Jiaotong University and CSR Qishuyan Institute Co., Ltd.), the authors present an integral mathematical model and particle swarm optimization (PSO) algorithm based on the life cycle cost (LCC) approach for the heating system planning (HSP) problem. The results show that the improved particle swarm optimization (IPSO) algorithm can more preferably solve the HSP problem than PSO algorithm.

In the paper by M. Tang et al. (National University of Defense Technology and Université Pierre et Marie Curie), they report that the flocking has some negative effects on the human, as the infectious disease H7N9 will easily be transmitted from the denser flocking birds to the human. Their paper focuses on the H7N9 virus transmission in the flocking birds and from the flocking birds to the human. Some interesting results have been shown: (1) only some simple rules could result in an emergence such as the flocking; (2) the minimum distance between birds could affect H7N9 virus transmission in the flocking birds and even affect the virus transmissions from the flocking birds to the human.

Y. Wang et al. (China University of Petroleum) present a memory-based multiagent coevolution algorithm for robust tracking the moving objects. Each agent can remember, retrieve, or forget the appearance of the object through its own memory system by its own experience. Experimental results show that their proposed method can deal with large appearance changes and heavy occlusions when tracking a moving object.

The paper by Q. Ni and J. Deng (Southeast University and Soochow University) analyzes the performance of PSO with the proposed random topologies and explores the relationship between population topology and the performance of PSO from the perspective of graph theory characteristics in population topologies. Further, in a relatively new PSO variant which named logistic dynamic particle optimization, an extensive simulation study is presented to discuss the effectiveness of the random topology and the design strategies of population topology.

Y. Zhou and H. Zheng (Guangxi University for Nationalities, Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis) propose a novel complex valued cuckoo search algorithm. They use complex-valued encoding to expand the information of nest individuals and denote the gene of individuals by plurality. The value of independent variables for objective function is determined by modules, and a sign of them is determined by angles. The position of nest is divided into real part gene and imaginary gene. Six typical functions are tested, and the usefulness of the proposed algorithm is verified.

The paper by R. Alwee et al. (Universiti Teknologi Malaysia) introduces a hybrid model that combines support vector regression (SVR) and autoregressive integrated moving average (ARIMA) to be applied in crime rates forecasting. Particle swarm optimization is used to estimate the parameters of the SVR and ARIMA models. The experimental results show that their proposed hybrid model is able to produce more accurate forecasting results as compared to the individual models.

Finally, K. S. Lim et al. (Universiti Teknologi Malaysia, Universiti Malaysia Pahang, and University of Malaya) describe an improved Vector Evaluated Particle Swarm Optimization algorithm by incorporating the nondominated solutions as the guidance for a swarm rather than using the best solution from another swarm. The results suggest that the improved Vector Evaluated Particle Swarm Optimization algorithm has impressive performance compared with the conventional Vector Evaluated Particle Swarm Optimization algorithm.

Aknowledgments

We would like to express their gratitude to all of the authors for their contributions, and the reviewers for their effort providing valuable comments and feedback. We hope this special issue offers a comprehensive and timely view of the area of applications of swarm intelligence and that it will offer stimulation for further research.

Yudong Zhang
Praveen Agarwal
Vishal Bhatnagar
Saeed Balochian
Jie Yan

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 636484, 11 pages
http://dx.doi.org/10.1155/2013/636484

Research Article

Discrete Particle Swarm Optimization with Scout Particles for Library Materials Acquisition

Yi-Ling Wu,[1] Tsu-Feng Ho,[2] Shyong Jian Shyu,[2] and Bertrand M. T. Lin[1]

[1] *Institute of Information Management, National Chiao Tung University, Hsinchu 30010, Taiwan*
[2] *Department of Computer Science and Information Engineering, Ming Chuan University, Taoyuan 33348, Taiwan*

Correspondence should be addressed to Tsu-Feng Ho; tfho@mail.mcu.edu.tw

Received 3 June 2013; Accepted 10 July 2013

Academic Editors: S. Balochian, V. Bhatnagar, and Y. Zhang

Materials acquisition is one of the critical challenges faced by academic libraries. This paper presents an integer programming model of the studied problem by considering how to select materials in order to maximize the average preference and the budget execution rate under some practical restrictions including departmental budget, limitation of the number of materials in each category and each language. To tackle the constrained problem, we propose a discrete particle swarm optimization (DPSO) with scout particles, where each particle, represented as a binary matrix, corresponds to a candidate solution to the problem. An initialization algorithm and a penalty function are designed to cope with the constraints, and the scout particles are employed to enhance the exploration within the solution space. To demonstrate the effectiveness and efficiency of the proposed DPSO, a series of computational experiments are designed and conducted. The results are statistically analyzed, and it is evinced that the proposed DPSO is an effective approach for the studied problem.

1. Introduction

In recent years, the price inflation of library materials, the shrinking of library budget, and the growth of electronic resources continue to challenge the acquisition librarians [1]. Complicating the effects of these challenges is the growth of scholarly and popular publications. With the great increase in publications, the librarians have not only to acquire the latest and the preferred materials within the limited budget but also to take the collection policy into consideration. Walters [2] reports that the annual inflation rate of academic books and periodicals were 1.4 and 8.5 percent. The research planning and review committee of the Association of College and Research Libraries (ACRL) [3] develops the 2010 top ten trends in academic libraries and finds that many libraries will face the budget pressure in the near future. These reaffirm the fact that the materials acquisition problem is exacerbated by the difficulty of aligning the library offerings with patron needs under the budget pressure.

Over the past few decades, researches on materials acquisition have been conducted and implemented with a number of operations research based models and approaches. Beilby and Mott Jr. [4] develop a linear goal programming model for acquisition planning of academic libraries, and incorporate with multiple collection development goals such as acquiring an adequate number of titles (at least 7,500 but not more than 10,500 titles), not exceeding the total acquisition budget ($200,000), and/or limiting periodical expenditures to 60% of the total acquisition expenditures. Wise and Perushek [5] introduce another model that takes into account more goals, like reaching the minimum limit for each subject fund, not surpassing the maximum limit for each subject fund, and so forth. Later, Wise and Perushek [6] not only address an important claim that the suggestions of collection development librarians and faculties must be taken into consideration but also elaborate another model to reflect the opinion of librarians and faculties. Ho et al. [7] present a model that maximizes the average preference of patrons subject to both the acquisition cost and the number of materials in each category.

In most of the cases, academic libraries are positioned to acquire materials for multiple departments, for example, Science, Business, Engineering, and so forth, within the budget of each department. Goyal [8] proposes an operations

research model of funds allocation to different departments of a university. The objective of this model is to maximize the total social benefits conveyed by the funds exercised for the purchase of materials among all departments, and the constraints of this model are the lower and upper limits of fund for each department and the total funds available. Arora and Klabjan [9] point out the critical concern about fairness in materials acquisition of academic libraries. They provide a model for maximizing the usage in the future time period subject to the bounds on the number of materials of each category and the lower and the upper bounds on the budgets of the library units. Existing researches on materials acquisition assume a single total budget or multiple department budgets. This study will investigate the scenario where each individual department has its own budget limit for the preferred materials that are to be acquired. This type of budget plan will introduce financial constraints that are much more complicated.

From the viewpoint of acquisition staffs, it is questionable if the patrons are satisfied with the decision outcome. Niyonsenga and Bizimana [10] indicate various factors related to the patron satisfactions with academic libraries services, such as a list of new acquisitions, lending services, serial collection. In this paper, we adopt the patron preferences of acquisitions to reflect the patron satisfactions. To allocate the budget as fairly as possible, we assume that the preferences are obtained from the patrons of all departments due to the different interests of the departments. Besides, a low budget execution rate may lead to a budget cut in the next fiscal year. Librarians sometimes are on the horns of a dilemma whether to purchase the less preferred materials or cause a low budget execution rate. Therefore, we concentrate on how to select materials to be acquired in order to maximize the average preference as well as the budget execution rate under the real-world restrictions including departmental budget and limitation of the number of materials in each category.

In the view of computational complexity, the materials acquisition problem is a generalized version of the knapsack problem which is known to be computationally intractable [11]. In other words, it is extremely time consuming and even unlikely to find an optimal solution when the problem size is large. By far, metaheuristics, such as genetic algorithm, ant colony optimization, and particle swarm optimization are successfully applied to cope with many hard optimization problems with impressive performances in obtaining solutions with in an effective and efficient way [12, 13]. This paper is devoted to tackling the studied problem by particle swarm optimization (PSO) that has earned a good reputation by the trustworthy merits including simplicity, efficiency, and effectiveness in producing quality solutions [14, 15]. Furthermore, to avoid premature convergence, we introduce a discrete particle swarm optimization with *scout particles*, introduced by Silva et al. [16], to enhance the exploration capability of the adopted swarms.

The rest of this paper is organized as follows. In Section 2, a mathematical model of the materials acquisition problem with departmental demands is proposed and followed by a greedy algorithm. Section 3 presents the fundamental concept and structure of the discrete particle swarm optimization (DPSO). In Section 4, we depict how the proposed DPSO with scout particles is tailored for the characteristics of the studied problem. A computational study is carried out to examine the performances of the proposed solution approaches. Our experimental settings and results of DPSO are presented in Section 5. We summarize the results of this study and give some concluding remarks in Section 6.

2. Problem Statements and Greedy Algorithm

A formal specification of the materials acquisition problem is presented in this section. Then, an integer programming model is developed to formulate the problem considered in a mathematical way.

2.1. Problem Specification. Consider a set of n materials to be acquired and a set of m departments. Each material is associated with a cost c_i and a preference value p_{ij} recommended by each department j for $1 \le i \le n$ and $1 \le j \le m$. Each department owns an amount B_j of budget for $1 \le j \le m$. Since one material may be recommended by more than one department, the acquisition cost would be apportioned by these recommending departments in proportion to their preferences. For instance, if a material with cost 100 is acquired to meet the recommendations from two departments j and j' with preferences 0.9 and 0.6, then departments j and j' should pay 40 ($= 100 \times (0.9/(0.9 + 0.6))$) and 60 ($= 100 \times (0.6/(0.9 + 0.6))$), respectively, from their budgets B_j and $B_{j'}$. We denote the actual expense by department j for material i as e_{ij}. To meet the acquisition requirements from various departments, q written languages (e.g., English, Japanese, Chinese, etc.) and r classified categories (e.g., Art, Science, Design, etc.) are considered such that the amount of materials belongs to a certain language and a specific category may be restricted into a range. In addition, the authority would expect the remainder of budget B_j, once granted, for department j to be the less the better after allocation. We thus define the *execution rate* to be the actual expenses of all departments divided by the budget of all departments.

The decision is to determine which materials should be acquired and which departments should cover the cost associated with these materials under the constraints of departmental budgets and the limitation of the amounts in each written language and each category. The objective is to maximize the combination of the average preference and the budget execution rate.

In Table 1, we summarize the notations that will be used in the integer programming model throughout the paper.

2.2. Problem Formulation. The materials acquisition problem is mathematically formulated as the following integer programming model:

$$\text{maximize} \quad O(x) = \rho \times \left(\frac{\sum_{j=1}^{m} \left(\sum_{i=1}^{n} x_{ij} p_{ij} / \sum_{i=1}^{n} x_{ij} \right)}{m} \right) + (1 - \rho) \left(\frac{\sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij} e_{ij}}{\sum_{j=1}^{m} B_j} \right) \tag{1}$$

Table 1: Notations.

Variable	Description
n	Number of materials
m	Number of departments
q	Number of categories
r	Number of languages
p_{ij}	Preference for material i recommended by department j, for $1 \le i \le n$ and $1 \le j \le m$
c_i	Cost of material i, for $1 \le i \le n$
B_j	Budget limit of department j, for $1 \le j \le m$
LU_l	Upper bound on the number of materials in language l, for $1 \le l \le r$
LL_l	Lower bound on the number of materials in language l, for $1 \le l \le r$
a_{il}	$a_{il} = 1$ if material i is in language q; $a_{il} = 0$ otherwise, for $1 \le i \le n$ and $1 \le l \le r$
CU_k	Upper bound on the number of materials in category k, for $1 \le k \le q$
CL_k	Lower bound on the number of materials in category k, for $1 \le k \le q$
b_{ik}	$b_{ik} = 1$ if material i belongs to category k; $b_{ik} = 0$ otherwise, for $1 \le i \le n$ and $1 \le l \le r$
x_{ij}	Decision variable: $x_{ij} = 1$ if material i is acquired for department j from which the cost will be charged; $x_{ij} = 0$ otherwise, for $1 \le i \le n$ and $1 \le j \le m$
z_i	Auxiliary variable: $z_i = 1$, if $\sum_{j=1}^{m} x_{ij} > 0$; otherwise $z_i = 0$, for $1 \le i \le n$ (z_i reveals whether material i is acquired or not)
e_{ij}	Actual expenses of material i by department j, for $1 \le i \le n$ and $1 \le j \le m$

subject to

$$e_{ij} \ge \left(\frac{x_{ij} p_{ij}}{\sum_{\hat{j}=1}^{m} x_{i\hat{j}} p_{i\hat{j}}} \right) \times c_i \quad \text{for } 1 \le i \le n,\ 1 \le j \le m, \tag{2}$$

$$\sum_{i=1}^{n} e_{ij} \le B_j \quad \text{for } 1 \le j \le m, \tag{3}$$

$$\sum_{i=1}^{n} x_{ij} - z_i M \le 0 \quad \text{for } 1 \le i \le n, \tag{4}$$

$$\sum_{i=1}^{n} x_{ij} + (1 - z_i) M > 0 \quad \text{for } 1 \le i \le n, \tag{5}$$

$$\sum_{i=1}^{n} z_i a_{il} \le LU_l \quad \text{for } 1 \le l \le r, \tag{6}$$

$$\sum_{i=1}^{n} z_i a_{il} \ge LL_l \quad \text{for } 1 \le l \le r, \tag{7}$$

$$\sum_{i=1}^{n} z_i b_{ik} \le CU_k \quad \text{for } 1 \le k \le q, \tag{8}$$

$$\sum_{i=1}^{n} z_i b_{ik} \ge CL_k \quad \text{for } 1 \le k \le q. \tag{9}$$

The objective function (1) is to maximize the weighted sum of the average preference and the budget execution rate, where ρ, $0 \le \rho \le 1$, is a parameter controlling the degree of importance between these two terms. The actual expense of material i apportioned by department j (e_{ij}) is given in constraints (2), where all the cost of materials will be apportioned according to the proportion of the preference (p_{ij}). Constraints (3) confine that the expense of any department j do not exceed its budget (B_j). To ease the amount computation of the acquired materials, we introduce an auxiliary variable z_i, which is 1 (0) if $\sum_{j=1}^{m} x_{ij} > 0$ (otherwise), to show whether material i is acquired or not. Using a sufficiently large positive number M, constraints (4) and (5) are deliberately designed to obtain the proper value of z_i. If $\sum_{j=1}^{m} x_{ij} \le 0$, constraint (4) becomes irrelevant, where z_i may be either 0 or 1, but constraints (5) pledge $z_i = 0$, which indicates that material i is not acquired. On the contrary ($\sum_{j=1}^{m} x_{ij} > 0$), constraints (5) would be redundant, yet constraint (4) promises $z_i = 1$, which means that material i is acquired. If the material i is acquired ($z_i = 1$); then constraints (6) and (7) will force the number of acquired materials in each language l to be larger than or equal to the lower bounds and not to exceed the upper bounds. If material i is not acquired ($z_i = 0$), constraints (6) and (7) will assure the number of acquired materials in each language l included no material i. Constraints (11) and (12) are similarly defined to abide by the lower bound and upper bound specified on the number of materials in each category k.

2.3. Greedy Algorithm. A greedy solution method, denoted by Algorithm Greedy as shown in the Algorithm 1, is designed to be the comparison counterpart for other approaches. First, to decide if each material i will be acquired or not, all the materials are sorted in nonincreasing order of the ratio $(\sum_{j=1}^{m} p_{ij})/c_i$. We thus assume the materials are reindexed in accordance with this sequencing rule. The first material, the one that attains the maximum $(\sum_{j=1}^{m} p_{ij})/c_i$ ratio, will be considered if the following two conditions are satisfied: (1) the upper bound on the number of languages LU_l is not exceeded, and (2) the upper bound on the number of categories CU_k is

```
Algorithm Greedy:
Sort all materials in nonincreasing order of the ratio (∑_{j=1}^{m} p_ij)/c_i;
for i := 1 to n do
   while (the upper bound on the number of materials in language LU_l is not exceeded, 1 ≤ l ≤ r)
      while (the upper bound on the number of materials in category CU_k is not exceeded, 1 ≤ k ≤ q)
            Sort the departments that propose material i in nonincreasing order of p_ij
            Let (j_1, j_2, ..., j_m) be the sorted sequence;
            for j := 1 to m do
               if (budget B_j is not exhausted)
                     Set x_ij = 1
            endfor
            Calculate the residual budget of all departments j with x_ij = 1 by
            deducing the apportioned cost of material i.
      endwhile
   endwhile
endfor
```

Algorithm 1: Greedy solution method.

not exceeded. Next, to determine which departments will apportion the cost of material i, all departments are sorted in nonincreasing order of p_{ij}, and let $(j_1, j_2, \ldots, j_m)$ be the sorted list. Material i will be acquired by department j ($x_{ij} = 1$), if the budget of this department is not exceeded.

3. Related Works of PSO

This section presents an overview on particle swarm optimization and describes two widely used topologies. What follows is a review on how to handle constraints and how to avoid premature convergence.

3.1. PSO. Particle swarm optimization (PSO) [14], introduced by Kennedy (a social psychologist) and Eberhart (an electrical engineer) in 1995 as an optimization method, is inspired by the observation on behavior of flocking birds and schooling fish. With the simplicity and lessened computation loads, PSO has been widely applied to many research areas, such as clustering and classification, communication networks, and scheduling [15, 17–19].

In foraging, birds flock together and arrange themselves in specific shapes or formations by sharing their information about food sources. The movement of each particle will be influenced by the experiences of itself and the peers. In the process of optimization, each particle s of flock S is associated with a *position*, a *velocity*, and a *fitness value*. A position, which is a vector in a search space, represents a potential solution to an optimization problem; a velocity, which is a vector, represents a change in the position; a fitness value, which is computed by the objective function, indicates how well the particle solves the problem.

To find an approximate solution, each particle s determines its movement iteratively by learning from its own experience and communication with its neighbors. The mechanism of coordination is encapsulated by the *velocity control* over all particles at each iteration t of the algorithm. For each particle s, the velocity at iteration $t + 1$ (V_s^{t+1}) is updated with (10), where P_s^t denotes the solution found by (position of) particle s at iteration t, $\overline{P}_s^t$ denotes the best solution found by particle s until iteration t, and $\widehat{P}_s^t$ denotes the best solution found by the neighbors of particle s. The *cognition learning* rate (c_1) and *social learning* rate (c_2) are introduced to control the influence of individual experience and their neighbors' experience, respectively. At the next iteration $t + 1$, the position of each particle is updated by (11). One has

$$V_s^{t+1} = V_s^t + c_1 r_1 \left(\overline{P}_s^t - P_s^t\right) + c_2 r_2 \left(\widehat{P}_s^t - P_s^t\right), \tag{10}$$

$$P_s^{t+1} = P_s^t + V_s^{t+1}. \tag{11}$$

For discrete optimization problems, Kennedy and Eberhart [20] also introduce a binary particle swarm optimization that changes the concept of velocity from adjustment of the position to the probability that determines whether a bit of a solution becomes one or zero. The velocity of each particle s at iteration t, V_s^{t+1}, is squashed in sigmoidal function as shown in (12); the position updating function is replaced by (13), where rand() is a random number drawn from the interval $[0, 1]$. One has

$$S\left(V_s^{t+1}\right) = \frac{1}{1 + e^{-(V_s^{t+1})}}, \tag{12}$$

$$P_s^{t+1} = \begin{cases} 1 & \text{if rand}() < S\left(V_s^{t+1}\right), \\ 0 & \text{otherwise,} \end{cases} \tag{13}$$

To better balance the exploration and exploitation, several variants of PSO algorithm have been proposed in the literature. A widely used method, proposed by Eberhart and Shi [21], is to introduce an *inertia weight* (w) to the velocity updating function shown in (14). The inertia weight is used

to adjust the influence of the current velocity on the new velocity:

$$V_s^{t+1} = wV_s^t + c_1 r_1 \left(\overline{P}_s^t - P_s^t\right) + c_2 r_2 \left(\widehat{P}_s^t - P_s^t\right). \quad (14)$$

3.2. Communication Topology. In the literature, several communication topologies have been extensively studied. Poli et al. [22] classify the communication structures into two categories: static topologies and dynamic topologies. Static topologies are that the number of neighbors does not change at all iterations of a run; dynamic topologies, on the other hand, are that the size of neighborhoods dynamically increases.

Local topology, global topology, and von Neumann topology are some well-known examples of static topology. As for dynamic topologies, the neighborhood size can be influenced by a dynamic hierarchy, a fitness distance ratio, or a randomized connection, just to name a few. The canonical PSO algorithm, proposed by Bratton and Kennedy [23], is equipped with global and local topologies.

A PSO with a global topology (or *gbest* topology) allows each particle to communicate with all other particles in the swarm, while a PSO with a local topology (or *lbest* topology) allows each particle to share information with only two other particles in the swarm. Therefore, a *gbest* PSO could lead to a faster convergence but might be trapped into a local optimal solution. Conversely, an *lbest* PSO could result in a slower rate of convergence but might be able to escape from a local optimal.

3.3. Constraint Handling. As reported in the literature, there are various different methods for handling constrained optimization problems. Several commonly used methods are based on penalty functions, rejection of infeasible solutions, repair algorithm, specialized operators, and behavioral memory [24–26]. In this paper, we focus on the method based on penalty function. Details concerning the penalty function for the studied problem are given in the next section.

When implementing penalty functions, the fitness evaluation for a solution is not just dependent on the objective function but incorporated the penalty function with the objective function. This method can be implemented as stationary or nonstationary. If there is an infeasible solution, the stationary penalty function simply adds a fixed penalty. Contrary to the stationary one, the nonstationary function adds a floating penalty which changes the penalty value according to the violated constrains and the iterations number. Parsopoulos and Vrahatis [25] note that the results obtained by nonstationary penalty functions are superior to the stationary one for the most of the time. A high penalty leads to a feasible solution even it is not approximate to the optimal solution, while a low penalty reduces the probability to obtain a feasible solution. Therefore, Coath and Halgamuge [24] point out that a fine-tuning of the parameters in the penalty function is necessary when using this method. The method based on the rejection of infeasible solution is to discard an infeasible solution even if it is closer to the optimal solution than some feasible ones. The repair algorithm, an extensively employed method in genetic algorithms (GA), is equipped to fix an infeasible solution, but the cost is more computationally expensive than other methods.

3.4. Avoiding Premature Convergence. Most of the global optimization methods suffer from premature convergence. One of the most used approaches to tackle this problem is to introduce diversity to the velocity or the position of a particle. As mutation operators are to the genetic algorithm, so is introduction of diversity to PSO algorithms. The focus of this paper is to introduce the diversity by employing scout particles. The details of how the proposed DPSO algorithm circumvents premature convergence are described in Section 4.

García-Villoria and Pastor [27] introduce the concept of diversity into the velocity updating function. The proposed dynamic diversity PSO (PSO-*c3dyn*) dynamically changes the diversity coefficients of all particles through iterations. The more heterogeneity of the population will be, the less diversity will be introduced to the velocity updating function, and vice versa. Blackwell and Bentley [28] incorporate diversity into the population by preventing the homogeneous particles from clustering tightly to each other in the search space. They provide collision-avoiding swarms that reduce the attraction of the swarm center and increase the coverage of a swarm in the search space. Silva et al. [16] attempt to apply the diversity to both the velocity and the population by a predator particle and several scout particles. A predator particle is intended to balance the exploitation and exploration of the swarm, while scout particles are designed to implement different exploration strategies. The closer the predator particle will be to the best particle, the higher probability of perturbation will be.

4. DPSO with Scout Particles

This section details how to tackle the materials acquisition problem by discrete particle swarm optimization with scout particles. The representation of a particle and the initialization method for the studied problem are described in Section 4.1. Then, Section 4.2 elaborates on the details of preventing premature convergence by deploying scout particles. Section 4.3 redefines a constraints handling mechanism for solving the constrained optimization problem.

4.1. Representation and Initialization. The solution of materials acquisition problem with n materials and m departments obtained by particle s at iteration t can be represented by an $n \times m$ binary matrix, proposed by Wu et al. [29], as shown in (15). Each entry of the matrix $(P_s^t)_{ij}$ indicates whether material i is acquired by department j or not. Note that each entry of the matrix $(P_s^t)_{ij}$ corresponds to the decision variable (x_{ij}) that was mentioned in Section 2.1:

$$P_s^t = \begin{bmatrix} P_{s(11)}^t & \cdots & P_{s(1m)}^t \\ \vdots & \ddots & \vdots \\ P_{s(n1)}^t & \cdots & P_{s(nm)}^t \end{bmatrix}. \quad (15)$$

Step 1. Compute the sum of the lower bound on the number of materials in language l ($\sum_{l=1}^{r} LL_l$); the sum of the lower bound on the number of materials in category k ($\sum_{k=1}^{q} CL_k$).
Step 2. **If** ($\sum_{l=1}^{r} LL_l$) < ($\sum_{k=1}^{q} CL_k$)
then randomly select a material i that is in language l,
randomly select a department j, and set $(P_s^t)_{ij} = 1$,
until all the lower bounds on the number of materials in all languages are reached.
Step 3. **If** ($\sum_{l=1}^{r} LL_l$) ≥ ($\sum_{k=1}^{q} CL_k$)
then randomly select a material i that belongs to category k,
randomly select a department j, and set $(P_s^t)_{ij} = 1$,
until all the lower bounds on the number of materials belong to all categories are reached.

Algorithm 2: Initialization procedure of DPSO.

The initial population is generated by setting a void velocity and randomly generated entries of matrix P_s^t for each particle s. To find feasible solutions for the initial population, an initialization procedure is designed and depicted in Algorithm 2. To determine which constraint should be satisfied first, the sum of the lower bounds on the numbers of materials in all languages $\sum_{l=1}^{r} LL_l$ and in all categories $\sum_{k=1}^{q} CL_k$ are computed. The one with less sum will be satisfied first by randomly selecting the material that belongs to language l or category k.

4.2. Constraints Handling. In the literature, repair operators and penalty functions are widely used approaches to handling constrained optimization problems. However, due to the computationally heavy load of repair operators, we focus on solely penalty functions. For each particle, the fitness value is evaluated by (16), where $O(x_{ij})$ is the objective value of the studied problem given in (1), and $H(x_{ij})$ is a penalty factor defined in (17). A feasible solution reflects its objective value as the fitness value, while an infeasible solution receives an objective value and a penalized value by (17). It can be seen from (17) that each term is associated with constrains (3), (6), (7), (8), and (9), as mentioned in Section 2.2. For instance, if a solution reports that the expense of any department j exceeds the budget B_j, addressed in constraints (3), then a positive penalty value can be subtracted from the fitness value to reflect the infeasibility. One has

$$F\left(x_{ij}\right) = O\left(x_{ij}\right) - H\left(x_{ij}\right), \tag{16}$$

$$\begin{aligned} H\left(x_{ij}\right) = {} & \sum_{j=1}^{m} \max\left\{0, \frac{\sum_{i=1}^{n} x_{ij} e_{ij} - B_j}{B_j}\right\} \\ & + \sum_{l=1}^{r} \max\left\{0, \frac{\sum_{i=1}^{n} y_i a_{il} - LU_l}{\left|\sum_{i=1}^{n} y_i a_{il} - LL_l\right|}\right\} \\ & + \sum_{l=1}^{r} \max\left\{0, \frac{LL_l - \sum_{i=1}^{n} y_i a_{il}}{\left|LU_l - \sum_{i=1}^{n} y_i a_{il}\right|}\right\} \\ & + \sum_{k=1}^{q} \max\left\{0, \frac{\sum_{i=1}^{n} y_i b_{ik} - UC_k}{\left|\sum_{i=1}^{n} y_i b_{ik} - CL_k\right|}\right\} \\ & + \sum_{k=1}^{q} \max\left\{0, \frac{CL_k - \sum_{i=1}^{n} y_i b_{ik}}{\left|CU_k - \sum_{i=1}^{n} y_i b_{ik}\right|}\right\}. \end{aligned} \tag{17}$$

4.3. Scout Particles. Premature convergence is a challenging problem faced by PSO algorithms throughout the optimization process. To avoid premature convergence in the DPSO algorithm for the studied problem, this paper employs scout particles to enhance the exploration. The concept is to send out scout particles to explore the search space and collect more extensive information of optimal solutions for other particles. If a scout particle finds a solution that is quite different from the best solution and the expected fitness value is better, the scout particle will share the information with some particles by affecting their velocities.

The DPSO procedure with scout particles is depicted in Figure 1. Firstly, in order to generate a feasible swarm, the particles are generated by the initialization procedure as mentioned in Section 4.1. Secondly, when the swarm has not yet converged, the regular particle s (P_s^t) flies through the search space by the following steps: fitness evaluation, velocity calculation, and position updating. If the swarm converges, on the other hand, scout particles $\widetilde{P}_s^t$ will be generated for exploration by randomly selecting a material to be acquired by all departments until the solution meets the lower bound and the upper bound on the number of languages and categories. In this paper, the convergence of DPSO is specified by the fitness variance.

The scout particles will share the information with the peer particles subject to a probability that depends on the velocity of each particle s. The larger velocity of particle s, the higher probability of the scout particle affecting the particle s by (19), where the diversity coefficient (c_3) is a prespecified parameter and r_3 is a random number drawn from the interval $[0, 1]$. Also, if the expected fitness value of the scout particle $\widetilde{P}_s^t$ is greater than the fitness of the best solution bound by particle $\overline{P}_s^t$, the particles will share

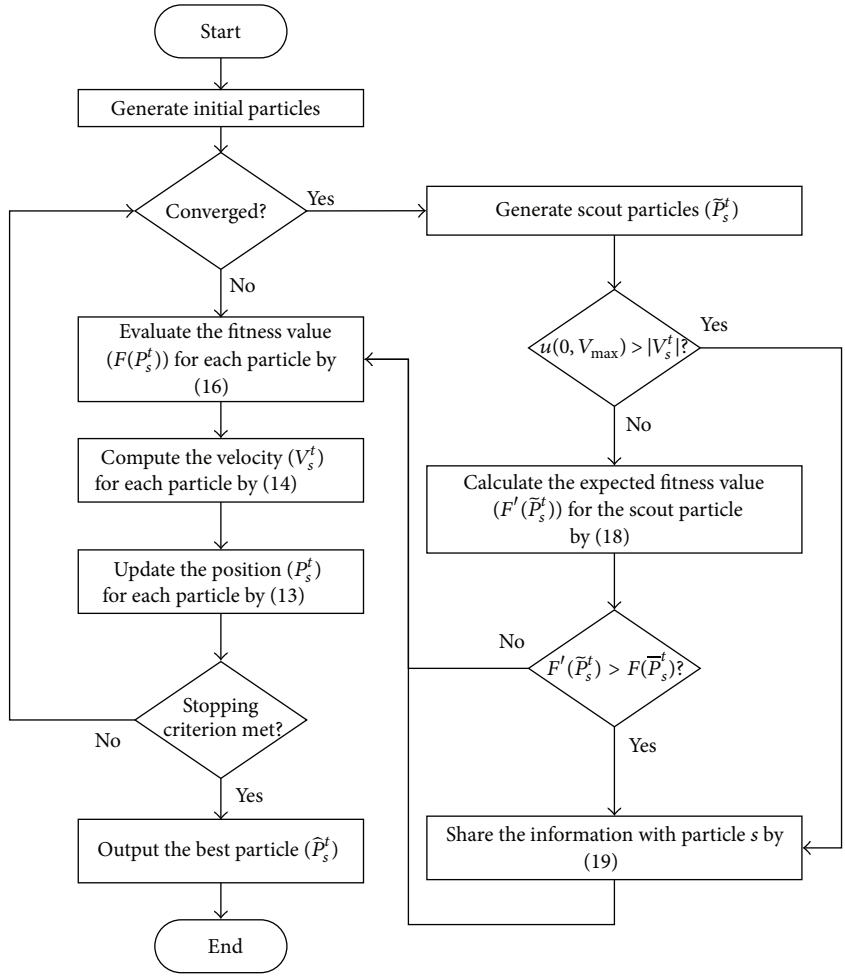


Figure 1: DPSO with scout particles.

information with other particles by (19). The expected fitness of the scout particle $\widetilde{P}_s^t$ is calculated by (18), where ρ is a nonnegative weight and p_i is the total preference of material i cast by all departments, $p_i = \sum_{j=1}^{m} p_{ij}$. One has

$$F'\left(\widetilde{P}_s^t\right) = \rho \times \frac{\sum_{i=1}^{n}\left(\widetilde{P}_s^t\right) p_i / \sum_{i=1}^{n}\left(\widetilde{P}_s^t\right)}{m} + (1-\rho) \times \frac{\sum_{i=1}^{n}\left(\widetilde{P}_s^t\right) c_i}{\sum_{j=1}^{m} B_j}, \tag{18}$$

$$V_s^{t+1} = wV_s^t + c_3 r_3 \left(\widetilde{P}_s^t - P_s^t\right). \tag{19}$$

5. Computational Experiments

To manifest the effectiveness and efficiency of the proposed DPSO of materials acquisition, a series of computational experiments were designed and conducted. The experiment setting and test instances are described in Section 5.1 and the computational results and analysis are given in Section 5.2.

5.1. Test Instances and Settings. Small-size test instances and large-size test instances are exhibited in Tables 2 and 3, respectively. The number of materials n, the number of departments m, the budget limits B_j of department j, the number of languages r, the lower bound on the number of materials LL_l in language l, the upper bound on the number of materials LU_l in language l, the number of categories q, the lower bound on the number of materials CL_k in category k, the upper bound on the number of materials CU_k in category k were tabulated. The small-size test instances (Case I), determined by the combinations of n, m, r, and q, were composed of 60 ($= 3 \times 5 \times 2 \times 2$) instances. The large-size test instances (Case II) were composed of 20 ($= 5 \times 2 \times 2$) instances, where n was 100,000.

The default values of the parameters in both DPSO and DPSO with scout particles algorithms were set as particle size $S = 30$, number of iterations $t = 500$, inertia weight $w = 0.9$, cognition learning rate $c_1 = 2.05$, social learning rate $c_2 = 2.05$, and diversity coefficient $c_3 = 0.5$. The number of scout particles was set to one. All of the programs were implemented in C#.net and run on a PC with an Intel Core i5-2400 3.1 GHz CPU and 4 G RAM. The stopping criteria of all

Table 2: Small-size test instances, Case I.

n	$\{m, \{B_j\}\}$	$\{r, \{LU_l\}, \{LL_l\}\}$	$\{q, \{CU_k\}, \{CL_k\}\}$
100	{1, {15000}}, {2, {6000, 9000}}, {3, {3000, 3000, 4000}}, {4, {3000, 3000, 4500, 4500}}, {5, {1500, 1500, 3000, 4500, 4500}}.	{2, {10, 20}, {5, 12}}, {3, {5, 10, 15}, {3, 3, 3}}.	{3, {6, 6, 12}, {3, 3, 6}}, {5, {3, 3, 6, 9, 9}, {3, 3, 3, 3, 3}}.
200	{1, {20000}}, {2, {8000, 12000}}, {3, {4000, 10000, 6000}}, {4, {3000, 3000, 3000, 4000, }}, {5, {2000, 2000, 2000, 8000, 4000, }}.	{2, {15, 25}, {5, 10, }}, {3, {10, 15, 15}, {5, 5, 5}}.	{3, {10, 10, 20}, {2, 4, 6}}, {5, {4, 4, 8, 12, 12}, {0, 0, 2, 3, 3}}.
300	{1, {30000}}, {2, {10000, 20000}}, {3, {6000, 6000, 18000, }}, {4, {6000, 6000, 9000, 9000}}, {5, {2000, 4000, 7000, 8000, 9000}}.	{2, {25, 35}, {10, 20}}, {3, {15, 15, 30}, {10, 10, 15}}.	{3, {10, 20, 30}, {5, 10, 10}}, {5, {10, 10, 10, 15, 15}, {5, 5, 5, 5, 5}}.

Table 3: Large-size test instances, Case II.

$\{m, \{B_j\}\}$ (unit of B_j: 10000)	$\{r, \{LU_l\}, \{LL_l\}\}$	$\{q, \{CU_k\}, \{CL_k\}\}$
{5, {80, 80, 100, 120, 120}}, {10, {40, 40, 40, 50, 50, 50, 50, 60, 60, 60}}, {15, {30, 30, 30, 30, 30, 33, 33, 33, 33, 33, 36, 36, 36, 36, 36}}, {20, {20, 20, 20, 20, 20, 25, 25, 25, 25, 25, 25, 25, 25, 25, 25, 30, 30, 30, 30}}, {25, {16, 16, 16, 16, 16, 18, 18, 18, 18, 18, 20, 20, 20, 20, 20, 22, 22, 22, 22, 22, 24, 24, 24, 24, 24}}.	{2, {4000, 6000}, {1000, 2000}}, {3, {3000, 3000, 4000}, {500, 500, 1000}}.	{5, {1000, 1000, 2000, 3000, 3000}, {200, 400, 800, 1000, 1200}}, {10, {600, 600, 800, 1000, 1000, 1000, 1200, 1400, 1400}, {100, 200, 300, 500, 500, 600, 700, 700, 800, 1000}}.

test cases were defined as no improvement on the incumbent solution can be achieved within 50 consecutive iterations.

5.2. Results and Analysis. To understand the effectiveness and efficiency of the proposed DPSO, we examine the four key features, including initialization, swarm topology, constraints handling, and scout particles. The following subsections detail the results and analysis (Tables 4–7). The rows labeled "*Average*" and "*Stdev*" in each table list the average and standard deviations of improvement and execution time for several observations. The next three rows in each table report the number of observations on the results of different DPSO algorithms for the test instances, the z-score of statistical test where the null hypothesis is that the different features of DPSO algorithm have the same improvement (or execution time), and the P value which is translated from z-score. Note that the number of observations for case I (resp., II) is set as 480 (resp., 160), the combinations 8 (= $2 \times 2 \times 2$) of features for 60 (resp., 20), for the purpose of evading the influence of other features. The significance level α is set at 0.05. Also, to facilitate a comparison of the effectiveness of the proposed DPSO algorithm across different test instances, the improvement in percentage over Algorithm Greedy, calculated as in (20), is employed instead of an absolute difference in objective value:

$$\text{improvement} = \left(\frac{\text{DPSO} - \text{greedy}}{\text{greedy}}\right)\%. \quad (20)$$

Table 4: Results of different initialization strategies on two test cases.

Case	Measure	Improvement		Execution time	
		Random	Greedy	Random	Greedy
I	Average	52.46%	52.11%	1.6455	1.5956
	Stdev	0.2805	0.2795	1.5258	1.3455
	Observations	480	480	480	480
	z-score	0.1942		0.5362	
	P value	0.8460		0.5918	
II	Average	71.32%	73.01%	779.9824	800.9922
	Stdev	0.3675	0.3722	318.31	324.77
	Observations	160	160	160	160
	z-score	−0.4090		−0.5843	
	P value	0.6825		0.5590	

5.2.1. Initialization. Results of different initialization strategies on the 60 small-size test instances (Case I) and 20 large-size test instances (Case II) are summarized in Table 4. The column labeled "*Random*" reports the results of DPSO algorithm that generates the initial swarms by the proposed initialization procedure in Section 4.1; the column labeled "*Greedy*" reports the results of DPSO algorithm that generates the initial swarms by both the abovementioned initialization procedure and the Algorithm Greedy in Section 2.3.

Table 5: Results of different swarm topologies on two test cases.

Case	Measure	Improvement		Execution time	
		Star	Ring	Star	Ring
I	Average	62.47%	42.10%	1.3193	1.9219
	Stdev	0.2244	0.2216	0.4285	1.9427
	Observations	480	480	480	480
	z-score	12.1029		−6.6364	
	P value	0.0000		0.0000	
II	Average	80.98%	63.35%	772.2568	808.7178
	Stdev	0.4147	0.2934	370.76	262.47
	Observations	160	160	160	160
	z-score	4.3889		−1.0202	
	P value	0.0000		0.3076	

Table 6: Results of different constraints handlings on two test cases.

Case	Measure	Improvement		Execution time	
		Accept	Reject	Accept	Reject
I	Average	52.81%	51.76%	1.6610	1.5802
	Stdev	0.2826	0.2773	1.4622	1.4135
	Observations	480	480	480	480
	z-score	0.5825		0.8703	
	P value	0.5603		0.3841	
II	Average	72.85%	71.48%	803.15	777.82
	Stdev	0.3693	0.3705	310.83	331.79
	Observations	160	160	160	160
	z-score	0.3304		0.7047	
	P value	0.7410		0.4810	

Table 7: Results of DPSO with and without scout particles on two test cases.

Case	Measure	Improvement		Execution time	
		Standard	Scout	Standard	Scout
I	Average	47.36%	57.21%	2.0065	1.2346
	Stdev	0.2443	0.3037	1.8070	0.7588
	Observations	480	480	480	480
	z-score	−5.5393		8.6285	
	P value	0.0000		0.0000	
II	Average	62.99%	81.34%	839.9745	741.0001
	Stdev	0.3015	0.4073	295.65	338.65
	Observations	160	160	160	160
	z-score	−4.5795		2.7849	
	P value	0.0000		0.0054	

It can be seen from Table 4 that the improvements achieved by two different initialization strategies are appealing. For case I, the improvement on the random strategy is slightly better than that on the greedy strategy (52.46% versus 52.11%); for case II, the greedy strategy performs slightly better (73.01% versus 71.32%). However, the difference in improvement between the "Random" and "Greedy" initializations for case I and case II yielded P values of 0.8460 and 0.6825 using z-test at α of 0.05. Therefore, the difference in improvement of two initialization strategies is not statistically significant. We could thus reason that the DPSO equipped with these different initialization strategies will lead to the same significant improvement rate.

Regarding the execution time, both initialization strategies can produce solution for small test instances (Case I) in a very short time. The difference in execution time between the "*Random*" and "*Greedy*" initialization on case I and II yielded a P value of 0.5918 and 0.5590 by z-test at $\alpha = 0.05$. It reveals that the difference is not statistically significant on both cases. This phenomenon is reasonable because both of the initialization strategies enable the diversity of initial swarms before they satisfy the stopping criterion. These results suggest that DPSO can obtain good solutions with these initialization strategies.

5.2.2. Swarm Topology. Results of different swarm topologies on the 60 small-size test instances (Case I) and 20 large-size test instances (Case II) are summarized in Table 5. The columns labeled "*Star*" and "*Ring*" list the results of DPSO algorithm with star topology and ring topology.

From Table 5, the improvements of both star and ring topologies on two test cases reached a high percentage (on average 62.25%), being quite attractive. For cases I and II, the difference in execution time yielded a P value less than 0.05 (P value = 0.0000), indicating that a statistically significant difference in improvement existed. Accordingly, we would suggest that star topology (*gbest*) is an effective swarm topology to deliver solutions with satisfactory qualities.

In Table 5, the results of execution time needed by different topologies reaffirm the fact that star topology (*gbest*) seem to have a faster convergence rate than the ring topology (*lbest*). For small-size test instances (Case I), the z-test of the difference in execution time between star topology (1.31 seconds) and ring topology (1.92 seconds) yielded a P value less than 0.05, indicating that a statistically significant difference in execution time exists; for large-size test instances (Case II), even though the star topology spent less computation time, the difference in execution time between the star topology (772.26 seconds) and ring topology (808.72 seconds) yielded a P value of 0.3076 by z-test at $\alpha = 0.05$, specifying that no statistically significant difference in execution time was found. This is reasonable because of the large standard deviation in the results of case II. The result suggests that the star topology spent less computational time to obtain attractive solutions to the studied problem.

5.2.3. Constraints Handling. Results of different constraints handling mechanisms on Cases I and II are shown in Table 6. The column labeled "*Accept*" reveals the results of DPSO algorithm that accept infeasible solutions as the best solution found by particle s at iteration t ($\overline{P}_s^t$); on the other hand, the column labeled "*Reject*" reveals those reject infeasible solutions.

As can be seen from Table 6, the improvements of two different constraint handling approaches do produce good solutions. For the small-size test instances (Case I), the average improvements of the "Accept" mechanism and the

"Reject" mechanism are 52.81% and 51.76%; for the large-size test instances (Case II), the average improvements are 72.85% and 71.48%. The results show that the "Accept" mechanism reaches slightly higher improvement than the "Reject" mechanism in both cases within a longer execution time. This is reasonable because the "Accept" mechanism has more chance to explore the infeasible solution space and takes more iteration to converge. However, to have a concise comparison of "Reject" mechanism and the "Accept" mechanism, the z-test yields P values of 0.5603 and 0.7410, which indicate that there is no statistical difference. The computational results and analysis shown in Table 6 suggest that DPSO with both constraints handling mechanisms can produce quality solutions.

5.2.4. Scout Particles. Results of DPSO and DPSO with scout particles on two test instances (small size and large size) are exhibited in Table 7. The column labeled "*Scout*" displays the results of DPSO algorithm with scout particles, while the column labeled "*Standard*" displays the results of DPSO algorithm without scouts.

For the improvement, the DPSO with scouts does produce better solutions than the standard DPSO on all test instances. It can be seen from Table 7 that the DPSO with scout particles reported 57.21% improvement rate on small-size test instances and 81.34% improvement rate on large-size test instances, while the standard DPSO showed 47.36% and 62.99%. The z-test of the difference in improvement yielded a P value less than 0.05 which indicates that a statistically significant difference in execution time existed. The effectiveness of the proposed DPSO can be attributed to the scout particles that decrease the chance to be trapped in local optimal by exploring the search space. This reveals that the proposed DPSO is an effective approach to the problem.

As for the execution time, the DPSO with scouts took less computation time than the standard DPSO on all test instances as well. In Table 7, the DPSO with scout particles took 1.23 seconds for solving the small-size test instances and 741 seconds 741 for large-size test instances. On the other hand, the elapsed times of the standard DPSO are 2.01 seconds and 839.97 seconds. For each case, the z-test yields a P value below 0.05, indicating that the difference in execution times is significant. This result evinces the efficiency of the DPSO with scouts by showing that the time elapsed is smaller than the standard DPSO. This phenomenon may be due to the fact that scout particles were evaluated by the expected fitness instead of the objective function.

6. Conclusions

In this paper, we have proposed an integer programming model for the materials acquisition problem, which is to maximize both the average preference and the budget execution rate being subject to some constraints of the budget, the required number of materials in each category and language. To solve the constrained problem, we have developed a DPSO algorithm and designed an initialization strategy to generate feasible particles. We have also conducted computational experiments of two sets test instances to demonstrate the effectiveness and efficiency of the proposed DPSO algorithm.

To better solve the studied problem, four different features of the proposed DPSO, including initialization strategies, swarm topology, constraints handling mechanism, and scout particles, are discussed. Firstly, we compare the results of employing the proposed initialization procedure, and the results of employing both the proposed Algorithm Greedy and initialization procedure. The computational results show that DPSO algorithm can obtain quality solutions with both the initialization strategies in a reasonable time. Secondly, we compare the results of performing star topology and ring topology. The results evince that star topology significantly outperforms ring topology in all test instances. Next, we compare the performances resulted from different constraint handling mechanisms. One mechanism is to accept the infeasible solutions as the best solution found by each particle, while the other is to reject the infeasible solutions as the best solution found by each particle. The computational results demonstrate that these two mechanisms reach the same performance. Lastly, we compare the results of standard DPSO and DPSO with the proposed scout particles. The results reveal that DPSO with scouts reaches higher improvement rates and takes shorter execution time. Accordingly, we would suggest that DPSO with the proposed initialization procedure, star topology, and scout particles is an effective approach to delivering attractive solutions in a reasonable time.

References

[1] J. Harrell, "Literature of acquisitions in review, 2008–9," *Library Resources and Technical Services*, vol. 56, no. 1, pp. 4–13, 2012.

[2] W. H. Walters, "Journal prices, book acquisitions, and sustainable college library collections," *College and Research Libraries*, vol. 69, no. 6, pp. 576–586, 2008.

[3] L. S. Connaway, K. Downing, Y. Du et al., "2010 top ten trends in academic libraries," *College and Research Libraries News*, vol. 71, no. 6, pp. 286–292, 2010.

[4] M. H. Beilby and T. H. Mott Jr., "Academic library acquisitions allocation based on multiple collection development goals," *Computers and Operations Research*, vol. 10, no. 4, pp. 335–343, 1983.

[5] K. Wise and D. E. Perushek, "Linear goal programming for academic library acquisitions allocations," *Library Acquisitions: Practice and Theory*, vol. 20, no. 3, pp. 311–327, 1996.

[6] K. Wise and D. E. Perushek, "Goal programming as a solution technique for the acquisitions allocation problem," *Library and Information Science Research*, vol. 22, no. 2, pp. 165–183, 2000.

[7] T.-F. Ho, S. J. Shyu, B. M. T. Lin, and Y.-L. Wu, "An evolutionary approach to library materials acquisition problems," in *Proceedings of the IEEE International Conference on Intelligent Systems (IS '10)*, pp. 450–455, London, UK, July 2010.

[8] S. K. Goyal, "Allocation of library funds to different departments of a university—an operational research approach," *College and Research Libraries*, vol. 34, pp. 219–222, 1973.

[9] A. Arora and D. Klabjan, "A model for budget allocation in multi-unit libraries," *Library Collections, Acquisition and Technical Services*, vol. 26, no. 4, pp. 423–438, 2002.

[10] T. Niyonsenga and B. Bizimana, "Measures of library use and user satisfaction with academic library services," *Library and Information Science Research*, vol. 18, no. 3, pp. 225–240, 1996.

[11] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freemaan and Company, New York, NY, USA, 1979.

[12] C.-J. Liao, C.-T. Tseng, and P. Luarn, "A discrete version of particle swarm optimization for flowshop scheduling problems," *Computers and Operations Research*, vol. 34, no. 10, pp. 3099–3111, 2007.

[13] 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.

[14] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, Australia, December 1995.

[15] R. Poli, "Analysis of the publications on the applications of particle swarm optimisation," *Journal of Artificial Evolution and Applications*, vol. 2008, Article ID 685175, 10 pages, 2008.

[16] A. Silva, A. Neves, and T. Goncalves, "An heterogeneous particle swarm optimizer with predator and scout particles," in *Autonomous and Intelligent Systems*, Lecture Notes in Computer Science, pp. 200–208, 2012.

[17] A. Hatamlou, "Black hole: a new heuristic optimization approach for data clustering," *Information Science*, vol. 222, pp. 175–184, 2013.

[18] C.-C. Chiu, M.-H. Ho, and S.-H. Liao, "PSO and APSO for optimizing coverage in indoor UWB communication system," *International Journal of RF and Microwave Computer-Aided Engineering*, vol. 23, no. 3, pp. 300–308, 2013.

[19] Y. Tian, D. Liu, D. Yuan, and K. Wang, "A discrete PSO for two-stage assembly scheduling problem," *International Journal of Advanced Manufacturing Technology*, vol. 66, no. 1-4, pp. 481–499, 2013.

[20] J. Kennedy and R. C. Eberhart, "Discrete binary version of the particle swarm algorithm," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pp. 4104–4108, Piscataway, NJ, USA, October 1997.

[21] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC '00)*, pp. 84–88, La Jolla, Calif, USA, July 2000.

[22] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization: an overview," *Swarm Intelligence*, vol. 1, pp. 33–57, 2007.

[23] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proceedings of the IEEE Swarm Intelligence Symposium (SIS '07)*, pp. 120–127, Honolulu, Hawaii, USA, April 2007.

[24] G. Coath and S. K. Halgamuge, "A comparison of constraint-handling methods for the application of particle swarm optimization to constrained nonlinear optimization problems," in *Proceedings of the Congress on Evolutionary Computation*, pp. 2419–2425, Canberra, Australia, December 2003.

[25] K. E. Parsopoulos and M. N. Vrahatis, "Particle swarm optimization method for constrained optimization problems," in *Intelligent Technologies—Theory and Applications: New Trends in Intelligent Technologies*, vol. 76, pp. 214–220, 2002.

[26] G. T. Pulido and C. A. Coello Coello, "A constraint-handling mechanism for particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC '04)*, pp. 1396–1403, Portland, Ore, USA, June 2004.

[27] A. García-Villoria and R. Pastor, "Introducing dynamic diversity into a discrete particle swarm optimization," *Computers and Operations Research*, vol. 36, no. 3, pp. 951–966, 2009.

[28] T. M. Blackwell and P. Bentley, "Don't push me! Collision-avoiding swarms," in *Proceedings of the Congress on Evolutionary Computation*, pp. 1691–1696, Honolulu, Hawaii, USA, May 2002.

[29] Y.-L. Wu, T.-F. Ho, S. J. Shyu, and B. M. T. Lin, "Discrete particle swarm optimization for materials acquisition in multi-unit libraries," in *Proceedings of the Congress on Evolutionary Computation*, pp. 1–7, Brisbane, Australia, June 2012.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 547656, 8 pages
http://dx.doi.org/10.1155/2013/547656

Research Article

A Multiuser Detector Based on Artificial Bee Colony Algorithm for DS-UWB Systems

Zhendong Yin, Xiaohui Liu, and Zhilu Wu

School of Electronics and Information Engineering, Harbin Institute of Technology, Harbin 150001, China

Correspondence should be addressed to Zhendong Yin; zgczr2005@yahoo.com.cn

Received 3 June 2013; Accepted 10 July 2013

Academic Editors: P. Agarwal, V. Bhatnagar, J. Yan, and Y. Zhang

Artificial Bee Colony (ABC) algorithm is an optimization algorithm based on the intelligent behavior of honey bee swarm. The ABC algorithm was developed to solve optimizing numerical problems and revealed premising results in processing time and solution quality. In ABC, a colony of artificial bees search for rich artificial food sources; the optimizing numerical problems are converted to the problem of finding the best parameter which minimizes an objective function. Then, the artificial bees randomly discover a population of initial solutions and then iteratively improve them by employing the behavior: moving towards better solutions by means of a neighbor search mechanism while abandoning poor solutions. In this paper, an efficient multiuser detector based on a suboptimal code mapping multiuser detector and artificial bee colony algorithm (SCM-ABC-MUD) is proposed and implemented in direct-sequence ultra-wideband (DS-UWB) systems under the additive white Gaussian noise (AWGN) channel. The simulation results demonstrate that the BER and the near-far effect resistance performances of this proposed algorithm are quite close to those of the optimum multiuser detector (OMD) while its computational complexity is much lower than that of OMD. Furthermore, the BER performance of SCM-ABC-MUD is not sensitive to the number of active users and can obtain a large system capacity.

1. Introduction

Since the concept of ultra-wideband (UWB) technology was put forward by FCC in the 1990s, UWB technique has drawn a lot of attention in the theoretic research, industrial application, and many other areas because of its attractive features such as high data transmission rate, low power density, high interference resistance, and strong multipath resolution [1–3].

Multiuser detection (MUD) is a method to eliminate the effect of multiple access interference (MAI). The multiple accesses are commonly divided into two paths: time hopping (TH-UWB) and direct sequence (DS-UWB) [4]. In recent years, the increase in demand for multiple access applications with high data transmission rates has prompted the development of multiuser detection (MUD) techniques in such systems to suppress the MAI and improve the system performance. Optimum multiuser detection (OMD) was provided by Verdu in 1986 [5]; OMD makes the BER performance of multiple access system approximate to single-user system. But it has a high computational complexity, so it has poor real-time characteristic confines in engineering. Therefore, suboptimal detectors which may approximate OMD's BER performance with an acceptable computational complexity have become a focus of research. A code-aided interference suppression method was introduced for NBI restriction in DS-UWB systems [6]. A multiuser frequency-domain (FD) turbo detector was employed which combines FD turbo equalization schemes with soft interference cancellation [7]. Adaptive MUD methods using the recursive least square (RLS) principles were proposed in [8, 9]. However, the trade-off problem between BER performance and computational complexity still exists.

Swarm intelligence is a research branch that models the population of interacting agents or swarms that are able to self-organize. Several modern heuristic algorithms have been developed for solving combinatorial and numeric optimization problems [10]. Artificial Bee Colony (ABC) algorithm is an optimization algorithm based on the intelligent behavior of honey bee swarm [11, 12], and some extended and efficient

ABC algorithms are presented to improve the performance of ABC [13, 14].

It has been demonstrated that ABC algorithm for numerical optimization problems has more superior performance than those based on heuristics algorithm [15]. ABC algorithm has been widely used in neural network training practice [16], path optimization [17], and back analysis [18]. However, no literatures have been reported that ABC was used in the multiuser detection (MUD) fields. In order to possess a good BER performance and a low time complexity, we investigate an efficient multiuser detector by selection of initial states based on code mapping for the Artificial Bee Colony algorithm (SCM-ABC-MUD) in DS-UWB systems. As a kind of swarm intelligence methods, ABC is selected here for its significant ability to search for the global optimal value and to adapt its searching space automatically [19, 20]. And its basic motivation is to find the global optimum by simulating the bees' behaviors [21]. This proposed algorithm makes use of the thought of ABC's iterative optimization and the result of the suboptimal detectors based on code mapping. Simulation results show that the BER performance and near-far effect (NFE) resistance capability of this algorithm are better than those of matched filter (MF), DEC, and MMSE detectors and even close to OMD.

The paper is organized as follows. In Section 2, the MUD model is introduced and a suboptimal detector based on code mapping is proposed. Then in Section 3, the basic principles of the Artificial Bee Colony (ABC) algorithm and the proposed SCM-ABC-MUD algorithm are illustrated. In Section 4, simulation experiments that compare the performance of different MUD algorithms are analyzed, followed by conclusions given in Section 5.

2. MUD Models and Methods

2.1. Multiuser DS-UWB System Model. In theory, a K-user synchronous DS-UWB system under the additive white Gaussian noise (AWGN) channel is considered which is not subjected to the frequency selective multipath. And assume each user employs the binary phase-shift key (BPSK) modulation. Then the kth user's transmit signal can be written as

$$x_k(t) = \sum_{i=1}^{M}\sum_{j=0}^{N_c-1} d_k(i)\, c_k\left(t-(i-1)T_s\right) p\left(t-(i-1)T_s-jT_c\right), \quad (1)$$

where M is the length of bits per packet and BPSK symbols $d_k(i) \in \{-1,1\}_{i=1}^{M}$ are spread with the specific PN codes $c_k(t)$, which are the binary bit stream valued only by -1 or 1. T_s is the symbol duration, T_c is the pulse repetition period, N_c equals to T_s/T_c, and $p(t)$ represents the transmitted pulse waveform generally characterized as the second derivative of Gaussian pulse

$$p(t) = \left[1-4\pi\left(\frac{t-t_d}{\tau_m}\right)^2\right]\exp\left[-2\pi\left(\frac{t-t_d}{\tau_m}\right)^2\right], \quad (2)$$

where t_d and τ_m are the pulse center and the pulse shape parameter.

The total received signal composed by different signals of all users is

$$r(t) = v(t) + n(t) = \sum_{k=1}^{K} A_k x_k(t) + n(t), \quad (3)$$

where A_k is the amplitude of the kth received signal and $n(t)$ is zero-mean additive white Gaussian noise with the unilateral power spectral density of N_0.

2.2. Classical Multiuser Detectors

2.2.1. Matched Filters (MFs). The traditional receiver of a DS-UWB system consists of a pulse demodulator and a set of matched filters (MFs) corresponding to each user. Let the output of a bank of single-user MFs be a K-dimensional vector $\mathbf{y} = [y_1, y_2, \ldots, y_K]^T$, the vector $\mathbf{b} = [b_1, b_2, \ldots, b_K]^T$ represent the output of sign detectors, the vector $\mathbf{d} = [d_1, d_2, \ldots, d_K]^T$ denotes the correct bits of each user, and the vector $\mathbf{n} = [n_1, n_2, \ldots, n_K]^T$ denotes the output of noise from matched filters which is a zero mean Gaussian random. So, the output of the MFs can be represented as follows:

$$\mathbf{y} = \mathbf{RAb} + \mathbf{n}, \quad (4)$$

$$\mathbf{b} = \operatorname{sgn}(\mathbf{y}), \quad (5)$$

where $R = (r_{ij})_{K\times K}$ denotes the cross-correlation matrix, in which $r_{ij} = \sum_{l=0}^{N_c-1} c_i(l)c_j(l)$, and $\mathbf{A} = \operatorname{diag}(A_1, A_2, \ldots, A_K)$ in which the diagonal element A_k $(k \in [1,K], k \in N)$ represents the signal amplitude of the kth user.

2.2.2. Optimum Multiuser Detection (OMD). According to the theory of OMD, the optimum detection result satisfies the following expression:

$$\mathbf{b}_{\mathrm{OMD}} = \arg\left\{\max_{b\in\{-1,1\}}\left(2\mathbf{b}^T\mathbf{Ay} - \mathbf{b}^T\mathbf{ARAb}\right)\right\}. \quad (6)$$

It is known that the selection of this optimal solution $\mathbf{b}_{\mathrm{OMD}}$ in the K-dimensional Euclidean solution space is generally a nondeterministic polynomial hard problem, but the computational complexity of the OMD method is $O(K^2)$, and K is the number of active users.

2.2.3. Suboptimal Multiuser Detection Based on Code Mapping (SCM). In order to get a suboptimal solution, the candidate bits set output from the matched filters mapped to a one-dimensional feature space using a mapping function.

Let

$$F(\mathbf{b}) = \frac{1}{2}\mathbf{b}^T\mathbf{ARAb} - \mathbf{b}^T\mathbf{Ay}. \quad (7)$$

According to (6), if the elements in $\mathbf{b}$ are all right, the value of $F(\mathbf{b})$ will achieve the minimum. Making a partial derivation of (7), we get

$$\frac{\partial F}{\partial \mathbf{b}} = \mathbf{Hb} - \mathbf{Ay}. \quad (8)$$

By expanding (8), we get a Kth-order linear equations given as follows:

$$\begin{aligned}
\frac{\partial F}{\partial b_1} &= \sum_{j=1}^{K} A_1 A_j r_{1j} b_j - A_1 y_1, \\
\frac{\partial F}{\partial b_2} &= \sum_{j=1}^{K} A_2 A_j r_{2j} b_j - A_2 y_2, \\
&\vdots \\
\frac{\partial F}{\partial b_K} &= \sum_{j=1}^{K} A_K A_j r_{Kj} b_j - A_K y_K.
\end{aligned} \tag{9}$$

Now, we take into account that MAI is the main interference resource which largely affects the performance of the system. Let $L(b_i) = \sum_{j=1}^{K} A_i A_j r_{ij} b_j - A_i y_i$, $i = 1, 2, \ldots, K$. Bringing the candidate codes set $\mathbf{b}$ into (9), there are two situations illustrated as follows.

(1) No wrong code in $\mathbf{b}$. Based on the theory of extreme value, if MAI is the only interference resource without AWGN and the elements in $\mathbf{b}$ are all correct, the result of (9) strictly equals $\mathbf{0}$. And, in the condition of high SNR, bringing (4) to $L(b_k)$, we can see that $L(b_k) = -A_i n_i \sim N(0, A_i^2 N_c N_0/2)$ $(k = 1, 2, \ldots, K)$ when there is no error bit in the candidate codes set $\mathbf{b}$.

(2) Wrong codes exist in $\mathbf{b}$. Supposing b_i $(i \in [1, K],\ i \in N)$ is the wrong code (in other words, $-b_i$ is the correct code), the other codes are correct. Substituting it to the ith equation of (9), we get

$$\begin{aligned}
L(b_i) &= \sum_{\substack{j=1 \\ j \neq i}}^{K} A_i A_j r_{ij} b_j - A_i y_i + A_i^2 r_{ii} b_i \\
&= \sum_{\substack{j=1 \\ j \neq i}}^{K} A_i A_j r_{ij} b_j - A_i y_i + A_i^2 r_{ii} (-b_i) + 2A_i^2 r_{ii} b_i \\
&= 2A_i^2 N_c b_i - A_i n_i.
\end{aligned} \tag{10}$$

As for k which does not equal i, we get

$$L(b_k) = 2A_i A_k r_{ik} b_i - A_k n_k, \qquad k = 1, 2, \ldots, K,\ k \neq i. \tag{11}$$

According to (10) and (11), it can be seen when the ith user's code is wrong, $|L(b_i)| \gg |L(b_k)|$, $k = 1, 2, \ldots, K$, $k \neq i$. Therefore, the function $L(\mathbf{b})$ can obviously differentiate the wrong codes and the right codes through the absolute value of it. In addition, $L(\mathbf{b})$ is a Kth-order linear equation which can get the result of $L(\mathbf{b})$ without complex computations.

It is accessible to map b into a one-dimensional feature space $|L(\mathbf{b})|$ to identify the wrong codes in the candidate set. Figure 1 shows an example of the feature space mapping in the scenario of 10 users, with 6 dB signal-to-noise ratio (SNR).

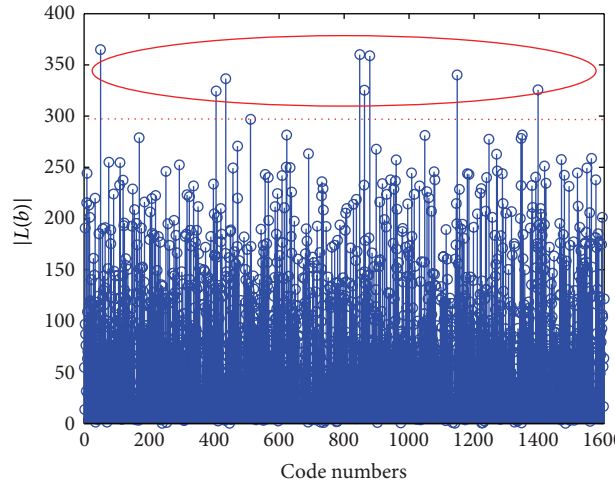


Figure 1: Relationship between code numbers and mapping function $|L(\mathbf{b})|$.

In Figure 1, it is clear that the wrong codes and the right ones have a significant difference in the feature space mapped by $|L(\mathbf{b})|$. Almost all the values of $|L(\mathbf{b})|$ corresponding to the wrong codes are greater than 280; in contrast the values of the right ones are smaller than 280. C-means clustering approach [22] is used to classify the candidate codes into right bits and wrong ones. The C-means clustering is a popular unsupervised clustering algorithm based on the partition of data, which can finish the classification without training samples.

In conclusion, the whole proposed approach can perform the multiuser detection with 3 steps. First, map the candidate bits set $\mathbf{b}$ into a one-dimensional feature space by mapping function $|L(\mathbf{b})|$. Second, aggregate and pick out the wrong bits among the candidate bits set by code clustering. And finally, correct the wrong bits to obtain a suboptimal solution.

3. The Proposed SCM-ABC-MUD Algorithm

3.1. The Basic Principle of ABC Algorithm. In the ABC algorithm, the colony of artificial bees consists of three kinds of bees: employed bees, onlookers, and scouts. In the algorithm, the number of employed bees is equal to the number of food sources around the hive. The employed bee whose food source is exhausted by the employed and onlooker bees becomes a scout. The position of a food source represents a possible solution of the optimization problem, and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. The number of the onlooker bees or the employed bees is equal to the number of food sources in the population.

The general scheme of the ABC algorithm is as follows.

Step 1. The first step is initialization. In this step, a set of food source positions is generated and distributed randomly. After initializing the solution population, the fitness value of each food source position is evaluated. In ABC algorithm, the fitness of each solution is the value of objective function. The ABC has some control parameters: food sources number (the number of food sources equal to the employed bees); limit,

the value of predetermined number of cycles; and MCN, the maximum cycle number. Initially, the values of these control parameters are assigned.

Step 2. After initialization, the population is evaluated and is subjected to repeated cycles of the search processes of the employed bees, the onlooker bees, and scout bees. Employed bees search for new food sources (V_m) having more nectar within the neighborhood of the food source (X_m) in their memory. They find a neighbor food source and then evaluate its profitability (fitness). They can determine a neighbor food source V_m using the formula given by

$$v_{mi} = x_{mi} + \theta_{mi}\left(x_{mi} - x_{ki}\right), \tag{12}$$

where x_k is a randomly selected food source, i is a randomly chosen parameter index, and θ_{mi} is random number between $[-1, 1]$.

Step 3. After producing the new food source V_m, its fitness is calculated and a greedy selection is applied between V_m and X_m. If the fitness of the new food source position is better than the old food source position, the employed bee moves from the old food source position to the new food source position, and the new food source takes place of old one in the population and becomes a new member.

The fitness value of the solution, $\mathrm{fit}_m(X_m)$, might be calculated for minimization problems using the following formula:

$$\mathrm{fit}_m\left(X_m\right) = \begin{cases} \dfrac{1}{1 + f_m\left(X_m\right)}, & \text{if } f_m\left(X_m\right) \geq 0, \\ 1 + \mathrm{abs}\left(f_m\left(X_m\right)\right), & \text{if } f_m\left(X_m\right) \leq 0, \end{cases} \tag{13}$$

where $f_m(X_m)$ is the objective function value of solution X_m.

Step 4. Unemployed bees consist of two groups of bees: onlooker bees and scouts. Employed bees share their food source information with onlooker bees waiting in the hive, and then onlooker bees probabilistically choose their food sources depending on the probability values calculated using the fitness values provided by employed bees.

Calculate the probability values P_i for the solutions using fitness of the solutions by

$$P_i = \frac{\mathrm{fit}_m\left(X_m\right)}{\sum_{m=1}^{\mathrm{SN}} \mathrm{fit}_m\left(X_m\right)}, \tag{14}$$

where SN represents the number of the food sources position.

After a food source X_m for an onlooker bee is probabilistically chosen, a neighborhood source V_m is determined by Step 2, and its fitness value is computed. As in Step 3, a greedy selection is applied between V_m and X_m. Hence, more onlookers are recruited to richer sources, and positive feedback behavior appears.

Step 5. If a predetermined number of trials cannot further improve a candidate solution represented by a food source then that food source, is considered abandoned and the employed bee associated with that food source becomes a scout. The new solution discovered by the scout can be defined by

$$x_{mi} = l_i + \mathrm{rand}(0, 1) * \left(u_i - l_i\right), \tag{15}$$

where l_i and u_i are the lower and upper bounds of the parameters x_{mi}. The abandoned food source is replaced by the randomly new food source.

Step 6. If a termination condition is reached, the process is stopped and memorizes the best solution achieved so far. Otherwise the algorithm returns to Step 2.

The general algorithmic structure of ABC optimization approach can be summarized as follows [23]:

(i) *Initialization Phase*

(ii) *Repeat*

Employed Bees Phase
Onlooker Bees Phase
Scout Bees Phase
Memorize the best solution achieved so far

(iii) *Until* (Cycle = Maximum Cycle Number).

3.2. The Discretization of Behavior Models. The mathematical model of OMD is considered as a combinatorial optimization problem, and ABC has a strong global searching capability to solve this problem. The optimization function for OMD is shown as (6), which is a discrete optimization function; for this reason, the behavior models of ABC should be discretized.

The discretization is defined as follows.

(1) The expression of artificial bee's state. In this solution space, the state of each artificial bee is encoded by −1 or +1. If there are K active users in this DS-UWB MA system, thus the state is a K-dimensional vector, like $X_0 = (x_1, x_2, \ldots, x_K)^T$, where $x_i \in \{-1, +1\}$ and $i = 1, 2, \ldots, K$.

(2) The food quality or the fitness function for artificial bees is the criterion of OMD in (6).

(3) X_c is the new food sources for each step; if the element of X_c is $x_{ci} > 0$, then assign $x_{ci} = +1$; otherwise, $x_{ci} = -1$.

(4) Initialization. The initial state of each artificial bee is selected randomly in the discrete space with 2^K likely solutions. But the ABC is an iterative optimization algorithm, which means that the selection of initial states has a great effect on the iteration times and the convergence rate. In this paper, we choose a suboptimal multiuser detection based on code mapping (SCM) and its variations as initial state values of artificial bees, which can ensure the artificial bees gather near the optimal value of the optimization problem because the suboptimal solution is quite close to the optimal value in the solution space. Besides, this selection method of initial states can speed up the convergence rate and reduce the iteration times for optimization.

3.3. The Specific Steps of the Initial Value Selection. The specific steps of the initial value selection are given as follows.

Step 1. Execute matched filter detection and code mapping to get a suboptimal solution first. Assign the result $b_{1\times K} = (b_1, b_2, \ldots, b_K)$ to the first artificial bee as its initial state value, where $b_i \in \{-1, +1\}$ and $i = 1, 2, \ldots, K$.

Step 2. Change an element b_i of the result $b_{1\times K}$ randomly, $b_i^* = -b_i$, and then assign this new result $b_{1\times K}^*$ which changes from $b_{1\times K}$ to the second artificial bee as its initial state value.

Step 3. Repeat Step 2 and assign these new results to other artificial bees.

Step 4. If all artificial bees have been assigned initial state values, the procedure of initial value selection comes to the end. And we can start the ABC to research the optimal value for MUD. In consideration of the previous statements, the overall structure of this proposed SCM-ABC-MUD detector is shown in Figure 2.

Figure 2 shows the block diagram of the algorithm. The implementation of this detector can be summarized as follows: firstly, the output of a bank of matched filter receivers is fed to suboptimal detectors based code mapping and clustering; secondly, the detection results of these suboptimal detectors are used to construct an initial solution space; finally, the ABC is executed in this space.

4. Simulation and Discussion

In order to analyze the proposed SCM-ABC-MUD algorithm, Monte Carlo simulations are utilized and the values of these majority parameters are assigned as Table 1.

4.1. The BER Performance versus SNR. The BER versus SNR curves with perfect power control in the AWNG are depicted in Figure 3. And there are 10 users in the system. As for ABC, we fix the number of initializing food sources (equal to employed bees) that is 3; limit, the value of predetermined number of cycles, that is 3; and MCN, the maximum cycle number, that is 20.

The performances of matched filter (MF), decor relating (DEC) [24], minimum mean square error (MMSE) [25], SCM-ABC-MUD, and OMD detectors are compared and depicted in Figure 3.

Figure 3 shows that the BER performance of SCM-ABC-MUD is superior to those of other suboptimal detectors including MF, DEC, and MMSE, and it even coincides with that of OMD. The reason is that this proposed SCM-ABC-MUD algorithm can make a search within a simplified solution space constructed by the solutions of these suboptimal detectors rather than a random search.

4.2. The BER Performance versus User Numbers K. The BER performances of these detectors with different number of active users K are analyzed in this experiment. The SNR of the AWGN channel is 5 dB. And the control parameters of ABC are same as the BER performance versus SNR. Figure 4 shows the result of this experiment. According to Figure 4, the performance of SCM-ABC-MUD is the best of the suboptimum MUD except OMD. While the number of users increases, ABC needs a new group of control parameters. However, in this experiment, they are unchanged, resulting in the performance difference between SCM-ABC-MUD and OMD.

Table 1: Simulation parameters.

System	DS-UWB
Modulation mode	BPSK
Spreading codes (SC)	m sequences
The length of SC	255
Communication channel	AWGN
The number of testing information bits	3200000
The width of UWB pulse	0.8 ns
The pulse repetition period	≈2 ns
Limit	3
Initializing food source number	3

4.3. The NFE Resistant Capability Comparison. In this experiment, the SNR of the first user's received signal is assigned at 5 dB all the time, while the SNR of other users changs from 0 dB to 10 dB with per step of 1 dB. Besides, other parameters are set as same as those in the BER performance versus SNR. Figure 5 shows the first user's BER performances of different MUD detectors. It can be seen from Figure 5 that OMD has the best near-far effect resistant ability, and the SCM-ABC-MUD is very close to OMD.

4.4. BER Performance versus Different Initial States. As an iterative optimization algorithm, the convergence rate is as important as the optimization capability. And the selection of initial states has a great effect on the iteration times and the convergence rate. In this experiment two initial state values of artificial bees are employed as a comparison. One has random initial state values which is named as ABC-MUD; the other is SCM-ABC-MUD which is assigned a group of initial state values using the suboptimal solution of the code mapping-based multiuser detection. The BER performance is studied for both algorithms with the maximum cycle number (MCN) of 5, 15, and 20, respectively. Figures 6 and 7 are the convergence rates of ABC-MUD and SCM-ABC-MUD separately.

From Figures 6 and 7, the BER performance of SCM-ABC-MUD is much better than ABC-MUD with the same iteration times, and the former can achieve OMD's BER performance with 20 times iteration which is the global optimal solution. It can be concluded that the convergence rate of SCM-ABC-MUD is much quicker than that of ABC-MUD. Meanwhile, ABC-MUD only improves its BER performance a little than matched filter (shown in Figure 3). It is because that SCM-ABC-MUD makes full use of the suboptimal solution, the initial value of SCM-ABC-MUD is more close to OMD

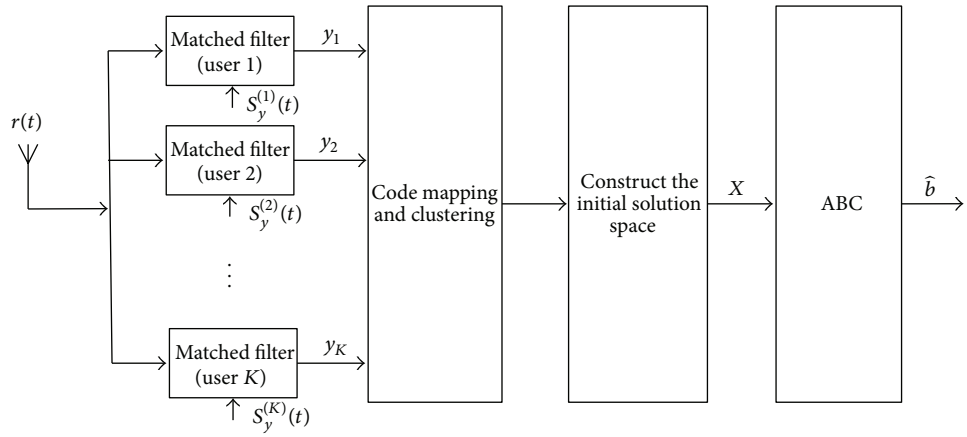


Figure 2: General schematic diagram of the SCM-ABC-MUD detector.

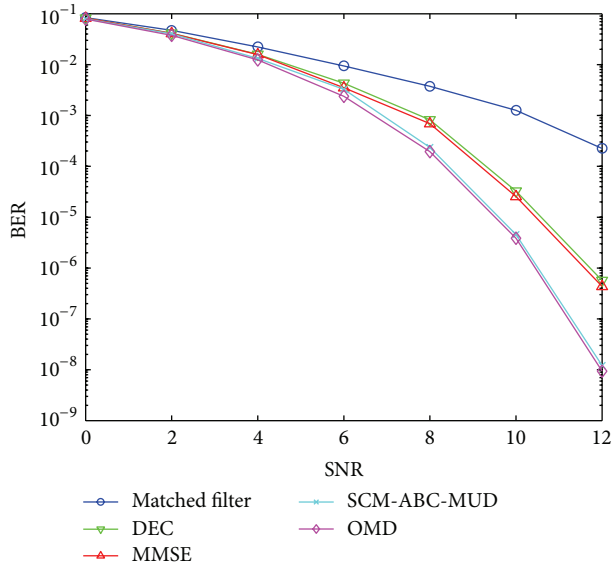


Figure 3: BER performance versus SNR.

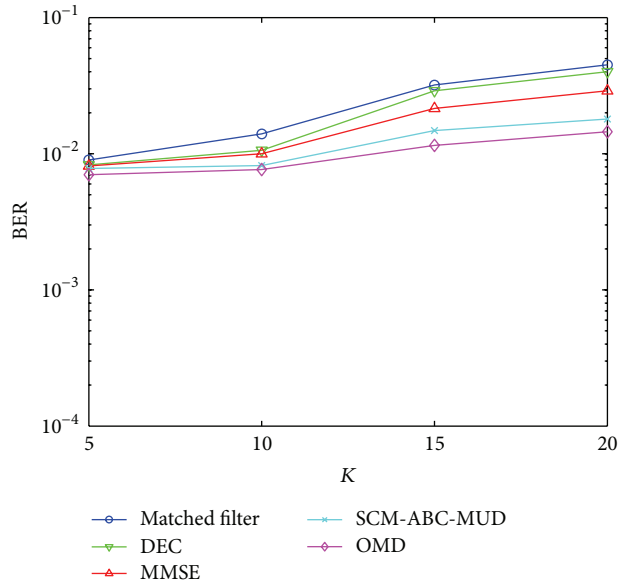


Figure 4: BER performance versus user numbers K.

than ABC-MUD, and so it can achieve the optimal value of OMD much quicker than ABC-MUD.

4.5. The Computational Complexity of the Proposed SCM-ABC-MUD. OMD can obtain the best BER performance while the computational complexity performance is too expensive to be implemented in practice. However, the computational complexity of SCM-ABC- MUD is much less than that of the OMD. In a K-user DS-UWB system, to detect the K-user vector $\mathbf{b}$, the computational complexity of iterations of the OMD is 2^K. The computational complexity of the proposed method includes three parts. The first part is the computational complexity of the code mapping; according to Section 2.2.3, the mapping method is just a calculation of $L(\mathbf{b})$, so the computational complexity of the first part can be neglected. The second part is the computational complexity of C-means clustering; the complexity of iterations of the code clustering is K. The third part is the complexity of the ABC algorithm. It depends on the convergence rate of SCM-ABC-MUD. In Section 4.4, we have obtained a conclusion that the SCM-ABC-MUD can achieve convergence with a much fewer iterations than OMD. Let the normalized operation time of per vector $\mathbf{b}$ using matched filter detector equal to 1 in the condition of 10 users. The simulation parameters are the same as Section 4.1. Table 2 lists the relative operation time using OMD, SCM-ABC-MUD, and matched filter.

From Table 2 we can see that the computational complexity of the SCM-ABC-MUD is in the same order of magnitude to the MMSE and DEC and far lower than OMD. This is because the iteration of SCM-ABC-MUD will be converged very soon and costs little quantity of computation. Hence, we can get a conclusion that the SCM-ABC-MUD can get good BER performance with low computational complexity.

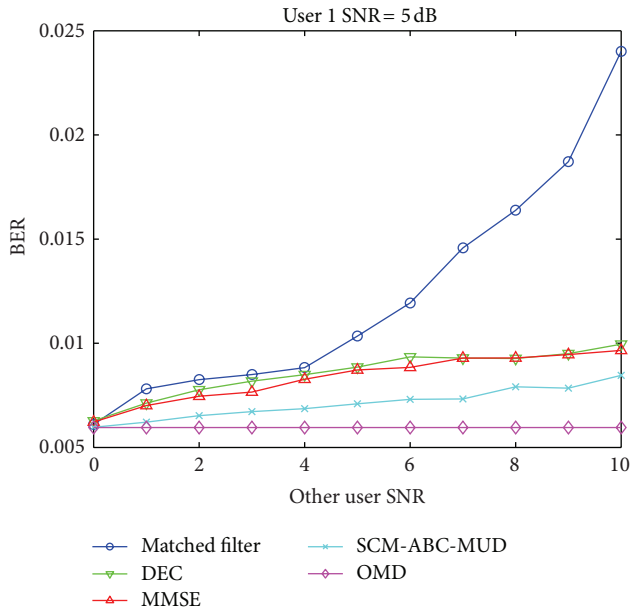


Figure 5: Near-far effect resistant of different MUD algorithms.

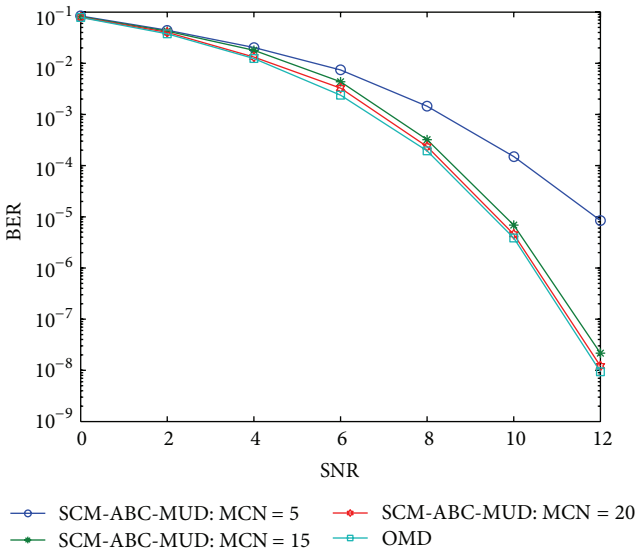


Figure 6: The BER performance with different iteration times of SCM-ABC-MUD.

5. Conclusions

In this paper, we firstly employed the Artificial Bee Colony algorithm in the DS-UWB MUD. In consideration of the high computational complexity of OMD, the proposed MUD is a hybrid method which combines ABC algorithm and a suboptimal solution of the code mapping-based MUD. First, the bits set output from the matched filters is mapped into a one-dimensional feature space to obtain a suboptimal solution; then the initial solution space is constructed based on the suboptimal solution; finally, the optimal solution is found by operating the different behaviors of artificial bees in solution space. The proposed multiuser detector can make full use

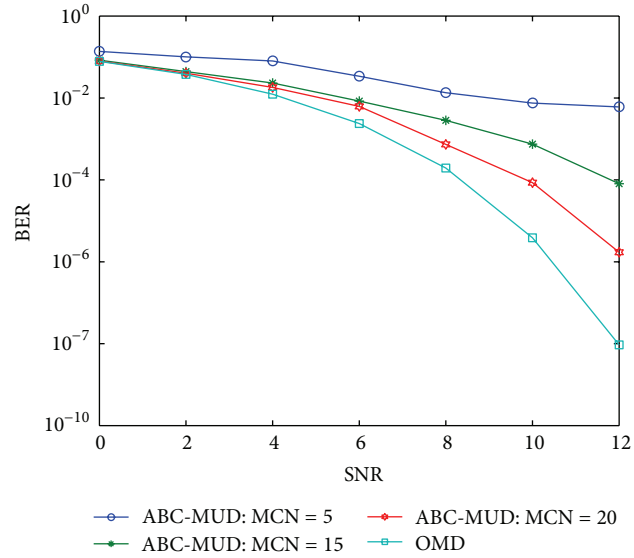


Figure 7: The BER performance with different iteration times of ABC-MUD.

Table 2: The comparison of computational complexity using different MUD algorithms.

OMD	MMSE	DEC	SCM-ABC-MUD	MF
58.9	1.38	1.30	1.21	1

of the suboptimal solution and advantages of ABC to study the optimal value in the solution space. Simulation results have indicated that the BER performance, user capacity, and the NFE resistant ability of this novel algorithm are quite close to those of OMD, and they are also superior to those of MF, DEC, and MMSE. Furthermore, the convergence rate of SCM-ABC-MUD is better than that of ABC-MUD. And the computational complexity of the SCM-ABC-MUD is much lower than that of OMD.

Acknowledgments

The research in this paper is supported by the National Natural Science Foundation of China (Grant no. 61102084), Foundation of China Academy of Space Technology (CAST), and the China Postdoctoral Science Foundation (Grant no. 2011M500665).

References

[1] M. Z. Win and R. A. Scholtz, "Impulse radio: how it works," *IEEE Communications Letters*, vol. 2, no. 2, pp. 36–38, 1998.

[2] X. Shen, M. Guizani, H. H. Chen et al., "Guest editorial ultra-wideband wireless communications-theory and applications," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 4, pp. 713–716, 2006.

[3] J. Ahmadi-Shokouh and R. C. Qiu, "Ultra-wideband (UWB) communications channel measurements—a tutorial review," *International Journal of Ultra Wideband Communications and Systems*, vol. 1, no. 1, pp. 11–31, 2009.

[4] I. Oppermann, M. Hämäläinen, and J. Iinatti, Eds., *UWB: Theory and Applications*, Wiley, 2005.

[5] S. Verdu, "Minimum probability of error for asynchronous Gaussian multiple-access channels," *IEEE Transactions on Information Theory*, vol. 32, no. 1, pp. 85–96, 1986.

[6] C. Wang, M. Ma, R. Ying et al., "Narrowband interference mitigation in DS-UWB systems," *IEEE Signal Processing Letters*, vol. 17, no. 5, pp. 429–432, 2010.

[7] P. Kaligineedi and V. K. Bhargava, "Frequency-domain turbo equalization and multiuser detection for DS-UWB systems," *IEEE Transactions on Wireless Communications*, vol. 7, no. 9, pp. 3280–3284, 2008.

[8] G. S. Biradar, S. N. Merchant, and U. B. Desai, "Performance of constrained blind adaptive DS-CDMA UWB multiuser detector in multipath channel with narrowband interference," in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM '08)*, pp. 1–5, November-December 2008.

[9] Q. Z. Ahmed and L. L. Yang, "Reduced-rank adaptive multiuser detection in hybrid direct-sequence time-hopping ultrawide bandwidth systems," *IEEE Transactions on Wireless Communications*, vol. 9, no. 1, pp. 156–167, 2010.

[10] M. Woodside-Oriakhi, C. Lucas, and J. E. Beasley, "Heuristic algorithms for the cardinality constrained efficient frontier," *European Journal of Operational Research*, vol. 213, no. 3, pp. 538–550, 2011.

[11] B. Basturk and D. Karaboga, "An artificial bee colony (ABC) algorithm for numeric function optimization," in *Proceedings of IEEE Swarm Intelligence Symposium*, pp. 12–14, May 2006.

[12] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[13] W. Zou, Y. Zhu, H. Chen et al., "A clustering approach using cooperative artificial bee colony algorithm," *Discrete Dynamics in Nature and Society*, vol. 2010, Article ID 459796, 16 pages, 2010.

[14] Y. Xu, P. Fan, and L. Yuan, "A simple and efficient artificial bee colony algorithm," *Mathematical Problems in Engineering*, vol. 2013, Article ID 526315, 9 pages, 2013.

[15] 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.

[16] D. Karaboga, B. Akay, and C. Ozturk, "Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks," in *Modeling Decisions for Artificial Intelligence*, pp. 318–329, Springer, Berlin, Germany, 2007.

[17] L. P. Wong, M. Y. H. Low, and C. S. Chong, "A bee colony optimization algorithm for traveling salesman problem," in *Proceedings of the 2nd Asia International Conference on Modelling and Simulation (AICMS '08)*, pp. 818–823, IEEE, May 2008.

[18] F. Kang, J. Li, and Q. Xu, "Structural inverse analysis by hybrid simplex artificial bee colony algorithms," *Computers and Structures*, vol. 87, no. 13-14, pp. 861–870, 2009.

[19] 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.

[20] B. Akay and D. Karaboga, "A modified artificial bee colony algorithm for real-parameter optimization," *Information Sciences*, vol. 192, pp. 120–142, 2012.

[21] B. Alatas, "Chaotic bee colony algorithms for global numerical optimization," *Expert Systems with Applications*, vol. 37, no. 8, pp. 5682–5687, 2010.

[22] C. Xu, P. Zhang, B. Li et al., "Vague C-means clustering algorithm," *Pattern Recognition Letters*, vol. 34, no. 5, pp. 505–510, 2012.

[23] D. Karaboga, "Artificial bee colony algorithm," *Scholarpedia*, vol. 5, no. 3, p. 6915, 2010.

[24] S. Im and E. J. Powers, "An iterative decorrelating receiver for DS-UWB multiple access systems using biphase modulation," in *Proceedings of IEEE Workshop on Signal Processing Systems Design and Implementation*, pp. 59–64, October 2004.

[25] G. C. Chung, "Multi-user access interference suppression for UWB system," in *Proceedings of the International Conference on Computer & Information Science (ICCIS '12)*, vol. 2, pp. 670–675, June 2012.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 419187, 10 pages
http://dx.doi.org/10.1155/2013/419187

Research Article

Feature Selection Method Based on Artificial Bee Colony Algorithm and Support Vector Machines for Medical Datasets Classification

Mustafa Serter Uzer,[1] Nihat Yilmaz,[1] and Onur Inan[2]

[1] *Electrical-Electronics Engineering, Faculty of Engineering, Selcuk University, Konya, Turkey*
[2] *Computer Engineering, Faculty of Engineering, Selcuk University, Konya, Turkey*

Correspondence should be addressed to Mustafa Serter Uzer; msuzer@selcuk.edu.tr

Received 25 May 2013; Accepted 6 July 2013

Academic Editors: J. Yan and Y. Zhang

This paper offers a hybrid approach that uses the artificial bee colony (ABC) algorithm for feature selection and support vector machines for classification. The purpose of this paper is to test the effect of elimination of the unimportant and obsolete features of the datasets on the success of the classification, using the SVM classifier. The developed approach conventionally used in liver diseases and diabetes diagnostics, which are commonly observed and reduce the quality of life, is developed. For the diagnosis of these diseases, hepatitis, liver disorders and diabetes datasets from the UCI database were used, and the proposed system reached a classification accuracies of 94.92%, 74.81%, and 79.29%, respectively. For these datasets, the classification accuracies were obtained by the help of the 10-fold cross-validation method. The results show that the performance of the method is highly successful compared to other results attained and seems very promising for pattern recognition applications.

1. Introduction

Pattern recognition and data mining are the techniques that allow for the acquirement of meaningful information from large-scale data using a computer program. Nowadays, these techniques are extensively used, particularly in the military, medical, and industrial application fields, since there is a continuously increasing amount and type of data in these areas, due to advanced data acquisition systems. For this reason, for the obtained data set, data reduction algorithms are needed for filtering, priority sorting, and providing redundant measurements to detect the feature selection. By using these algorithms, quality data is obtained, which in turn raises the quality of the analyzing systems or the success of the recognition systems. In particular, medical applications with ever-increasing popularity and use of advanced technology are the most important field in which these algorithms are used. Many new algorithms developed in the field of medicine are tested on the disease data presented for the common use of all the scientists, and their performances are compared. The datasets from UCI database are very popular for this purpose. The algorithm developed and tested on hepatitis, liver disorders, and diabetes data from UCI was compared with studies in the literature that use the same datasets. These data sets consist of diseases that are commonly encountered in society and significantly reduce the quality of life of patients. The selected data sets are comprised of a variety of test and analysis device data and personal information about the patients. The main objective our work is the integration of the developed systems to these test and analysis devices and to provide a fully automatic assistance to the physician in the creation of diagnosis systems for the diseases. The diagnosis systems, which can be easily used during routine controls, will make the timely information and the early treatment of patients possible.

For the dataset recognition aiming diagnosis of the diseases, we propose a two-stage approach. The first stage has used the clustering with ABC algorithm as selection criteria for feature selection, and, thus, more effective feature selection methods have been constituted. Hence, it has been

made possible both to select the related features faster and to reduce the feature vector dimensions. In the second stage, the reduced data was given to the SVM classifier and the accuracy rates were determined. The k-fold cross-validation method was used for improving the classifier reliability. The datasets we have worked on have been described in the Background section. As it is seen from the results, the performance of the proposed method is highly successful compared to other results attained and seems very promising for pattern recognition applications.

1.1. Background. The developed approach has been tested for the diagnosis of liver diseases and diabetes, which are commonly seen in the society and both reduce the quality of life. In the developed system, the hepatitis and liver disorders datasets were used for the diagnosis of liver disease, and the Diabetes dataset was used for the diagnosis of diabetes.

The liver disease diagnostics studies using the Hepatitis dataset were as follows: Polat and Güneş [1] proposed a new diagnostic method of hepatitis disease based on a hybrid feature selection (FS) method and artificial immune recognition system (AIRS) using fuzzy resource allocation mechanism. The obtained classification accuracy of the proposed system was 92.59%. A machine learning system studied by Polat and Güneş [2] was conducted to identify hepatitis disease. At first, the feature number of dataset on hepatitis disease was reduced from 19 to 10 by using in the feature selection (FS) subprogram and C4.5 decision tree algorithm. Then, fuzzy weighted preprocessing was used for weighting the dataset after normalizing between 0 and 1. AIRS classifier system was used while classifying the weighted input values. The classification accuracy of their system was 94.12%. Principal component analysis (PCA) and artificial immune recognition system (AIRS) were conducted for hepatitis disease prediction in the study by Polat and Güneş [3]. Classification accuracy, 94.12%, was obtained with the proposed system using 10-fold cross-validation. A method which had an accuracy value of 96.8% for hepatitis dataset was proposed by Kahramanli and Allahverdi [4], and in this method extracting rules from trained hybrid neural network was presented by using artificial immune systems (AISs) algorithm. An automatic diagnosis system using linear discriminant analysis (LDA) and adaptive network based on fuzzy inference system (ANFIS) was proposed by Dogantekin et al. [5] for hepatitis diseases. This automatic diagnosis system of hepatitis disease diagnostics was obtained with a classification accuracy of about 94.16%. Bascil and Temurtas [6] realized a hepatitis disease diagnosis based on a multilayer neural network structure that used the Levenberg- Marquardt algorithm as training algorithm for the weights update with a classification accuracy of 91.87% from 10-fold cross-validation.

The studies for the diagnosis of liver disease in using the liver disorders dataset were as follows: by Lee and Mangasarian [7], smoothing methods were applied to generate and solve an unconstrained smooth reformulation of the support vector machine for pattern classification using a completely arbitrary kernel. They termed such reformulation a smooth support vector machine (SSVM). Correct classification rate of the proposed system with CV-10 was 70.33% for liver disorders dataset. In Van Gestel et al.'s [8] article, the Bayesian evidence framework was combined with the LS-SVM classifier formulation. Correct classification rate of proposed system with CV-10 was 69.7% for liver disorders dataset. Gonçalves et al. [9] a new neuro-fuzzy model, especially created for record classification and rule extraction of databases, named as inverted hierarchical neuro-fuzzy BSP System (HNFB). Correct classification rate of this system was 73.33% for liver disorders dataset. Özşen and Güneş [10] aimed to contribute to an artificial immune system AIS by attaching this aspect and used the Euclidean distance, Manhattan distance, and hybrid similarity measure with simple AIS. Correct classification rate of the proposed system with AWAIS was 70.17%, with hybrid similarity measure 60.57%, with the Manhattan distance 60.21%, with the Euclidean distance 60.21% for liver disorders. Li et al. [11] proposed a nonlinear transformation method based on fuzzy to find classification information in the original data attribute values for a small dataset and used a support vector machine (SVM) as a classifier. Correct classification rate of the proposed system was 70.85% for liver disorders. Chen et al. [12] proposed an analytical approach by taking an integration of particle swarm optimization (PSO) and the 1-NN method. Correct classification rate of proposed system with 5-fold cross-validation was 68.99% for liver disorders dataset. A hybrid model based on integrating a case-based reasoning approach and a particle swarm optimization model were proposed by Chang et al. [13] for medical data classification.

Another disease that we selected is diabetes. Some of the most important studies conducted on this dataset are as follows: Şahan et al. [14] proposed attribute weighted artificial immune system (AWAIS) with weighting attributes due to their important degrees in class discrimination and using them for the Euclidean distances calculation. AWAIS had a classification accuracy of 75.87 using 10-fold cross-validation method for diabetes dataset. Polat and Güneş [15] worked on diabetes disease using principal component analysis (PCA) and adaptive neuro-fuzzy inference system (ANFIS). The obtained test classification accuracy was 89.47% by using the 10-fold cross-validation. Polat et al. [16] proposed a new learning system which is cascade and used generalized discriminant analysis and least square support vector machine. The classification accuracy was obtained as 82.05%. Kahramanli and Allahverdi [17] presented a hybrid neural network that achieves accuracy value of 84.24% using artificial neural network (ANN) and fuzzy neural network (FNN) together. Patil et al. [18] proposed hybrid prediction model (HPM) which uses Simple k-means clustering algorithm for verifying the chosen class labels and then using the classification algorithm on the result set. Accuracy value of HPM was 92.38%. Isa and Mamat [19] presented a modified hybrid multilayer perceptron (HMLP) network for improving the conventional one, and the average correct classification rate of the proposed system was 80.59%. Aibinu et al. [20] proposed a new biomedical signal classification method using complex-valued pseudo autoregressive (CAR) modeling approach. The presented technique obtained a classification accuracy of 81.28%.

(1) Start with the empty set $Y_0 = \{\varnothing\}$
(2) Select the next best feature
$x^+ = \arg\max[J(Y_k + x)]$; $x \notin Y_k$
(3) Update $Y_{k+1} = Y_k + x^+$; $k = k + 1$
(4) Goto 2

Pseudocode 1: Pseudo code for SFS [22].

2. Preliminaries

2.1. Feature Selection. Feature selection provides a smaller but more distinguishing subset compared to the starting data, selecting the distinguishing features from a set of features and eliminating the irrelevant ones. Reducing the dimension of the data is aimed by finding a small important features set. This results in both reduced processing time and increased classification accuracy.

The algorithm developed in this study was based on the sequential forward selection (SFS) algorithm, which is popular in these algorithms. SFS is a method of feature selection offered by Whitney [21]. Sequential forward selection is the simplest greedy search algorithm which starts from the empty set and sequentially adds the feature x^+ for obtaining results in the highest objective function $J(Y_k + x^+)$ when combined with the features Y_k that have already been selected. Pseudo code is given Pseudocode 1 for SFS [22].

In summary, SFS begins with zero attributes and then evaluates the whole feature subsets with only one feature, and the best performing one adds this subset to the best performing feature for subsets of the next larger size. This cycle repeats until there is no improvement in the current subset [23].

The objection function is critical for this algorithm. Finding the highest value of this function is an optimization problem. Clustering is an ideal method for the detection of feature differentiation. The developed method can be summarized using the ABC algorithm for feature selection aiming clustering problem adaptation.

2.1.1. Clustering with Optimization. Clustering is a grouping process running on the multi-dimentional data by using similarities. Distance criteria are used to evaluate similarities in samples set. Clustering problems can be expressed as the placement of every object into one K cluster for a given N number of objects and minimizing the sum of squares of the Euclidean distances between the centers of these objects in the cluster to which they belong. The function that uses the clustering algorithm is given in (1) [24] for minimizing:

$$J(w, z) = \sum_{i=1}^{N} \sum_{j=1}^{K} w_{ij} \left\| x_i - z_j \right\|^2. \quad (1)$$

Here, N is the number of samples, K is the number of clusters, x_i $(i = 1, \ldots, N)$ is the place of the ith sample, and the center of the jth sample z_j $(j = 1, \ldots, N)$ can be obtained by (2):

$$z_j = \frac{1}{N_j} \sum_{i=1}^{N} w_{ij} x_i. \quad (2)$$

Here, N_j is the number of samples in the jth cluster, and w_{ij} is the relationship of j cluster and x_i sample with a value of 1 or 0. If the sample i (x_i) belongs to the j cluster, w_{ij} is 1, otherwise that it is 0.

The clustering process that separates objects into groups can be performed by supervised or unsupervised learning. Training data in unsupervised clustering (also known as automatic clustering) does not need to set class tags. In supervised clustering, however, it should be specified so that the classes can learn the tags. In this study, the datasets used should contain class information since supervised clustering was used. Therefore, the optimization aims to find the centers of clusters by making the objective function minimize, which is the total of the samples distances to centers [24]. In this study, the sum of distances between all training cluster samples and the cluster center $(p_i^{\mathrm{CL_{known}}(x_j)})$ that samples belong to in the n-dimensional Euclidean space are minimized for adaptation [24]. Consider the following:

$$f_i = \frac{1}{D_{\mathrm{Train}}} \sum_{j=1}^{D_{\mathrm{Train}}} d\left(x_j, p_i^{\mathrm{CL_{known}}(x_j)}\right). \quad (3)$$

Here, D_{Train} is the number of training samples, and the total expression in the cost function is for normalizing the number to a value between 0.0 and 1.0. The $p_i^{\mathrm{CL_{known}}(x_j)}$ value indicates the center of the class that belongs to the sample that is used according to training data. Here, the ABC algorithm was chosen as the optimization method for clustering. Thus, ABC, as a new clustering method, can also be used in the feature selection algorithms.

2.2. Artificial Bee Colony (ABC) Algorithm. Artificial bee colony (ABC) algorithm, as a population-based stochastic optimization proposed by Karaboga in [24–26], realize the intelligent foraging behavior of honey bee swarms. It can be used for classification, clustering and optimization studies. Pseudocode of the ABC algorithm is given as Pseudocode 2.

An artificial group of bees in the ABC algorithm consists of three different groups: employed bees, onlooker bees, and scout bees. In this algorithm, the number of bees employed in the colony also equals the number of onlooker bees. Additionally, the number of employed bees or onlooker bees equals the number of solutions in the population. An onlooker bee is the bee that waits in the dance area to make the food source selection decision. An onlooker bee is named employed bee once it goes to a food source. An employed bee that has consumed the food source turns into a scout bee, and its duty is to perform a random search to discover new resources. Food supply position—which represents the solution to the optimization problem—and the amount of nectar

(1) Load training samples
(2) Generate the initial population z_i, $i = 1, \ldots, \mathrm{SN}$
(3) Evaluate the fitness (f_i) of the population
(4) set cycle to 1
(5) **repeat**
(6) **FOR** each employed bee {
Produce new solution v_i by using (6)
Calculate the value f_i
Apply greedy selection process}
(7) Calculate the probability values p_i for the solutions (z_i) by (5)
(8) **FOR** each onlooker bee {
Select a solution z_i depending on p_i
Produce new solution v_i
Calculate the value f_i
Apply greedy selection process}
(9) **If** there is an abandoned solution for he scout
then replace it with a new solution which will
be randomly produced by (7)
(10) Memorize the best solution so far
(11) cycle = cycle + 1
(12) until cycle = MCN

Pseudocode 2: Pseudo-code of the ABC algorithm [24].

in the food source depends on the quality of the associated solution. This value is calculated in (4).

$$\mathrm{fit}_i = \frac{1}{1 + f_i} \tag{4}$$

SN in the algorithm indicates the size of the population. At first, the ABC algorithm produces a distributed initial population $P(C = 0)$ of SN solutions (food source positions) randomly, where SN means the size of population. Each z_i solution is a D-dimensional vector for $i = 1, 2, 3, \ldots, \mathrm{SN}$. Here, D is the numbers of cluster products and input size for each dataset. After startup, an investigation is repeated on employed bees, onlooker bees, and scout bees processes until the number of population of positions ($C = 1, 2, \ldots, \mathrm{MCN}$) is completed. Here, MCN is the maximum cycle number.

An employed bee makes a small change in position due to the local knowledge in its memory, and a new source is generated. This bee makes a comparison of the nectar amount (fitness amount) of a new source with the nectar amount of previous source and decides which one is higher. If the new position is higher than the old one then it is assimilated into its memory and the old one is forgotten. Otherwise, the position of the previous one stays in its memory. All employed bees that complete the task of research share the position and nectar food source information with the onlooker bees that are in the dance area.

An onlooker bee evaluates the nectar information of all employed bees and chooses a food source depending on the probability of the nectar amount. This probability value (p_i) is calculated in (5). Just like the employed bees, the onlooker bee modifies the situation from memory and it checks the nectar amount of the candidate source. If its nectar amount is higher than the previous one and the new position is assimilated into memory and the old one is forgotten, then

$$p_i = \frac{\mathrm{fit}_i}{\sum_{n=1}^{\mathrm{SN}} \mathrm{fit}_n}, \tag{5}$$

where SN is the number of food sources which is equal to the number of employed bees and the fitness of the fit_i solution given in (4). The f_i given in (3) is the cost function of the cluster problem. ABC uses (6) for producing a candidate food position:

$$v_{ij} = z_{ij} + \phi_{ij}\left(z_{ij} - z_{kj}\right). \tag{6}$$

Here, $k \in \{1, 2, \ldots, \mathrm{SN}\}$ and $j \in \{1, 2, \ldots, D\}$ are randomly selected indexes. k is a random value different from i. ϕ_{ij} is a random number between $[-1, 1]$ which controls the production of neighboring food sources around z_{ij} and represents comparison of two food sources to a bee.

While onlooker and employed bees perform exploitation in the search area, scout bees control the discovery process and replace the consumed nectar food source with a new food source in the ABC algorithm. If the position cannot be improved as a previously determined cycle number, this food source is accepted as abandoned. The previously determined cycle number is defined as the "*limit*" for abandonment. In this case, there are three control parameters in ABC: the number of food sources (SN) which is equal to the number of employed and onlooker bees, the maximum cycle number (MCN), and the limit value.

If an abandoned source is assumed to be z_i and $j \in \{1, 2, \ldots, D\}$, the scout looks for a new source to replace z_i. This process is described by (7):

$$z_i^j = z_{\min}^j + \mathrm{rand}\,(0, 1)\left(z_{\max}^j - z_{\min}^j\right). \tag{7}$$

After (v_{ij}) which is each candidate position is produced, the position is evaluated by ABC and its performance is compared with previous one. The performance is compared with the previous one. If the new food source has an equal amount or more nectar than the old one, the new one takes place instead of the old food source in memory. Otherwise, the old one stays in its place in memory. So a greedy selection mechanism is used to make selections among the old source and one of the candidates.

2.3. Support Vector Machines (SVMs). SVM is an effective supervised learning algorithm used in classification and regression analyses for applications like pattern recognition, data mining, and machine learning application. SVM was developed in 1995 by Cortes and Vapnik [27]. Many studies have been conducted on SVM: a flexible support vector machine for regression, an evaluation of flyrock phenomenon based on blasting operation by using support vector machine [28, 29].

In this algorithm, there are two different categories separated by a linear plane. The training of the algorithm is determining the process for the parameters of this linear plane. In multiclass applications, the problem is categorized into groups as belonging either to one class or to others. SVM's use in pattern recognition is described below.

An n-dimensional pattern (object) x has n coordinates, $x = (x_1, x_2, \ldots, x_n)$, where each x is a real number, $x_i \in R$ for $i = 1, 2, \ldots, n$. Each pattern x_j belongs to a class $y_j \in \{-1, +1\}$. Consider a training set T of m patterns together with their classes, $T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$. Consider a dot product space S, in which the patterns x are embedded, $x_1, x_2, \ldots, x_m \in S$. Any hyperplane in the space S can be written as

$$\{x \in S \mid w \cdot x + b = 0\}, \quad w \in S,\ b \in R. \tag{8}$$

The dot product $w \cdot x$ is defined by

$$w \cdot x = \sum_{i=1}^{n} w_i x_i. \tag{9}$$

A training set of patterns can be separated as linear if there exists at least one linear classifier expressed by the pair (w, b) which correctly classifies all training patterns as can be seen in Figure 1. This linear classifier is represented by the hyperplane $H(w \cdot x + b = 0)$ and defines a region for class +1 patterns ($w \cdot x + b > 0$) and another region for class −1 patterns ($w \cdot x + b < 0$).

After the training process, the classifier becomes ready for prediction of the class membership on new patterns, different from training. The class of a pattern x_k is found from the following equation:

$$\text{class}(x_k) = \begin{cases} +1 & \text{if } w \cdot x_k + b > 0 \\ -1 & \text{if } w \cdot x_k + b < 0. \end{cases} \tag{10}$$

Thus, the classification of new patterns relies on only the sign of the expression $w \cdot x + b$ [30].

Sequential Minimal optimization is used in the training stage of SVM. SMO algorithm is a popular optimization method used to train the support vector machine (SVM). The dual presentation of an SVM primal optimization problem is indicated in (11):

$$\max_{\alpha} \quad \Psi(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j k(x_i, x_j) \alpha_i \alpha_j$$
$$\text{subject to} \quad \sum_{i=1}^{N} y_i \alpha_i = 0, \quad 0 \le \alpha_i \le C,\ i = 1, \ldots, n, \tag{11}$$

where x_i is a training sample, $y_i \in \{-1, +1\}$ is the corresponding target value, α_i is the Lagrange multiplier, and C is a real value cost parameter [31].

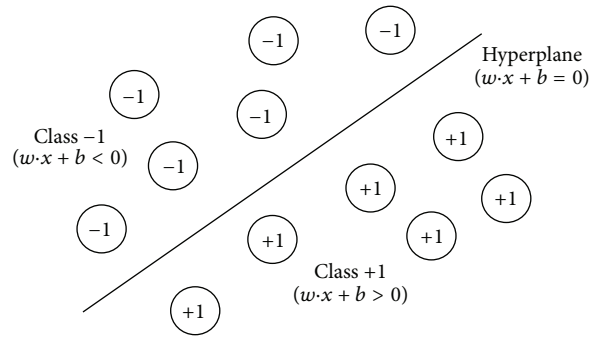


Figure 1: Linear classifier defined by the hyperplane $H(w \cdot x + b = 0)$.

2.4. Performance Evaluation. Four criteria for performance evaluation of hepatitis, liver disorders and diabetes datasets were used. These criteria are classification accuracy, confusion matrix, analysis of sensitivity and specificity, and k-fold cross-validation.

2.4.1. Classification Accuracy. In this study, the classification accuracies for the datasets are measured with the following the equation:

$$\text{accuracy}(T) = \frac{\sum_{i=1}^{N} \text{assess}(t_i)}{N}, \quad t_i \in T,$$
$$\text{assess}(t_i) = \begin{cases} 1, & \text{if classify}(t_i) \equiv \text{correctclassification}, \\ 0, & \text{otherwise}, \end{cases} \tag{12}$$

where T is the classified set of data items (the test set) and N is the number of testing samples of the dataset. We will also show the accuracy of our performed k-fold cross-validation (CV) experiment.

2.4.2. Confusion Matrix. The confusion matrix includes four classification performance indices: true positive, false positive, false negative, and true negative as given in Table 1. They are also usually used in the two-class classification problem to evaluate the performance.

2.4.3. Analysis of Sensitivity and Specificity. The following expressions were used for calculating sensitivity, specificity,

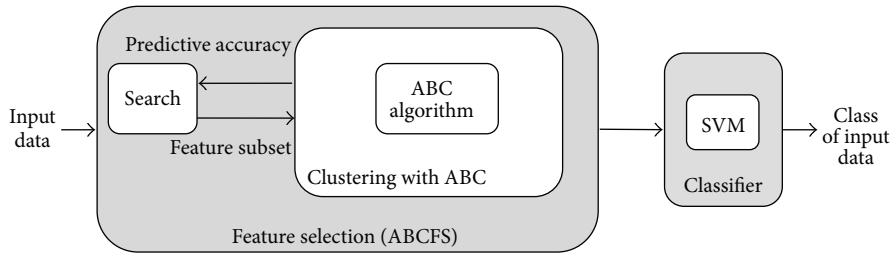


Figure 2: Block diagram of the proposed system.

Table 1: The four classification performance indices included in the confusion matrix.

Actual class	Predicted class	
	Positive	Negative
Positive	True positive (TP)	False negative (FN)
Negative	False positive (FP)	True negative (TN)

positive predictive value, and negative predictive value; we use [32]:

$$\begin{aligned}
\text{Sensitivity } (\%) &= \frac{\text{TP}}{\text{TP} + \text{FN}} \times 100, \\
\text{Specificity } (\%) &= \frac{\text{TN}}{\text{TN} + \text{FP}} \times 100, \\
\text{Positive predctive value } (\%) &= \frac{\text{TP}}{\text{TP} + \text{FP}} \times 100, \\
\text{Negative predicitive value } (\%) &= \frac{\text{TN}}{\text{TN} + \text{FN}} \times 100.
\end{aligned} \tag{13}$$

2.4.4. k-Fold Cross-Validation. k-fold cross-validation is used for the test result to be more valuable [33]. In k-fold cross-validation, the original sample is divided into random k subsamples, one of which is retained as the validation data for model testing and the remaining k-1 sub-samples are used for training The cross-validation process is then repeated k times (the folds), with each of the k sub-samples used exactly once as the validation data. The process is repeated k times (the folds), with each of the k sub-samples used only once as the validation data. The average of k results from the folds gives the test accuracy of the algorithm [34].

3. Experimental Work

Less distinctive features of the data set affect the classification negatively. Such data especially decrease the speed and the system performance significantly. With the proposed system, using the feature selection algorithm, the features with less discriminant data were eliminated. The reduced data set increased the testing success of the classifier and the rate of the system. From Figure 2, the proposed system has two phases. At the first phase, as selection criteria, clustering with ABC algorithm was used for feature selection, and, thus, a more effective feature selection method was constituted. Hence, it has been made possible both to select the related features in a shorter period of time and to reduce the dimension of the feature vector. At second stage, the obtained reduced data is supplied to the SVM classifier to determine the accuracy rates. The k-fold cross-validation was used for the classifier reliability improvement.

In this study, ABCFS + SVM system is suggested in order to solve the three classification problem named as Hepatitis dataset, Liver Disorders dataset, Diabetes dataset, respectively.

3.1. Datasets. We used the dataset from the UCI machine learning database [35], which is commonly used among researchers for classification, that gives us a chance to compare the performance of our method with others. The datasets of this work can be defined shortly as follows.

3.1.1. Hepatitis Dataset. This dataset was donated by Jozef Stefan Institute, Yugoslavia. The purpose of the dataset is to predict the presence or absence of hepatitis disease from the different medical tests results of a patient. This database contains 19 attributes. There are 13 binary and 6 discrete values. Hepatitis dataset includes 155 samples from two different classes (32 "die" cases, 123 "live" cases). This dataset contains missing attribute values. We substituted the missing data by frequently encountered values of own class. Attributes of symptoms that are obtained from patient are given in Table 2 [3, 35].

3.1.2. Liver Disorders Dataset. The liver disorders dataset is named as BUPA liver disorders. The liver disorders database includes 6 features, that is, MCV, alkphos, SGPT, SGOT, gammaGT and drinks. There are 345 data in total and each sample is taken from an unmarried man. Two hundred of them are chosen for one class with the remaining 145 are in the other. The first 5 features are all blood tests which are sensitive to liver disorders that arise from excessive alcohol consumption. This dataset is donated by Richard S. Forsyth et al. in 1990. The attributes are given in Table 3 [13].

3.1.3. Diabetes Dataset. This dataset contains 768 samples, where each sample has 8 features which are eight clinical findings. All patients of the dataset are Pima Indian women in which the youngest one is 21 years old and living near Phoenix, Arizona, USA. The binary target variable can take "0" or "1." If it takes "1," it means a positive test for Diabetes, or if it takes "0," it means a negative test. There are 268 different cases in class "1" and 500 different cases in class "0." The features and parameters are given in Table 4 [16].

Table 2: Range values and attribute names for hepatitis dataset [35].

The number of attribute	The name of attribute	Interval of attribute
1	Age	7–78
2	Sex	Male, Female
3	Steroid	No, Yes
4	Antivirals	No, Yes
5	Fatigue	No, Yes
6	Malaise	No, Yes
7	Anorexia	No, Yes
8	Liver big	No, Yes
9	Liver firm	No, Yes
10	Spleen palpable	No, Yes
11	Spiders	No, Yes
12	Ascites	No, Yes
13	Varices	No, Yes
14	Bilirubin	0.3–8
15	Alk phosphate	26–295
16	SGOT	14–648
17	Albumin	2.1–6.4
18	Protime	0–100
19	Histology	No, Yes

Table 3: Range values and attribute names for liver disorders dataset [35].

The number of attribute	The name of attribute	Description of the attribute	Interval of attribute
1	MCV	Mean corpuscular volume	65–103
2	Alkphos	Alkaline phosphatase	23–138
3	SGPT	Alamine aminotransferase	4–155
4	SGOT	Aspartate aminotransferase	5–82
5	gammaGT	Gamma-glutamyl transpeptidase	5–297
6	Drinks	Number of half-pint equivalents of alcoholic beverages drunk per day	0–20

3.2. Feature Selection with ABC. In the system, a searching process runs to find the best feature subset same like sequential forward selection algorithm. Prediction accuracy for feature selection is found by ABC clustering. Pseudocode of the developed feature selection algorithm based on ABC is given in Pseudocode 3.

In Pseudocode 3, n is sample count and p is desired feature count which is selected as providing the highest performance criteria. While *data* represents the entire dataset, *Data_all* includes the features that are considered chosen. *Train_data_all* is generated by taking 75% of the data found in all classes of *Data_all*. *Test_data_all* is generated by taking 25% of the data that are found in all classes of *Data_all*.

Table 4: Features and parameters of the diabetes dataset.

Features	Mean	Standard deviation	Min	Max
Number of times pregnant	3.8	3.4	0	17
Plasma glucose concentration, 2 h in an oral glucose tolerance test	120.9	32.0	0	199
Diastolic blood pressure (mm Hg)	69.1	19.4	0	122
Triceps skinfold thickness (mm)	20.5	16.0	0	99
2-hour serum insulin (mu U/mL)	79.8	115.2	0	846
Body mass index (kg/m^2)	32.0	7.9	0	67.1
Diabetes pedigree function	0.5	0.3	0.078	2.42
Age (years)	33.2	11.8	21	81

Table 5: List of datasets.

Databases	Number of classes	Samples	Number of features	Number of selected features	Selected features
Hepatitis	2	155	19	11	12, 14, 13, 15, 18, 1, 17, 5, 16, 2, 4
Liver disorders	2	345	6	5	5, 3, 2, 4, 1
Diabetes	2	768	8	6	2, 8, 6, 7, 4, 5

Table 6: List of classification parameters.

Parameters	Value
Method	SVM
Optimization algorithm	SMO
Validation method	k-fold cross-validation (10-fold CV)
Kernel_Function	Linear
TolKKT	$1.0000e-003$
MaxIter	15000
KernelCacheLimit	5000
The initial value	Random

Table 7: Performance of classification for the hepatitis, liver disorders, and diabetes datasets.

Performance criteria	Hepatitis dataset	Liver disorders dataset	Diabetes dataset
Classification accuracy (%)	94.92	74.81	79.29
Sensitivity (%)	97.13	88.22	89.84
Specificity (%)	88.33	56.68	59.61
Positive predictive value (%)	96.91	73.99	80.63
Negative predictive value (%)	88.33	78.57	75.65

Thus, a uniform dispersion was obtained according to the classes. Train data that belongs to each class (*train_data_class*) is trained by the ABC algorithm, which has been modified to cluster. At the end of training, 10 feature vectors named food and representing each class are obtained. Goodness of the chosen feature cluster is described by the food values

```
n = sample_count;
Selected_features = {∅}
For p = 1 to desired_feature_count
  For c = 1 to feature_count
    Data_all = data(Selected_features + feature(c));
    For i = 1 to class_number
      Train_data_class(i) = partition(rand(Data_all(class == i)), 0.75)
      Test_data_class(i) = partition(rand(Data_all(class = i)), others);
        [foods(i)] = Modified_ABC_algortihm (train_data_class(i), performance_function);
    End for i
    Test_data_all = merge(test_data_class);
    For i = 1: size(Test_data_all)
      For k = 1: class_count
        For j = 1: count(foods(k))
          distance(k, j) = oklid_distance(foods(j, k)-test_data(i));
        End for j
        min_dist(k) = min(distance(k));
      End for k
      [result_of_test_data(i), class_of_test_data(i)] = min(min_dist);
    End for i
    Performance_criteria(feature(c))
= sum(class_of_test_data(i) == class_of_test_data_expected);
  End for c
  Best_feature(c) = arg max(performance_criteria(feature(c))
  Selected_features = Selected_fetures + best_feature(c)
  End for p
```

Pseudocode 3: Pseudo-code of developed feature selection algorithm based on ABC.

Table 8: Classification accuracies obtained by our method and other classifiers for the hepatitis dataset.

Author (year)	Method	Classification accuracy (%)
Polat and Güneş (2006) [1]	FS-AIRS with fuzzy res. (10-fold CV)	92.59
Polat and Güneş (2007) [2]	FS-Fuzzy-AIRS (10-fold CV)	94.12
Polat and Güneş (2007) [3]	AIRS (10-fold CV)	76.00
	PCA-AIRS (10-fold CV)	94.12
Kahramanli and Allahverdi (2009) [4]	Hybrid system (ANN and AIS) (without k-fold CV)	96.8
Dogantekin et al. (2009) [5]	LDA-ANFIS	94.16
Bascil and Temurtas (2011) [6]	MLNN (MLP) + LM (10-fold CV)	91.87
Our study	ABCFS + SVM (10-fold CV)	**94.92**

Table 9: Classification accuracies obtained by our method and other classifiers for the liver disorders dataset.

Author (year)	Method	Classification accuracy (%)
Lee and Mangasarian (2001) [7]	SSVM (10-fold CV)	70.33
van Gestel et al. (2002) [8]	SVM with GP (10-fold CV)	69.7
Gonçalves et al. (2006) [9]	HNFB-1 method	73.33
Özşen and Güneş (2008) [10]	AWAIS (10-fold CV)	70.17
	AIS with hybrid similarity measure (10-fold CV)	60.57
	AIS with Manhattan distance (10-fold CV)	60.21
	AIS with Euclidean distance (10-fold CV)	60.00
Li et al. (2011) [11]	A fuzzy-based nonlinear transformation method + SVM	70.85
Chen et al. (2012) [12]	(PSO) + 1-NN method (5-fold CV)	68.99
Chang et al. (2012) [13]	CBR + PSO (train: 75%-test: 25%)	76.81
Our study	ABCFS + SVM (train: 75%-test: 25%)	**82.55**
	ABCFS + SVM (10-fold CV)	**74.81**

Table 10: Classification accuracies obtained by our method and other classifiers for diabetes dataset.

Author (year)	Method	Classification accuracy (%)
Şahan et al. (2005) [14]	AWAIS (10-fold CV)	75.87
Polat and Güneş (2007) [15]	Combining PCA and ANFIS	89.47
Polat et al. (2008) [16]	LS-SVM (10-fold CV)	78.21
	GDA-LS-SVM (10-fold CV)	82.05
Kahramanli andAllahverdi (2008) [17]	Hybrid system (ANN and FNN)	84.2
Patil et al. (2010) [18]	Hybrid prediction model (HPM) with reduced dataset	92.38
Isa and Mamat (2011) [19]	Clustered-HMLP	80.59
Aibinu et al. (2011) [20]	AR1 + NN (3-fold CV)	81.28
Our study	ABCFS + SVM (train: 75%-test: 25%)	**86.97**
	ABCFS + SVM (10-fold CV)	**79.29**

accuracy representing the test dataset. The error value is found by taking the difference between the test data class and the food value class having a minimum Euclidean distance to the test data class.

The performance value shows the suitability of the added property. The most appropriate property value does not belong to the chosen properties cluster. This process is repeated by starting from an empty cluster up until the desired feature number. The decline in the value of rising performance trend is for determining the maximum number of features. In summary, in ABCFS, it starts from selecting the feature set as empty, then adds the feature(c) that results in the highest objective function.

We selected colony size 20, maximum cycle/generation number (MCN) 300, and limit value 200. The algorithm was run 100 times. Performance value is found by taking the average of these algorithm results.

The datasets used for evaluating ABCFS performance and their features are as follows: the number of classes, the number of samples, the number of features and the number of selected features, which are given in Table 5.

3.3. SVM Classification Parameters. The reliability of the classifier was provided by the k-fold cross-validation method. While this classifier was used, the training was performed according to the parameters in Table 6.

4. Experimental Results and Discussion

ABCFS + SVM method test results developed for the hepatitis dataset, liver disorders dataset and diabetes datasets are given in Pseudocode 3. These test results contain the classification performance values achieved by the developed methodology by the help of 10-fold cross-validation. The performance values include average classification accuracy, sensitivity, specificity, positive predictive value, and negative predictive value of the proposed system which are given in Table 7. The results of the study show that the average correctness rate of the studies performed so far on all used datasets by employing the method of k-fold cross-validation is a very promising result.

For the hepatitis dataset, the comparisons with the other systems are given in Table 8.

For the liver disorders dataset, the comparisons with the other systems are given in Table 9.

For the diabetes dataset, the comparisons with the other systems are given in Table 10.

5. Conclusions

This study was designed for use in the diagnosis of liver and diabetes. In these databases that were used, there are some redundant and low-distinctive features. These features are very important factors affecting the success of the classifier and the system processing time. In the system we have developed, the elimination of these redundant features increased the system speed and success. The artificial bee Colony (ABC) algorithm, which is a very popular optimization method, was used for the feature selection process in the study. The ABC-based feature selection algorithm that was developed in this study is the first example of the ABC algorithm used in the field of feature selection. The databases that are subjected to feature selection are classified using SVM. In order to achieve a reliable performance of the classifier, the 10-fold cross-validation method was used. The system results were compared with the literature articles that use the same databases. Classification accuracy of the proposed system reached 94.92%, 74.81%, and 79.29% for hepatitis dataset, liver disorders dataset and diabetes dataset, respectively. Obtained results show that the performance of the proposed method is highly successful compared to other results attained and seems very promising for pattern recognition applications.

Acknowledgment

The authors would like to thank Selcuk University Scientific Research Projects Coordinatorship for the support of this paper.

References

[1] K. Polat and S. Güneş, "Hepatitis disease diagnosis using a new hybrid system based on feature selection (FS) and artificial immune recognition system with fuzzy resource allocation," *Digital Signal Processing*, vol. 16, no. 6, pp. 889–901, 2006.

[2] K. Polat and S. Güneş, "Medical decision support system based on artificial immune recognition immune system (AIRS), fuzzy

weighted pre-processing and feature selection," *Expert Systems with Applications*, vol. 33, no. 2, pp. 484–490, 2007.

[3] K. Polat and S. Güneş, "Prediction of hepatitis disease based on principal component analysis and artificial immune recognition system," *Applied Mathematics and Computation*, vol. 189, no. 2, pp. 1282–1291, 2007.

[4] H. Kahramanli and N. Allahverdi, "Extracting rules for classification problems: AIS based approach," *Expert Systems with Applications*, vol. 36, no. 7, pp. 10494–10502, 2009.

[5] E. Dogantekin, A. Dogantekin, and D. Avci, "Automatic hepatitis diagnosis system based on linear discriminant analysis and adaptive network based on fuzzy inference system," *Expert Systems with Applications*, vol. 36, no. 8, pp. 11282–11286, 2009.

[6] M. S. Bascil and F. Temurtas, "A study on hepatitis disease diagnosis using multilayer neural network with Levenberg Marquardt training algorithm," *Journal of Medical Systems*, vol. 35, no. 3, pp. 433–436, 2011.

[7] Y. J. Lee and O. L. Mangasarian, "SSVM: a smooth support vector machine for classification," *Computational Optimization and Applications*, vol. 20, no. 1, pp. 5–22, 2001.

[8] T. van Gestel, J. A. K. Suykens, G. Lanckriet, A. Lambrechts, B. de Moor, and J. Vandewalle, "Bayesian framework for least-squares support vector machine classifiers, Gaussian processes, and Kernel Fisher discriminant analysis," *Neural Computation*, vol. 14, no. 5, pp. 1115–1147, 2002.

[9] L. B. Gonçalves, M. M. B. R. Vellasco, M. A. C. Pacheco, and F. J. de Souza, "Inverted Hierarchical Neuro-Fuzzy BSP system: a novel neuro-fuzzy model for pattern classification and rule extraction in databases," *IEEE Transactions on Systems, Man and Cybernetics C*, vol. 36, no. 2, pp. 236–248, 2006.

[10] S. Özşen and S. Güneş, "Effect of feature-type in selecting distance measure for an artificial immune system as a pattern recognizer," *Digital Signal Processing*, vol. 18, no. 4, pp. 635–645, 2008.

[11] D. C. Li, C. W. Liu, and S. C. Hu, "A fuzzy-based data transformation for feature extraction to increase classification performance with small medical data sets," *Artificial Intelligence in Medicine*, vol. 52, no. 1, pp. 45–52, 2011.

[12] L. F. Chen, C. T. Su, K. H. Chen, and P. C. Wang, "Particle swarm optimization for feature selection with application in obstructive sleep apnea diagnosis," *Neural Computing and Applications*, vol. 21, no. 8, pp. 2087–2096, 2012.

[13] P. C. Chang, J. J. Lin, and C. H. Liu, "An attribute weight assignment and particle swarm optimization algorithm for medical database classifications," *Computer Methods and Programs in Biomedicine*, vol. 107, no. 3, pp. 382–392, 2012.

[14] S. Şahan, K. Polat, H. Kodaz, and S. Güneş, "The medical applications of attribute weighted artificial immune system (AWAIS): diagnosis of heart and diabetes diseases," in *Artificial Immune Systems*, vol. 3627 of *Lecture Notes in Computer Science*, pp. 456–468, 2005.

[15] K. Polat and S. Güneş, "An expert system approach based on principal component analysis and adaptive neuro-fuzzy inference system to diagnosis of diabetes disease," *Digital Signal Processing*, vol. 17, no. 4, pp. 702–710, 2007.

[16] K. Polat, S. Güneş, and A. Arslan, "A cascade learning system for classification of diabetes disease: generalized discriminant analysis and least square support vector machine," *Expert Systems with Applications*, vol. 34, no. 1, pp. 482–487, 2008.

[17] H. Kahramanli and N. Allahverdi, "Design of a hybrid system for the diabetes and heart diseases," *Expert Systems with Applications*, vol. 35, no. 1-2, pp. 82–89, 2008.

[18] B. M. Patil, R. C. Joshi, and D. Toshniwal, "Hybrid prediction model for t ype-2 diabetic patients," *Expert Systems with Applications*, vol. 37, no. 12, pp. 8102–8108, 2010.

[19] N. A. M. Isa and W. M. F. W. Mamat, "Clustered-hybrid multilayer perceptron network for pattern recognition application," *Applied Soft Computing Journal*, vol. 11, no. 1, pp. 1457–1466, 2011.

[20] A. M. Aibinu, M. J. E. Salami, and A. A. Shafie, "A novel signal diagnosis technique using pseudo complex-valued autoregressive technique," *Expert Systems with Applications*, vol. 38, no. 8, pp. 9063–9069, 2011.

[21] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognition Letters*, vol. 15, no. 11, pp. 1119–1125, 1994.

[22] L. Ladha and T. Deepa, "Feature selection methods and algorithms," *International Journal on Computer Science and Engineering*, vol. 3, no. 5, pp. 1787–1797, 2011.

[23] M. Sasikala and N. Kumaravel, "Comparison of feature selection techniques for detection of malignant tumor in brain images," in *Proceedings of the Annual IEEE INDICON '05*, pp. 212–215, December 2005.

[24] 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.

[25] 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.

[26] B. Akay and D. Karaboga, "A modified artificial bee colony algorithm for real-parameter optimization," *Information Sciences*, vol. 192, pp. 120–142, 2012.

[27] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[28] H. Amini, R. Gholami, M. Monjezi, S. R. Torabi, and J. Zadhesh, "Evaluation of flyrock phenomenon due to blasting operation by support vector machine," *Neural Computing and Applications*, vol. 21, no. 8, pp. 2077–2085, 2012.

[29] X. B. Chen, J. Yang, and J. Liang, "A flexible support vector machine for regression," *Neural Computing and Applications*, vol. 21, no. 8, pp. 2005–2013, 2012.

[30] O. Ivanciuc, *Reviews in Computational Chemistry*, edited by K. B. Lipkowitz and T. R. Cundari, 2007.

[31] T. W. Kuan, J. F. Wang, J. C. Wang, P. C. Lin, and G. H. Gu, "VLSI design of an SVM learning core on sequential minimal optimization algorithm," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 4, pp. 673–683, 2012.

[32] K. Polat and S. Güneş, "Breast cancer diagnosis using least square support vector machine," *Digital Signal Processing*, vol. 17, no. 4, pp. 694–701, 2007.

[33] D. François, F. Rossi, V. Wertz, and M. Verleysen, "Resampling methods for parameter-free and robust feature selection with mutual information," *Neurocomputing*, vol. 70, no. 7–9, pp. 1276–1288, 2007.

[34] N. A. Diamantidis, D. Karlis, and E. A. Giakoumakis, "Unsupervised stratification of cross-validation for accuracy estimation," *Artificial Intelligence*, vol. 116, no. 1-2, pp. 1–16, 2000.

[35] C. L. Blake and C. J. Merz, *University of California at Irvine Repository of Machine Learning Databases*, 1998, http://www.ics.uci.edu/~mlearn/MLRepository.html.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 160687, 8 pages
http://dx.doi.org/10.1155/2013/160687

Research Article

QPSO-Based Adaptive DNA Computing Algorithm

Mehmet Karakose[1] and Ugur Cigdem[2]

[1] *Computer Engineering Department, Firat University, Elazig, Turkey*
[2] *Computer Programming Department, Gaziosmanpaşa University, Tokat, Turkey*

Correspondence should be addressed to Mehmet Karakose; mkarakose@firat.edu.tr

Received 3 May 2013; Accepted 20 June 2013

Academic Editors: P. Agarwal and Y. Zhang

DNA (deoxyribonucleic acid) computing that is a new computation model based on DNA molecules for information storage has been increasingly used for optimization and data analysis in recent years. However, DNA computing algorithm has some limitations in terms of convergence speed, adaptability, and effectiveness. In this paper, a new approach for improvement of DNA computing is proposed. This new approach aims to perform DNA computing algorithm with adaptive parameters towards the desired goal using quantum-behaved particle swarm optimization (QPSO). Some contributions provided by the proposed QPSO based on adaptive DNA computing algorithm are as follows: (1) parameters of population size, crossover rate, maximum number of operations, enzyme and virus mutation rate, and fitness function of DNA computing algorithm are simultaneously tuned for adaptive process, (2) adaptive algorithm is performed using QPSO algorithm for goal-driven progress, faster operation, and flexibility in data, and (3) numerical realization of DNA computing algorithm with proposed approach is implemented in system identification. Two experiments with different systems were carried out to evaluate the performance of the proposed approach with comparative results. Experimental results obtained with Matlab and FPGA demonstrate ability to provide effective optimization, considerable convergence speed, and high accuracy according to DNA computing algorithm.

1. Introduction

DNA computing is a new field of research which performs computing using the biomolecular structure of DNA molecules. The first study performed in the field of DNA computing was the solution of the problem of traveling salesmen composed of 7 cities by Adleman using real DNA molecules [1]. The application was realized by creating solution environment in the biology laboratory and using biochemical reactions. The cities and distances that make up the problem of traveling salesmen were coded using DNA series and all ways that might be solution were created using polymer chain reaction. Upon the success of the study performed a new algorithm for the solution of multivariable problems was acquired. In the continuation of this study Lipton solved satisfiability problem included in NP (nonpolynomial) problem class using DNA computing in a similar way [2]. Lipton demonstrated the application and that DNA computing can be used for the solution of the problems containing logical equations as well. After this study Lin et al. [3] performed the design and optimization of PI (proportional integral) parameters using DNA computing. Ding and Ren used similar DNA computing algorithm with the abovementioned study for setting turbid inspecting parameters [4]. Kim and Lee applied DNA computing algorithm with a different method for setting the PID parameters [5]. In the study performed DNA molecules were used in coding and setting the PI parameters. The results of the application through computer simulation indicated that high success was acquired in this field. Wang et al. compared DNA computers and electronic computers and suggested that DNA computers were more advantageous [6]. As a result of the applications given above, DNA computing was developed rapidly and used in many scientific studies [7–15]. It has been used frequently, particularly in NP problems, coloring problems of graphics in setting the inspecting parameters, arithmetic operations, signal processing problems, and ciphering the data [16–24]. DNA computing algorithm performs computing using the natural characteristics of DNA molecules. Those characteristics include parallel operations, storing high amount of

data, providing energy saving, and having significant role in computing. Those characteristics are listed quite effectively in the solution of complex and difficult problems. In order to use the DNA computing more effectively, scientists dealt with the design of total natural DNA computers composed of DNA molecules [25].

In Section 2 of this study, DNA computing algorithm is explained in detail. In Section 3 QPSO algorithm is mentioned. In Section 4 DNA computing is made applicable with QPSO algorithm; parameters that increase the effectiveness of DNA computing algorithm are found using QPSO and used for the proposed algorithm. And in the last section the acquired simulation results are given. It is understood from the simulation results that the proposed method produces better values and is more successful.

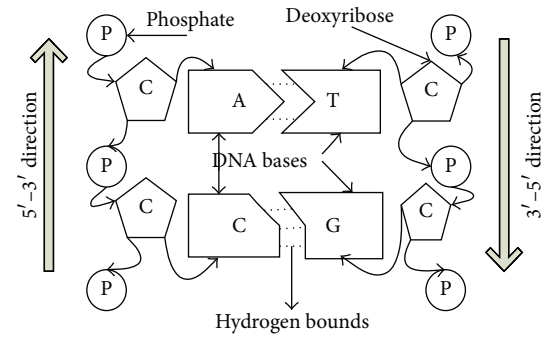


Figure 1: The general structure of the DNA molecule.

2. DNA Computing Algorithm

DNA computing is a new optimization algorithm performing computing using DNA molecules which store genetic information of the living things. While performing the computing using DNA, DNA bases that make up the ground of the DNA molecules are used. DNA bases are Adenine (A), Guanine (G), Cytosine (C), and Thymine (T) and in this study they have been converted into numerical values including A-0, G-1, C-2, and T-3 [9]. Adenine and Thymine and Guanine and Cytosine complete each other [26–28]. DNA molecules have a structure due to the double and triple hydrogen ties among those bases and in the computing process where number of hydrogen ties is used. In Particular the completion situations of DNA bases and numbers of hydrogen ties are used in the computing performed in solution environments. There are 2 hydrogen ties between Adenine and Thymine and 3 hydrogen ties between Guanine and Cytosine [27]. DNA molecules exist in the form of single serial and double spiral serial. Using the single DNA serials synthesis and reproduction of DNA molecules is realized. Double spiral DNA serials are created according to the Watson-Crick complementation rule. According to this rule Adenine and Thymine and Guanine and Cytosine combine. No emergence is possible. DNA molecules have many advantages including performing parallel operations, storing high amount of data, and using those data for many years without any corruption. These advantages are used sufficiently while performing the computing; the algorithm produces better results. In Figure 1, the general structure of the DNA molecule was shown.

Parallel computing provides a quick conclusion for operations. It is possible to complete the problems that require very big operation volumes and take a long time in the solution with DNA molecules. Some similar problems have many variables and parameters. Coding these variables and parameters and modeling the system generally require data. DNA molecules make great contributions to the solution of these problems with their capability to store data. While DNA computing is applying, the errors are made in the sequence of DNA serials to affect the result negatively. As a result of DNA computing, the values acquired with other optimization algorithm are not deemed to be final results. The studies performed on DNA computing have been implemented in two different manners. The most commonly used method among these is the computing performed in solution environment [26–28]. The cost of performing the computing in solution environment is more because machines are needed to produce DNA serials and DNA synthesis. Furthermore solution should be prepared and solution tubes should be used. Systems named gel electrophoresis are needed for acquiring and analyzing the results [22, 23]. And another algorithm used for DNA computing is the numerical DNA computing algorithm. In this algorithm DNA serials are created in electronic computers and they are converted into numerical values for computing. Although the cost of computing performed electronically is less, the mentioned advantages of DNA molecules cannot be applied on problems completely because when the DNA molecules are used in solution environment, they only can use the abovementioned characteristics.

Numerical DNA computing algorithm is similar to the genetic algorithms; however it is different from genetic algorithms. It uses A, T, G, and C bases rather than dual number of the systems and the solution set of the problems is composed of these bases [29]. Furthermore DNA computing has two new mutation operations. These are enzyme and virus mutations and provide great advantage while computing is performed. Enzyme mutation is the operation of deletion in one or more DNA parts from any DNA serial. And virus mutation is the operation of adding one or more DNA parts to any DNA serial [9, 19, 20]. Enzyme and virus mutations provide continuous renewal of the population and prevent focusing on local optimum points, thus give the algorithm a global search capacity. In Table 1, the conversion of DNA serials into numerical data has been given.

3. QPSO Algorithm

QPSO is an algorithm having the capacity of global searching. This algorithm has been developed inspired by living things that move as masses including insects, bees and were used for the solution of many problems in the literature. In the QPSO algorithm which is a population-based algorithm, individuals act together and compose the masses. While the speed of population is used in PSO algorithm, the next position of the

Table 1: Coding table of DNA computing algorithm.

	DNA coding system A, G, C, T	Quartet coding system 0, 1, 2, 3	Binary coding system 00, 01, 10, 11	Decimal coding system	Minimum value of K_p and K_i	Maximum value of K_p and K_i
K_p	AGT, TAA, TTT	013, 300, 333	000111, 110000, 111111	$0 * 16 + 1 * 4 + 3 * 1 = 7$ $3 * 16 + 0 * 4 + 0 * 1 = 48$ $3 * 16 + 3 * 4 + 3 * 1 = 63$	0	63
K_i	GTT, CCA, AGC	133, 220, 012	011111, 101000, 000110	$1 * 1 + 3 * 1/4 + 3 * 1/16 = 1.9375$ $2 * 1 + 2 * 1/4 * 0 * 1/16 = 2.5$ $0 * 1 + 1 * 1/4 + 2 * 1/16 = 0.375$	0	4

population is determined by taking the average of the best results acquired by the particles that make up the population in QPSO locally. In order to update the speeds and positions of the particles in QPSO algorithm, (1), (2), and (3) are used [30].

While preparing the algorithm the best information acquired by each particle ($p$best) and the best information realized by the mass ($g$best = min ($p$best)) variables are used. Using these two variables the movement point of the mass P value is calculated using (1). The $Q1$ and $Q2$ variables in (1) represent random numbers produced in the interval 0-1. In determining the next position of the mass QPSO algorithm uses the variable of $m$best which is the best arithmetic average of mass instead of $g$best. $m$best value is calculated using (2). In (2) n represents the population size. After computing the $m$best value for the next step of the population the x value is computed by using (3). The β variable in (3) is a value to be determined by the applier. And the u variable is another one which takes value at the interval of 0-1:

$$p = \frac{(Q1 * p\text{best} + Q2 * g\text{best})}{Q1 + Q2}, \quad (1)$$

$$m\text{best} = \frac{\left(\sum_{m=1}^{n} p\text{best}_i\right)}{n}, \quad (2)$$

$$x(i+1) = p \pm \beta \cdot |m\text{best} - x(i)| \cdot \ln\left(\frac{1}{u}\right). \quad (3)$$

4. QPSO-Based Adaptive DNA Computing Algorithm

In order to increase the global search capacity and effectiveness of DNA computing algorithm in problem solving, parameters should be applicable. The size of the population suitable for this algorithm (Pb), maximum operation number *(N)*, Crossover rate (Mu), enzyme proportion (E), and virus proportion (V) affect the local and global search capacity of the algorithm positively. Making all these parameters adaptable simultaneously can sometimes give negative results. The real target aimed by adapting the parameters is to increase the diversity in the population and prevent the focusing on local optimum points. Furthermore the adaptable algorithms can easily be applied for the solution of many optimization problems. In this study it is aimed to find the optimum values of the parameters considering the conformity values acquired by the population in all iterations of the algorithm. Using the equation depending on conformity function together with the QPSO algorithm, the parameter values are determined. The acquired values are taken as optimized parameter values when optimum solution is achieved. These parameters which are fixed at the beginning are made dynamic and increasing the effectiveness of the algorithm is aimed. In Figure 2, the steps of QPSO based on numerical DNA computing algorithm are shown.

As shown in Figure 3, the proposed approach can be modeled to find the parameters values of DNA computing algorithm using a QPSO algorithm. In the proposed model variables of K_p and K_i are coded with DNA serials at the length of 3 bases. K_p and K_i values are created randomly using A, G, C, and T bases and converted into numerical data using 0, 1, 2, and 3 values, respectively. The serials are converted into first system of 4 then system of 10 and used in numerical environment. The parameter values used for making adaptive DNA computing are explained as follows.

Population Size (Pb). Population size is one of the most significant characteristics that contribute to the solution of the problem. The size of the problem and its complex structure are taken into consideration while determining the value of those parameters. When the population size is increased, a broader solution area is created. The time of solution gets longer. Furthermore unnecessary expansion of the solution area causes big time losses and the solution time for the problem increases rapidly. For this reason optimum value should be selected in the applications. In setting of the population size, (4) is used. The f variable given in (4) is the conformity function used for the algorithm. $f(n)$ gives the conformity value of the nth element of the population, and $f(1)$ gives the conformity value of the first element of the population. Max(f) gives the maximum value of the population and min(f) gives the minimum value. These two values provide information about the distribution of the values belonging to population elements. The number 15 given in (4) is the base value and it is used for the purpose of the probability of the size of the population to be zero. It is determined with the method of trial and error:

$$\text{Pop\_size} = \frac{(\max(f) - \min(f))}{(f(n) - f(1))} + 15. \quad (4)$$

Maximum Number of Operations (N). Maximum number of operations gives the number of processes of the cycle

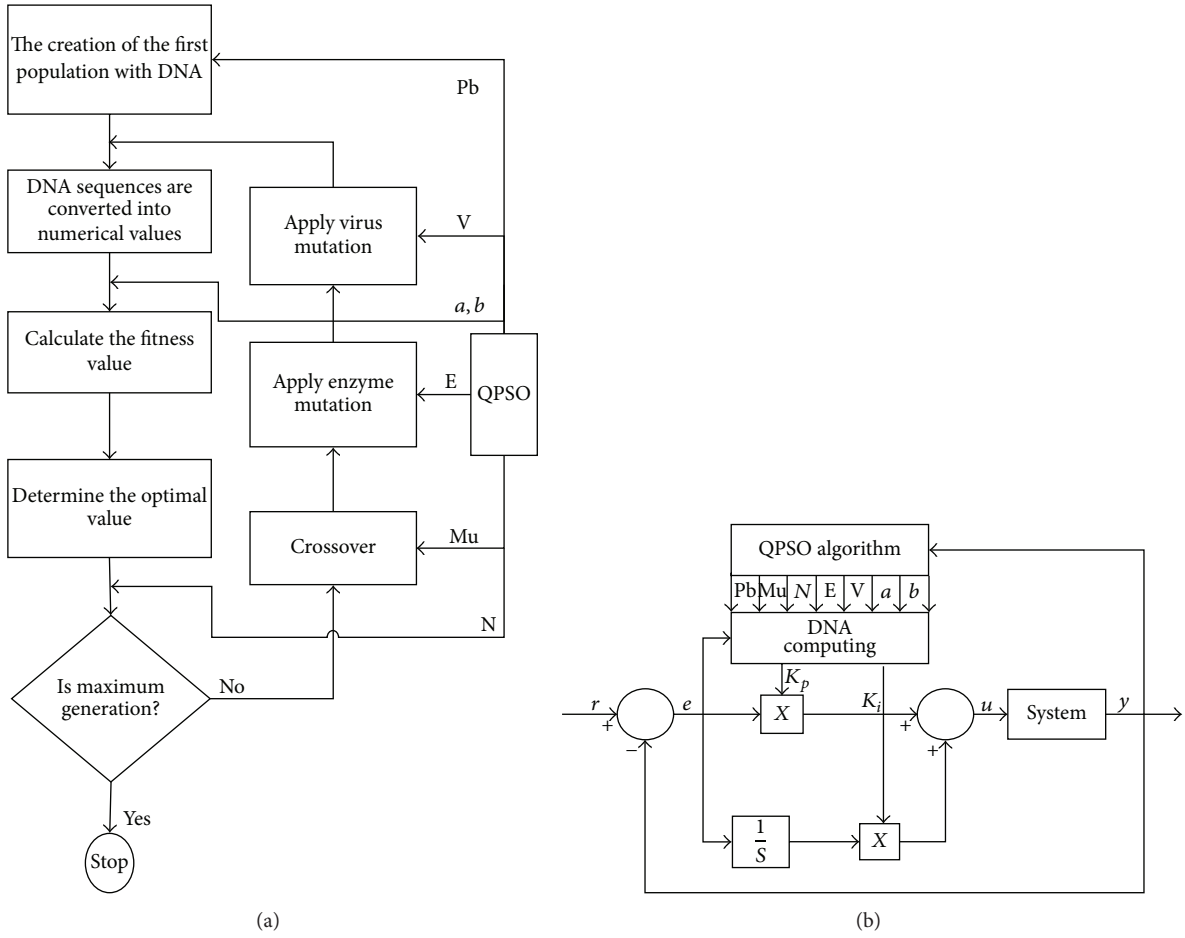


Figure 2: Block diagram of the proposed approach.

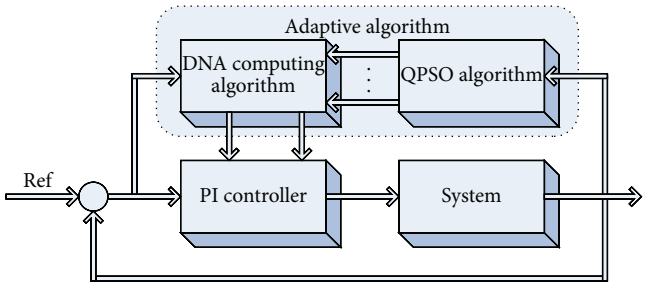


Figure 3: Schematic diagram of the model.

in the DNA computing algorithm. When the maximum number of operations is increased, the completing time of the cycles is extended and the number of the performed operations increases. In Particular the number of elements that composes the population and the number of the variables used in the problem are big; the completion time of the algorithm increases rapidly and this causes loss of time and economic losses. Keeping the maximum number of operations in the lowest level decreases the efficiency of the algorithm. The best values should be determined considering all those situations. In this study, in order to find maximum number of operations, (5) has been used. The number 5 given in (5) is the base value and it is used for the purpose of the probability of the size of the maximum number of operations to be zero and it is determined with the method of trial and error:

$$\text{Maximum\_generation\_number} = \left(f(n) + f(1)\right) * \text{Pb} + 5. \quad (5)$$

Crossover Rate (Mu). With Crossover, new elements are created by using the existing population elements. While determining the Crossover rate, the number and selection of the elements to be crossed are taken into account. When the Crossover rate is kept high, the number of elements to be selected increases and the change of the set that shall be created newly is provided. However it is probable that the new created elements may be values close to the parent elements. Furthermore the number of elements to be crossed being high prolongs the duration of completion of the algorithm. In this study (6) has been used for the purpose of finding the Crossover rate. When (6) is used, crossover rate in all

Table 2: Comparison results of DNA and adaptive DNA computing algorithms.

Algorithm	K_p	K_i	Settling time	Overshoot %
DNA-PI	30	0.1250	0.08	14
Adaptive DNA-PI	17	0.4375	0.08	~0

Table 3: Values of DNA and adaptive DNA computing parameters.

Parameters	DNA computing	Adaptive DNA computing
Population size	80	60
Maximum generations	20	40
Crossover rate	%100	%32
Enzyme and virus rate	%30	%12
a and b	15, 12	11.8203, 1.9700

iterations is applied at a proportion of 80%:

$$\text{Crossover\_rate} = \text{Pop\_size} * 0.8. \quad (6)$$

Enzyme (E) and Virus (V) Proportion. The most significant two parameters used in this study are enzyme and virus mutations. While enzyme mutation provides the deletion of a certain proportion of elements from the population, virus mutation provides the addition of a certain proportion of elements to the population. These two parameters are only used with DNA computing unlike others. The implementing adaptive DNA computing enzyme and virus proportions should be selected well because when the enzyme mutation is not applied in the correct time, it may lead to the loss of elements with good conformity values. One should be careful that the algorithm does not create unnecessary solution area while implementing virus mutation. Equation (7) has been used in the study performed in order to determine enzyme and virus proportions. With the change of the population size, the variation of population at a proportion of 30% is provided in ever iteration. In the selection of the equations found through the trial and error method were taken into consideration and the values acquired by running the system many times were examined in detail and the equations have been given their final forms:

$$\text{e\_v} = \text{Pop\_size} * 0.3. \quad (7)$$

The algorithmic steps of DNA computing algorithm are explained in detail as follows.

Step 1. The first population is created with DNA serials randomly.

Step 2. The serials created were firstly converted into system of four and then into decimal system as it is explained in Table 3 and converted into numerical data. The operations of coding and converting are given in Table 3 in detail. The population elements converted into numerical data are sent to the simulation environment and the error values created in the system are determined.

Step 3. The error values from the simulation are placed in the conformity function and conformity values are obtained. The population elements that minimize sum conformity values are used as the new K_p and K_i values. Equation (8) has been selected for the determination of the conformity value. This conformity function may change in accordance with the preference of the applier as it is explained above.

Step 4. The adaptive parameter values to be used for the adaptive DNA computing algorithm and cycle are found using QPSO and sent to the system. Equation (9) is used for the QPSO conformity function. In the selection of the conformity function the results found through the method of trial and error were taken into consideration and the values acquired by running the system many times were examined in detail and the equations have been given their final forms.

Step 5. The elements of the population created firstly are subjected to Crossover process and new population is created. Using Steps 2 and 3 new K_p and K_i are determined.

Step 6. The population applied enzyme and virus mutation and new population is created. Applying Steps 2, 3, and 4 new K_p and K_i values are determined.

Step 7. The best conformity values found in the above steps are compared. K_p and K_i values in the step where the conformity value is minimum are sent to the system.

Step 8. Those steps are applied till the maximum number of operations is achieved. At the end of maximum operations the most suitable K_p and K_i values are determined:

$$f\left(K_p, K_i\right) = \text{sum}\left(\text{abs}\left(e\right)\right), \quad (8)$$

$$f\_\text{qpso} = a * \text{sum}\left(e^2\right) + b * \max\left(\text{yout}\right), \quad (9)$$

$$a = \text{average}\left(f\_\text{qpso}\right) * c1, \quad (10)$$

$$b = \text{average}\left(f\_\text{qpso}\right) * c2. \quad (11)$$

5. Experimental Results

A transfer function given with equation (12) that is modeling position control of a DC motor was used for the performance measurement of the proposed method. DNA computing algorithm was applied for setting the PI parameters and Matlab m-file was written. For finding the simulation results, the model given in Figure 3 was created in the Simulink environment and the results were obtained. The parameter values used for DNA computing are given in Table 3:

$$G\left(s\right) = \frac{2.2}{8.96e-6s^3 + 7.27e-3s^2 + 0.945s}. \quad (12)$$

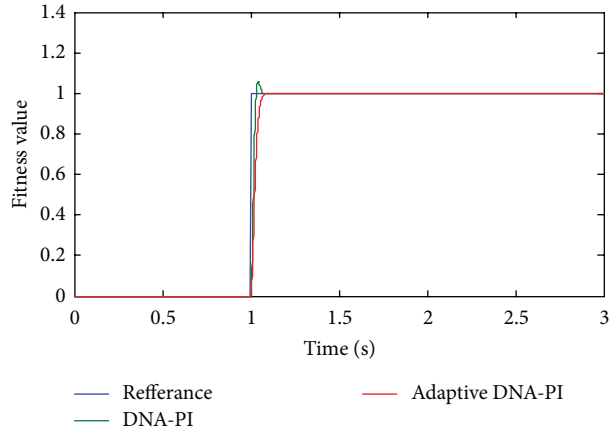


Figure 4: Adaptive algorithm results.

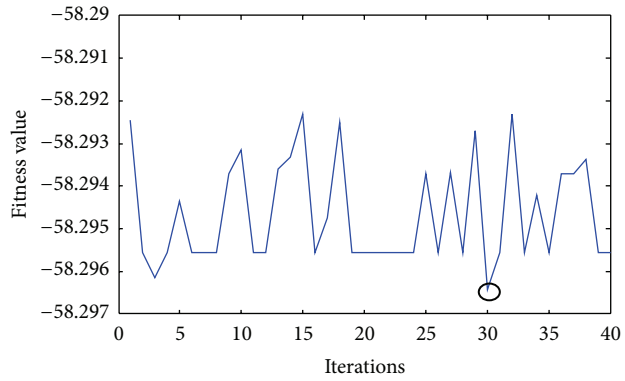


Figure 5: The fitness values of adaptive DNA computing algorithm.

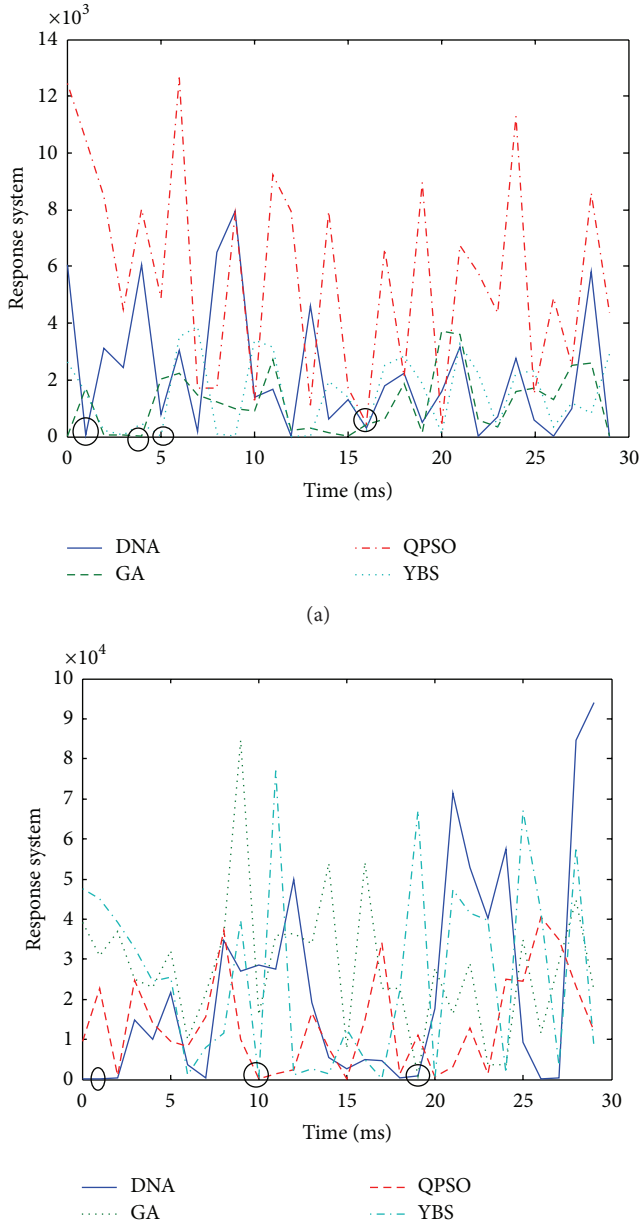


Figure 6: Comparison results.

In the application performed, QPSO-based DNA computing algorithm was used for the optimization of the PI parameters. Although various conformity functions were used in the optimization of the PI parameters, in this study the sum of absolute value of error was selected as the conformity function. With the use of this function which is sensitive even to the errors with minimal values it was targeted that upper excess, increase, and setting times would give better results. In the Matlab/Simulink study performed the size of the population used in the application was taken as 80, maximum number of operations as 20, and reference value as 1. For the detection of K_p and K_i values (8) has been used as the conformity function. While performing coding with DNA computing algorithm K_p and K_i values were coded with DNA basis using data of 6 bytes. Firstly 80 individuals in the population were used and each individual was represented with data of 12 bytes. The first 6 bytes of those data of 12 bytes were used for K_p and the other 6 bytes were used for K_i. In every iteration enzyme and virus mutation as much as 30% of the population was applied and the change of the individual was provided as much as population size $*$ 0.3 and the population was renewed. In the application performed, the population elements created in the algorithms were sent to the system and determination of the PI elements has been provided. The results produced by the system as a result of running the program many times are given in Table 3 and Figure 4.

As it is given in Table 2, using the adaptive DNA computing algorithm K_p is found to be 17, K_i 0.4375, placement time 0.08 seconds, and maximum excess approximately 0%. In Figure 4, the comparison of the results found with adaptive DNA computing to DNA computing is given. The maximum excess value found with DNA computing algorithm made adaptive is approximately 0% while the maximum excess value found with DNA computing is computed as approximately 14%. Those results indicate that adaptive DNA computing gets better results.

In Table 3 DNA computing parameters made adaptive with the QPSO algorithm and DNA computing parameters are given. With the activation of the algorithm with QPSO parameter values have changed. With the QPSO algorithm population size was revised as 60, maximum number of operations as 40, Crossover rate as 32%, and enzyme and virus proportion as 12%. New values were applied by being sent to DNA computing algorithm. The values of a and b used for the conformity function were 15 and 12 when the algorithm started and with the QPSO algorithm they were acquired as 11.8203 and 1.9700 and used for the system. The performance of the proposed approach and comparison results are shown in Figures 5 and 6, respectively. As shown in these figures, the proposed approach finds faster than other algorithms.

6. Conclusions

DNA computing that the key advantage is parallelism has the natural capabilities of DNA and large information storage abilities. But this computation model has some problems for numerical implementation in real world applications. In particular, the parameters of the DNA computing algorithm are determined by trial and error in the literature. This determination method is usually very difficult, inefficient, and inflexible. A new QPSO-based adaptive DNA computing algorithm is proposed and implemented for optimization problems in this study. The seven parameters of DNA computing algorithm has been simultaneously and online tuned with QPSO for adaptive operation in the proposed approach. Experimental results obtained with computation of parameters of PI controller for system identification application consistently show that proposed approach has effective optimization, high accuracy and high convergence speed according to traditional DNA computing algorithm. In addition, general search capability of the method is not dependent on local optimum points and applicability of the proposed approach is easier, simpler, and faster than traditional DNA computing algorithm.

References

[1] L. M. Adleman, "Molecular computation of solutions to combinatorial problems," *Science*, vol. 266, no. 5187, pp. 1021–1024, 1994.

[2] R. J. Lipton, "Using DNA to solve NP-complete problems," *Science*, vol. 268, no. 5210, pp. 542–545, 1995.

[3] C.-L. Lin, H.-Y. Jan, and T.-S. Hwang, "Structure variable PID control design based on DNA coding method," in *Proceedings of the IEEE International Symposium on Industrial Electronics (IEEE-ISlE '04)*, pp. 423–428, May 2004.

[4] Y. Ding and L. Ren, "DNA genetic algorithm for design of the generalized membership-type Takagi-Sugeno fuzzy control system," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, vol. 5, pp. 3862–3867, October 2000.

[5] J. J. Kim and J. J. Lee, "PID controller design using double helix structured DNA algorithms with a recovery function," *Artificial Life and Robotics*, vol. 12, no. 1-2, pp. 241–244, 2008.

[6] Y. Wang, G. Cui, and Z. Wang, "Research on DNA computer with coprocessor organizational model," in *Proceedings of the 8th International Conference on High-Performance Computing in Asia-Pacific Region (HPC Asia '05)*, vol. 6, pp. 544–549, December 2005.

[7] X. Zhang, Y. Niu, and Y. Wang, "DNA computing in microreactors: a solution to the minimum vertex cover problem," in *Proceedings of the 6th International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA '11)*, pp. 236–240, September 2011.

[8] R. Sridhar and S. Balasubramaniam, "GIS integrated DNA computing for solving travelling salesman problem," in *Proceedings of the IEEE Symposium on Computers & Informatics (ISCI '11)*, pp. 402–406, mys, March 2011.

[9] U. Çiğdem and M. Karaköse, "Using of DNA computing for tuning of parameters of PI and fuzzy controllers," in *Proceedings of the Automatic Control National Symposium*, pp. 665–670, Izmir, Turkey, 2011.

[10] F. Li, Z. Li, and J. Xu, "DNA computing model based on photoelectric detection system with magnetic beads," in *Proceedings of the 6th International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA '11)*, pp. 170–175, September 2011.

[11] Y. Huang and L. He, "DNA computing research progress and application," in *Proceedings of the 6th International Conference on Computer Science and Education (ICCSE '11)*, pp. 232–235, August 2011.

[12] J. Xu, X. Qiang, Y. Yang et al., "An unenumerative DNA computing model for vertex coloring problem," *IEEE Transactions on Nanobioscience*, vol. 10, no. 2, pp. 94–98, 2011.

[13] S. Mitra, R. Das, and Y. Hayashi, "Genetic networks and soft computing," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 8, no. 1, pp. 94–107, 2011.

[14] H. Jiao, Y. Zhong, L. Zhang, and P. Li, "Unsupervised remote sensing image classification using an artificial DNA computing," in *Proceedings of the 2011 7th International Conference on Natural Computation (ICNC '11)*, pp. 1341–1345, July 2011.

[15] Z. Yin, C. Hua, and B. Song, "Plasmid DNA computing model of 0-1 programming problem," in *Proceedings of the IEEE 5th International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA '10)*, pp. 148–151, September 2010.

[16] J. Xu, X. Qiang, K. Zhang et al., "A parallel type of DNA computing model for graph vertex coloring problem," in *Proceedings of the IEEE 5th International Conference on Bio-Inspired Computing: Theories and Applications (BIC-TA '10)*, pp. 231–235, September 2010.

[17] Y. Huang, Z. Yin, and Y. Tian, "Design of PID controller based on DNA computing," in *Proceedings of the International Conference on Artificial Intelligence and Computational Intelligence (AICI '10)*, pp. 195–198, October 2010.

[18] Z. Yin, B. Song, C. Zhen, and C. Hua, "Molecular beacon-based DNA computing model for maximum independent set problem," in *Proceedings of the International Conference on Intelligent Computation Technology and Automation (ICICTA '10)*, pp. 732–735, May 2010.

[19] C.-L. Lin, H.-Y. Jan, and T.-S. Hwang, "Structure variable PID control design based on DNA coding method," in *Proceedings of the IEEE International Symposium on Industrial Electronics*, pp. 423–428, May 2004.

[20] C.-L. Lin, H.-Y. Jan, and T.-H. Huang, "Self-organizing PID control design based on DNA computing method," in *Proceedings of the IEEE International Conference on Control Applications*, pp. 568–573, September 2004.

[21] C. V. Henkel, *Experimental DNA computing [Ph.D. thesis]*, Leiden University, 2005.

[22] Z. F. Qiu, *Advance the DNA computing [Ph.D. thesis]*, Texas A&M University, College Station, Tex, USA, 2003.

[23] M. O. Rahman, A. Hussain, E. Scavino, M. A. Hannan, and H. Basri, "Object identification using DNA computing algorithm," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1–7, 2012.

[24] J. M. Chaves-González and M. A. Vega-Rodríguez, "DNA sequence design for reliable DNA computing by using a multiobjective approach," in *Proceedings of the 13th IEEE International Symposium on Computational Intelligence and Informatics*, pp. 73–79, 2012.

[25] H. Zhang and X. Liu, "A general object oriented description for DNA computing technique," in *Proceedings of the International Conference on Information Technology and Computer Science (ITCS '09)*, pp. 3–6, July 2009.

[26] H. Jiao, Y. Zhong, and L. Zhang, "Artificial DNA computing-based spectral encoding and matching algorithm for hyperspectral remote sensing data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 10, pp. 4085–4105, 2012.

[27] W. Liu, S. Sun, and Y. Guo, "A DNA computing model of perceptron," in *Proceedings of the Pacific-Asia Conference on Circuits, Communications and System (PACCS '09)*, pp. 710–712, May 2009.

[28] Q. Zhang, X. Xue, and X. Wei, "A novel image encryption algorithm based on DNA subsequence operation," *The Scientific World Journal*, vol. 2012, Article ID 286741, 10 pages, 2012.

[29] J. Xu, X. Qiang, Y. Yang et al., "An unenumerative DNA computing model for vertex coloring problem," *IEEE Transactions on Nanobioscience*, vol. 10, no. 2, pp. 94–98, 2011.

[30] M. Xi, J. Sun, and W. Xu, "Parameter optimization of PID controller based on Quantum-Behaved Particle Swarm Optimization Algorithm," *Complex Systems and Applications-Modelling*, vol. 14, supplement 2, pp. 603–607, 2007.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 370172, 11 pages
http://dx.doi.org/10.1155/2013/370172

Research Article

An Improved Marriage in Honey Bees Optimization Algorithm for Single Objective Unconstrained Optimization

Yuksel Celik[1] and Erkan Ulker[2]

[1] *Department of Computer Programming, Karamanoglu Mehmetbey University, Karaman, Turkey*
[2] *Computer Engineering Department, Selcuk University, Konya, Turkey*

Correspondence should be addressed to Yuksel Celik; celik_yuksel@hotmail.com

Received 5 May 2013; Accepted 11 June 2013

Academic Editors: P. Agarwal, V. Bhatnagar, and Y. Zhang

Marriage in honey bees optimization (MBO) is a metaheuristic optimization algorithm developed by inspiration of the mating and fertilization process of honey bees and is a kind of swarm intelligence optimizations. In this study we propose improved marriage in honey bees optimization (IMBO) by adding Levy flight algorithm for queen mating flight and neighboring for worker drone improving. The IMBO algorithm's performance and its success are tested on the well-known six unconstrained test functions and compared with other metaheuristic optimization algorithms.

1. Introduction

Optimization means to find the best among the possible designs of a system. In other words, for the purpose of minimizing or maximizing a real function, selecting real or integer values from an identified range, placing these into the function and systematically examining or solving the problem is referred to as optimization. For the solution of optimization problems, mathematical and heuristic optimization techniques are used. In problems with wide and large solution space, heuristic algorithms heuristically produce the closest results to the solution, without scanning the whole solution space and within very short durations. Metaheuristic algorithms are quite effective in solving global optimization problems [1]. The main metaheuristic algorithms are genetic algorithm (GA) [2], simulated annealing (SA) [3], particle swarm optimization (PSO) [4], ant colony optimization (ACO) [5], differential evolution (DE) [6], marriage in honey bees optimization (MBO) [7, 8], artificial bee colony algorithm (ABC) [9] and evolutionary algorithms (EAs) [9, 10].

Performance of algorithms carrying out nature-inspired or evolutionary calculations can be monitored with their application on the test functions of such algorithms. Karaboga and Basturk implemented the artificial bee colony (ABC) algorithm, which they proposed from the inspiration of the food searching activities of honey bees, on unconstrained test functions [11]. Digalakis and Margaritis developed two algorithms titled as the generational replacement model (GRM) and the steady state replacement model by making modifications on the genetic algorithm and monitored their performances on unconstrained test functions [12]. By combining the GA and SA algorithms, Hassan et al. proposed the geno-simulated annealing (GSA) algorithm and implemented it on the most commonly used unconstrained test functions [13]. In order to obtain a better performance in the multidimensional search space, Chatterjee et al. suggested the nonlinear variation of the known PSO, the non-PSO algorithm, and measured its performance on several unconstrained test functions [14]. By integrating the opposition-based learning (OBL) approach for population initialization and generation jumping in the DE algorithm, Rahnamayan et al. proposed the opposition-based DE (ODE) algorithm and compared the results they obtained from implementing the algorithm on the known unconstrained test functions with DE [15]. It is difficult to exhibit a good performance on all test functions. Rather than expecting the developed algorithm to provide accurate results on all types of problems, it is more reasonable to determine the types of problems where the algorithm functions well and decide on the algorithm to be used on a specific problem.

Test functions determine whether the algorithm will be caught to the local minimum and whether it has a wide search function in the search space during the solution.

In 2001, Abbass [8] proposed the MBO algorithm, which is a swarm intelligence based and metaheuristic algorithm predicated on the marriage and fertilization of honey bees. Later on, Abbass and Teo used the annealing approach in the MBO algorithm for determining the gene pool of male bees [16]. Chang made modifications on MBO for solving combinatorial problems and implemented this to the solution. Again, for the solution of infinite horizon-discounted cost stochastic dynamic programming problems, he implemented MBO on the solution by adapting his algorithm he titled as "Honey Bees Policy Iteration" (HBPI) [17]. In 2007, Afshar et al. proposed MBO algorithm as honey bee mating optimization (HBMO) algorithm and implemented it on water resources management applications [18]. Marinakis et al. implemented HBMO algorithm by obtaining HBMOTSP in order to solve Euclidan travelling salesman problem (TSP) [19]. Chiu and Kuo [20] proposed a clustering method which integrates particle swarm optimization with honey bee mating optimization. Simulations for three benchmark test functions (MSE, intra-cluster distance, and intercluster distance) are performed.

In the original MBO algorithm, annealing algorithm is used during the queen bee's mating flight, mating with drones, generation of new genotype, and adding these into the spermatheca. In the present study, we used Levy flight [1] instead of the annealing algorithm. Also, during the improvement of the genotype of worker bees, we applied single neighborhood and single inheritance from the queen. We tested the IMBO algorithm we developed on the most commonly known six unconstrained numeric test functions, and we compared the results with the PSO and DE [21] algorithms from the literature.

This paper is organized as follows: in Section 2, the MBO algorithm and unconstrained test problems are described in detail. Section 3 presents the proposed unconstrained test problems solution procedure using IMBO. Section 4 compares the empirical studies and unconstrained test results of IMBO, MBO, and other optimization algorithms. Section 5 is the conclusion of the paper.

2. Material and Method

2.1. The Marriage in Honey Bee Optimization (MBO) Algorithm

2.1.1. Honey Bee Colony. Bees take the first place among the insects that can be characterized as swarm and that possess swarm intelligence. A typical bee colony is composed of 3 types of bees. These are the queen, drone (male bee), and workers (female worker). The queen's life is a couple of years old, and she is the mother of the colony. She is the only bee capable of laying eggs.

Drones are produced from unfertilized eggs and are the fathers of the colony. Their numbers are around a couple of hundreds. Worker bees are produced from fertilized eggs, and all procedures such as feeding the colony and the queen, maintaining broods, building combs, and searching and collecting food are made by these bees. Their numbers are around 10–60 thousand [22].

Mating flight happens only once during the life of the queen bee. Mating starts with the dance of the queen. Drones follow and mate with the queen during the flight. Mating of a drone with the queen depends of the queen's speed and their fitness. Sperms of the drone are stored in the spermatheca of the queen. The gene pool of future generations is created here. The queen lays approximately two thousand fertilized eggs a day (two hundred thousand a year). After her spermatheca is discharged, she lays unfertilized eggs [23].

2.1.2. Honey Bee Optimization Algorithm. Mating flight can be explained as the queen's acceptance of some of the drones she meets in a solution space, mating and the improvement of the broods generated from these. The queen has a certain amount of energy at the start of the flight and turns back to the nest when her energy falls to minimum or when her spermatheca is full. After going back to the nest, broods are generated and these are improved by the worker bees crossover and mutation.

Mating of the drone with the queen bee takes place according to the probability of the following annealing function [8]:

$$\text{prob}\, f\,(Q, D) = e^{-\text{difference/speed}}, \tag{1}$$

where (Q, D) is the probability of the drone to be added to the spermatheca of the Q queen (probability of the drone and queen to mate) and $\Delta(f)$ is the absolute difference between D's fitness and Q's fitness. $f(Q)$ and $S(t)$ are the speed of the queen at t time. This part is as the annealing function. In cases where at first the queen's speed is high or the fitness of the drone is as good as the queen's fitness, mating probability is high. Formulations of the time-dependent speed $S(t)$ and energy $E(t)$ of the queen in each pass within the search space are as follows:

$$\begin{aligned} S\,(t+1) &= \alpha \times S\,(t)\,, \\ E\,(t+1) &= E\,(t) - \gamma. \end{aligned} \tag{2}$$

Here, α is the factor of $\in [0, 1]$ and γ is the amount of energy reduction in each pass. On the basis of (1) and (2), the original MBO algorithm was proposed by Abbas [8] as shown in Algorithm 1.

2.2. Unconstrained Numerical Benchmark Functions. Performance of evolutionary calculating algorithms can be monitored by implementing the algorithm on test functions. A well-defined problem set is useful for measuring the performances of optimization algorithms. By their structures, test functions are divided into two groups as constrained and unconstrained test functions. Unconstrained test functions can be classified as unimodal and multimodal. While unimodal functions have a single optimum within the search space, multimodal functions have more than one optimum. If the function is predicated on a continuous mathematical objective function within the defined search space, then it

Initialize workers
Randomly generate the queens
Apply local search to get a good queen
For a pre-defined maximum number of mating-flights
  For each queen in the queen list
    Initialize energy (E), speed (S), and position
    The queen moves between states
    And probabilistically chooses drones prob$f$ $(Q, D) = e^{-(\text{difference/speed})}$
    If a drone is selected, then
      Add its sperm to the queen's spermatheca
      $S(t+1) = \alpha \times S(t)$
      $E(t+1) = E(t) - \gamma$
    End if
    Update the queen's internal energy and speed
  End for each
  Generate broods by crossover and mutation
  Use workers to improve the broods
  Update workers' fitness
  While the best brood is better than the worst queen
    Replace the least-fittest queen with the best brood
    Remove the best brood from the brood list
  End while
End for

Algorithm 1: Original MBO algorithm [8].

is a continuous benchmark function. However, if the bit strings are not defined and continuous, then the function is described as a discreet benchmark function [24]. Alcayde et al. [25] approach a novel extension of the well-known Pareto archived evolution strategy (PAES) which combines simulated annealing and tabu search. They applied this several mathematical problems show that this hybridization allows an improvement in the quality of the nondominated solutions in comparison with PAES Some of the most commonly known test functions are as follows. We have solved well-known six unconstrained single objective numeric benchmark function. The details of the benchmark functions are given in Table 1.

3. IMBO for Unconstrained Test Functions

In the original MBO mating possibility of the queen bee in the mating flight is calculated through the annealing function. In the proposed study Improved MBO (IMBO) algorithm was obtained by improving the MBO algorithm through the replacement of the annealing algorithm with the Levy flight algorithm in order to enable the queen to make a better search in the search space. Flight behaviors of many animals and insects exhibited the typical characteristics of Levy flight [26]. In addition, there are many studies to which Levy flight was successfully adapted. Pavlyukevich solved a problem of nonconvex stochastic optimization with the help of simulated annealing of Levy flights of a variable stability index [27]. In biological phenomena, Viswanathan et al. used Levy flight in the search of biologic organisms for target organisms [28]. Reynolds conducted a study by integrating Levy flight algorithm with the honey bees' strategies of searching food [29]. Tran et al. proposed Levy flight optimization (LFO) for global optimization problems, implemented it on the test functions, and compared the results they obtained with simulated annealing (SA) [30]. By adapting Levy flight algorithm instead of the gaussian random walk in the group search optimizer (GSO) algorithm developed for Artificial neural network (ANN), Shan applied the algorithm on a set of 5 optimization benchmark functions [31].

In general terms, Levy flight is a random walk. The steps in this random walk are obtained from Levy distribution [1]. Levy flight is implemented in 2 steps. While the first is a random selection of direction, the second is the selection of a step suitable for Levy distribution. While direction has to be selected from a homogenous distribution region, step selection is a harder process. Although there are several methods for step selection, the most effective and simplistic one is the Mantegna algorithm.

Mantegna algorithm is calculated as shown in the following equation:

$$s = \frac{u}{|v|^{1/\beta}}. \tag{3}$$

Here, the v is obtained by taking the magnitude of the genotype as basis.

u on the other hand is calculated as shown in the following equation;

$$\sigma_u = \left\{ \frac{\Gamma(1+\beta)\sin(\pi\beta/2)}{\Gamma\left[(1+\beta)/2\right]\beta 2^{(\beta-1)/2}} \right\}^{1/\beta}, \quad \sigma_u = 1. \tag{4}$$

Table 1: Unconstrained test functions.

No.	Function name	Formula	Range	Optimum $f(x)$
1	Sphere	$f(x) = \sum_{i=1}^{n} x^2$	$[5.12, 5.12]^n$	0
2	Rosenbrock	$f(\overrightarrow{x}) = \sum_{i=1}^{n-1} 100\left(x_{i+1} - x^2\right)^2 + \left(x_i - 1\right)^2$	$[-2.04, 2.048]^n$	0
3	Rastrigin	$f(\overrightarrow{x}) = 10 \cdot n + \sum_{i=1}^{n} \left(x_i^2 - 10 \cdot \cos\left(2\pi x_i\right)\right)$	$[-5.12, 5.12]^n$	0
4	Griewank	$f(\overrightarrow{x}) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^n$	0
5	Schwefel	$f(\overrightarrow{x}) = 418.9829 \cdot n + \sum_{i=1}^{n} x_i \cdot \sin\left(\sqrt{\lvert x_i \rvert}\right)$	$[-500, 500]^n$	$-418.9829 * n$
6	Ackley	$f(\overrightarrow{x}) = 20 + e - 20 \cdot e^{-0.2 \cdot \sqrt{(1/n)\sum_{i=1}^{n} x_i^2}} - e^{(1/n)\sum_{i=1}^{n} \cos(2\pi x_i)}$	$[-32.768, 32.768]^n$	0

While in this equation β is $0 \leq \beta \leq 2$, is the Γ is the Gamma function, and calculated as follows:

$$\Gamma(z) = \int_0^{\infty} t^{z-1} e^{-t} dt. \tag{5}$$

In consequence, the direction of the next step is determined with the u and v parameters, and step length is found by placing u and v into their place in the Mantegna algorithm (3). Based on S, new genotype is generated as much as random genotype size, and the generated genotype is added to the previous step. Consider

$$s = \propto_0 \left(x_j^{(t)} - x_i^{(t)}\right) \oplus \text{Lévy}(\beta) \sim 0.01 \frac{u}{|v|^{1/\beta}} \left(x_j^{(t)} - x_i^{(t)}\right). \tag{6}$$

Creation of the new genotype of this step is completed by subjecting the new solution set obtained, that is, the genotype to the maximum and minimum controls defined for the test problem and adjusting deviations if any. Accordingly, through these implemented equations, the queen bee moves from the previous position to the next position, or towards the direction obtained from Levy distribution and by the step length obtained from Mantegna algorithm as follows:

$$L_{ij} = x_{ij} + s * \text{rand}\left(\text{size}\left(x_{ij}\right)\right). \tag{7}$$

In the crossover operator, the genotype of the queen bee and all genotypes in the current population are crossed over. Crossover was carried out by randomly calculating the number of elements subjected to crossover within Hamming distance on the genotypes to be crossed over.

In the improvement of the genotype (broods) by the worker bees single neighborhood and single inheritance from the queen was used. Consider

$$W_{ij} = x_{ij} + \left(x_{ij} - x_{kj}\right) \emptyset_{ij}, \tag{8}$$

where $\emptyset_{ij}$ is a random (0,1) value, x_i is the current brood genotype, x_k is the queen genotype, j is a random value number of genotype. In this way, and it was observed that the developed IMBO algorithm exhibits better performance than the other metaheuristic optimization algorithms.

The MBO algorithm we modified is shown in Algorithm 2.

4. Experimental Results

In this study, we used Sphere, Rosenbrock, Rastrigin, Griewank, Schwefel, and Ackley unconstrained test problems; a program in the MatLab 2009 programming language was developed for the tests of MBO and IMBO. Genotype sizes of 10, 50, 100, 300, 500, 800, and 1000 were taken for each test. Population size (Spermatheca Size) was accepted as $M = 100$. At the end of each generation, mean values and standard deviations were calculated for test functions. Each genotype was developed for 10,000 generations. Each improvement was run for 30 times. Global minimum variance graphs of each test function for IMBO are presented in Figure 1.

Examining the six test functions presented in Figure 1 shows that, at first, global optimum values were far from the solution value in direct proportion to genotype size. Accordingly, for large genotypes, or in other words in cases where the number of entry parameters is high, convergence of the proposed algorithm to the optimum solution takes a longer time. The test results for MBO and IMBO are given in Tables 2 and 3.

When Tables 2 and 3 were examined, it is seen that genotype size increases in all functions of MBO IMBO algorithms and the global minimum values get away from the optimal minimum values. In Table 2, the MBO algorithm Sphere function, 10, 50, 100 the size of genotype reached an optimum value while the other genotype sizes converge to the optimal solution. It was observed that Rosenbrock function minimum is reached optimal sizes in all genotypes. It was observed that rastrigin function 10, 50, 100, Griewank function 50, 100 genotype sizes in the optimal solution was reached. It was observed that except Schwefel function, other function and genotype sizes, the optimal solution was reached close to values.

When Table 3 was examined, it was seen that, while the size of genotype increased, IMBO algorithm Sphere, Rastrigin, Griewank, Schwefel, and Ackley function, were getting away from the optimal minimum. It was seen that Sphere function, 10, 50, 100 the size of genotype reached an optimum value while the other genotype sizes converges to the optimal solution. It was observed that, except for the size of 10 genotypes Rosenbrock function, but all other

Initialize
Randomly generate the queens
Apply local search to get a good queen
For a pre-defined maximum number of mating-flights (Maximum Iteration)
  While spermatheca not full
    Add its sperm to the queen's spermatheca by **Levy Flight** (7)
  End while
  Generate broods by crossover and mutation
  Use workers to improve the broods by **single inheritance single neighborhood** (8)
  Update workers' fitness
  While the best brood is better than the worst queen
    Replace the least-fittest queen with the best brood
    Remove the best brood from the brood list
  End while
End for

Algorithm 2: Proposed IMBO algorithm.

Table 2: Test results of the MBO algorithm for the genotype sizes of 10, 50, 100, 300, 500, 800, and 1000; number of runs = 30; SD: standard deviation, AV: global minimum average; generation = 10000.

Genotype size		10	50	100	300	500	800	1000
Sphere	AV	0	0	0	$1.97E-26$	$1.21E-13$	$2.06E-06$	$8.26E-04$
	SD	$9.81E-01$	$3.36E+01$	$1.41E+02$	$1.19E+03$	$2.66E+03$	$6.82E+03$	$1.84E+04$
Rosenbrock	AV	0	0	0	0	0	0	0
	SD	$4.73E-03$	$8.67E-03$	$8.29E-03$	$1.55E-02$	$7.28E-03$	$8.09E-03$	0
Rastrigin	AV	0	0	0	$6.0633E-13$	$7.88048E-12$	$3.33194E-05$	0.002606
	SD	0	0	0	$3.95279E-13$	$1.80594E-12$	$3.09109E-05$	0.002843
Griewank	AV	0.010377126	0	0	$3.40468E-16$	$7.26498E-08$	$1.1388E-06$	0.000141
	SD	0.026549182	0.32563139	1.09610828	8.02049039	70.78602045	36.67034057	108.177558
Schwefel	AV	$-9.99E+223$	$-2.96E+211$	$-1.00E+183$	$-2.37E+306$	$-1.02E+305$	$-1.10E+305$	$-1.10E+305$
	SD	$3.81E+22$	$1.94E+16$	$5.13E+26$	$4.31E+35$	$7.31E+35$	$3.39E+32$	$7.89E+35$
Ackley	AV	$4.08562E-15$	$8.23045E-15$	$7.3922E-09$	$8.49609E-14$	$9.13234E-10$	0.000227233	0.000452387
	SD	0.006649746	0.01783530	0.61180347	0.05005151	0.507695929	1.288113922	0.287375496

Table 3: Test results of the IMBO algorithm for the genotype sizes of 10, 50, 100, 300, 500, 800, and 1000; number of runs = 30; SD: standard deviation, AV: global minimum average; generation = 10000.

Genotype size		10	50	100	300	500	800	1000
Sphere	AV	0	0	0	$1.38E-28$	$1.47E-13$	$2.78E-07$	0.0001224
	SD	0	0	0	$8.97E-29$	$1.90E-15$	$8.50E-08$	$3.09E-05$
Rosenbrock	AV	$2.25E-07$	0	0	0	0	0	0
	SD	$2.24E-06$	0	0.1425029	0	0	0.2440021	0
Rastrigin	AV	0	0	$4.54E-15$	$5.68E-13$	$3.60E-12$	$6.13E-07$	0.0002386
	SD	0	0	$3.18E-14$	$4.78E-13$	$1.25E-12$	$3.14E-07$	$9.13E-05$
Griewank	AV	0.0002219	0	0	$3.48E-16$	$9.15E-13$	$1.78E-07$	$3.26E-05$
	SD	0.0012617	0	0	$3.85E-17$	$1.88E-14$	$5.43E-08$	$6.73E-06$
Schwefel	AV	$-1.21E+23$	$-1.08E+25$	$-6.17E+29$	$-1.21E+40$	$-1.76E+36$	$-2.96E+39$	$-1.64E+39$
	SD	$7.28E+44$	$1.06E+26$	$3.52E+30$	$6.52E+40$	$5.78E+36$	$2.14E+40$	$1.49E+40$
Ackley	AV	$3.73E-15$	$8.13E-15$	$1.57E-14$	$7.49E-14$	$1.76E-05$	$3.30E-05$	0.0006221
	SD	$1.42E-15$	$9.94E-16$	$2.30E-15$	$5.11E-15$	$3.09E-05$	$9.57E-06$	0.0001513

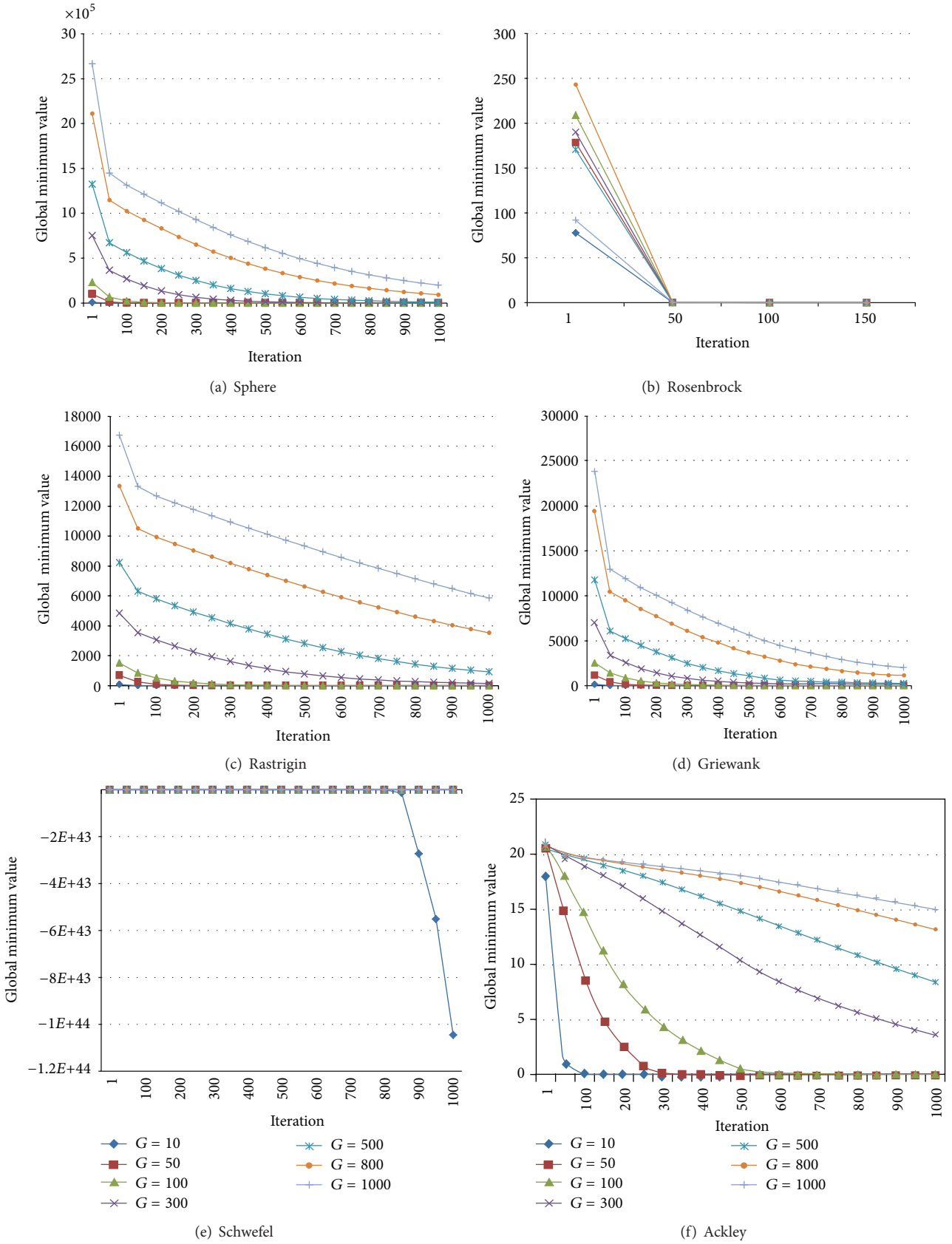


FIGURE 1: Mean global minimum convergence graphs of benchmark functions in 1000 generations for genotype sizes (G) of 10, 50, 100, 300, 500, 800 and 1000.

Table 4: Success rates of MBO when compared with those of the IMBO algorithm. + indicates that the algorithm is better while − indicates that it is worse than the other. If both algorithms show similar performance, they are both +.

Genotype size	10		50		100		300		500		800		1000	
Function	MBO	IMBO	MBO	IMBO	MBO	IMBO	MBO	IMBO	MBO	IMBO	MBO	IMBO	MBO	IMBO
Sphere	+	+	+	+	+	+	−	+	+	−	−	+	−	+
Rosenbrock	+	−	+	+	+	+	+	+	+	+	+	+	+	+
Rastrigin	+	+	+	+	+	−	−	+	−	+	−	+	−	+
Griewank	−	+	+	+	+	+	+	−	−	+	−	+	−	+
Schwefel	−	+	−	+	−	+	−	+	−	+	−	+	−	+
Ackley	−	+	−	+	−	+	−	+	+	−	−	+	+	−
Total	3	5	4	6	4	5	2	5	3	4	1	6	2	5

Table 5: CPU time results of the MBO algorithm for the genotype sizes of 10, 50, 100, 300, 500, 800, and 1000; number of runs = 1; iteration = 10000.

Genotype size	Sphere	Rosenbrock	Rastrigin	Griewank	Schwefel	Ackley
10	02:45:095	02:36:827	03:12.552	05:00.894	03:25.141	03:20.882
50	05:39:832	05:35.729	06:15.572	08:22.590	07:30.842	06:20.845
100	15:47:269	17:13.912	17:36.329	18:34.517	17:29.746	15:55:318
300	1:09:07:083	1:07:07.602	1:12:21.617	1:16:02.748	1:17:11.700	1:16:04.994
500	2:03:31:669	2:03:54.118	2:08:22.112	2:03:11.670	2:06:25.689	2:09:27.164
800	3:23:52:834	2:51:14.834	2:53:25.547	3:00:11.898	3:06:41.792	2:52:34.737
1000	4:31:56.752	3:35:18.224	3:40:03.753	3:45:45.894	3:54:20.884	3:38:12.446

Table 6: CPU time results of the IMBO algorithm for the genotype sizes of 10, 50, 100, 300, 500, 800, and 1000; number of runs = 1; iteration = 10000.

Genotype size	Sphere	Rosenbrock	Rastrigin	Griewank	Schwefel	Ackley
10	35.303	31.949	40.716	01:08.110	46.831	42.245
50	43.134	39.749	49.592	01:20.714	01:21.869	48.141
100	54.475	50.466	01:00.980	01:36.439	01:57.109	58.983
300	1:42.213	01:39.778	01:52.445	02:45.033	04:41.472	01:41.151
500	2:32.662	02:11.836	02:31.180	03:26.841	07:30.062	02:26.812
800	3:49.415	03:16.452	03:49.633	05:00.754	11:15.094	03:49.227
1000	4:48.789	4:02:082	04:37.822	06:05.650	14:15.259	04:42.845

genotypes sizes optimal minimum was reached. Rastrigin function 10, 50, Griewank function 50, 100 to the optimal solution was observed to have reached the optimal solution. The other functions and sizes of genotype were observed to have reached the values close to the optimal solution.

Comparative results of the best and mean solutions of the MBO and IMBO algorithms are presented in Table 4.

According to Table 4, it is seen that, when compared with the MBO algorithm and IMBO algorithm according to genotype, IMBO exhibited better performance than MBO in all genotypes sizes. When it was thought that total better or equal cases were represented with "+" mark, MBO algorithm, a total of 19 "+" available, and IMBO algorithm, a total of 36 pieces of the "+" were available. Accordingly, IMBO's MBO algorithm demonstrates a better performance.

CPU time results of the all genotype sizes for MBO are given in Table 5 and for IMBO are given in Table 6.

In Tables 5 and 6, it was seen that, when CPU time values were analyzed, depending upon the size in the same proportion as genotype problem, solving time took a long time. Again in these tables, when solution CPU time of MBO and IMBO algorithm was analyzed, IMBO algorithm solves problems with less CPU time than MBO algorithm.

For 10, 50, 100, and 1000 problem sizes of unconstrained numeric six benchmark functions, comparisons were made between test results of IMBO algorithm and the algorithms in literature, including DE, PSO, ABC [32], bee swarm optimization, (BSO) [33], bee and foraging algorithm (BFA) [34], teaching-learning-based optimization (TLBO) [35], bumble bees mating optimization (BBMO) [36] and honey bees mating optimization algorithm (HBMO) [36]. Table 7 presents the comparison between experimental test results obtained for 10-sized genotype (problem) on unconstrained test functions of IMBO algorithm and the results for the same problem size in literature including PSO, DE, ABC, BFA and BSO optimization algorithms; while the comparison of success of each algorithm and IMBO algorithm is given in Table 8. Table 9 shows the comparison between experimental

Table 7: The mean solutions obtained by the PSO, DE, ABC, BFA, BSO, and IMBO algorithms for 6 test functions over 30 independent runs and total success numbers of algorithms. Genotype size: 10; (—): not available value, SD: standard deviation, AV: global minimum average.

Problem		PSO	DE	ABC	IMBO	BFA	BSO
Sphere	ORT	$4.13E-17$	$4.41E-17$	$4.88E-17$	0	0.000031	$8.475E-123$
	SD	$7.71E-18$	$8.09E-18$	$5.21E-18$	0	0.00024	$3.953E-122$
Rosenbrock	ORT	0.425645397	$4.22E-17$	0.013107593	$2.25E-07$	7.2084513	$3.617E-7$
	SD	1.187984439	$1.09E-17$	0.008658828	$2.24E-06$	9.436551	$5.081E-5$
Rastrigin	ORT	7.362692992	0.099495906	$4.76E-17$	0	0.003821	$4.171E-64$
	SD	2.485404145	0.298487717	$4.40E-18$	0	0.006513	$7.834E-64$
Griewank	ORT	0.059270504	0.008127282	$5.10E-19$	0.000221881	3.209850	$3.823E-46$
	SD	0.03371245	0.009476456	$1.93E-19$	0.00126167	4.298031	$6.679E-46$
Schwefel	ORT	−2654.033431	−4166.141206	−4189.828873	$-1.21E+23$	—	—
	SD	246.5263242	47.37533385	$9.09E-13$	$7.289E+44$	—	—
Ackley	ORT	$4.67E-17$	$4.86E-17$	$1.71E-16$	$3.73E-15$	0.000085	$7.105E-19$
	SD	$8.06E-18$	$6.55E-18$	$3.57E-17$	$1.42E-15$	0.000237	$5.482E-18$

Table 8: Comparative results of IMBO with PSO, DE, ABC, BFA, and BSO algorithms over 30 independent runs for genotype size 50. + indicates that the algorithm is better while − indicates that it is worse than the other, (—): not available value. If both algorithms show similar performance, they are both +.

Problem	IMBO-PSO		IMBO-DE		IMBO-ABC		IMBO-BFA		IMBO-BSO	
	IMBO	PSO	IMBO	DE	IMBO	ABC	IMBO	BFA	IMBO	BSO
Sphere	+	−	+	−	+	−	+	−	+	−
Rosenbrock	+	−	−	+	+	−	+	−	+	−
Rastrigin	+	−	+	−	+	−	+	−	+	−
Griewank	+	−	+	−	−	+	+	−	−	+
Schwefel	−	+	−	+	−	+	—	—	—	—
Ackley	−	+	−	+	−	+	+	−	−	+
Total	4	2	3	3	3	3	5	0	3	2

Table 9: The mean solutions obtained by the TLBO, HBMO, BBMO, and IMBO algorithms for 6 test functions over 30 independent runs and total success numbers of algorithms. Genotype size: 50; (—): not available value, SD: standard deviation, AV: global minimum average.

Problem		IMBO	TLBO	HBMO	BBMO
Sphere	AV	0	0.00	0.67	0.00
	SD	0	0.00	—	—
Rosenbrock	AV	0	47.0162	46.07	24.37
	SD	0.142502861	$3.56E-01$	—	—
Rastrigin	AV	$4.54747E-15$	$2.03E-12$	4.03	$1.59E-08$
	SD	$3.18323E-14$	$5.46E-12$	—	—
Griewank	AV	0	0.00	$1.44E-02$	0.00
	SD	0	0.00	—	—
Schwefel	AV	$-6.17561E+29$	−20437.84	—	—
	SD	$3.52272E+30$	$1.48E+02$	—	—
Ackley	AV	$1.57E-14$	$3.55E-15$	—	—
	SD	$2.30E-15$	$8.32E-31$	—	—

test results obtained for 50-sized genotype (problem) on unconstrained test functions of IMBO algorithm and the results for the same problem size in literature including TLBO, HBMO and BBMO optimization algorithms, while the comparison of success of each algorithm and IMBO algorithm is given in Table 10. Table 11 shows the comparison between experimental test results for 100-sized genotype (problem) on unconstrained test functions of IMBO algorithm and the results for the same problem size in literature including PSO, DE, and ABC optimization algorithms, while the comparison of success of each algorithm and IMBO algorithm is given in Table 12. Table 13 shows the comparison between experimental test results obtained for 1000-sized genotype (problem) on unconstrained test functions of

Table 10: Comparative results of IMBO with TLBO, HBMO, and BBMO algorithms over 30 independent runs for genotype size 50. + indicates that the algorithm is better while − indicates that it is worse than the other, (—): not available value. If both algorithms show similar performance, they are both +.

Problem	IMBO-TLBO		IMBO-HBMO		IMBO-BBMO	
	IMBO	TLBO	IMBO	HBMO	IMBO	BBMO
Sphere	+	−	+	−	+	+
Rosenbrock	+	−	+	−	+	−
Rastrigin	+	−	+	−	+	−
Griewank	+	+	+	−	+	+
Schwefel	−	+	—	—	—	—
Ackley	−	+	—	—	—	—
Total	4	3	4	0	4	2

Table 11: The mean solutions obtained by the DE, PSO, ABC, and IMBO algorithms for 6 test functions over 30 independent runs and total success numbers of algorithms. Genotype size: 100; (—): not available value, SD: standard deviation, AV: global minimum average.

Problem		PSO	DE	ABC	IMBO
Sphere	AV	$5.14E-16$	$8.84E-17$	$1.08E-15$	0
	SD	$3.12E-16$	$4.29E-17$	$1.04E-16$	0
Rosenbrock	AV	113.143751	132.3488752	0.054865327	0
	SD	48.99432331	41.72265261	0.045566135	0.142502861
Rastrigin	AV	148.2486456	133.1138439	$1.08E-15$	$4.54747E-15$
	SD	17.76489083	106.6728854	$8.99E-17$	$3.18323E-14$
Griewank	AV	0.048643996	0.000739604	$4.92E-17$	0
	SD	0.063166266	0.002218812	$4.25E-18$	0
Schwefel	AV	−20100.36156	−31182.49983	−41898.28873	$-6.17561E+29$
	SD	1763.156655	2078.47339	$3.30E-10$	$3.52272E+30$
Ackley	AV	0.732022399	$2.14E-16$	$4.21E-15$	$1.57E-14$
	SD	0.755456829	$4.53E-17$	$3.09E-16$	$2.30E-15$

Table 12: Comparative results of IMBO with DE, PSO, and ABC algorithms over 30 independent runs for genotype size 100. + indicates that the algorithm is better while − indicates that it is worse than the other. If both algorithms show similar performance, they are both +.

Problem	IMBO-PSO		IMBO-DE		IMBO-ABC	
	IMBO	PSO	IMBO	DE	IMBO	ABC
Sphere	+	−	+	−	+	
Rosenbrock	+	−	+	−	+	
Rastrigin	+	−	+	−	−	+
Griewank	+	−	+	−	+	−
Schwefel	−	+	−	+	−	+
Ackley	+	−	+	−	−	+
Total	5	1	5	1	3	3

IMBO algorithm and the results for the same problem size in the literature including PSO, DE, and ABC optimization algorithms, while the comparison of success of each algorithm and IMBO algorithm is given in Table 14.

Tables 7, 11, and 13 demonstrate that, as the problem size increases in ABC, DE, and PSO, the solution becomes more distant and difficult to reach. However, the results obtained with IMBO showed that, despite the increasing problem size, optimum value could be obtained or converged very closely. There are big differences among the results obtained for 10, 100, and 1000 genotype sizes in DE and PSO; however, this difference is smaller in IMBO algorithm, which indicates that IMBO performs better even in large problem sizes. In Tables 7 and 8, it is seen that IMBO performs equally to DE and ABC and better than PSO, BFA and BSO. In Tables 9 and 10 showing the comparison of IMBO with LBO, HBMO, and BBMO for genotype (problem) size 50, it is seen that IMBO performs better than all the other algorithms. In Tables 11 and 12 showing the comparison of IMBO with DE, PSO and ABC algorithms on problem size 100, it is seen that IMBO performs equally to ABC and better than DE and PSE. In Tables 13 and 14 showing the comparison of IMBO with DE, PSO and ABC

Table 13: The mean solutions obtained by the DE, PSO, ABC, and IMBO algorithms for 6 test functions over 30 independent runs and total success numbers of algorithms. Genotype size: 1000; (—): not available value, SD: standard deviation, AV: global minimum average.

Problem		PSO	DE	ABC	IMBO
Sphere	AV	9723.034942	329214.6744	0.058275686	0.000122371
	SD	3920.944041	917847.3604	0.021093306	$3.09627E-05$
Rosenbrock	AV	1679629.019	14373397912	2603.968539	0
	SD	648462.4744	361340776.6	599.4022496	0
Rastrigin	AV	2722.799729	1674.782779	735.8480014	0.000238574
	SD	83.14754621	96.86409615	24.75231998	$9.13969E-05$
Griewank	AV	86.03568115	266.1639753	0.10290266	$3.2663E-05$
	SD	29.1502045	335.3504904	0.068217103	$6.73212E-06$
Schwefel	AV	−187704.1438	−252854.5198	−350890.8062	$-1.64729E+39$
	SD	1109795553	17724.02042	2279.801625	$1.49786E+40$
Ackley	AV	8.741445965	17.47129372	3.200412604	0.000622099
	SD	0.784830594	3.815946124	0.133628837	0.000151332

Table 14: Comparative results of IMBO with DE, PSO, and ABC algorithms over 30 independent runs for genotype size 1000. + indicates that the algorithm is better while − indicates that it is worse than the other. If both algorithms show similar performance, they are both +.

Problem	IMBO-PSO		IMBO-DE		IMBO-ABC	
	IMBO	PSO	IMBO	DE	IMBO	ABC
Sphere	+	−	+	−	+	−
Rosenbrock	+	−	+	−	+	−
Rastrigin	+	−	+	−	+	−
Griewank	+	−	+	−	+	−
Schwefel	−	+	−	+	−	+
Ackley	+	−	+	−	+	−
Total	5	1	5	1	5	1

on problem size 1000, IMBO is seen to perform better than all the other algorithms.

5. Conclusion

In the proposed study, we developed a new IMBO by replacing annealing algorithm in the queen bee's mating flight with the Levy flight algorithm and using single inheritance and single neighborhood in the genotype improvement stage. We tested the MBO algorithm we improved on the most commonly known six unconstrained numeric benchmark functions. We compared the results obtained with the results of other metaheuristic optimization algorithms in the literature for the same test functions.

In order to observe the improvement of IMBO, the experimental test results of MBO and IMBO were compared for 10, 50, 100, 300, 500, 800, and 1000 problem sizes. Consequently, IMBO algorithm was concluded to perform better than MBO algorithm. Furthermore, according to CPU time of problem solving process, IMBO algorithm works in shorter CPU times. The test results obtained with IMBO were compared with the results of DE, ABC, PSO, BSO, BFA; TLBO, BBMO and HBMO in the literature.

Accordingly, IMBO is observed to perform equally to or a little better than other algorithms in comparison with small genotype size, while IMBO performs much better than other algorithms with large genotype size. A total of 14 comparisons were made between experimental results of IMBO and other optimization algorithms in literature, and it showed better performances in 11 comparisons, and equal performances in 3 comparisons.

In future studies, different improvements can be made on the MBO algorithm and tests can be made on different test functions. Also, comparisons can be made with other metaheuristic optimization algorithms not used in the present study.

References

[1] X.-S. Yang, "Levy flight," in *Nature-Inspired Metaheuristic Algorithms*, pp. 14–17, Luniver Press, 2nd edition, 2010.

[2] D. Bunnag and M. Sun, "Genetic algorithm for constrained global optimization in continuous variables," *Applied Mathematics and Computation*, vol. 171, no. 1, pp. 604–636, 2005.

[3] H. E. Romeijn and R. L. Smith, "Simulated annealing for constrained global optimization," *Journal of Global Optimization*, vol. 5, no. 2, pp. 101–126, 1994.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.

[5] J. Holland, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, 1975.

[6] 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.

[7] O. B. Haddad, A. Afshar, and M. A. Mariño, "Honey-bees mating optimization (HBMO) algorithm: a new heuristic approach for water resources optimization," *Water Resources Management*, vol. 20, no. 5, pp. 661–680, 2006.

[8] H. A. Abbass, "MBO: Marriage in honey bees optimization a haplometrosis polygynous swarming approach," in *Proceedings of the Congress on Evolutionary Computation (CEC '01)*, pp. 207–214, May 2001.

[9] T. Thomas, *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*, Oxford University Press, New York, NY, USA, 1996.

[10] C. Coello Coello and G. B. Lamont, *Evolutionary Algorithms for Solving Multi-Objective Problems*, Genetic Algorithms and Evolutionary Computation, Kluwer Academic Publishers, Boston, Mass, USA, 2nd edition, 2007.

[11] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 687–697, 2008.

[12] J. G. Digalakis and K. G. Margaritis, "On benchmarking functions for genetic algorithms," *International Journal of Computer Mathematics*, vol. 77, no. 4, pp. 481–506, 2001.

[13] M. M. Hassan, F. Karray, M. S. Kamel, and A. Ahmadi, "An integral approach for Geno-Simulated Annealing," in *Proceedings of the 10th International Conference on Hybrid Intelligent Systems (HIS '10)*, pp. 165–170, August 2010.

[14] A. Chatterjee and P. Siarry, "Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization," *Computers and Operations Research*, vol. 33, no. 3, pp. 859–871, 2006.

[15] R. S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 64–79, 2008.

[16] H. A. Abbass and J. Teo, "A true annealing approach to the marriage in honey-bees optimization algorithm," *International Journal of Computational Intelligence and Aplications*, vol. 3, pp. 199–208, 2003.

[17] H. S. Chang, "Converging marriage in honey-bees optimization and application to stochastic dynamic programming," *Journal of Global Optimization*, vol. 35, no. 3, pp. 423–441, 2006.

[18] A. Afshar, O. Bozorg Haddad, M. A. Mariño, and B. J. Adams, "Honey-bee mating optimization (HBMO) algorithm for optimal reservoir operation," *Journal of the Franklin Institute*, vol. 344, no. 5, pp. 452–462, 2007.

[19] Y. Marinakis, M. Marinaki, and G. Dounias, "Honey bees mating optimization algorithm for the Euclidean traveling salesman problem," *Information Sciences*, vol. 181, no. 20, pp. 4684–4698, 2011.

[20] C.-Y. Chiu and T. Kuo, "Applying honey-bee mating optimization and particle swarm optimization for clustering problems," *Journal of the Chinese Institute of Industrial Engineers*, vol. 26, no. 5, pp. 426–431, 2009.

[21] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[22] R. F. A. Moritzl and C. Brandesl, "Behavior genetics of honeybees (Apis mellifera L.)," in *Neurobiology and Behavior of Honeybees*, pp. 21–35, Springer, Berlin, Germany, 1987.

[23] R. F. A. Moritz and E. E. Southwick, *Bees as Super-Organisms*, Springer, Berlin, Germany, 1992.

[24] B. Bilgin, E. Özcan, and E. E. Korkmaz, "An experimental study on hyper-heuristics and exam scheduling," in *Practice and Theory of Automated Timetabling VI*, vol. 3867 of *Lecture Notes in Computer Science*, pp. 394–412, Springer, 2007.

[25] A. Alcayde, R. Baños, C. Gil, F. G. Montoya, J. Moreno-Garcia, and J. Gómez, "Annealing-tabu PAES: a multi-objective hybrid meta-heuristic," *Optimization*, vol. 60, no. 12, pp. 1473–1491, 2011.

[26] X.-S. Yang and S. Deb, "Multiobjective cuckoo search for design optimization," *Computers and Operations Research*, vol. 40, no. 6, pp. 1616–1624, 2013.

[27] I. Pavlyukevich, "Lévy flights, non-local search and simulated annealing," *Journal of Computational Physics*, vol. 226, no. 2, pp. 1830–1844, 2007.

[28] G. M. Viswanathan, F. Bartumeus, and S. V. Buldyrev, "Levy Flight random searches in biological phenomena," *Physica A*, vol. 314, pp. 208–213, 2002.

[29] A. M. Reynolds, "Cooperative random Lévy flight searches and the flight patterns of honeybees," *Physics Letters A*, vol. 354, no. 5-6, pp. 384–388, 2006.

[30] T. T. Tran, T. T. Nguyen, and H. L. Nguyen, "Global optimization using levy flight," in *Proceedings of the 3rd National Symposium on Research, Development and Application of Information and Communication Technology (ICT.rda '06)*, Hanoi, Vietnam, September 2004.

[31] S. He, "Training artificial neural networks using lévy group search optimizer," *Journal of Multiple-Valued Logic and Soft Computing*, vol. 16, no. 6, pp. 527–545, 2010.

[32] B. Akay, *Nimerik Optimizasyon Problemlerinde Yapay Arı Kolonisi (Artifical Bee Colony, ABC) Algoritmasının Performans Analizi*, Kayseri Üniversitesi, Fen Bilimleri Enstitüsü, Kayseri, Turkey, 2009.

[33] R. Akbari, A. Mohammadi, and K. Ziarati, "A novel bee swarm optimization algorithm for numerical function optimization," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 10, pp. 3142–3155, 2010.

[34] K. Sundareswaran and V. T. Sreedevi, "Development of novel optimization procedure based on honey bee foraging behavior," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics (SMC '08)*, pp. 1220–1225, October 2008.

[35] R. Venkata Rao and V. Patel, "An improved teaching-learning-based optimization algorithm for solving unconstrained optimization problems," *Scientia Iranica*. In press.

[36] Y. Marinakis, M. Marinaki, and N. Matsatsinis, "A bumble bees mating optimization algorithm for global unconstrained optimization problems," *Studies in Computational Intelligence*, vol. 284, pp. 305–318, 2010.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 805343, 12 pages
http://dx.doi.org/10.1155/2013/805343

Research Article

Immunity-Based Optimal Estimation Approach for a New Real Time Group Elevator Dynamic Control Application for Energy and Time Saving

Mehmet Baygin[1] and Mehmet Karakose[2]

[1] *Computer Engineering Department, Ardahan University, Ardahan, Turkey*
[2] *Computer Engineering Department, Firat University, Elazig, Turkey*

Correspondence should be addressed to Mehmet Baygin; mehmetbaygin@ardahan.edu.tr

Received 3 May 2013; Accepted 13 June 2013

Academic Editors: P. Agarwal, S. Balochian, V. Bhatnagar, and Y. Zhang

Nowadays, the increasing use of group elevator control systems owing to increasing building heights makes the development of high-performance algorithms necessary in terms of time and energy saving. Although there are many studies in the literature about this topic, they are still not effective enough because they are not able to evaluate all features of system. In this paper, a new approach of immune system-based optimal estimate is studied for dynamic control of group elevator systems. The method is mainly based on estimation of optimal way by optimizing all calls with genetic, immune system and DNA computing algorithms, and it is evaluated with a fuzzy system. The system has a dynamic feature in terms of the situation of calls and the option of the most appropriate algorithm, and it also adaptively works in terms of parameters such as the number of floors and cabins. This new approach which provides both time and energy saving was carried out in real time. The experimental results comparatively demonstrate the effects of method. With dynamic and adaptive control approach in this study carried out, a significant progress on group elevator control systems has been achieved in terms of time and energy efficiency according to traditional methods.

1. Introduction

The scheduling of the group elevator systems, which has been used for vertical transportation at high-rise buildings, has an important place in our daily lives. Aiming to give rapid service to the crowd population in the building, such systems have very complex structure. One of the issues that people usually complain inside building is vertical transportation service of the building [1]. The studies on this topic continue for a long time, and important developments are provided.

The group elevator systems are structures, which elevate the population in the building from one floor to another by using more than one cabin in the widest sense. Generally managed from one control center in a coordinated manner, such systems are finding optimal ways according to the condition of calls, and they apply such ways on cabins. However, the width of the optimal way space makes this problem hard to solve in the best way. Because there are the option of n^p different ways in a building having n storey and p cabin [2, 3]. Actually, what is expected from the group elevator systems is the average performance of such cabins instead of individual performance of elevator. In addition, the basic criteria at such performances are to minimize the average waiting time of passengers and energy consumption values of cabins [3, 4]. Generally, by considering such parameters, the most optimum way is selected from the way space, and these are applied to the system. Actually, there are many factors, which affect average waiting time; these can be counted as lack of some information like number of passengers, elevator speed, and so on. For instance, the passengers have to wait new coming elevator for a long time because they missed the elevator a few minutes ago. In addition to this case, although no one uses elevator because of call of any empty floor, the passengers often wait for closing

the doors [5]. All of this and similar circumstances are caused by lack of traffic information, and those waiting directly affect the average waiting time. However, despite all such circumstances, average waiting time can be approximately calculated as shown in the following equation [6, 7]:

$$\begin{aligned} \text{AWT} &= 0.4\,\text{INT}, \quad \text{for car loads} < 50\%, \\ \text{AWT} &= \left[0.4 + \frac{1.8\,P}{\text{CC}} - 0.77^2\right]\text{INT}, \quad \text{for car loads} > 50\%, \end{aligned} \tag{1}$$

where AWT, average waiting time, P, numbers of passenger, INT, interval (the main floor average time), and CC, car capacity.

For the most suitable car allocation at the past, although some mathematical approaches are thought, today soft computing techniques and artificial intelligence algorithms are often applied to this kind of system. In addition, such calculations can be made in shortest time with advanced computer systems, and the most optimum car can be allocated to the users [3].

One other criterion of the group elevator systems is energy consumption values in cabins. Although the energy consumption of the elevator occurs at the moment of stop start, it consists of two sections as travel and standby position. The consumption during travel (stop-travel-start) compromises more than 70% of total energy consumption [8]. The most important problem here is irregular energy consumption of cabins. For example, working of elevators without any pattern causes some cabins to operate more and to wear off. This parameter is considered in applications to be made for the removal of this condition, and load distribution of cabins is attempted to be equalized in parallel with average waiting time. Generally, it is very hard to determine daily energy consumption of elevator. However, Schroeder makes many measurements in many elevators and creates a formula to calculate this value. In (2), according to the Schroeder method, daily energy consumption amount of an elevator has been given. In this equation, TP shows general course time, and this term varies depending on drive mechanism type and speed of elevator [7]. The strongest point in the Schroeder method is to determine the ST value. Because the incoming calls are not known earlier, accordingly, how much the lifts operate during day is not predetermined. For this reason, this value is determined by either making certain measurements or by using certain approaches [7, 9]. Because the energy consumption of elevators is mostly caused by stop-start movements, this value directly affects the equation [9]

$$\text{EC} = N * \frac{R * \text{ST} * \text{TP}}{3600}, \tag{2}$$

where EC is energy consumption (kWh/day), N is cabin number, R is motor ratings (kW), ST is daily start number, and TP = time factor.

The most important factor affecting both the average waiting time of passengers and energy consumption in group elevator systems is traffic time of buildings [2, 9]. This is very important issue because passengers use elevators much in certain hours, and they rarely use cabins in rest of their times. In the observations, 4 different traffic times are determined in any building, and such traffic hours and passenger density created depending on those hours are considered in either real world or computer simulations. Depending on those hours, the first of created density is uppeak, and it includes the hours when people come to work at mornings at the office building [1, 4, 6, 8]. The calls made in traffic are generally upward position, and these upward calls reach peak points at mornings. The second traffic movement is downpeak. This situation includes arrival time from work, and calls from floors are generally downward [5, 9]. Other traffic time is seen at lunch times. It contains the time frame at which building population is going out at lunch break and is coming back to work [10]. Traffic condition is random situation between floors. This traffic condition is caused by passing of people inside building between floors, and it is traffic condition, which is seen during working hours [3, 8].

There are various studies made in literature on group elevator systems. Either reduction of average waiting time of passengers or reduction of energy consumption is wide in literature. In such studies on which certain traffic conditions are considered, the soft computing techniques like genetic algorithms, fuzzy logic, particle swarm optimization, and artificial intelligence [4, 6, 8]. However, sheer number of parameters in elevator systems, width of solution space, and failure to determine incoming calls earlier make scheduling problem hard to solve [2, 3]. Despite these lacks, an important development has been performed from the first day of appearance of group elevator systems, and still continues to develop. Some studies, which have been made on literature within scope of working has been examined, such studies have been classified as aim, used method, traffic condition, number of elevator, and simulation forms. However, some lacks have been found in studies performed in literature in this classification. This study aims at elimination of lacks in literature with this study, and brings a new point of view to the solution of scheduling problems for group elevator systems. Two examples have been given to the studies which have been made in the literature Figure 1.

As seen in Figure 1, the procedures in the studies made in literature are performed over one algorithm with certain number of cabins. These limitations bring many problems. For example, subjecting to only one algorithm does not guarantee that the application always gives best results. Because each algorithm has different ways to reach better, and for this reason, the best results, which have been found by each algorithm, are different from each other. In this kind of system, it is evident that better results can be obtained by using different algorithms together. One other lack in the studies made on group elevator systems is limited number of cabins. This situation is directly associated with use of only one algorithm in the system. The largest lack here is that the algorithm selected gives better results in the designated number of algorithm. However, the results are reverse from one expected when you exclude the number of cabins. This study makes clear all this and similar conditions, and the studies made on literature gains new dimension. The artificial immune systems, genetic algorithms, and DNA

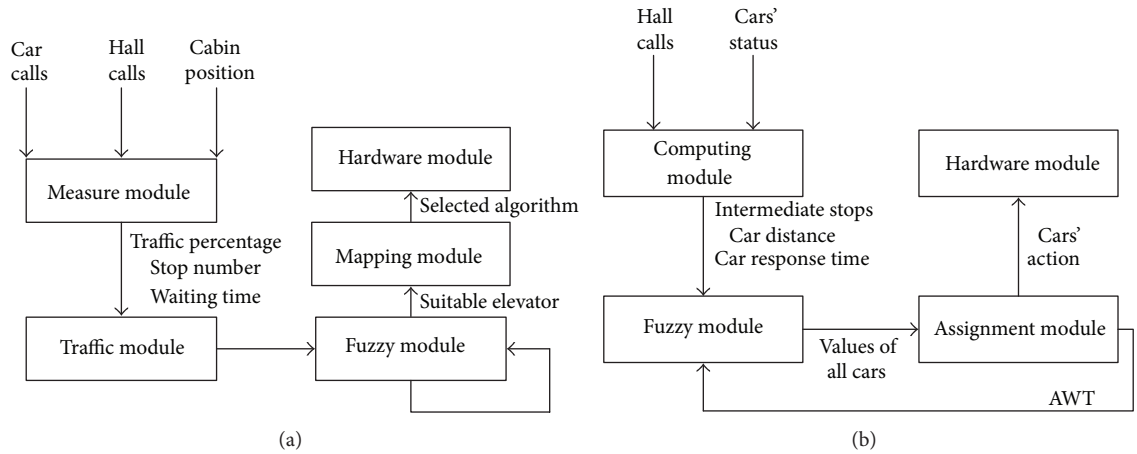


Figure 1: The closest two studies in literature (a) [2] (b) [4].

computing algorithm have been used in the study performed for reduction of average waiting time of passengers and providing energy efficiency in group elevator systems. In addition to these methods, an estimation algorithm has been developed, the relation of such methods has been provided with fuzzy logic, and the most suitable algorithm has been aimed according to the calls and cabin conditions of the system in adaptive way. The results which have been obtained from fuzzy logic are sent to the microprocessor-based experiment set, and the performance and accuracy of the system have been observed. The system procedure and methods used for control algorithm, which is recommended in 2nd section within scope of this study, have been examined; the simulation results and test results which have been obtained from these algorithms in 3rd section have been given. The obtained results are assessed in 4th and final section of the study.

2. Proposed Approach

The group elevator systems have a very complex structure. The sheer number of input parameters and many options as output makes these systems hard to solve. Especially the conditions as unknown building population or unknown floor by a passenger are basic factors in creation of wide solution space. However the developed electronic and computer world is useful for similar problems and used to solve this. Especially, making computational procedures about milliseconds makes detection of most suitable option from wide solution space.

There are some basic principles required to perform by system while group elevator systems are in service. These principles, which have been given in list below, are considered in this study.

(i) The elevator cannot pass up cabin call. It must complete these until all calls in present direction, and the cabin calls in reverse direction must be ignored in this process. After present direction of elevator returns reverse, the rest of calls must be completed.

(ii) If there is enough space inside cabin before change of direction of system, it can collect floor calls in same direction.

(iii) If there is no one at the point where floor call is made, this floor call can be ignored.

(iv) It is required to do less stop-start movement as much as possible to provide energy efficiency of system. To perform this, no more than one cabin goes to the same floor call.

(v) It is required to do feasibility study before for population inside building and frequency to use elevator by this population in order to increase efficiency of methods to be used in the system.

This study aims at reduction of average waiting time of passengers and providing energy efficiency. The proposed control approach primarily consists of 3 main modules. The first module is optimization module, and calls are evaluated here according to the 4 different optimization methods. The results which have been obtained from the assessment made according to the average waiting time and energy efficiency are sent to the 2nd module, the fuzzy module. In this module, the method to be applied on call pool is determined, and optimum way must be given as exit. The optimal way obtained from the fuzzy module is sent to the hardware module and is tested on experiment set. A flow diagram is given in Figure 2, which summarizes the system.

2.1. Optimization Module. A group elevator system is modeled with modules designated within scope of this study. The optimization methods, which are often used in wide problems where solution space is wide, are used within scope of study. Three different optimization methods and 1 estimation algorithm are used in this performed optimization module. As optimization methods, clonal selection algorithm, genetic

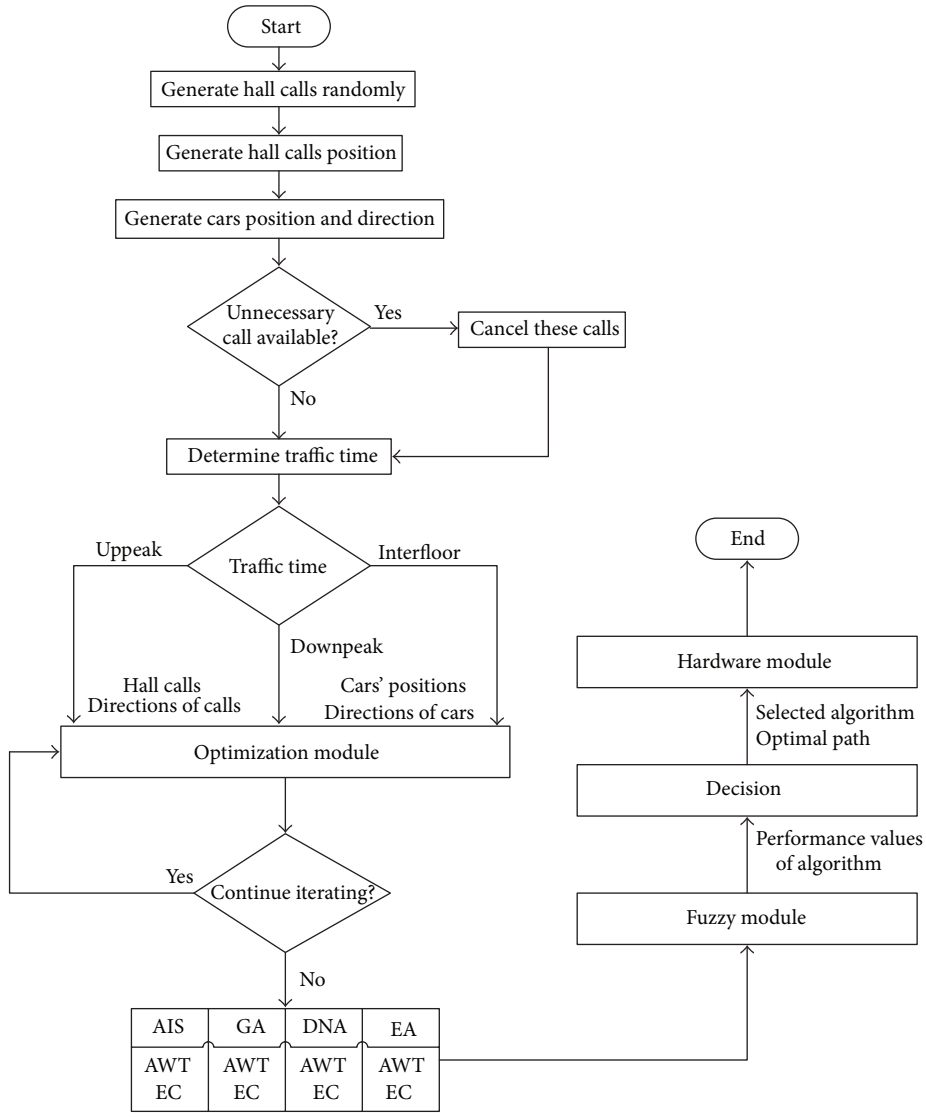


Figure 2: Flow chart of the proposed approach.

algorithms, and DNA computing algorithm are used from artificial immune system.

2.1.1. Clonal Selection Algorithm. The artificial immune algorithms which are proposed by taking human immune system as example incorporate negative and clonal selection algorithms. While negative selection algorithm is used in determination of undesired situations, the clonal selection algorithms are often used in optimization and pattern recognition problems [11, 12]. The pseudocode of the clonal selection algorithm, which has been used for optimization purpose in this study, has been given below.

Step 1. The antibodies compromising body antibody repertoire constitute the initial solution set.

Step 2. Degree of similarity of antibodies is calculated.

Step 3. n pieces of highest similar antibody are selected.

Step 4. Proportional to the degree of similarity of the n pieces of the selected antibody, the high degree of similarity cloning of antibody is carried out.

Step 5. Antibodies with a high degree of similarity are exposed to the mutation as a manner to become less.

Step 6. The degrees of similarity of mutated clones are determined.

Step 7. n pieces of highest similar antibody are selected again.

Step 8. Change of d pieces of antibodies in lowest degree of similarity with newly produced antibodies is realized.

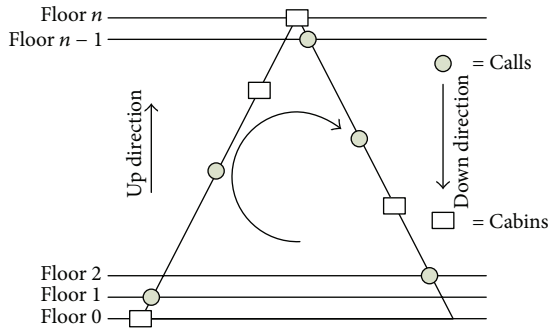


Figure 3: Proposed estimation algorithm.

2.1.2. Genetic Algorithm. Genetic algorithms are search and optimization methods are aiming to find the best result from the problem space in the widest sense [13, 14]. Offering extremely fast and efficient solutions, this method has been used in this study as one of the optimization methods, which have been used during simulation. A pseudocode, which summarizes the operating principle of genetic algorithms, is given below.

Step 1. Initial population of randomly generated sequences of binary numbers.

Step 2. A certain amount of element is selected for the solution.

Step 3. Crossing is applied to new population.

Step 4. Mutation process is applied for the same population.

Step 5. Affinity values of elements of these populations are found.

Step 6. Step 2 is repeated until it reaches the maximum number of transactions.

2.1.3. DNA Computing Algorithm. DNA computing algorithm is a method, which is mainly used in literature, and it is soft computing technique used in solution of the NP hard problems [15, 16]. The operating principle of this method used in this study as assessment criteria is similar to the genetic algorithms, and the pseudocode which summarizes this algorithm is given below.

Step 1. First population is created.

Step 2. DNA sequences are converted to numeric values.

Step 3. Affinity value is calculated for each element.

Step 4. Crossing process is applied to individuals in the population and the new population is obtained.

Step 5. Mutation of the enzyme is applied to new population.

Step 6. Mutation of the virus instead of the deleted elements is applied.

Step 7. Affinity value of population is determined. If newly found population value is better than original value, it is changed and it is continued from 3rd step until maximum procedure number is reached.

2.1.4. Proposed Estimation Algorithm. Within the scope of the study, an estimation algorithm has been proposed in order to bring a different point of view to group elevator systems. Together with this algorithm, it has been aimed to enable the energy efficiency of the cabins. Since the results obtained by optimization methods used bring overload to some cabins, it causes some cars work more and some cars work less. This is an undesired situation and it both directly reflects to average waiting time and energy consumption values of the cabins. In principle, it is expected from an elevator system that energy consumption values should resemble each other so that all the cabins can work efficiently to decrease waiting time of the passengers. For this reason, a diagram, which summarizes the estimation algorithm performed, has been given in Figure 3.

Proposed estimation algorithm considers the longest route that the elevators can follow. As it is shown in Figure 3, left side of the triangle shows upper direction, and right side of the triangle shows the down direction. The rectangles present in these sides show the instant position of the cabins and rounds show the calls made. The longest route that the elevators can follow is to go to the top floor and turn back to the ground floor, and this situation has been shown with an arrow in the triangle. In other words, the cabins in the triangle move clockwise and calls settle into the edges of the triangle according to the directions. Proposed estimation algorithm is based on answering calls available to the current direction of movement of the cabins. Pseudocode, which summarizes working principle of this algorithm, has been given below.

Step 1. Determine number of user floor, number of cabins, position of cabins, and their directions.

Step 2. Randomly directed calls are created.

Step 3. Calls at ground floor are always arranged upwards and calls at top floor of building are arranged downwards.

Step 4. Cabins and randomly produced calls are placed on triangle according to directions.

Step 5. The arrival time to calls is calculated according to affinity function of cabins.

Step 6. Calls having low arrival time are shared to proper cabins and these calls are deleted from call pools.

Step 7. If there are remaining calls, proceed from the 5th step.

Step 8. Proceed from Step 2 until maximum iteration number is reached.

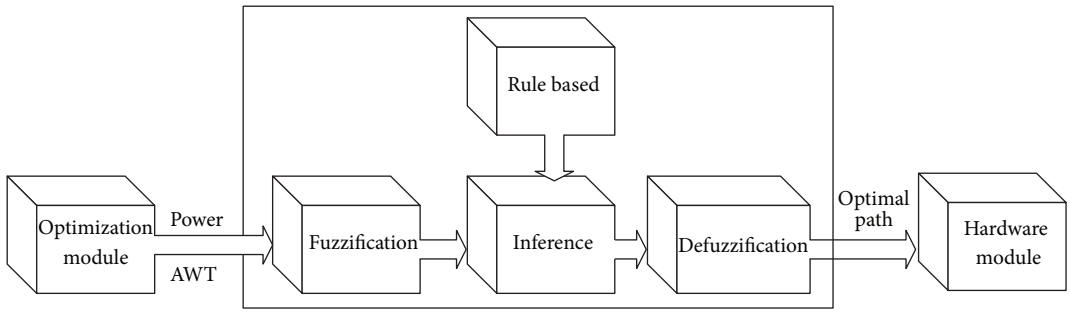
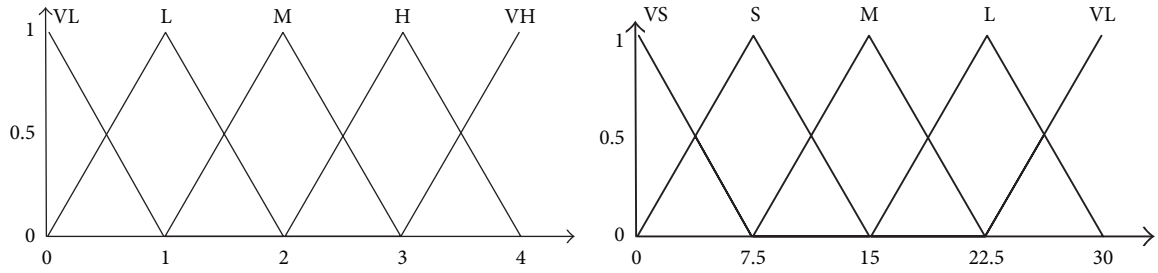


Figure 4: Fuzzy logic module.



(a) Membership functions for energy consumption

(b) Membership functions for average waiting time

Figure 5: Input membership functions.

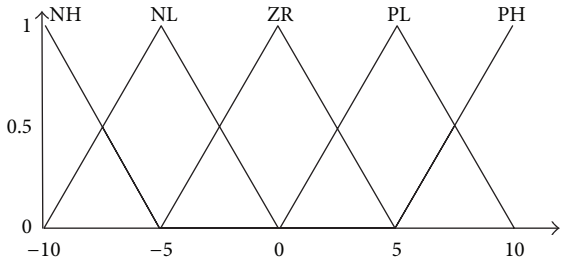


Figure 6: Output membership functions.

Table 1: Rule base of fuzzy logic module.

Input	Energy consumption				
	VL	L	M	H	VH
Average waiting time					
VS	NH	NH	NL	NL	ZR
S	NH	NL	NL	ZR	PL
M	NL	NL	ZR	PL	PL
L	NL	ZR	PL	PL	PH
VL	ZR	PL	PL	PH	PH

2.2. Fuzzy Module. Another part in proposed control algorithm for group elevator systems is fuzzy module. The diagram which summarizes fuzzy systems consisting of generally four fundamental parts has been given in Figure 4 [17, 18].

Fuzzy module in the system takes energy consumption values and average waiting time coming from optimization module and sends to fuzzy part. In step of fuzzification, there is energy membership input function and average waiting time membership input function. The input functions have been given in Figure 5.

The values obtained from membership input function are sent to inference part to be evaluated. Here, Mamdani method is used and the system has defined 25 different rules for this procedure. The values obtained from inference part are sent to clarification part to be converted into reel values. In this part, optimal route is defined according to exit membership functions defined. In Table 1 given below, rule base used in fuzzy module has been given and membership exit function has been shown in Figure 6.

2.3. Hardware Module. The final part of the practice performed within the scope of the study is hardware part. In this part, optimal path coming from fuzzy logic module has been evaluated in the microprocessor-based experiment set and the accuracy of the system is tested. Also, led mechanism connected to experiment set represents a building and the cabins in the building. The practice made with this module has been brought into practicable condition by taking from being simulation. A diagram, which summarizes a hardware module used in the system, is as given in Figure 7.

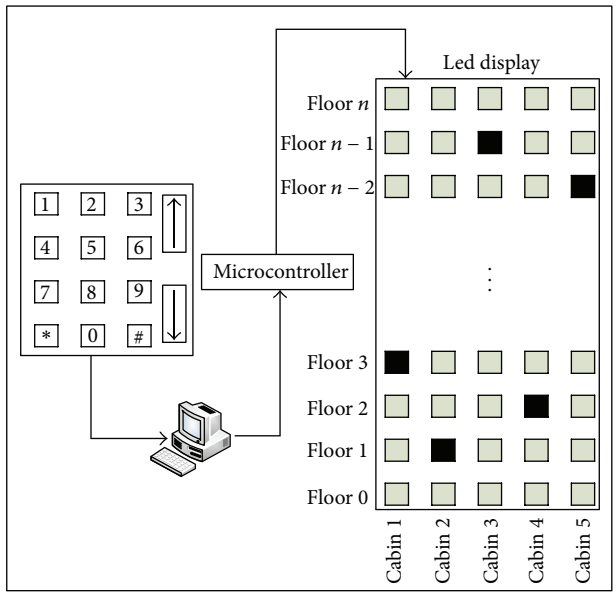


Figure 7: Hardware module.

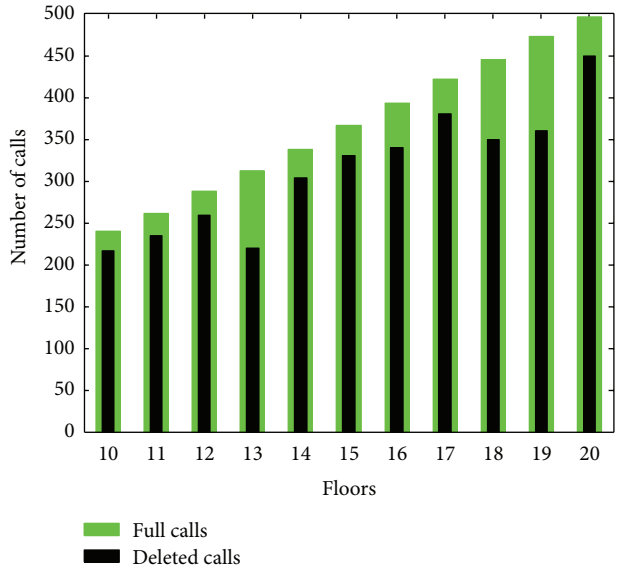


Figure 8: Calls according to number of floors.

3. Experimental Results

Within the scope of the study done, it has been aimed to decrease the average waiting time of the passengers and enable energy efficiency with proposed control approach. In direction of these purposes, the application performed has been performed by using 3 modules and the results are perceptibly observed. Simulation stage of the application has been prepared in a computer environment with Windows 7 operating system including 2,53 GHZ speed, 4 GB RAM, Intel Core i5. The study done has been performed in Matlab R2009b platform. An experiment set has been used for hardware stage of the application. This experiment set has been worked in 20 MHZ speed, and RS-232 serial communication line in 9600 bandwidth has been used for communication. The results coming from simulation stage are transmitted to microprocessor-based experiment set through serial communication line. In addition to this, in order to be able to define the starting status of the cabins and let them make call after the system starts, a numeric keypad is used. Building and cabin specifications used as base at simulation stage are as given in Tables 2 and 3.

With the application performed within the scope of the study, total average waiting times and energy efficiencies of the cabins have been calculated. Random 500 numbers of calls have been created within the frame of the application, and approximately 80% of these calls have been arranged as to be downside and upside to be able to obtain up and down traffic condition. For traffic condition in inter floors, half of 500 numbers of calls have been arranged so as to be downward and the other half has been arranged so as to be upward. The distribution of the calls according to the floor has been shown in Figure 8.

Depending on the number of floors, the number of the calls increases. In the study done, if there is nobody in the floor to be made, the cabins are enabled to pass over this call.

Table 2: Building parameters.

Number of elevators	From 10 to 20
Number of cabins	From 2 to 5
Floor height	3,5 m
Building type	Office
Building population	500
Building occupancy	80%

Table 3: Cabin parameters.

Cabin rate	13 person
Velocity	1,6 m/sn
Door open	3 sn
Door close	4 sn
Passenger transfer time	1,2 sn
One floor passing time	2,2 sn

In order to perform this purpose, 20% of the current calls have been cancelled, computational procedures are started after this point. The cancellation of invalid calls provides an important advantage regarding energy efficiency and average waiting time. In Table 4, average waiting time changing to the floor situations and algorithms have been shown.

The performance of algorithms according to the number of traffic peak hours and floors is shown in Figure 9. In the application performed, 3 different traffic times, genetic algorithm, DNA computing algorithm, artificial immune system algorithm, and estimation algorithm have, respectively, been applied. At practice stage, total 11 different floor conditions have been considered, and algorithms through 2, 3, 4, and 5 cabins for each floor have been applied. Average waiting times obtained for each cabin and floor have been added separately and divided into total cabin number. When Table 4 is examined, it has been seen that waiting time increases

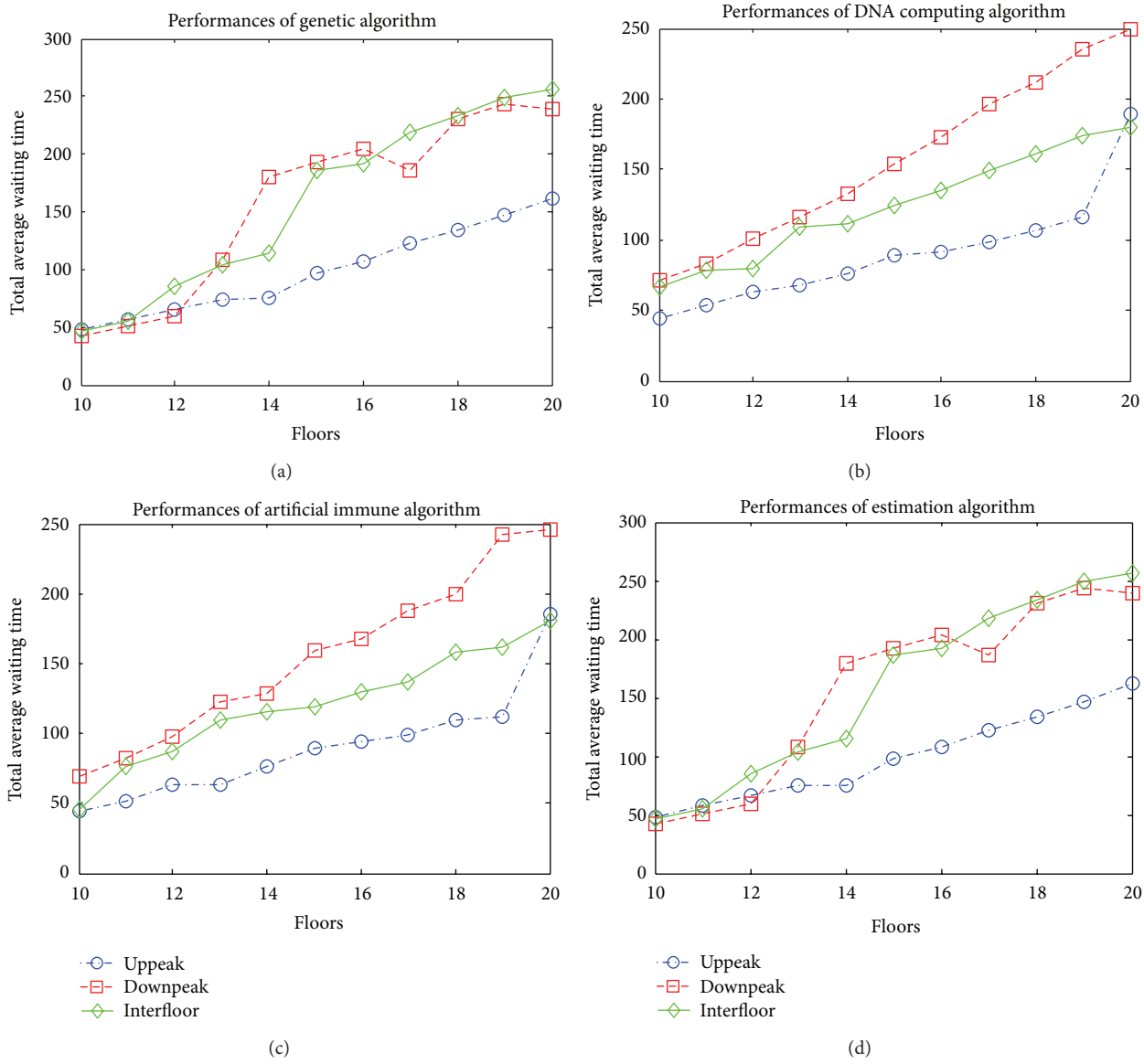


Figure 9: Total average waiting time performances of algorithms.

in parallel with the floor number. Furthermore, the performances that algorithms have shown under different traffic conditions are different from each other. For example, while artificial immune system algorithm gives the best result in uppeak traffic condition, estimation algorithm in downpeak traffic algorithm shows the best performance. There are so many factors which affect average waiting times in group elevator systems. As the complete of factors such as number of passengers or hall calls are not known, it cannot completely be calculated regarding which algorithm that provides the best solution. With this study done, a system with adaptive structure has been created and significant results have been obtained from the point of waiting time and energy efficiency.

Another parameter within the scope of the study done is the energy efficiency of the cabins. One of the most important disadvantages of the group elevator systems is the lowness in energy performances arising from working principles. 70% of the energy that elevator systems consumed occurs while the cabins stop and start. Furthermore, this stop and start movements are seen more frequent in some cabins compared to the others. This situation causes some cabins to consume more energy and wear down. In principle, it is expected from group elevator systems that total energy to be consumed is provided to be stably distributed to the cabins according to the stop-and-start number. Because of working principles of optimization methods, some cabins are overloaded. With the study performed, this situation is aimed to be removed and the energy is provided to be distributed as efficiently. An example of the energy distributions that the cabins consumed has been shown in Figure 10 according to the traffic and algorithm conditions.

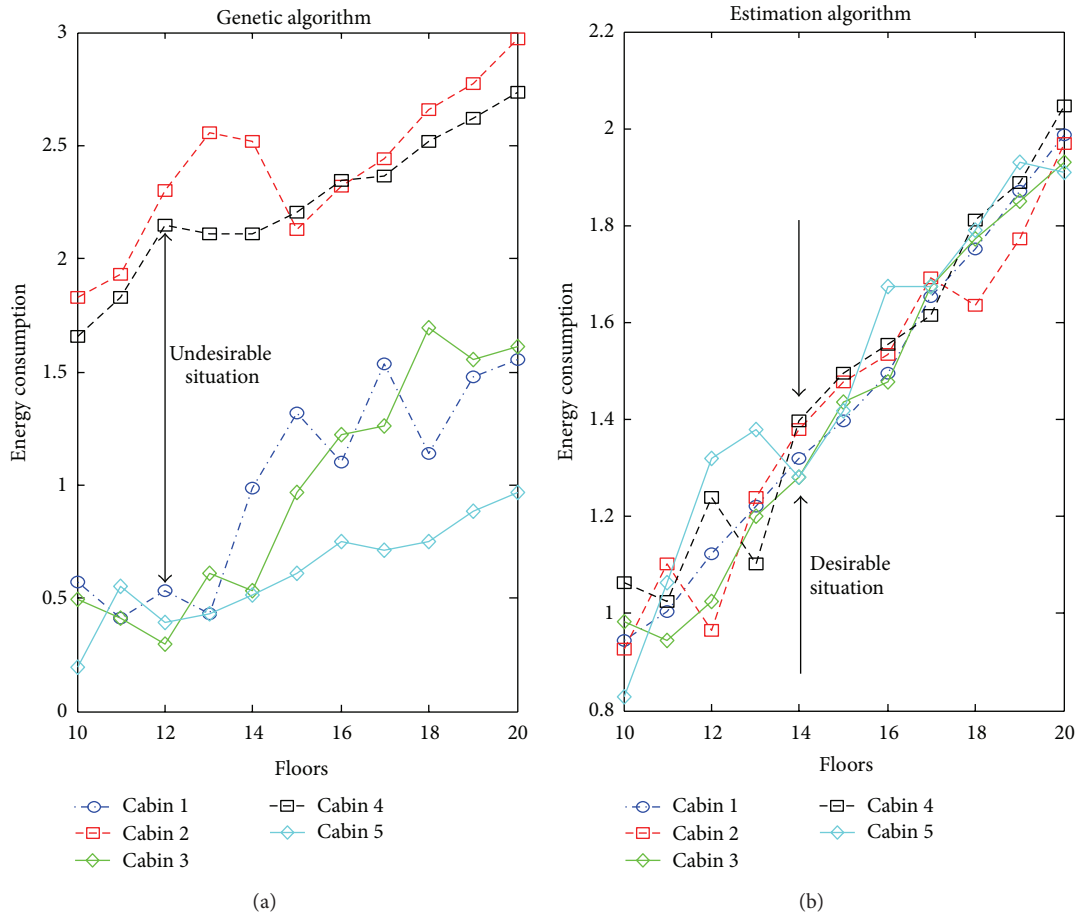


Figure 10: Energy consumption of cabins according to floors.

Table 4: Average waiting time of passengers (s).

Floor	GA			DNA			AIS			EA		
	Uppeak	Downpeak	Inter floor	Uppeak	Downpeak	Inter floor	Uppeak	Downpeak	Inter floor	Uppeak	Downpeak	Inter floor
10	52,1	75,1	68,7	**44,8**	71,5	66,2	44,4	69,9	**45,7**	48,5	**42,7**	46,8
11	59,2	87,7	79,7	53,3	82,7	78,3	**51,2**	82,2	76,3	57,2	**50,4**	**55,4**
12	**62,3**	101,5	91	63,5	100,6	**79,6**	62,8	97,5	86,5	65,9	**59,2**	85,5
13	72,9	117,2	**103,7**	67,6	**115,7**	108,5	**63,6**	122,8	109,2	74,5	108,3	104,1
14	78,3	135,9	**110,9**	75,9	133,1	111,4	76,4	**128,8**	115,4	**75,4**	179,7	114,3
15	**88,3**	159,9	127,6	89,4	**154,1**	124,7	89,2	159	**119**	97,6	192,6	186,6
16	93,8	174,6	137,9	**91,1**	172,8	134,7	93,9	**167,9**	129,2	107,3	203,9	192,2
17	**97,5**	198,4	147,2	99	196,2	149,5	98,4	188,3	**137,2**	122,5	**185,8**	218,4
18	109,6	211,1	**154,3**	**107,1**	211,3	161,1	109,1	**200**	157,6	133,8	231,1	233,6
19	115,7	**234,4**	171,6	115,7	235,3	174,5	**112,2**	242,4	**161,3**	147,1	243	248,6
20	188,5	250	186,8	189,3	249	**179,6**	185	245,9	181,2	**162,1**	**239,3**	256,1

As it can be understood from the figure, there is an unbalanced load distribution between the cabins according to the optimization methods. With the proposed estimation algorithm, this situation is prevented and it has been enabled to make the cabins share the calls equally. The equal distribution of the calls to all the cabins provides a significant efficiency from the point of energy. By this means, a development occurs in the usage times of the cabins and the wearing times of the cabins extremely decrease.

It is expected from group elevator systems that average waiting time of the system and energy efficiency should be provided at the same time. Because of the fact that in

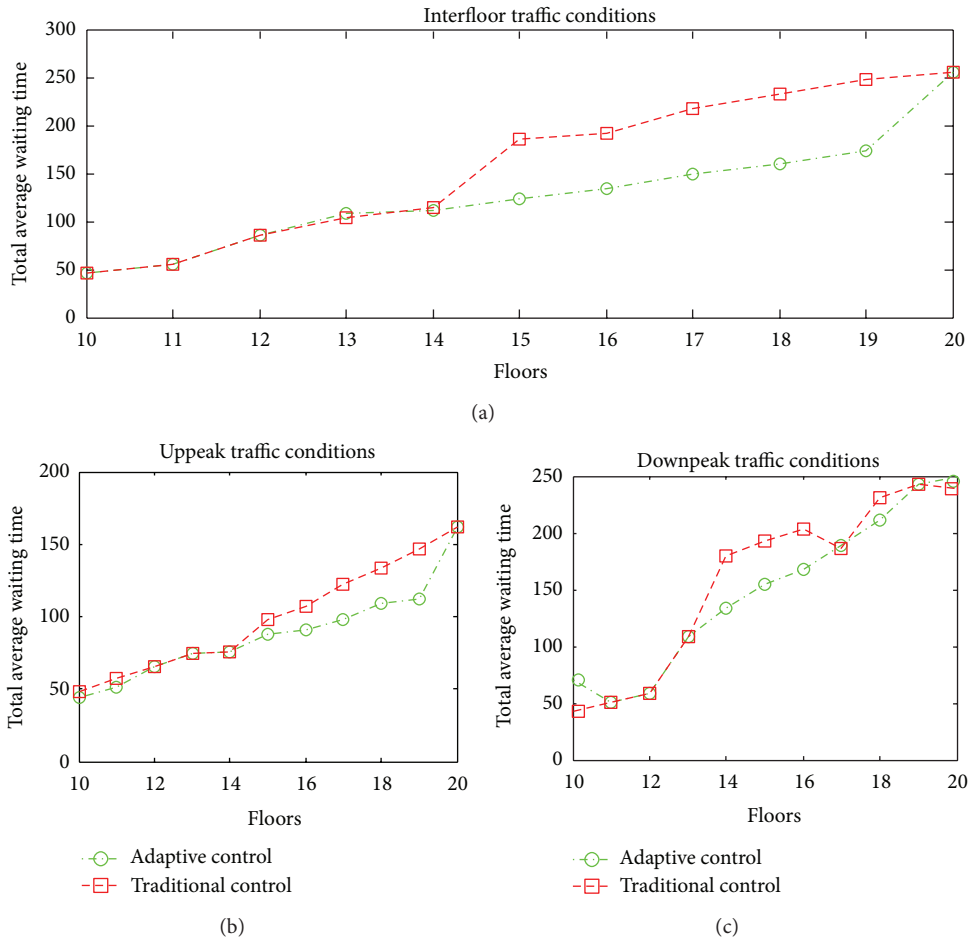


Figure 11: Comparative results for classical methods and the proposed approach.

Table 5: The results obtained from the fuzzy controller.

Floor	GA			DNA			AIS			EA		
	Uppeak	Downpeak	Inter floor	Uppeak	Downpeak	Inter floor	Uppeak	Downpeak	Inter floor	Uppeak	Downpeak	Inter floor
10	−27,5	−26,7	−27,2	−26,1	−22,7	−26,1	**−28,0**	**−27,5**	**−28,0**	−25,8	−26,0	−25,9
11	−24,8	−21,1	−22,7	−24,7	−20,1	−24,7	**−26,6**	−23,4	−24,4	−25,2	**−25,2**	**−25,2**
12	−24,8	−10,5	−21,1	−22,0	−14,2	−22,0	−23,9	−16,5	−20,2	**−25,0**	**−25,0**	**−25,0**
13	−21,4	−3,1	−10,3	−19,5	−6,11	**−19,5**	−20,9	−2,84	−7,30	**−25,1**	**−7,96**	−10,8
14	−18,5	0,25	−6,17	−18,6	**0,03**	**−18,6**	−16,7	2,92	0,35	**−22,1**	15,1	−3,17
15	**−17,5**	9,56	2,31	−16,3	**6,76**	**−16,3**	−12,2	8,09	1,22	−16,5	19,6	17,3
16	−11,7	15,2	6,91	**−16,4**	12,3	**−16,4**	−16,1	**11,3**	2,26	−4,37	24,3	19,8
17	**−6,97**	25,7	8,54	−6,5	22,2	**−6,58**	−6,4	**17,8**	6,64	8,4	20,1	24,5
18	0,33	26,9	12,6	0,6	**24,0**	**0,64**	**−1,1**	26,3	11,6	11,6	25,1	25,0
19	4,9	26,3	16,2	3,3	25,3	**3,30**	**0,7**	25,4	15,5	15,7	**23,1**	22,7
20	22,19	**24,1**	21,4	22,5	25,2	22,5	21,6	25,3	20,3	**18,7**	27,8	**20,1**

some circumstances, while average waiting time is good, energy efficiency shows inverse proportion to this time. To prevent this situation, a fuzzy system has been designed. The fuzzy system designed takes energy values and average waiting times coming from 4 different algorithms to the traffic conditions as input parameter. Then, it evaluates these parameters and provides the most optimal path to be determined. The basic purpose in fuzzy system designed is to find the lowest average waiting time and the best result of energy efficiency. In Table 5, the values rising from fuzzy logic

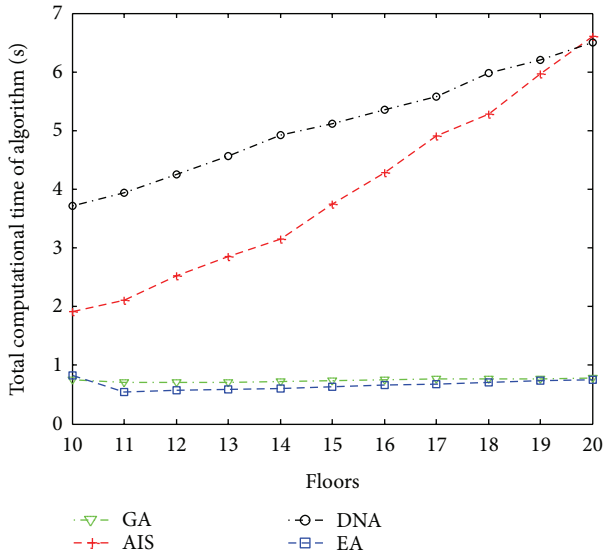


Figure 12: Total computational time of algorithms.

and the methods to be applied according to these values have been shown.

Thanks to this performed study, an adaptive approach is developed related to group elevator control systems. It is aimed at making this approach reduces the average waiting time of passengers and provides energy efficiency. The proposed adaptive approach and four different algorithms are combined and fuzzy logic module of the system and the results from these different algorithms are provided to evaluate. In Figure 11, traditional methods on group elevator control systems and changing total average waiting times depending on number of floors of proposed adaptive approach.

These findings obtained in simulation platform are sent to the experiment set and the accuracy of the system has been tested. Figure 12 shows the computing time passing from simulation platform to the experiment set according to the floor number and traffic condition for each algorithm separately.

4. Conclusions

The elevator systems have a structure that constantly renews itself since day one of the emergence. Today, the increase of high-rise buildings is the biggest factor for this situation. Group elevator control systems which were developed in order to provide faster service to resident are computerized to provide both time saving and energy efficiency. Group elevator control systems which have very large place in the literature are not incomplete in terms of efficiency because all of parameters are not taken into consideration.

In this study, an immune system-based approach for the control of group elevator control systems is conducted, and reducing of the average waiting times of passengers and providing of energy efficiency of system are aimed to provide. In the scope of this study, the adaptability of this application to all building structure is aimed to provide by creating building models from 10 floors to 20 floors and from 2 cabins to 5 cabins. In the next step, calls randomly generated according to traffic hours are sent to optimization module which includes immune system algorithm, genetic algorithm, and DNA computing algorithm to determine average waiting times and energy consumption values of cabins. The average waiting time and cabin energy consumption values that they are acquired from optimization module due to each algorithm are sent to fuzzy system module which is designed in the scope of the study to determine system performance. The optimum way for formed building models and traffic hours are aimed to determine by providing obtained values of fuzzy system. Finally, this optimum way obtained from fuzzy module is tested with hardware module.

This adaptive and dynamic control approach for elevator systems provides efficiency approximately between 5% and 20% on average waiting times of passengers in comparison with traditional methods. In addition to this, a significant gain was achieved related to energy efficiency which is another expected parameter in elevator systems, provided that calls to cabins are as evenly distributed as possible. Future studies target to further increase the efficiency of the system and perform these applications which are performed on a real elevator system.

References

[1] P. B. Luh, B. Xiong, and S. C. Chang, "Group elevator scheduling with advance information for normal and emergency modes," *IEEE Transactions on Automation Science and Engineering*, vol. 5, no. 2, pp. 245–258, 2008.

[2] D. M. Muñoz, C. H. Lianos, M. A. Rincón et al., "Distributed approach to group control of elevator systems using fuzzy logic and FPGA implementation of dispatching algorithms," *Engineering Applications of Artificial Intelligence*, vol. 21, no. 8, pp. 1309–1320, 2008.

[3] P. Cortes, L. Onieva, J. Munuzuri, and J. Guadix, "A viral system algorithm to optimize the car dispatching in elevator group control systems of tall buildings," *Computers & Industrial Engineering*, vol. 64, no. 1, pp. 403–411, 2013.

[4] J. Jamaludin, N. A. Rahim, and W. P. Hew, "An elevator group control system with a self-tuning fuzzy logic group controller," *IEEE Transactions on Industrial Electronics*, vol. 57, no. 12, pp. 4188–4198, 2010.

[5] J. Sun, Q. C. Zhao, and P. B. Luh, "Optimization of group elevator scheduling with advance information," *IEEE Transactions on Automation Science and Engineering*, vol. 7, no. 2, pp. 352–363, 2010.

[6] G. Barney, Ed., *Elevator Traffic Handbook: Theory and Practice*, Taylor & Francis, New York, NY, USA, 2003.

[7] M. Baygin and M. Karakose, "A new intelligent group elevator control approach," in *Proceedings of the 15th International Symposium (MECHATRONIKA '12)*, pp. 1–6, December 2012.

[8] Y. Liu, Z. Hu, Q. Su, and J. Huo, "Energy saving of elevator group control based on optimal zoning strategy with interfloor traffic," in *Proceedings of the 3rd International Conference on Information Management, Innovation Management and Industrial Engineering (ICIII '10)*, vol. 3, pp. 328–331, November 2010.

[9] J. Wang, X. Mu, J. Xu, and S. Wang, "Design and optimization of dispatching rules for elevator group control aiming at energy saving," in *Proceedings of the International Conference on Information Science and Technology (ICIST '12)*, pp. 124–127, March 2012.

[10] P. E. Utgoff and M. E. Connell, "Real-time combinatorial optimization for elevator group dispatching," *IEEE Transactions on Systems, Man, and Cybernetics A*, vol. 42, no. 1, pp. 130–146, 2012.

[11] I. Aydin, M. Karakose, and E. Akin, "An adaptive artificial immune system for fault classification," *Journal of Intelligent Manufacturing*, vol. 23, no. 5, pp. 1489–1499, 2010.

[12] I. Aydin, M. Karakose, and E. Akin, "Artificial immune classifier with swarm learning," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 8, pp. 1291–1302, 2010.

[13] B. Bolat and P. Cortés, "Genetic and tabu search approaches for optimizing the hall call—car allocation problem in elevator group systems," *Applied Soft Computing*, vol. 11, no. 2, pp. 1792–1800, 2011.

[14] K. Hirasawa, T. Eguchi, Z. Jin et al., "A double-deck elevator group supervisory control system using genetic network programming," *IEEE Transactions on Systems, Man and Cybernetics C*, vol. 38, no. 4, pp. 535–550, 2008.

[15] M. S. Muhammad, Z. Ibrahim, S. Ueda, O. Ono, and M. Khalid, *DNA Computing for Complex Scheduling Problem*, Springer, Berlin, Germany, 2005.

[16] H. Jiao, Y. Zhong, and L. Zhang, "Artificial DNA computing-based spectral encoding and matching algorithm for hyperspectral remote sensing data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 10, pp. 4085–4104, 2012.

[17] M. M. Rashid, N. A. Rashid, A. Farouq et al., "Design and implementation of fuzzy based controller for modern elevator group," in *Proceedings of IEEE Symposium on Industrial Electronics and Applications (ISIEA '11)*, pp. 63–68, September 2011.

[18] T. Chen, Y. Hsu, A. Lee, and P. Hong, "Real-time self-tuning approach for intelligent elevator group control system," in *Proceedings of the 5th International Conference on New Trends in Information Science and Service Science (NISS '11)*, pp. 420–424, October 2011.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 581846, 7 pages
http://dx.doi.org/10.1155/2013/581846

Research Article

Reinforcement Learning Based Artificial Immune Classifier

Mehmet Karakose

Computer Engineering Department, Firat University, Elazig, Turkey

Correspondence should be addressed to Mehmet Karakose; mkarakose@firat.edu.tr

Received 3 May 2013; Accepted 16 June 2013

Academic Editors: P. Agarwal, S. Balochian, and Y. Zhang

One of the widely used methods for classification that is a decision-making process is artificial immune systems. Artificial immune systems based on natural immunity system can be successfully applied for classification, optimization, recognition, and learning in real-world problems. In this study, a reinforcement learning based artificial immune classifier is proposed as a new approach. This approach uses reinforcement learning to find better antibody with immune operators. The proposed new approach has many contributions according to other methods in the literature such as effectiveness, less memory cell, high accuracy, speed, and data adaptability. The performance of the proposed approach is demonstrated by simulation and experimental results using real data in Matlab and FPGA. Some benchmark data and remote image data are used for experimental results. The comparative results with supervised/unsupervised based artificial immune system, negative selection classifier, and resource limited artificial immune classifier are given to demonstrate the effectiveness of the proposed new method.

1. Introduction

Artificial immune systems are one of the algorithms that forms the basis of classification methods and they can be used in many areas of daily life. The basis of these algorithms is based on the human immune system, and they have been usually used in popular areas such as the classification, anomaly detection, and the computer virus detection. Human immune system known as the natural immune system is an effective mechanism for protecting the human body against foreign cells. By utilizing these features of natural immune systems, artificial immune systems have been developed for solving the scientific and engineering problems. Artificial immune systems are used in many applications such as optimization, classification, and pattern recognition. In classification algorithms, there are various algorithms that are different from artificial immune systems. These algorithms are based on maximum likelihood and minimum distance. They are used on parallel and piped images [1]. These type algorithms used in the classification of images generally need the preclassified data. In classification methods, there are two basic learning method such as supervised and unsupervised learning algorithms. Block diagrams of some artificial immune classifiers in the literature are given in Figure 1.

Supervised and unsupervised learning algorithms are widely used methods in classification problems [2–13]. Some algorithms such as K-Means [4], ISODATA, Fuzzy C-Means, and AIS (Artificial Immune System) [2] are known as unsupervised classification methods, and they cluster data without the need for the training data. Zhong et al. [2] proposed unsupervised artificial immune classification method for multi-hyperspectral remote sensing images. Four different structures such as water, plants, roads, and buildings are classified by using unsupervised artificial immune system. Afterwards, the obtained results with the results of other methods such as K-Means, ISODATA, Fuzzy C-Means, and SOM methods were compared. In another study, Zhong et al. [3] proposed supervised artificial immune classification for remote sensing images. The effectiveness of the proposed algorithm is verified via simulation results. Another study of Zhong and Zhang [5] supervised adaptive artificial immune network was proposed for multi/hyperspectral remote sensing images. Aydin et al. [7] proposed an adaptive artificial immune algorithm for fault diagnosis of induction motors faults. The classification was made by using three-phase current signals taken from a real induction motor. Stator and broken rotor bar faults are detected by using this classification method.

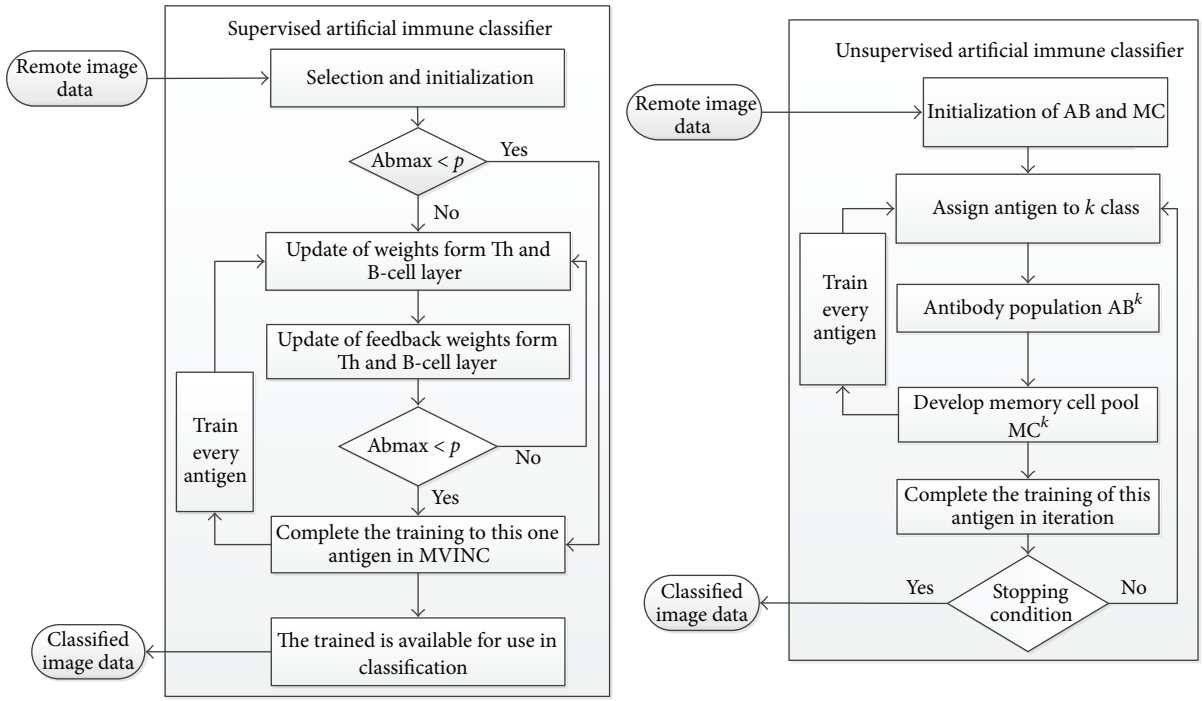


Figure 1: Artificial immune classifiers with supervised-unsupervised learning models [2, 3].

Supervised and unsupervised learning methods have some advantages over one another, though they have some disadvantages according to reinforcement learning. Reinforcement learning presents a wide solution framework in planning and control. This learning method aims to obtain most suitable results with award values for each action of an agent. The purpose of the training process continues until agent reaches the awards. Q learning algorithm is a reinforcement learning algorithm. Q learning algorithm is based on status and action value of Q and converges according to award value of action and Q values of next status. The reinforcement learning has been used for many applications such as fuzzy systems, neural networks, and classification applications [8–11].

This paper presents a new artificial immune classifier based on reinforcement learning. The proposed approach has a self-learning structure using clonal selection and memory cells. Especially this algorithm is trained by reinforcement learning with feedback of mutation rate. Thus best antibodies are obtained to recognize antigens. In this paper, Section 2 presents algorithms for artificial immune systems. Section 3 provides details of the proposed new approach. Section 4 presents the validation of the proposed approach using experimental results and Section 5 gives the conclusions.

2. Artificial Immune Systems

Artificial immune system was exposed by inspiring the human immune system. The method is used in many fields, especially engineering [12]. Artificial immune systems have been developed to detect and destroy viruses with the emergence of computer viruses and used in the antivirus detection process. Immune systems are effective protection mechanisms that protect the human body from foreign antigens or pathogens [13]. The artificial immune algorithm has been uncovered by considering the importance of the artificial immune system in human life. The base of the artificial immune algorithm is the same with artificial immune system. Furthermore, the method has become indispensable for optimization and classification problems. Different types of methods have emerged with creation of the artificial immune algorithm. Some of these methods have become important algorithms which form the basis of the artificial immune algorithm. Artificial immune systems basically consist of two sections. These important algorithms are clonal selection algorithm and negative selection algorithm.

The clonal selection algorithm is based on the cloning of immune cells in type B antigen recognition. The structures that will be used as a set of data structures for the algorithm have been applied in the clonal selection structure. The pseudocode of the clonal selection algorithm is given in Pseudocode 1. The algorithm that used in pattern recognition is shown in Pseudocode 1. The similarities between antibodies and antigens are found and the clones are formed from antibodies with high similarity.

The negative selection algorithm is often used in determination of the undesirable situations. Particularly, this method is use in the operating of virus detection software. The implementation and the operation of the algorithm are quite easy. First, a set of is produced with this method and this candidate detector set is matched with the self-set. If there is a matching, these candidates are transferred to the

```
Start
(1)  Ag determination antigen set
(2)  k the number of steps, N population size
(3)  ab_b the number of antibody for cloning
(4)  ab_k the number of low similarity elements for end of iteration
(5)  P, randomly production of n sized population
(6)  j = 0;
(7)  While j < g do
(8)        similarity (P, Ag) ← computation between antibody and antigen
(9)        P_1 ← Selection(P, ab_b) //selection of best n1 antibody for cloning
(10)       C ← Cloning(P_1, Similarity(P_1)) //clones from P_1
(11)       C_1 ← Mutation(C, Similarity(C)) //Mutation for C by similarity
(12)       Similarity(C_1, Aj) ← similarity between clone set(C_1) and antigen
(13)       M ← Selection(C_1)
(14)       P ← Displacement(P, at_k)
(15)       j ← j + 1
     End while
```

Pseudocode 1: The pseudocode of clonal selection algorithm.

```
Start
(1)  Creation of self sample set (K)
(2)  While (for end of training)
(3)     Production of random detectors (P)
(4)     If(match detector with self sample set)
(5)           Remove detector from set
(6)           Else
(7)           Add detector to set
(8)     End else
(9)  End while
(10) Creation of test sample set (K)
(11) While (for end of testing)
(12)    If(match detector with test sample set)
(13)          Anomaly detection according to active detectors
(14)    End if
(15) End while
```

Pseudocode 2: The pseudocode of negative selection algorithm.

main detector set. Afterwards, the detector set compared by entering the test data. If this test data is recognized by any detector, the desired operation is performed. The training and the test phase of the negative selection algorithm are expressed with pseudocode given in Pseudocode 2.

3. The Proposed Approach

The reinforcement based artificial immune classifier is proposed for classification in this study. Reinforced artificial immune classifier method can be easily used in all areas in which using the computational intelligence is convenient and can produce the best results in used areas. This method is welded from reinforcement learning algorithm which is a machine learning method. Successful results were obtained by using reinforcement learning algorithm in machine learning. The difference of reinforcement learning algorithm from other algorithms is to calculate all probabilities of current states to identify the next state. So, each step that algorithm operates is used as reinforcement for the next operation. The reinforced artificial immune classifier is composed by using this algorithm in combination with an artificial immune system. The flow chart of proposed method is given in Figure 2.

Q learning algorithm is a reinforcement learning algorithm. According to the Q learning algorithm, the existing situation and the Q value of its action converge with a certain coefficient to the award value of the action and Q values of the arriving situations. In the Q learning algorithm, the situation of the algorithm for step and the best possible step for next

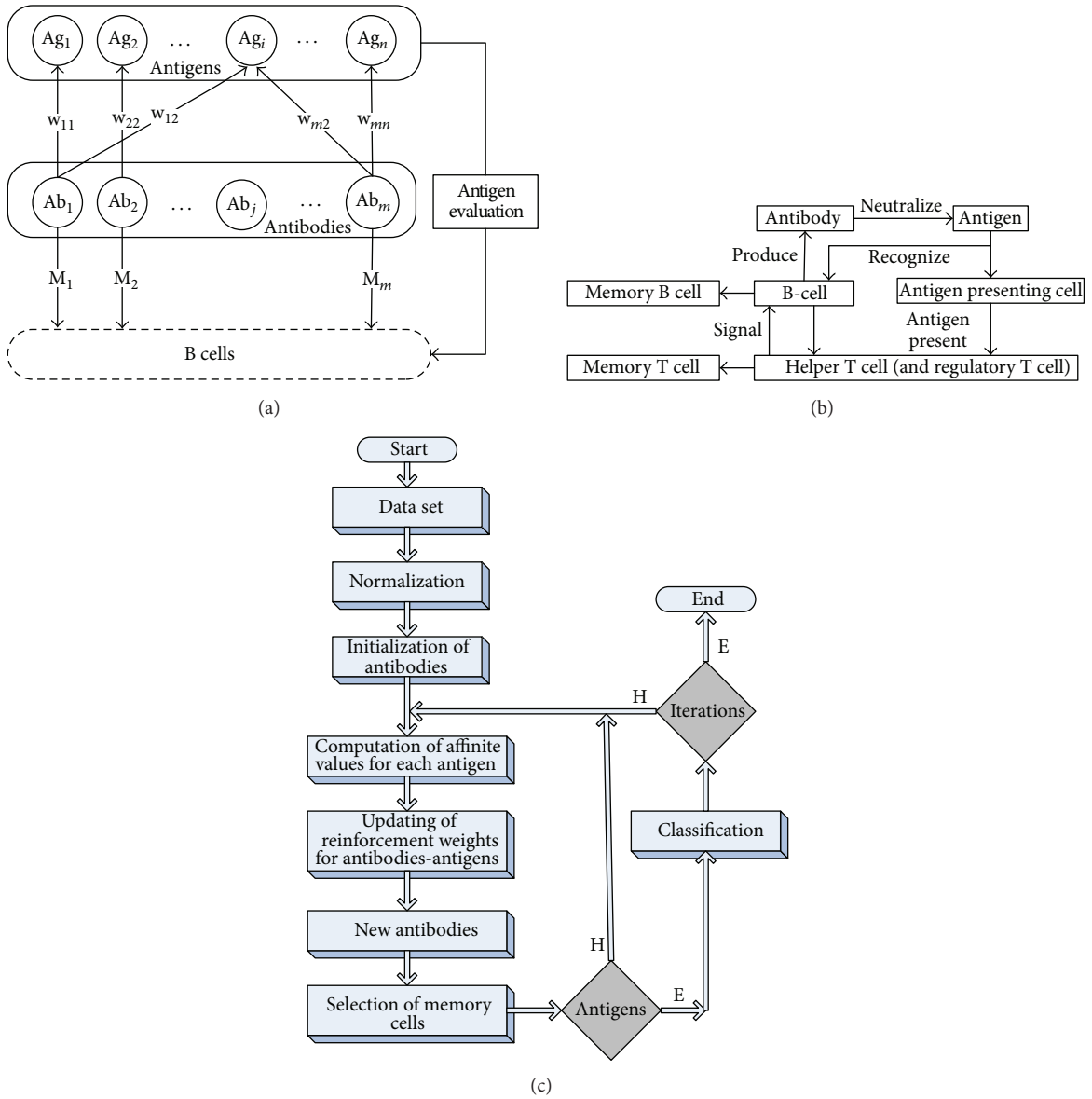


Figure 2: Block diagram and flow chart of the proposed approach.

situations are determined by considering this situation. The Q value of the next situation in algorithm is calculated by (1)

$$\begin{aligned} &Q(\text{status}, \text{action}) \\ &\quad = R(\text{status}, \text{action}) \\ &\qquad + \sigma \max\left(Q(\text{next status}, \text{all mutation actions})\right). \end{aligned} \tag{1}$$

The R value given in (1) indicates the current status of the algorithm. σ max value is the most high affinity value of the steps for the next situation. The pseudocode of Q learning algorithm is given in Pseudocode 3.

Q learning is a commonly used algorithm in reinforcement learning methods. A new reinforced artificial immune

```
Start
(1)   Import of R award status
(2)   Initizialiton of Q values
(3)     For all Q status
(4)     While (arrive to goal status)
(5)       Random start value
(6)       All actions and computation of next status
(7)       Next maximum Q value status
(8)       From current status to next status
(9)     End while
(10)    Update Q status
(11)  End of algorithm
```

Pseudocode 3: The pseudocode of Q learning algorithm.

```
Start
(1)  Loading of training data
(2)  Random starting of antibody set
(3)       While (maximum iteration step)
(4)            Computation of affinity value for each antigen
(5)            Update of reinforcement weights between each antibody and antigen
(6)            Creation of clones of antibodies
(7)            Selection of memory cells
(8)       End while
(9)       Classification of Q memory cells
(10) End of algorithm
```

Pseudocode 4: The pseudocode of reinforcement learning based artificial immune classifier.

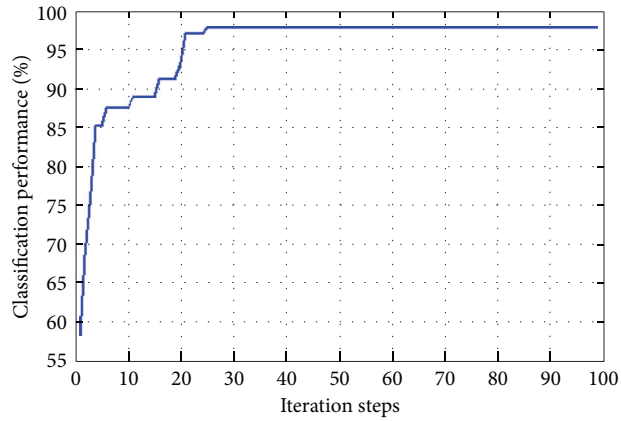


Figure 3: A test result for iris data.

classifier was proposed by applying the algorithm given the steps in Figure 2 to artificial immune system. Each antigen is recognized by an antibody (Ab). The connection between antigens and antibodies is updated according to Q learning algorithm in reinforced learning. Antibodies turn to the antigens is achieved through the clonal selection and mutation operators in immune selection. Antibodies that recognize the antigens are formed B-cell cells as memory cell. The pseudo code of proposed algorithm is given in Pseudocode 4.

In this study, the set of antigens have been identified as training data. Each antigen is represented by Ag, and the set of each antigen is expressed by AG. The set of antibodes is started in solution space at random locations. Each antibody is represented by Ab, and the set of antibodes is expressed by AB. The affinity of any i. Antibody is calculated by two factors given in (2) and (3):

$$D_s = \sum_{j \in \{d\}} \frac{1}{d_{ij} + 1}, \tag{2}$$

$$Y_s = \sum_{j \in \{y\}} \frac{1}{d_{ij} + 1}. \tag{3}$$

The set of d indicates the number of antigens that are correctly classified by i antibody in the previous equations. The set of misclassified samples with the same antibody is expressed with y in the second equation. According to these two equations, the affinity account is maintained as (4)

$$\text{ab} \cdot \text{af} = \begin{cases} \dfrac{D_s}{N} + 2.0, & \text{If } \{d\} \neq 0 \text{ ve } \{y\} = 0, \\ \dfrac{D_s - Y_s}{D_s + Y_s} + 1.0, & \text{If } \{d\} \neq 0 \text{ ve } \{y\} \neq 0, \\ 0, & \text{If } \{d\} = 0 \text{ ve } \{y\} = 0. \end{cases} \tag{4}$$

Each antigen is recognized by an antibody. Each connection of an antibody with antigens is updated as (5)

$$w_{ij}(t+1) = w_{ij}(t) + \alpha\left(r_{ij}(t+1) - w_{ij}(t)\right). \tag{5}$$

In (5), α indicates learning constant and $r_{ij}(t)$ indicates the award obtained in step t. The affinity value is used as an award here. Learning constant α is determined in $[0, 1]$. If an antibody recognizes an antigen, the antibody memory cell turns into (B cell). The weight value between the antibody turned into the cell and B cell, M_k is 1. This value determines the excitation level of related B cell. The antibodies transformed memory cell before is updated according to (6)

$$M_k = \beta M_k. \tag{6}$$

β is the reduction factor used in (6) and gets value in $[0, 1]$. The clones of antibodies are generated proportional with their affinity. The clone are not created for antibodies whose affinities are over 2.0. The clones are formed as (7) for antibodies whose affinities are under 2.0

$$N_c = \sum_{k=1}^{n} \text{round}\left(c_{\text{rate}} * \text{ab}_k \cdot \text{aff}\right). \tag{7}$$

The mutation is applied on the clones obtained in (7) according to affinity values. The mutation process is maintained as claimed in (8)

$$\text{Ab}_c(i) = \text{Ab}_c(i) + \text{rand}(-s, s), \quad \text{if } d < \text{mutate}_{\text{rate}}. \tag{8}$$

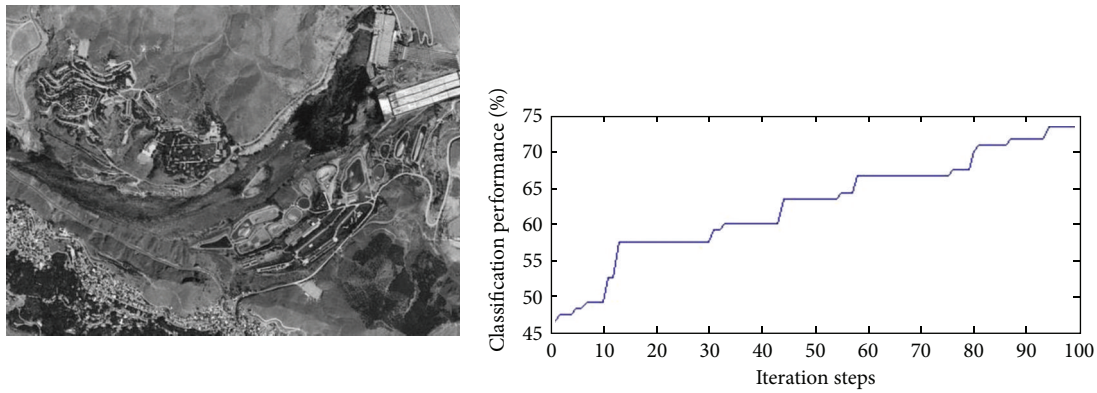


Figure 4: (a) Example remote image (400 × 400 pixel). (b) Performance graphic.

(d) number is a random number generated in $[0, 1]$. If this number is lower than mutation possibility, the mutation is applied to this antibody gene. The best mutated clone of an antibody is selected as a memory cell. According to the obtained memory cells, the performed classification process depending on the nearest neighbor algorithm is given in (9):

$$\text{Classification performance} = \frac{\text{Successful classification samples}}{\text{Total samples}} * 100. \quad (9)$$

4. Experimental Results

The proposed reinforcement learning based artificial immune classifier has been implemented using software written in Matlab m-file and hardware in FPGA. The Iris data and remote image data were used to evaluate performances of the proposed approach and other artificial immune classifiers.

Iteration steps: firstly, performance of the proposed approach has been tested using the iris data. The iris controled the diameter and size of the pupil is a thin and circular structure in the eye. 768 × 4 matrix obtained by processing of iris images is used for classification and results is compared other artificial immune classification methods. In this process, data is divided into 10 parts, 9 parts are used for training, and one part is also used for testing of algorithm. Performance of algorithm is obtained by average of results computed with 10 test processes. The classification performance of the proposed approach for iris data is shown in Figure 3. The comparative results between the proposed approach and other artificial immune classifiers have been given in Table 1. Conventional artificial immune classifiers have large classification time for large data size. In addition, they use many memory cells for classification process. As shown in Table 1, the proposed approach has more effective classification performance and less memory cells according to other methods.

Secondly, performance of the proposed approach has been tested using the remote image data. 400 × 400 pixel color image has been used for this purpose. If image has plant, water, building, and ways areas, performance and effectiveness of the proposed approach can be well presented. Figure 4 and Table 2 show an image used for experiments and comparative classification performance of the proposed approach.

Table 1: Comparative results for the proposed approach.

Classification methods	Iris	Wine	Sonar
	Classification performances		
Proposed classifier	**97.33 ± 3.06**	**97.6 ± 3.98**	**92.17 ± 2.92**
[5]	96.3 ± 4.7	96.1 ± 4.7	—
[3]	94.8	—	89.1
[2]	95.7	—	84.9
	Number of memory cells		
Proposed classifier	**21**	**30**	**60**
[5]	100	100	—
[3]	28	—	71
[2]	32	—	177

5. Conclusions

Artificial immune systems have an important role for classification methods that are commonly used in real-world applications. However, there are many approaches to improve the performance of artificial immune systems in the literature. For this purpose, supervised and unsupervised learning algorithms are used in many studies. But artificial immune systems that used supervised or unsupervised learning algorithms have many disadvantages in terms of today's data features. Therefore, a new artificial immune classifier based on reinforcement learning has been proposed for classification applications in this study. The proposed approach that used reinforcement learning algorithm for computation of memory cells of artificial immune system aimed to obtain a new classifier that is faster in real time, has less memory cells, has higher accuracy in results, and more effectively. Performance and effectiveness of the proposed approach has been shown using some benchmark data and

Table 2: Comparison of results for the proposed approach.

Class	Samples	Award function values	[5]	[1]	Proposed approach
Water	950	00111100, 0010111, 00011000	97%	96%	**99%**
Plant	350	00110000, 00111000, 010110	82%	78%	**84%**
Road	250	00000011, 00010110, 00000110	85%	—	**88%**
Building	300	10100110, 0110011, 0101111	80%	82%	**85%**

remote image data. Comparative results between proposed approach and some methods in the literature have been given with experimental results. The reinforcement learning based artificial immune classifier provides higher accuracy with less memory cells as seen in the experimental results.

References

[1] D. Landgrebe, "Hyperspectral image data analysis," *IEEE Signal Processing Magazine*, vol. 19, no. 1, pp. 17–28, 2002.

[2] Y. Zhong, L. Zhang, B. Huang, and P. Li, "An unsupervised artificial immune classifier for multi/hyperspectral remote sensing imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 44, no. 2, pp. 420–431, 2006.

[3] Y. Zhong, L. Zhang, J. Gong, and P. Li, "A supervised artificial immune classifier for remote-sensing imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 12, pp. 3957–3966, 2007.

[4] S. Arunprasath, S. Chandrasekar, K. Venkatalakshmi, and S. M. Shalinie, "Classification of remote sensed image using rapid genetic k-means algorithm," in *Proceedings of the IEEE International Conference on Communication Control and Computing Technologies (ICCCCT '10)*, pp. 677–682, October 2010.

[5] Y. Zhong and L. Zhang, "An adaptive artificial immune network for supervised classification of multi-/hyperspectral remote sensing imagery," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 3, pp. 894–909, 2012.

[6] X. H. Quah, C. Quek, and G. Leedham, "Pattern classification using fuzzy adaptive learning control network and reinforcement learning," *International Conference on Neural Information Processing*, vol. 3, pp. 1439–1443, 2002.

[7] I. Aydin, M. Karakose, and E. Akin, "An adaptive artificial immune system for fault classification," *Journal of Intelligent Manufacturing*, vol. 23, pp. 1489–1499, 2012.

[8] J. Shen, G. Gu, and H. Liu, "Automatic option generation in hierarchical reinforcement learning via immune clustering," in *Proceedings of the 1st International Symposium on Systems and Control in Aerospace and Astronautics*, pp. 497–500, January 2006.

[9] W.-C. Wong, S.-Y. Cho, and C. Quek, "R-POPTVR: a novel reinforcement-based POPTVR fuzzy neural network for pattern classification," *IEEE Transactions on Neural Networks*, vol. 20, no. 11, pp. 1740–1755, 2009.

[10] R. Ramos Da Silva and R. A. F. Romero, "Relational reinforcement learning and recurrent neural network with state classification to solve joint attention," in *Proceedings of the International Joint Conference on Neural Network (IJCNN '11)*, pp. 1222–1229, August 2011.

[11] M. A. Wiering, H. van Hasselt, A.-D. Pietersma, and L. Schomaker, "Reinforcement learning algorithms for solving classification problems," in *Proceedings of the IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL '11)*, pp. 91–96, April 2011.

[12] I. Aydin, M. Karakose, and E. Akin, "Artificial immune classifier with swarm learning," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 8, pp. 1291–1302, 2010.

[13] R. Zhang, T. Li, X. Xiao, and Y. Shi, "A danger-theory-based immune network optimization algorithm," *The Scientific World Journal*, vol. 2013, Article ID 810320, 13 pages, 2013.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 969734, 12 pages
http://dx.doi.org/10.1155/2013/969734

Research Article

An Adaptive Cauchy Differential Evolution Algorithm for Global Numerical Optimization

Tae Jong Choi,[1] Chang Wook Ahn,[1] and Jinung An[2]

[1] *Department of Computer Engineering, Sungkyunkwan University (SKKU), 2066 Seobu-ro, Suwon 440-746, Republic of Korea*
[2] *Robot Research Division, Daegu Gyeongbuk Institute of Science and Technology (DGIST), 50-1 Sang-ri, Hyeonpung-meyeon, Daegu 711-873, Republic of Korea*

Correspondence should be addressed to Chang Wook Ahn; cwan@skku.edu

Received 3 May 2013; Accepted 6 June 2013

Academic Editors: P. Agarwal, S. Balochian, V. Bhatnagar, J. Yan, and Y. Zhang

Adaptation of control parameters, such as scaling factor (F), crossover rate (CR), and population size (NP), appropriately is one of the major problems of Differential Evolution (DE) literature. Well-designed adaptive or self-adaptive parameter control method can highly improve the performance of DE. Although there are many suggestions for adapting the control parameters, it is still a challenging task to properly adapt the control parameters for problem. In this paper, we present an adaptive parameter control DE algorithm. In the proposed algorithm, each individual has its own control parameters. The control parameters of each individual are adapted based on the average parameter value of successfully evolved individuals' parameter values by using the Cauchy distribution. Through this, the control parameters of each individual are assigned either near the average parameter value or far from that of the average parameter value which might be better parameter value for next generation. The experimental results show that the proposed algorithm is more robust than the standard DE algorithm and several state-of-the-art adaptive DE algorithms in solving various unimodal and multimodal problems.

1. Introduction

Differential Evolution (DE) is a powerful population based search technique for optimizing problems. Many researchers have used the DE in practical fields because this technique has good convergence properties and is easy to apply [1]. The DE has three control parameters such as scaling factor (F), crossover rate (CR), and population size (NP). The performance of DE is largely influenced by what parameter values are assigned to these control parameters. Therefore, in order to have a good optimization performance, finding suitable parameter values is of crucial importance [2, 3]. In the DE, the control parameters are usually adjusted by using the trial-and-error search method. However, the value assigned by the trial-and-error method might be efficient for solving one type of problems and inefficient for solving other problems [4]. Moreover, it requires a lot of computational resources. As a solution of this problem, parameter adaptation has been utilized. According to Eiben et al. [5, 6], the parameter adaptation can be categorized into three classes as follows.

(1) Deterministic parameter control: the control parameters are adapted by some deterministic rule.

(2) Adaptive parameter control: the control parameters are adapted by some form of feedback from evolutionary search.

(3) Self-adaptive parameter control: the control parameters are adapted by the evolution-of-evolution technique. The control parameters are encapsulated in each individual as additional chromosomes and undergo evolutionary procedure.

The well designed adaptive or self-adaptive parameter control method can improve the performance of DE. Therefore, the adaptive and self-adaptive parameter controls are more applicable than the trial-and-error search method. So far, many adaptive and self-adaptive DE algorithms have been proposed and they have shown that the adaptive and self-adaptive DE algorithms have more robust performance than standard DE algorithm for many benchmark functions

[4, 7, 8]. Although there are many suggestions for adapting control parameters, it is still a challenging task to properly adapt the control parameters for problem. Based on various experiments, we found out that the parameter adaptation should be performed in every generation and the control parameters of each individual should be adapted based on the average parameter value of successfully evolved individuals' parameter values by using the Cauchy distribution.

The Cauchy distribution is one of the long tail distributions. The Cauchy distribution generates large step from peak location with higher probability. Many evolutionary algorithms have used this long tail property as an escaping local minima method. The proposed algorithm also, but in different manner, utilizes the Cauchy distribution for the parameter adaptation. In the proposed algorithm, each individual has its own control parameters. The control parameters of each individual are adapted based on the average parameter value of successfully evolved individuals' parameter values by using the Cauchy distribution. It is because the successfully evolved individuals are led by appropriate parameter values. That is to say, the appropriate parameter values make the individuals take the good region for solving problem. However, there is a possibility that the current appropriate parameter values might be inappropriate parameter values in next generation. Therefore, we cannot assure that the parameter adaptation based on the average parameter value is good for making well-suited parameter values for future generations. In view of the above considerations, the parameter adaptation of proposed algorithm utilizes the Cauchy distribution as a large step method. According to it, the control parameters of each individual are assigned either near the average parameter value or far from that of the average parameter value which might be better parameter value for next generation. The experimental results show that the proposed algorithm is more robust than standard DE algorithm and some adaptive and self-adaptive DE algorithms such as jDE [4], SaDE [7], and MDE [9] on solving multimodal problems as well as unimodal problems.

The rest of this paper proceeds as follows. In Section 2, we introduce basic operations of standard DE algorithm and some adaptive and self-adaptive parameter control DE algorithms. In Section 3, the Cauchy distribution is described. The proposed algorithm is explained in detail in Section 4. Section 5 presents the experimental results. We conclude this paper in Section 6.

2. Related Work

2.1. DE Algorithm. In the DE, a parent vector is called "target vector," a mutant vector is that generated by mixing donor vectors, and an offspring obtained by making crossover between target vector and mutant vector is called "trial vector." A target vector generates a trial vector which is moved around in search space by using the mutation and the crossover operations. If the fitness value of the trial vector is better than or equal to the fitness value of the target vector, the trial vector is accepted and included in the population of next generation, otherwise it is discarded and the target vector remains for the next generation. This cycle of operations is repeated until some specific termination conditions are not satisfied.

2.1.1. Initialization. The population of the DE consists of NP individuals. Each individual is a D-dimensional parameter vectors, denoted as $X_{i,G} = x^1_{i,G}, \ldots, x^D_{i,G}$ where $i = 1, \ldots, \mathrm{NP}$. In the initialization stage, first of all, the DE designates the search space of the test problem by prescribing the minimum ($X_{\min} = x^1_{\min}, \ldots, x^D_{\min}$) and the maximum ($X_{\max} = x^1_{\max}, \ldots, x^D_{\max}$) parameter bounds. After that, the parameter vectors of the each individual are initialized as follows:

$$x_{i,j,0} = x_{j,\mathrm{MIN}} + \mathrm{rand}\,[0,1] \cdot \left(x_{j,\mathrm{MAX}} - x_{j,\mathrm{MIN}}\right), \quad (1)$$

where rand$[0, 1]$ is the uniform distributed random number lying between 0 and 1. By doing this, all the individuals are randomly scattered in the search space. After initialization, the DE executes a loop of the operations: mutation, crossover, and selection.

2.1.2. Mutation Operation. The mutation is the first operation to generate child individuals from their parent individuals. So far, a lot of mutation strategies have been proposed. Here, we explain an example, called DE/rand/1/bin, which was introduced by Storn and Price [1]. First of all, the mutation strategy randomly select three mutually exclusive individuals among $[1, \mathrm{NP}]$. They are called the "donor vectors", denoted as $X_{r_1,G}$, $X_{r_2,G}$, and $X_{r_3,G}$. A mutant vector $V_{i,G}$ is generated by adding the scaling difference of $X_{r_2,G}$ and $X_{r_3,G}$ to $X_{r_1,G}$.

$$V_{i,G} = X_{r_1,G} + F \cdot \left(X_{r_2,G} - X_{r_3,G}\right), \quad (2)$$

where F is a scaling factor for amplifying the difference value between $X_{r_2,G}$ and $X_{r_3,G}$.

2.1.3. Crossover Operation. The crossover generates the trial vectors by making a crossover between target vector and mutant vector. There are two crossover operations which are commonly used. They are the binomial and the exponential crossovers. Here, we describe the binomial crossover. At the beginning, a random number is selected. If the random number is less than or equal to the crossover rate CR, the first element of the trial vector is occupied by the first element of the mutant vector. Otherwise, the element is occupied by the target vector. This procedure is repeated D times for each individual. The trial vector $U_{i,G} = u^1_{i,G}, \ldots, u^D_{i,G}$ is generated as follows:

$$u_{i,j,G} = \begin{cases} v_{i,j,G} & \text{if } \left(\mathrm{rand}\,[0,1) \le \mathrm{CR} \text{ or } j = j_{\mathrm{rand}}\right) \\ x_{i,j,G} & \text{otherwise.} \end{cases} \quad (3)$$

Prior to crossover, the DE select another random number j_{rand} lying between $[1, D]$. The random number is used to guarantee that at least one element of the trial vector is occupied by the mutant vector.

2.1.4. Selection Operation. The selection is the last operation of the DE iterations. It compares the fitness value of the target and the trial vectors. If the fitness value of the trial vector is better than or equal to the fitness value of the target vector, the trial vector is accepted and forms part of the population, otherwise it is discarded and the target vector remains for the next generation. these procedures are formulated as follows:

$$X_{i,G+1} = \begin{cases} U_{i,G} & \text{if } \left(f\left(U_{i,G}\right) \le f\left(X_{i,G}\right)\right), \\ X_{i,G} & \text{otherwise.} \end{cases} \quad (4)$$

2.2. jDE. Brest et al. [4] proposed a self-adaptive parameter control DE (called jDE) based on DE/rand/1/bin. In jDE, the control parameters, F and CR, are encapsulated in each individual as additional chromosomes. Therefore, all individuals have their own control parameters, denoted as F_i and CR_i. jDE utilizes four additional parameters: τ_1, τ_2, F_l, and F_u. The first two parameters are used to determine whether the control parameters need to be updated or not and the last two parameters are used to designate the range of the control parameter F_i. At the beginning, the values of F_i and CR_i are initialized by 0.5 and 0.9, respectively. Then, the control parameters F_i and CR_i for next generation are adapted as follows:

$$F_{i,G+1} = \begin{cases} F_l + \mathrm{rand}_2\left[0,1\right] \cdot F_u & \text{if } \left(\mathrm{rand}_1\left[0,1\right] < \tau_1\right), \\ F_{i,G} & \text{otherwise,} \end{cases}$$
$$\mathrm{CR}_{i,G+1} = \begin{cases} \mathrm{rand}_4\left[0,1\right] & \text{if } \left(\mathrm{rand}_3\left[0,1\right] < \tau_2\right), \\ \mathrm{CR}_{i,G} & \text{otherwise.} \end{cases} \quad (5)$$

The author used $\tau_1 = 0.1$, $\tau_2 = 0.1$, $F_l = 0.1$, and $F_u = 0.9$. This procedure is executed before applying the mutation operation. Therefore, newly generated control parameters affect the mutation and the crossover operations.

2.3. DESAP. Teo [10] proposed the first self-adaptive population size DE (called DESAP) based on the self-adaptive Pareto DE [11]. DESAP can self-adapt not only the scaling factor F and the crossover rate CR but also the population size NP. This algorithm utilizes additional parameters such as η_i, δ_i, and π_i. These parameters are encapsulated in each individual as additional chromosomes and also participated in the mutation and the crossover operations for evolving itself. The newly generated control parameters are selected when the fitness value of the trial vector is lower than or equal to the fitness value of the target vector. DESAP is divided into two algorithms (i.e., DESAP-abs and DESAP-rel) according to the equation of the population size for the next generation. DESAP has shown the effectiveness of the self-adaptive population size technique.

2.4. JADE. Zhang and Sanderson [12, 13] proposed a new mutation strategy called DE/current-to-$p$best which is lower greedy than DE/current-to-best/1. This strategy utilizes not the best individual of the population but the randomly selected one from the top $100p\%$ ($p \in (0,1]$) individuals. In addition, an external archive scheme was proposed by storing the set of parameter vectors of recently discarded individuals. These parameter vectors provide the additional information about promising progress direction and increase the population diversity. The following equations represent the DE/current-to-$p$best with and without the external archive strategy:

(1) DE/current-to-$p$best with archive:

$$V_{i,G} = X_{i,G} + F_i \cdot \left(X^p_{\mathrm{best},G} - X_{i,G}\right) + F_i \cdot \left(X_{r1,G} - \widetilde{X}_{r2,G}\right), \quad (6)$$

(2) DE/current-to-$p$best without archive:

$$V_{i,G} = X_{i,G} + F_i \cdot \left(X^p_{\mathrm{best},G} - X_{i,G}\right) + F_i \cdot \left(X_{r1,G} - X_{r2,G}\right), \quad (7)$$

where $\widetilde{X}_{r2,G}$ is an individual randomly selected from the population or external archive.

In terms of parameter adaptation, JADE adapts the crossover rate CR_i, as follows:

$$\mathrm{CR}_i = \mathrm{rnd}\, n_i\left(\mu_{\mathrm{CR}}, 0.1\right), \quad (8)$$

where $\mathrm{rnd}\, n_i$ is the Gaussian distributed random number generator. After that, the crossover rate CR_i is truncated to $[0,1]$. Moreover, μ_{CR} that is a mean value to generate CR_i is modified as follows:

$$\mu_{\mathrm{CR}} = (1-c) \cdot \mu_{\mathrm{CR}} + c \cdot \mathrm{mean}_A\left(S_{\mathrm{CR}}\right), \quad (9)$$

where c is a constant value in $[0,1]$, mean_A stands for the arithmetic mean, and S_{CR} contains the successfully evolved crossover rates of individuals after the selection operation. Similarly, the scaling factor F_i is adapted as follows:

$$F_i = \mathrm{rnd}\, c_i\left(\mu_F, 0.1\right), \quad (10)$$

where $\mathrm{rnd}\, c_i$ is the Cauchy distributed random number generator. After that, the scaling factor F_i is truncated to 1 if $F_i \ge 1$ or regenerated if $F_i \le 0$. Also, μ_F that is a mean value to generate F_i is modified as follows:

$$\mu_F = (1-c) \cdot \mu_F + c \cdot \mathrm{mean}_L\left(S_F\right), \quad (11)$$

where c is a constant value in $[0,1]$, mean_L stands for the Lehmer mean, and S_F contains the successfully evolved scaling factors of individuals after the selection operation.

2.5. MDE. Ali and Pant [9] proposed a Modified Differential Evolution (MDE). This algorithm utilizes the Cauchy distribution as another mutation operation. In the selection operation, all individuals are monitored and the results of the selection operation are stored in the failure counter. If some individuals consequently fail to be selected as an individual for the next generation over MFC (Maximum Failure Counter), MDE assumes that these individuals were felled into some local minima. Therefore, the algorithm applies the Cauchy distributed mutation to these individuals instead of the mutation and the crossover operations to escape the local minima. After that, the failure counter is initialized by 0. MDE has shown the good performance for the higher dimensional problems, compared with DE/rand/1/bin.

3. Analysis of the Cauchy Distribution

The Cauchy distribution is a continuous probability distribution and it has two parameters x_0 and γ. x_0 is the peak location of the distribution and γ stands for the halfwidth at halfmaximum (HWHM) of the distribution. The value of γ determines the shape of the Cauchy distribution. If γ is assigned a lower value, the peak of the probability density function would be higher and its width would be narrower. On the other hand, if γ is assigned a higher value, the probability density function would have a lower peak and a wider width. The Cauchy distribution generates a large step from the peak with a higher probability. In general, many evolutionary algorithms have used this long tail property as an escaping local minima technique. The probability density function and the cumulative distribution function of the Cauchy distribution are defined by

$$f(x; x_0, \gamma) = \frac{1}{\pi\gamma\left[1 + \left((x - x_0)/\gamma\right)^2\right]} = \frac{1}{\pi}\left[\frac{\gamma}{(x - x_0)^2 + \gamma^2}\right], \quad (12)$$

$$F(x; x_0, \gamma) = \frac{1}{\pi}\arctan\left((x - x_0)/\gamma\right) + \frac{1}{2}.$$

Figure 1 illustrates the various probability density functions of the Cauchy distribution. Here, L and S denote the location (x_0) and the scaling factor (γ), respectively. In addition, $L = 0$ and $S = 1$ generate the standard Cauchy distribution.

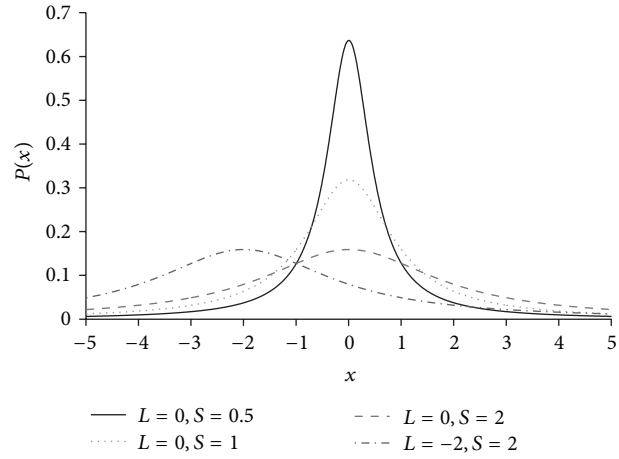


Figure 1: The various probability density functions of the Cauchy distribution.

4. Adaptive Cauchy DE

4.1. When Parameter Adaptation Should Be Performed? Finding appropriate moments of adapting control parameters is important problem for improving the DE performance. In this section, we explain when parameter adaptation should be performed.

Looking for previous studies, jDE utilizes self-adaptive method which allows each individual to maintain suitable control parameter values by itself. However, the parameter adaptation of jDE depends on the predefined probabilities (τ_1 and τ_2). Therefore, this method does not guarantee the adequacy of maintained control parameter values for current generation. In other words, it is possible that some individuals maintain unsuitable control parameter values. In SaDE, the scaling factor is calculated in every generation by using Gaussian distribution with the predefined mean value and all individuals utilize it. The crossover rate of SaDE, each individual has its own crossover rate and they are calculated by using Gaussian distribution with the median value of accumulated information about selection operation results during learning period as a mean value. This method is performed in every end of learning period. The parameter adaptation of SaDE has two problems. First, although each individual has different state during the DE iteration, the scaling factor adaptation of SaDE does not consider it. Therefore, many individuals might be utilized unsuitable control parameter values. Second, the selection operation results of past generations might become unnecessary or even noisy information for adapting the crossover rate. In addition, similar to jDE, during the learning period, it is possible that some individuals maintain unsuitable control parameter values.

Parameter adaptation should be performed whenever current control parameter values are not suitable for finding optimal value. We can utilize the selection operation results for distinguishing that an individual has suitable control parameters or not because the DE is based on the elitism. We find out that parameter adaptation should be performed in every generation. This means that every generation is appropriate moments of adapting control parameters. The reasons are follows.

(1) If an individual has good control parameter values, the child individual can succeed in the selection operation and it can locate better region for finding optimal value than the region of its parent individual. At this moment, the characteristic of child individual region might differ from the characteristic of its parent region. It means that there is a possibility of the existing of more suitable control parameter values than the previous control parameter values for new region. Therefore, although an individual succeeds in the selection operation, we should apply parameter adaptation for finding more suitable control parameter values.

(2) On the contrary, if an individual does not have good parameter values, the child individual might fail to evolve in the selection operation and then it remains the same region with its parent individual. This indicates that the individual needs more suitable control parameter values for escaping the region. Therefore, if an individual fails to evolve itself, we should also apply parameter adaptation.

Table 1: Benchmark functions used in the performance evaluation.

Benchmark function	Dim	Search space	Global min.
$F_1(x) = \sum_{i=1}^{D} x_i^2$	30	$[-100, 100]^D$	0
$F_2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	30	$[-10, 10]^D$	0
$F_3(x) = \max_i (\lvert x_i \rvert, 1 \le i \le D)$	30	$[-100, 100]^D$	0
$F_4(x) = \sum_{i=1}^{D} (\lfloor x_i + 0.5 \rfloor)^2$	30	$[-100, 100]^D$	0
$F_5(x) = \sum_{i=1}^{D} i x_i^4 + \text{random}[0, 1)$	30	$[-1.28, 1.28]^D$	0
$F_6(x) = \sum_{i=1}^{D} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30	$[-500, 500]^D$	−12569.5
$F_7(x) = \sum_{i=1}^{D} \left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$	30	$[-5.12, 5.12]^D$	0
$F_8(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right) + 20 + \exp(1)$	30	$[-32, 32]^D$	0
$F_9(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	$[-600, 600]^D$	0
$F_{10}(x) = \frac{\pi}{D}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_D - 1)^2\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$	30	$[-50, 50]^D$	0
$F_{11}(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2\left[1 + \sin^2(3\pi x_{i+1})\right] + (x_D - 1)^2\left[1 + \sin^2(2\pi x_D)\right]\right\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	30	$[-50, 50]^D$	0
$F_{12}(x) = f_{12}(x_n, x_1) + \sum_{i=1}^{D-1} f_{12}(x_i, x_{i+1})$ $f_{12}(x, y) = (x^2 + y^2)^{0.25}\left[\sin^2\left(50(x^2 + y^2)^{0.1}\right) + 1\right]$	30	$[-100, 100]^D$	0
$F_{13}(x) = \sum_{i=1}^{D-1}\left(x_i^2 + 2x_{i+1}^2 - 0.3\cos(3\pi x_i) - 0.4\cos(4\pi x_{i+1}) + 0.7\right)$	30	$[-15, 15]^D$	0
$F_{14}(x) = \sum_{i=1}^{D-1}\left(x_i^2 + x_{i+1}^2\right)^{0.25}\left[\sin^2\left(50\left(x_i^2 + x_{i+1}^2\right)^{0.1}\right) + 1\right]$	30	$[-100, 100]^D$	0

As a result, because the individuals of DE are evolved for exploring new regions, it is hard to assure that the previous suitable control parameter values are still suitable until satisfying some probabilities or during some periods. Therefore, parameter adaptation should be performed in every generation.

4.2. How Parameter Adaptation Should Be Performed? Finding proper method of adapting control parameters is also important problem for improving the DE performance. In this section, we explain how parameter adaptation should be performed. When performing parameter adaptation, we can utilize the successfully evolved individuals' control parameter values for parameter adaptation. It is because the successfully evolved individuals are led by good parameter values. That is to say, good parameter values make the individuals take the good region for solving problem.

Looking for previous studies, jDE adapts control parameter values by using the uniform distribution. However, the randomly generated control parameter values might not be suitable for finding better region. In SaDE, the successfully evolved individuals' crossover rates are stored in CR_Memory. After learning period, the parameter adaptation of SaDE extracts the median value from the CR_Memory as a mean value of the Gaussian distribution. In general, the median function is not largely influenced by outliers. However, the outliers give us the information about a new possibility of better control parameter values. Therefore, we should consider the outliers together.

As a result, the successfully evolved individuals' control parameter values based parameter adaptation is proper

Table 2: The experiment result of comparison of adaptive Cauchy DE with other DE algorithms.

	GEN	Adaptive Cauchy DE		DE/rand/1/bin		jDE		SaDE		MDE	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F_1	1500	**5.0E − 36**	**9.4E − 36**	7.9E − 14	6.8E − 14	2.6E − 28	4.0E − 28	1.8E − 20	2.3E − 20	7.0E − 17	2.8E − 17
F_2	2000	**2.4E − 30**	**1.6E − 30**	1.2E − 09	6.7E − 10	1.8E − 23	1.8E − 23	6.2E − 15	3.2E − 15	4.8E − 13	1.3E − 13
F_3	5000	2.9E − 12	1.1E − 12	3.3E − 02	7.2E − 02	**2.0E − 15**	**3.3E − 15**	7.8E − 10	2.0E − 10	2.0E − 08	8.5E − 09
F_4	1500	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00
F_5	3000	**3.0E − 03**	**6.9E − 04**	4.6E − 03	1.0E − 03	3.1E − 03	8.5E − 04	4.5E − 03	1.2E − 03	8.8E − 03	1.8E − 03
F_6	9000	**−12569.5**	**7.3E − 12**	−11095.3	5.2E + 02	**−12569.5**	**7.3E − 12**	**−12569.5**	**7.3E − 12**	−11482.1	3.0E + 02
F_7	5000	**0.0E + 00**	**0.0E + 00**	7.1E + 01	2.9E + 01	**0.0E + 00**	**0.0E + 00**	**0.0E + 00**	**0.0E + 00**	4.0E + 01	5.6E + 00
F_8	1500	**3.3E − 15**	**7.0E − 16**	9.2E − 08	4.0E − 08	8.2E − 15	2.3E − 15	5.3E − 11	3.7E − 11	4.0E − 09	9.2E − 10
F_9	2000	**0.0E + 00**	**0.0E + 00**	3.9E − 04	2.0E − 03	**0.0E + 00**	**0.0E + 00**	**0.0E + 00**	**0.0E + 00**	7.4E − 03	1.1E − 02
F_{10}	1500	**1.6E − 32**	**5.5E − 48**	6.5E − 15	5.6E − 15	7.0E − 30	7.2E − 30	3.3E − 20	4.2E − 20	9.9E − 18	1.5E − 17
F_{11}	1500	**1.3E − 32**	**1.1E − 47**	5.9E − 14	4.9E − 14	1.2E − 28	1.4E − 28	8.5E − 20	1.6E − 19	2.6E − 17	1.6E − 17
F_{12}	3000	**6.1E − 22**	**5.0E − 22**	2.1E − 07	1.3E − 07	6.3E − 17	7.0E − 17	3.5E − 12	2.9E − 12	7.4E − 11	3.9E − 11
F_{13}	1000	**5.0E − 20**	**3.4E − 20**	7.3E − 04	6.6E − 04	1.4E − 15	6.9E − 16	1.5E − 07	3.4E − 08	3.4E − 05	1.3E − 05
F_{14}	3000	**3.5E − 22**	**2.5E − 22**	2.3E − 05	1.7E − 05	7.5E − 17	1.4E − 16	1.8E − 10	2.2E − 10	1.2E − 08	3.7E − 09

Table 3: The success rate of comparison of adaptive Cauchy DE with other DE algorithms.

Success rate	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}
Adaptive Cauchy DE	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
DE/rand/1/bin	100%	100%	38%	100%	100%	0%	0%	100%	96%	100%	100%	100%	0%	18%
jDE	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
SaDE	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
MDE	100%	100%	100%	100%	72%	0%	0%	100%	58%	100%	100%	100%	0%	100%

Table 4: The experiment result of comparison of adaptive Cauchy DE with FEP and CEP.

	GEN	Adaptive Cauchy DE		DE/rand/1/bin		FEP		CEP	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std
F_1	1500	**5.0E − 36**	**9.4E − 36**	7.9E − 14	6.8E − 14	5.7E − 04	1.3E − 04	2.2E − 04	5.9E − 04
F_2	2000	**2.4E − 30**	**1.6E − 30**	1.2E − 09	6.7E − 10	8.1E − 03	7.7E − 04	2.6E − 03	1.7E − 04
F_3	5000	**2.9E − 12**	**1.1E − 12**	3.3E − 02	7.2E − 02	3.0E − 01	5.0E − 01	2.0E + 00	1.2E + 00
F_4	1500	**0.0E + 00**	**0.0E + 00**	**0.0E + 00**	**0.0E + 00**	**0.0E + 00**	**0.0E + 00**	5.8E + 02	1.1E + 03
F_5	3000	**3.0E − 03**	**6.9E − 04**	4.6E − 03	1.0E − 03	7.6E − 03	2.6E − 03	1.8E − 02	6.4E − 03
F_6	9000	**−12569.5**	**7.3E − 12**	−11095.3	5.2E + 02	−12554.5	5.3E + 01	−7917.1	6.3E + 02
F_7	5000	**0.0E + 00**	**0.0E + 00**	7.1E + 01	2.9E + 01	4.6E − 02	1.2E − 02	8.9E + 01	2.3E + 01
F_8	1500	**3.3E − 15**	**7.0E − 16**	9.2E − 08	4.0E − 08	1.8E − 02	2.1E − 03	9.2E + 00	2.8E + 00
F_9	2000	**0.0E + 00**	**0.0E + 00**	3.9E − 04	2.0E − 03	1.6E − 02	2.2E − 02	8.6E − 02	1.2E − 01
F_{10}	1500	**1.6E − 32**	**5.5E − 48**	6.5E − 15	5.6E − 15	9.2E − 06	3.6E − 06	1.8E + 00	2.4E + 00
F_{11}	1500	**1.3E − 32**	**1.1E − 47**	5.9E − 14	4.9E − 14	1.6E − 04	7.3E − 05	1.4E + 00	3.7E + 00

method to adapting control parameter values. This method should also consider the outliers.

4.3. The Proposed Algorithm. The proposed algorithm makes use of DE/rand/1/bin as a basic framework, in which the mutation is one of weaker greedy mutation strategies. In general, this mutation strategy is not so efficient in solving the unimodal problems since its lack of the fast convergence property makes the population slowly converge into the global minimum. However, if the control parameters are adapted suitably, this strategy can also demonstrate a good performance property in the unimodal and the multimodal problems.

The proposed algorithm adjusts two control parameters, F and CR, except for NP. The control parameter NP does not seriously affect the performance of DE more than the other two control parameters. Prior to explaining the adaptation procedures, the characteristics of these parameters are described. The control parameter F is related to the convergence speed of DE. Therefore, a higher value of F encourages the exploration power which is generally useful in the early stage of DE. On the other hand, a lower value of

Table 5: The experiment result of comparison of adaptive Cauchy DE with adaptive LEP and best Lévy.

	GEN	Adaptive Cauchy DE		DE/rand/1/bin		Adaptive LEP		Best Lévy	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std
F_1	1500	**4.7E − 36**	**5.1E − 36**	7.2E − 14	7.9E − 14	6.3E − 04	7.6E − 05	6.6E − 04	6.4E − 05
F_6	1500	**−12569.5**	**7.3E − 12**	−6506.71	6.7E + 02	−11469.2	5.8E + 01	−11898.2	5.2E + 01
F_7	1500	**0.0E + 00**	**0.0E + 00**	1.7E + 02	1.2E + 01	5.9E + 00	2.1E + 00	1.3E + 01	2.3E + 00
F_8	1500	**3.2E − 15**	**5.0E − 16**	9.1E − 08	3.7E − 08	1.9E − 02	1.0E − 03	3.1E − 02	2.0E − 03
F_9	1500	**0.0E + 00**	**0.0E + 00**	2.2E − 13	1.4E − 13	2.4E − 02	2.8E − 02	1.8E − 02	1.7E − 02
F_{10}	1500	**1.6E − 32**	**5.5E − 48**	7.5E − 15	7.2E − 15	6.0E − 06	1.0E − 06	3.0E − 05	4.0E − 06
F_{11}	1500	**1.3E − 32**	**1.1E − 47**	5.4E − 14	4.9E − 14	9.8E − 05	1.2E − 05	2.6E − 04	3.0E − 05

Table 6: The experiment result of comparison of various failure counters.

	GEN	$FC_F = 0$, $FC_{CR} = 0$		$FC_F = 0$, $FC_{CR} = 1$		$FC_F = 1$, $FC_{CR} = 0$		$FC_F = 1$, $FC_{CR} = 1$		$FC_F = 2$, $FC_{CR} = 2$	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
F_1	1500	5.0E − 36	9.4E − 36	4.0E − 31	5.2E − 31	**6.9E − 40**	**6.8E − 40**	2.1E − 35	4.1E − 35	1.4E − 30	2.0E − 30
F_2	2000	2.4E − 30	1.6E − 30	1.7E − 26	1.5E − 26	**8.7E − 34**	**5.2E − 34**	3.4E − 30	3.1E − 30	2.0E − 26	1.4E − 26
F_3	5000	**2.9E − 12**	**1.1E − 12**	1.9E − 04	3.0E − 05	2.5E + 00	2.9E + 00	4.7E − 01	9.7E − 01	1.1E + 00	1.5E + 00
F_4	1500	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00
F_5	3000	**3.0E − 03**	**6.9E − 04**	3.1E − 03	8.4E − 04	3.1E − 03	8.9E − 04	3.0E − 03	8.0E − 04	3.1E − 03	8.3E − 04
F_6	9000	**−12569.5**	**7.3E − 12**	**−12569.5**	**7.3E − 12**	**−12569.5**	**7.3E − 12**	**−12569.5**	**7.3E − 12**	−12567.1	1.7E + 01
F_7	5000	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00
F_8	1500	**3.3E − 15**	**7.0E − 16**	1.4E − 14	4.5E − 15	3.1E − 15	0.0E + 00	3.0E − 07	2.1E − 06	1.5E − 14	2.6E − 15
F_9	2000	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00
F_{10}	1500	**1.6E − 32**	**5.5E − 48**	9.6E − 32	6.6E − 32	4.8E − 07	3.4E − 06	1.6E − 32	3.6E − 34	1.7E − 31	1.6E − 31
F_{11}	1500	**1.3E − 32**	**1.1E − 47**	7.3E − 31	1.1E − 30	1.4E − 32	6.9E − 34	1.3E − 32	1.1E − 47	8.5E − 31	7.8E − 31
F_{12}	3000	6.1E − 22	5.0E − 22	1.6E − 18	1.8E − 18	**1.4E − 23**	**6.9E − 23**	2.0E − 21	2.1E − 21	1.4E − 18	1.4E − 18
F_{13}	1000	5.0E − 20	3.4E − 20	1.7E − 16	1.4E − 16	**5.8E − 23**	**3.3E − 23**	8.6E − 20	5.9E − 20	4.7E − 17	4.1E − 17
F_{14}	3000	3.5E − 22	2.5E − 22	7.2E − 19	7.1E − 19	**9.5E − 25**	**1.1E − 24**	8.0E − 22	8.9E − 22	8.3E − 19	6.6E − 19

F promotes the exploitation power that is usually desirable in the later stage of DE. Moreover, the value of control parameter CR is related to the diversity of population.

The parameter adaptation of proposed algorithm utilizes F_Memory and CR_Memory. The successfully evolved individuals' scaling factors and crossover rates are stored in these memories. When performing parameter adaptation, arithmetic mean function is applied to extract mean values and these are actual parameter values of the Cauchy distribution as location parameters. The Cauchy distribution is one of the long tail distributions. The Cauchy distribution generates the large step from the peak location with higher probability. There is a possibility that the current appropriate parameter values might be the inappropriate parameter values in next generation. Therefore, we cannot assure that the average parameter value is still the well-suited parameter value for the future generations. In view of the above considerations, the parameter adaptation of the proposed algorithm utilizes the Cauchy distribution as a large step method. Through this, the control parameters of each individual are assigned either near the average parameter value or far from that of the average parameter value which might be the better parameter value of the next generation.

The details of the proposed algorithm are given as follows. First of all, all individuals have their own control parameters, F_i and CR_i where i is the individual's index. At the initialization stage, these parameters are initialized as 0.5 and 0.9, respectively. The mutation and crossover operations used in DE/rand/1/bin are employed. In the selection operation, if the trial vector is selected as an individual for the next generation, the control parameter values of this individual are stored in the F_Memory and CR_Memory. After the selection operation, the parameter adaptation is carried out.

The parameter F_i is adapted by the Cauchy distribution with the average parameter value. After that, the F_i is truncated to 0.1 or 1 if the F_i is less than 0.1 or greater than 1. The adaptation of the scaling factor is performed as follows:

$$F_{i,G+1} = C\left(0, \gamma_F\right) + F_{\text{avg},G}, \tag{13}$$

where $F_{\text{avg},G}$ is the average parameter value of the accumulated information in the F_Memory as the location parameter of the Cauchy distribution. The γ_F is scaling factor of the equation and is assigned 0.1.

Similarly, the CR_i is adapted by the Cauchy distribution with the average parameter value. After that, the CR_i is

truncated to 0 or 1 if the CR_i is less than 0 or greater than 1. The adaptation of the crossover rate is given as follows:

$$CR_{i,G+1} = C(0, \gamma_{CR}) + CR_{avg,G}, \quad (14)$$

where $CR_{avg,G}$ is the average parameter value of the accumulated information in the CR_Memory as the location parameter of the Cauchy distribution. The γ_F is scaling factor of the equation and is assigned 0.1. Algorithm 1 describes the pseudocode of the proposed algorithm.

When performing parameter adaptation, if there is no successfully evolved individual, then the average parameter values are assigned the average parameter values of last generation.

5. Performance Evaluation

5.1. Benchmark Functions. The performance of proposed algorithm was evaluated by fourteen benchmark functions. The first eleven benchmark functions are from [14, 15] and the rest benchmark functions are Extended f_{12} (F_{12}), Bohachevsky (F_{13}), and Schaffer (F_{14}). The functions are shown in Table 1.

The characteristics of the benchmark functions are described as follows: F_1–F_3 are continuous unimodal functions, F_4 is a discontinuous step function, F_5 is a noise quadratic function, and F_6–F_{14} are continuous multimodal functions that the number of local minima exponentially increases when their dimension grows. A more detailed description of each function is given in [14, 15].

5.2. Experiment Setup. The proposed algorithm was compared to standard DE algorithm and several state-of-art adaptive DE algorithms. The five algorithms in comparison are listed as follows:

(1) adaptive Cauchy DE;

(2) DE/rand/1/bin with $F = 0.5$ and $CR = 0.9$ [1, 4, 16, 17];

(3) jDE [4];

(4) SaDE [7];

(5) MDE with MFC = 5 [9].

All of the used parameter values are the recommended or utilized parameter values by their authors. The population size NP is fixed by 100 in all experiments. The maximum number of generations is assigned by 1500 for F_1, F_4, F_8, F_{10}, and F_{11}; 2000 for F_2 and F_9; 3000 for F_5, F_{12}, and F_{14}; 5000 for F_3 and F_7; 9000 for F_6; 1000 for F_{13}. All experiment results were run 50 times, independently. For clarity, the result of the best algorithm is marked in boldface. If the difference between the global minimum and the best fitness is lower than 10^{-5} (In F_5, 10^{-2}), we countered the experiment is successful.

5.3. Comparison of Adaptive Cauchy DE with Adaptive DE Algorithms. The mean and the standard deviation of experiment results obtained by adaptive Cauchy DE and the compared DE algorithms for F_1–F_{14} for $D = 30$ are summarized in Table 2.

The proposed algorithm shows better performance on solving the unimodal problems as well as in the multimodal problems except F_3 benchmark function. jDE shows the best performance in F_3 benchmark function. The proposed

```
/* Initialization */
Generate the initial population
Evaluate the initial population
FOR i = 0 to NP DO
  F_i = 0.5
  CR_i = 0.9
END FOR

WHILE The termination condition is not satisfied DO
  /* Mutation operation */
  FOR i = 0 to NP DO
    Generate a mutant vector V_{i,G}
    Randomly select three donor vectors X_{r1}, X_{r2}, X_{r3}
    V_{i,G} = X_{r1,G} + F_i · (X_{r2,G} − X_{r3,G})
  END FOR

  /* Crossover operation */
  FOR i = 0 to NP DO
    Generate a trial vector U_{i,G}
    Select a random number j_rand lying between [1, D]
    FOR j = 0 to D DO
      IF rand[0, 1) ≤ CR_i or j == j_rand THEN
        u_{i,j,G} = v_{i,j,G}
      ELSE
        u_{i,j,G} = x_{i,j,G}
      END IF
    END FOR
  END FOR

  /* Selection operation */
  k = 0
  FOR i = 0 to NP DO
    IF f(U_{i,G}) ≤ f(X_{i,G}) THEN
      X_{i,G+1} = U_{i,G}
      F_Memory[k] = F_i
      CR_Memory[k] = CR_i
      k+ = 1
    ELSE
      X_{i,G+1} = X_{i,G}
    END IF
  END FOR

  /* Parameter adaptation */
  IF k ≠ 0 THEN
    F_{avg,G} = mean(F_Memory, k)
    CR_{avg,G} = mean(CR_Memory, k)
  END IF
  FOR i = 0 to NP DO
    F_{i,G+1} = C(0, γ_F) + F_{avg,G}
    CR_{i,G+1} = C(0, γ_CR) + CR_{avg,G}
  END FOR
END WHILE
```

Algorithm 1: Adaptive Cauchy DE.

Table 7: The success rate of comparison of various failure counters.

Success rate	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}
$FC_F = 0$, $FC_{CR} = 0$	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
$FC_F = 0$, $FC_{CR} = 1$	100%	100%	0%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
$FC_F = 1$, $FC_{CR} = 0$	100%	100%	0%	100%	100%	100%	100%	100%	100%	98%	100%	100%	100%	100%
$FC_F = 1$, $FC_{CR} = 1$	100%	100%	0%	100%	100%	100%	100%	98%	100%	100%	100%	100%	100%	100%
$FC_F = 2$, $FC_{CR} = 2$	100%	100%	0%	100%	100%	98%	100%	100%	100%	100%	100%	100%	100%	100%

Table 8: The experiment result of comparison of various mathematical functions for utilizing success memories.

	GEN	Arithmetic mean		Median		Best individual		Itself	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std
F_1	1500	**5.0E − 36**	**9.4E − 36**	2.9E − 27	4.0E − 27	2.4E − 31	8.2E − 31	4.2E − 19	2.5E − 19
F_2	2000	**2.4E − 30**	**1.6E − 30**	5.5E − 23	4.4E − 23	8.1E − 28	2.4E − 27	2.9E − 16	1.2E − 16
F_3	5000	**2.9E − 12**	**1.1E − 12**	2.1E + 00	2.5E + 00	8.1E + 00	7.5E + 00	4.7E − 04	3.3E − 03
F_4	1500	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00
F_5	3000	**3.0E − 03**	**6.9E − 04**	4.6E − 03	1.2E − 03	4.1E − 03	1.4E − 03	4.4E − 03	1.1E − 03
F_6	9000	**−12569.5**	**7.3E − 12**	**−12569.5**	**7.3E − 12**	**−12569.5**	**7.3E − 12**	−12567.1	1.7E + 01
F_7	5000	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00	0.0E + 00
F_8	1500	**3.3E − 15**	**7.0E − 16**	3.0E − 14	8.4E − 15	7.4E − 15	2.7E − 14	1.7E − 10	6.7E − 11
F_9	2000	**0.0E + 00**	**0.0E + 00**	0.0E + 00	0.0E + 00	3.5E − 04	1.7E − 03	3.9E − 04	2.0E − 03
F_{10}	1500	**1.6E − 32**	**5.5E − 48**	4.8E − 29	4.1E − 29	5.0E + 00	3.5E + 01	1.2E − 20	1.2E − 20
F_{11}	1500	**1.3E − 32**	**1.1E − 47**	8.0E − 28	1.1E − 27	1.1E − 02	7.5E − 02	1.4E − 19	1.2E − 19
F_{12}	3000	**6.1E − 22**	**5.0E − 22**	1.1E − 16	9.2E − 17	4.1E + 03	2.7E + 04	4.0E − 11	2.8E − 11
F_{13}	1000	**5.0E − 20**	**3.4E − 20**	2.3E − 15	8.7E − 16	1.3E − 02	3.5E − 02	1.2E − 10	5.1E − 11
F_{14}	3000	**3.5E − 22**	**2.5E − 22**	1.5E − 15	1.8E − 15	2.3E − 01	5.3E − 01	1.1E − 09	6.6E − 10

algorithm outperformed all multimodal problems. The second best algorithm is jDE. Although SaDE utilizes strategy adaptation as well as parameter adaptation, jDE shows better results than SaDE in all benchmark functions. It means that parameter adaptation is more important to improve the performance of DE. MDE shows better performance than DE/rand/1/bin in several unimodal and multimodal problems.

Table 3 shows the success rate of comparison results. The success rate is obtained by a mount of successful counter divided by a mount of experiment runs (50). The proposed algorithm and two adaptive DE algorithms (jDE and SaDE) show perfect success rates. However, DE/rand/1/bin and MDE show lower success rate than the proposed algorithm and they totally failed to find global optimum in several benchmark functions.

Figure 2 shows the average best graphs of adaptive Cauchy DE and the compared DE algorithms.

5.4. Comparison of Adaptive Cauchy DE and FEP and CEP. The mean deviation and the standard deviation of experiment results obtained by adaptive Cauchy DE, DE/rand/1/bin, FEP (Fast Evolutionary Programming), and CEP (Classic Evolutionary Programming) for F_1–F_{11} for $D = 30$ are summarized in Table 4. The results of FEP and CEP are taken from [13, Tables 2-4].

The proposed algorithm shows better performance on solving all benchmark functions than DE/rand/1/bin, FEP, and CEP. The second best algorithm is DE/rand/1/bin. However, DE/rand/1/bin shows lower performance than FEP in several benchmark functions (F_6 and F_7).

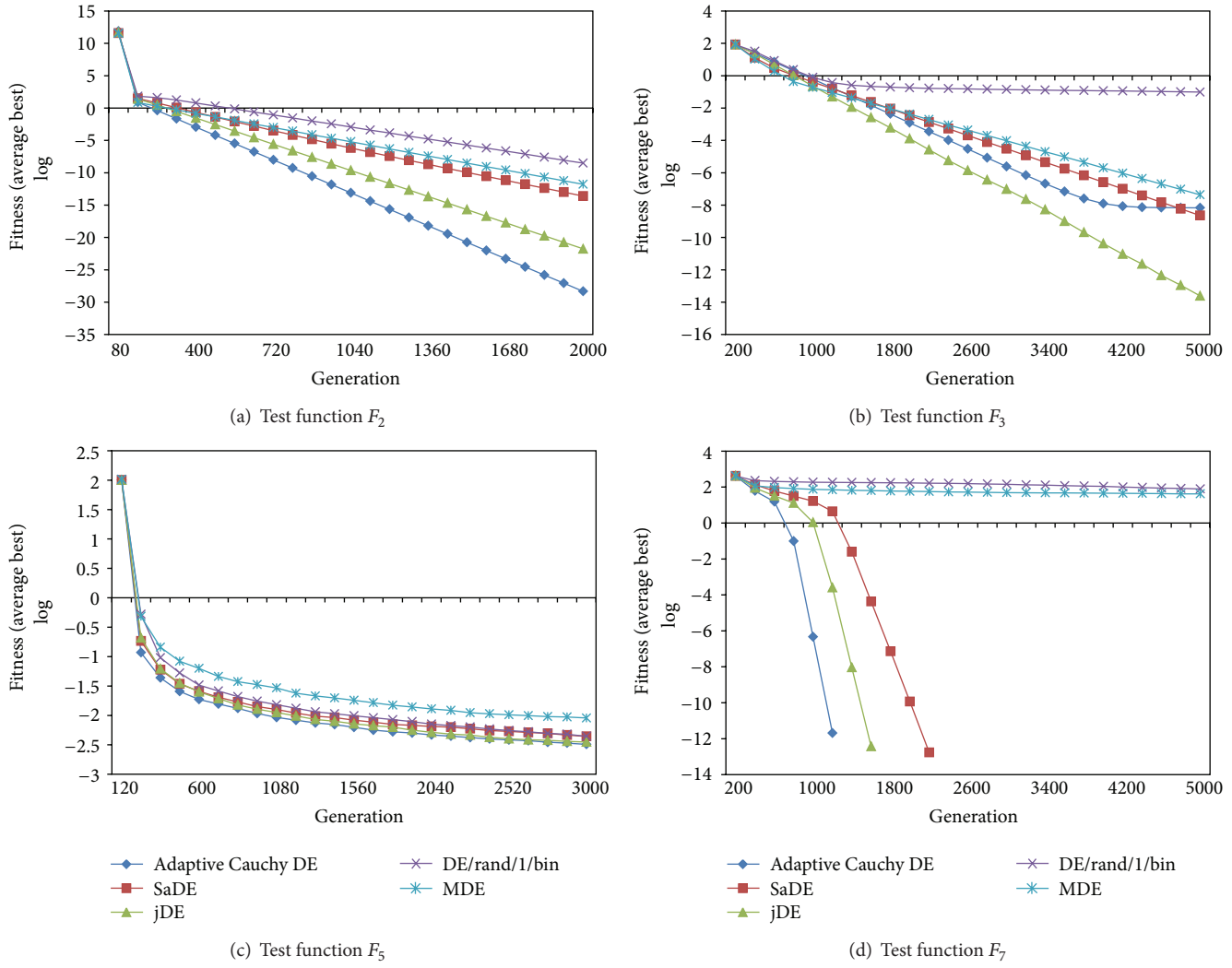
5.5. Comparison of Adaptive Cauchy DE and Adaptive LEP and Best Lévy. The mean deviation and the standard deviation of experiment results obtained by adaptive Cauchy DE, DE/rand/1/bin, adaptive LEP, and best Lévy for F_1 and F_6–F_{11} for $D = 30$ are summarized in Table 5. The results of adaptive LEP and best Lévy are taken from [18, Table 3]. The population size NP is fixed by 100 in all experiments. The maximum number of generations is assigned by 1500 for all benchmark functions.

The proposed algorithm shows better performance on solving all benchmark functions than DE/rand/1/bin, FEP, and CEP. The second best algorithm is DE/rand/1/bin again. However, DE/rand/1/bin shows lower performance than adaptive LEP and best Lévy in several benchmark functions (F_6 and F_7).

5.6. Parameter Study. Tables 6 and 7 show that various failure counter experiment results. The goal of this experiments is finding appropriate moments of parameter adaptation. $FC_F = n$ means if an individual fails to evolve itself consequently n

Table 9: The success rate of comparison of various mathematical functions for utilizing success memories.

Success rate	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}
Arithmetic mean	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
Median	100%	100%	0%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
Best individual	100%	100%	0%	100%	100%	100%	100%	100%	96%	90%	96%	92%	74%	82%
Itself	100%	100%	90%	100%	100%	98%	100%	100%	96%	100%	100%	100%	100%	100%



(a) Test function F_2

(b) Test function F_3

(c) Test function F_5

(d) Test function F_7

Figure 2: Average best graphs of Adaptive Cauchy DE with the compared DE algorithms.

times, the scaling factor of individual is adapted. Similarly, $FC_{CR} = n$ means if an individual fails to evolve itself consequently n times, the crossover rate of individual is adapted. For example, if $FC_F = 1$ and $FC_{CR} = 0$, then an individual's scaling factor is adapted when the individual fails to evolve itself, the last selection operation and the crossover of individual aer adapted in every generation.

The results show that adapting control parameters $FC_F = 0$ with $FC_{CR} = 0$ and $FC_F = 1$ with $FC_{CR} = 0$ had good performance in the comparison. However, when comparing success rate, $FC_F = 0$ with $FC_{CR} = 0$ had higher success rate than $FC_F = 1$ with $FC_{CR} = 0$ in F_3 benchmark function.

Note that when failure counter is increasing, the performance of algorithm decreased. Therefore, parameter adaptation should be performed in every generation. This is because the individuals of DE are evolved for exploring new regions. Therefore, it is hard to assure that the previous suitable control parameter values are still suitable until satisfying some probabilities or during some periods.

Tables 8 and 9 show the various parameter adaptation method experiment results. The goal of these experiments is finding proper method of utilizing F_Memory and CR_Memory for parameter adaptation. Arithmetic mean indicates that the proposed algorithm utilized arithmetic

Table 10: The experiment result of comparison of Cauchy distribution with Gaussian distribution for parameter adaptation.

	GEN	Cauchy $\gamma = 0.1$		Cauchy $\gamma = 0.3$		Gaussian Std = 0.1		Gaussian Std = 0.3	
		Mean	Std	Mean	Std	Mean	Std	Mean	Std
F_1	1500	**5.0E − 36**	**9.4E − 36**	$1.6E-28$	$1.2E-28$	$2.8E-32$	$3.4E-32$	$2.8E-30$	$2.8E-30$
F_2	2000	**2.4E − 30**	**1.6E − 30**	$3.1E-24$	$1.3E-24$	$1.9E-27$	$2.1E-27$	$7.4E-26$	$4.7E-26$
F_3	5000	**2.9E − 12**	**1.1E − 12**	$1.3E-02$	$6.7E-02$	$1.3E-01$	$5.6E-01$	$5.3E-12$	$9.0E-12$
F_4	1500	$0.0E+00$	$0.0E+00$	$0.0E+00$	$0.0E+00$	$0.0E+00$	$0.0E+00$	$0.0E+00$	$0.0E+00$
F_5	3000	**3.0E − 03**	**6.9E − 04**	$4.0E-03$	$1.3E-03$	$3.4E-03$	$1.6E-03$	$3.1E-03$	$8.4E-04$
F_6	9000	−12569.5	$7.3E-12$	−12569.5	$7.3E-12$	−12569.5	$7.3E-12$	−12569.5	$7.3E-12$
F_7	5000	$0.0E+00$	$0.0E+00$	$0.0E+00$	$0.0E+00$	$0.0E+00$	$0.0E+00$	$0.0E+00$	$0.0E+00$
F_8	1500	**3.3E − 15**	**7.0E − 16**	$8.7E-15$	$2.7E-15$	$4.2E-15$	$1.6E-15$	$6.3E-15$	$1.1E-15$
F_9	2000	**0.0E + 00**	**0.0E + 00**	$2.0E-04$	$1.4E-03$	**0.0E + 00**	**0.0E + 00**	**0.0E + 00**	**0.0E + 00**
F_{10}	1500	**1.6E − 32**	**5.5E − 48**	$6.2E-30$	$4.9E-30$	$1.9E-32$	$6.7E-33$	$2.9E-31$	$1.9E-31$
F_{11}	1500	**1.3E − 32**	**1.1E − 47**	$1.0E-28$	$9.4E-29$	$6.0E-32$	$7.6E-32$	$2.3E-30$	$2.8E-30$
F_{12}	3000	**6.1E − 22**	**5.0E − 22**	$3.7E-17$	$2.9E-17$	$1.5E-19$	$2.1E-19$	$3.5E-18$	$2.6E-18$
F_{13}	1000	**5.0E − 20**	**3.4E − 20**	$2.7E-16$	$1.1E-16$	$1.6E-17$	$8.0E-18$	$6.7E-17$	$2.8E-17$
F_{14}	3000	**3.5E − 22**	**2.5E − 22**	$4.4E-17$	$6.6E-17$	$2.1E-19$	$2.4E-19$	$2.9E-18$	$2.5E-18$

Table 11: The success rate of comparison Cauchy distribution with Gaussian distribution for parameter adaptation.

Success rate	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}
Cauchy $\gamma = 0.1$	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
Cauchy $\gamma = 0.3$	100%	100%	78%	100%	100%	100%	100%	100%	98%	100%	100%	100%	100%	100%
Gaussian Std = 0.1	100%	100%	72%	100%	98%	100%	100%	100%	100%	100%	100%	100%	100%	100%
Gaussian Std = 0.3	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%

mean function to extract mean values from F_Memory and CR_Memory and each individual's control parameter values are adapted based on the mean values as location parameters of the Cauchy distribution for parameter adaptation. Similarly, median indicates that the proposed algorithm utilized median function to extract median values from F_Memory and CR_Memory and each individual's control parameter values are adapted based on the median values as location parameters of the Cauchy distribution for parameter adaptation. Best individual indicates that each individual's control parameter values are adapted based on the best individual's control parameter values as location parameter of the Cauchy distribution. Finally, itself indicates that each individual's control parameter values are adapted based on its own control parameter values as location parameter of the Cauchy distribution.

The results show that adapting control parameters based on arithmetic mean function had good performance than median function. This is because the outliers give us the information about a new possibility of better control parameter values. Therefore, the arithmetic mean function is more applicable than median function. Parameter adaptation based on its own control parameter values shows good success rate in the comparison. However, the performance was lower than that of other methods. Parameter adaptation based on best individual's control parameter values shows good performance in only unimodal problems.

Tables 10 and 11 show the comparison results of the Cauchy distribution with the Gaussian distribution for parameter adapation method. The goal of these experiments is finding which distribution property (short or long tail) is more suitable for parameter adaptation. Cauchy $\gamma = 0.1$ indicates that parameter adaptation is performed based on the Cauchy distribution and the scaling parameter of distribution is assigned 0.1. Gaussian Std = 0.1 indicates that parameter adaptation is performed based on the Gaussian distribution and the standard deviation parameter of distribution is assigned 0.1.

The experiment results show that the Cauchy distribution with $\gamma = 0.1$ had good performance than others. This is because, the Cauchy distribution generates the large step from the peak location with higher probability. Therefore, the control parameters of each individual are assigned either near the average parameter value or far from that of the average parameter value which might be the better parameter value of the next generation.

6. Conclusion

The parameters of DE should be adequately assigned to attain better performance. But finding suitable values demands a lot of computational resources. In this sense, we present a new DE algorithm which utilizes success memories of scaling factors and crossover rates to properly adjust the control parameters of DE; the control parameters are adapted at each generation based on the Cauchy distribution with mean values of success memories. Experimental results showed that the adaptive Cauchy DE algorithm generally achieves

better performance than existing DE variants on various multimodal and unimodal test problems. The results also supported the claim that a long tail distribution is more reliable than a short tail distribution in adjusting the control parameters.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) Grant funded by the Korea government (MSIP) (no. 2012-013-735). This work was also supported by the DGIST R&D Program of the Ministry of Education, Science and Technology of Korea (13-BD-01).

References

[1] 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.

[2] R. Gamperle, S. D. Muller, and P. Koumoutsakos, "A parameter study for differential evolution," in *Proceedings of the International Conference on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation (WSEAS '02)*, pp. 293–298, 2002.

[3] J. Zhang and A. C. Sanderson, "An approximate Gaussian model of differential evolution with spherical fitness functions," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '07)*, pp. 2220–2228, September 2007.

[4] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[5] Á. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 124–141, 1999.

[6] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, Springer, Berlin, Germany, 2003.

[7] A. K. Qin and P. N. Suganthan, "Self-adaptive differential evolution algorithm for numerical optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '05)*, vol. 2, pp. 1785–1791, September 2005.

[8] J. Brest, B. Bošković, S. Greiner, V. Žumer, and M. S. Maučec, "Performance comparison of self-adaptive and adaptive differential evolution algorithms," *Soft Computing*, vol. 11, no. 7, pp. 617–629, 2007.

[9] M. Ali and M. Pant, "Improving the performance of differential evolution algorithm using Cauchy mutation," *Soft Computing*, vol. 15, no. 5, pp. 991–1007, 2011.

[10] J. Teo, "Exploring dynamic self-adaptive populations in differential evolution," *Soft Computing*, vol. 10, no. 8, pp. 673–686, 2006.

[11] H. A. Abbass, "The self-adaptive Pareto differential evolution algorithm," in *Proceedings of the Congress on Evolutionary Computation (CEC '02)*, pp. 831–836, May 2002.

[12] 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.

[13] J. Zhang and A. C. Sanderson, *Adaptive Differential Evolution: A Robust Approach to Multimodal Problem Optimization*, Springer, Berlin, Germany, 2009.

[14] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.

[15] X. Yao, Y. Liu, K. H. Liang, and G. Lin, "Fast evolutionary algorithms," in *Advances in Evolutionary Computing*, pp. 45–94, Springer, New York, NY, USA, 2003.

[16] E. Mezura-Montes, J. Velázquez-Reyes, and C. A. Coello Coello, "A comparative study of differential evolution variants for global optimization," in *Proceedings of the 8th Annual Genetic and Evolutionary Computation Conference (GECCO '06)*, pp. 485–492, July 2006.

[17] J. Vesterstrøm and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Proceedings of the Congress on Evolutionary Computation (CEC '04)*, vol. 2, pp. 1980–1987, June 2004.

[18] E. Cuevas, D. Zaldivar, and M. Pérez-Cisneros, "A novel multi-threshold segmentation approach based on differential evolution optimization," *Expert Systems with Applications*, vol. 37, no. 7, pp. 5265–5271, 2010.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 718345, 11 pages
http://dx.doi.org/10.1155/2013/718345

Research Article

Application of Particle Swarm Optimization Algorithm in the Heating System Planning Problem

Rong-Jiang Ma,[1] Nan-Yang Yu,[1] and Jun-Yi Hu[1,2]

[1] *School of Mechanical Engineering, Southwest Jiaotong University, Chengdu 610031, China*
[2] *CSR Qishuyan Institute Co., Ltd., Changzhou 213011, China*

Correspondence should be addressed to Rong-Jiang Ma; swjtumrj@139.com

Received 2 May 2013; Accepted 13 June 2013

Academic Editors: P. Agarwal, S. Balochian, and Y. Zhang

Based on the life cycle cost (LCC) approach, this paper presents an integral mathematical model and particle swarm optimization (PSO) algorithm for the heating system planning (HSP) problem. The proposed mathematical model minimizes the cost of heating system as the objective for a given life cycle time. For the particularity of HSP problem, the general particle swarm optimization algorithm was improved. An actual case study was calculated to check its feasibility in practical use. The results show that the improved particle swarm optimization (IPSO) algorithm can more preferably solve the HSP problem than PSO algorithm. Moreover, the results also present the potential to provide useful information when making decisions in the practical planning process. Therefore, it is believed that if this approach is applied correctly and in combination with other elements, it can become a powerful and effective optimization tool for HSP problem.

1. Introduction

Humanity faces serious energy and environment problems at present. The environment is increasingly threatened. For instance, with the increase of greenhouse gas emissions in the atmosphere the environments have already reached concerning levels in terms of their potential to cause climate change. Air pollution, acid precipitation, and stratospheric ozone depletion are other serious environmental concerns. The severity of climate change impacts shows the increasing trend if significant action is not taken to reduce greenhouse gas emissions [1]. An important action to address energy and environmental challenges lies in the intelligent and efficient use of energy, including reducing energy waste and using low-carbon fuels.

In China, heating utilities have been developed rapidly, but the energy consumption of production and transport is still too much, which accounts for 21.5% of building energy consumption; building energy consumption accounts for 20.9% of social total energy consumption [2]. With the perfection of the systematic reform, the adjustment of energy structure in China and the requirement of environmental protection, heating energy structure had been changing, and it had been promoting the development of heating mode. It has very important significance to analyze, evaluate, and select heating mode correctly which suits its local characteristics. With the speeding up of urbanization, more and more heating systems will be built due to the importance of infrastructure in urban area. The research on optimal plan of heating system is very imperative for saving project investment, decreasing heating energy consumption, and improving enterprise benefit.

Sustainable development of heating system requires application of planning procedures, which includes optimization of both demand and supply sides of heating. Because the heat source site selection and heating pipe network optimizing plan have an important role in the HSP, there are many scholars concerning this subject and lots of optimization methods have been proposed. The methods of HSP can be classified into three separate categories [3]: planning by models, planning by analogy, and planning by inquiry. The planning by models can be based either on econometric or optimization models. Econometric models utilize mathematical or statistical methods and relay on statistical

data. Optimization model allows for the identification of best possible solution—minimization or maximization of objective function, with the predefined set of constrains which describes the space of acceptable solutions. The planning by analogy utilizes the simulation of heating system. That kind of HSP is usually used for the verification of planning results which were achieved by other planning methods [4]. The planning by inquiry is used in the case when other aforementioned methods are not reliable. Good example of planning by inquiry is DELPHI method, which is based on the questioning of group of heating, ventilating, and air conditioning (HVAC) experts or municipal planners and statistical evaluation of their answers [5]. All the methods of HSP listed earlier have a limited transparency, especially for decision makers who do not have good mathematical background. Those methods do not give opportunity to create decision makers preference model or define that model a priori. Hence, many scholars have carried out extensive and deep research on optimization method of HSP. Shen and He [6] investigated optimal planning method of central heating system of water boiler and then put forward optimal planning model and solving method. According to the method, it can be determined the size, location, and so forth of regional heating plant and intermediate heat exchanger station, but there were no further discussions about how to design the pipe network. Wang et al. [7–9] investigated design method of central heating system with double "duct-station," proposing two-step optimization method, but this method was only applicable to double "duct-station" system. For solving the problem, the study in [8] used fully stratified sequence method [10–12] simultaneously taking into account only one objective function for each layer of heating source layout optimization and pipe network system. However, for HSP problem, there are kinds of complex logic even iterative relationships between/among objective functions of layers. This method usually can provide the optimal solution of each layer, but it cannot ensure that the solution of the objective function for last layer is the optimal solution of the whole system exactly. Shi and Li [13] first applied genetic algorithm (GA) for solving the heating source location problem in the study. This method described the cost of the heat source and heating substation as the function of heat load and described the cost of the heating network as the function of heat load and pipe length simply. So the calculated result by this method and actual situation often put in certain error. Shi et al. [14, 15] and Mu et al. [16] put forward the relatively consistent mathematical model for heating system optimization, based on the life cycle cost method, but formula or method for some of parameters was not given clearly and integrally in the model. It seems that limitation is inevitable in the process of the practical application of these methods. But we noticed the life cycle cost (LCC) and particle swarm optimization (PSO) algorithm in the more extensive research areas.

Life cycle cost (LCC) has been applied since the 1960s when the United States' Department of Defense stimulated the development and application of LCC to enhance its cost effectiveness. Defense systems, such as an aircraft or a special land vehicle, are ideal for LCC analyses since the Department of Defense mainly controls the entire life cycle [17]. LCC may be defined as "the cost of acquisition, ownership, and disposal of a product over a defined period of its life cycle" [18, 19]. LCC is a standard engineering economic approach used for choosing among alternative products or designs that approximately provide the same service to the customer [20]. In many cases it may not be necessary to perform a complete LCC analysis, but rather to estimate the differences between the alternatives for the major cost elements [21]. The LCC process may also provide information, for example, in the assessment of the economic viability of products and projects, in the identification of the cost drivers and cost efficiency improvements, and in evaluations of different strategies for product operation, maintenance, and inspection, and so on [22].

There are two popular swarm inspired methods in computational intelligence areas: ant colony optimization (ACO) and particle swarm optimization (PSO). ACO was inspired by the behaviors of ants and has many successful applications in discrete optimization problems. The particle swarm concept originated as a simulation of simplified social system. The original intent was to graphically simulate the choreography of a bird block or fish school. However, it was found that particle swarm model can be used as an optimizer. A substantial review of the properties of the global optimization problems has been given by Parsopoulos and Vrahatis [23]. As one of the global optimization problems, PSO has been widely used in various kinds of planning problems, especially in the area of substation locating and sizing [24–27]. But in area of heating supply, PSO is mainly applied in heating load forecasting [28, 29], but rarely used in HSP.

The main objective of this paper is to discuss the usefulness of the PSO algorithm for solving the HSP problem. Therefore, based on the LCC approach, an integral mathematical model is presented and PSO algorithm is introduced and improved for solving the problem. In the end, the results of the case study suggest the effectiveness of improved particle swarm optimization (IPSO) application to the optimal planning method for heating system.

2. Mathematical Formulation

2.1. Problem Definition and Assumptions. LCC is related to the systems engineering process, because economic considerations are very important in the process of creating systems. Life cycle economic analyses should be done early in the system or product life cycle, because the outcome of the systems engineering process cannot be influenced very much when the design is completed. Thus, LCC involves evaluation of all future costs related to all of the phases in the system life cycle including design, construction and/or production, distribution, operation, maintenance and support, retirement, and material disposal, and so on [30].

Cost models may range from simple to complex and are essentially predictive in nature. Parameters, such as the system's physical environment, usage demand, reliability, maintainability, labor, energy, taxes, inflation, and the time value of money, may have a great influence on the life cycle costs [17].

The main objective of this paper is to discuss the usefulness of the PSO algorithm for owners in making sustainable heating system investment decisions and to improve their decision-bases for municipal administration. Therefore, we apply LCC approach to describe the HSP problem.

Moreover, HSP considered in this study works under the following definition and assumptions.

(i) A heat consuming installation can connect with any heat source but cannot connect with two or more heat sources at the same time.

(ii) The indirect connection between heat consuming installation and heat source is not allowed.

(iii) A heat source must be connected with more than one heat consuming installation; otherwise, it will be closed.

(iv) Any connection between any two heat sources is not allowed.

(v) The location of heat consuming installation is fixed.

(vi) A heat source can be sited in a given region.

(vii) The elevation difference between heat consuming installation and heat source is ignored.

(viii) Heating system planning and optimization can be achieved by changing the number and the heating capacity of heat source and the distance between the heat source and heat consuming installation.

(ix) The measure between heat source and heat consuming installation is simplified to the Manhattan (or city block) distance.

(x) There is no functional difference between any two heat sources and their products.

2.2. Notation. The notations used in the mathematical formulations are given as follows.

Indices

i: Optional heating source

k: Heating equipment

j: Heat consuming installation

r: Heat load distributing segment.

Parameters

m: Number of heat source

k_i: Number of heating equipment which could be installed at the heating source i; $k_i = \{1, 2, \ldots, P_i\}$

n: Number of heat consuming installation

n_r: Number of heat load distributing segments

F_i: Life cycle fixed cost of the heat source i

F_{ik}: Life cycle fixed cost of the heating equipment k which is in the heat source i

C_{ikjr}: Variable production and transport discounted costs within life cycle of heating equipment k to satisfy the heat load distributing segment r of heat consuming installation j, which is in the heat source i; $C_{ikjr} = P_{ikjr} + t_{ikjr}$, where P_{ikjr} is the variable production discounted cost within life cycle of specific heat load; t_{ikjr} is the transport discounted cost within life cycle per specific heat load

X_{ikjr}: Continuous variable, the load of the heat load distributing segment r of heat consuming installation j, which is supplied by heating equipment k of the heat source i

Q_{jr}: Load of the heat load distributing segment r of heat consuming installation j

S_{ik}: Maximum supply capacity of heating equipment k of the heat source i

$Q_i^{\max}$: Maximum supply capacity of heat source i

C_{zd}: Major repair depreciation discounted costs within life cycle of heat source i

C_{rg}: Labor discounted cost within life cycle of heat source i

u: Coefficient of sum; $u = [(1 + r)^y - 1]/r(1 + r)^y$, where r is the standard discount rate, and y is the life cycle

P_{rl}: Price of fuel

Q_w: Calorific value of fuel

η: Thermal efficiency of heat source

E: Water and electricity consumption costs of specific heat load

h_r: Duration of heat load distributing segment r

β: Sulfur content in fuel

λ: Standard emission charge for SO_2

$t_{rw}(j)$: Pipe network discounted cost per specific heat load, which is supplied by heat source i to heat consuming installation j

$C(L_j)$: The discounted cost of pipe segment L_j

$C_{zd}(L_j)$: The major repair depreciation discounted costs within life cycle of pipe segment L_j

$C_{sr}(L_j)$: The heat loss discounted costs within life cycle of pipe segment L_j

$Q(L_j)$: The heat load-bearing of pipe segment L_j

$C_{dl}(L_j)$: The power consumption discounted cost within life cycle of pipe segment L_j per specific heat load

$t_{rz}(j)$: The transport discounted cost within life cycle per specific heat load, which is supplied to heat consuming installation j

$C_{zd}(j)$: The major repair depreciation discounted costs within life cycle of heat consuming installation j

Q_{ij}: The heat load of pipe network for heat consuming installation j

$C_{dl}(j)$: The power consumption discounted cost within life cycle of heat consuming installation j per specific heat load

$C_{rg}(j)$: The labor discounted cost within life cycle of heat consuming installation j

$a[d(L_j)]^b$: Investment of $d(L_j)$ meters diameter double-pipe per meter length, where a and b are coefficients of pipe laying

γ: Rate of major repair depreciation

ρ: Rate of gross fixed capital formation

ω: Conversion coefficient of the units

R: Specific frictional resistance

$l_{dl}(L_j)$: Equivalent length of local resistance for pipe segment L_j

H_{gl}: Heating period

P_d: Electricity price for industrial uses

η_{xb}: Efficiency of circulating water pump

t_g, t_h: Supply/return water temperature of pipe segment

ξ: Conversion coefficient of the units

k: Heat transfer coefficient

ε: Local heat loss coefficient of pipe fittings

P_{sr}: Annual costs of heat loss

$t_{g,pj}$: Annual mean supply water temperature of pipe segment

$t_{h,pj}$: Annual mean return water temperature of pipe segment

$t_{hj,pj}$: Annual mean temperature

c_1, c_2: Comprehensive coefficient of investment

Q_{ij}: Heat load of pipe network

α: Correction factor

μ: Conversion coefficient of the units

ΔP_j: Pressure difference between supply and return water of pipe network for heat consuming installation j

S_{gz}: Average annual wages of operating personnel and manager

n_{yg}: Number of operating personnel and manager per 1 MW heat load

Ω: Conversion coefficient of the units.

Decision Variables

Y_{ik}: 1, if the equipment k is installed or set up in the heat source i; 0, if the equipment k is not installed or set up in the heat source i

Z_i: 1, if the heat source i is set up; 0, if the heat source i is not set up.

2.3. Mathematical Model of HSP. In this study, the problem is summarized into a multisource, multifacility, single-commodity, multiraw material plant location problem, and a mixed 0-1 integer planning model has been formulated. The cost model of the heat source and the heat-transmission network concerned in the optimization model are considered in this study. The objective function of heating system planning problem is to minimize the total heat production cost. The proposed mathematical model formulation for HSP problem can be found as follows.

Minimize

$$\mathrm{LCC} = \sum_{i=1}^{m} F_i Z_i + \sum_{i=1}^{m}\sum_{k=1}^{k_i} F_{ik} Y_{ik} + \sum_{i=1}^{m}\sum_{k=1}^{k_i}\sum_{j=1}^{n}\sum_{r=1}^{n_r} C_{ikjr} X_{ikjr} \quad (1)$$

subject to

$$\sum_{i=1}^{m}\sum_{k=1}^{k_i} X_{ikjr} = Q_{jr}, \quad j = 1, 2, \ldots, n;\ r = 1, 2, \ldots, n_r, \quad (2)$$

$$\sum_{j=1}^{n}\sum_{r=1}^{n_r} X_{ikjr} \le S_{ik} Y_{ik}, \quad i = 1, 2, \ldots, m;\ k = 1, 2, \ldots, k_i, \quad (3)$$

$$\sum_{k=1}^{k_i} S_{ik} Y_{ik} \le Q_i^{\max}, \quad i = 1, 2, \ldots, m, \quad (4)$$

$$\sum_{k=1}^{k_i} Y_{ik} \le P_i Z_i, \quad i = 1, 2, \ldots, m, \quad (5)$$

$$Z_i, Y_{ik} = 0 \text{ or } 1, \qquad X_{ikjr} \ge 0. \quad (6)$$

Objective function (1) minimizes the discounted costs within life cycle of heating system as the general objective; it is an index of dynamic economy evaluation, where F_i, F_{ik}, and C_{ikjr} are composed of respective discounted costs together. Constraint (2) is each heat consuming installation's heat load, which is heat consumption for each user and the requirements of the heating quantity and quality. Constraint (3) means that each of the heating equipment in the heating system bear heat load should not exceed the maximum heating capacity. Constraint (4) means the maximum heating capacity of heating source, which is allowed under the restrictions of objective conditions. Constraint (5) means that only open heating source first can install equipment in it. In the model, there are two decision variables, in which Z_i is related to heating source, and Y_{ik} is related to heating equipment.

Because the piecewise function of heat load duration curve is introduced in the process of solving the model, this model can be applied to any form of heating system.

2.4. Formulation of Heating System Cost Model

2.4.1. The Heating Source Cost Model. The heating source cost model is aimed to resolve the calculation problem of F_{ik},

in the objective function (1), and P_{ikjr}, which is a part of C_{ikjr} in the objective function (1). It consists of fixed costs and variable costs, the former refers to all necessary costs of heating source, so long as open a heating source or install a piece of heating equipment, the latter only associated with the size of the heat load and running status of equipment. Consider

$$F_{ik} = C_{zd} + C_{rg}, \tag{7}$$

$$P_{ikjr} = u\left(\frac{0.36P_{rl} + 0.72\beta\lambda}{Q_w\eta} + 0.36E\right)h_r. \tag{8}$$

Equation (7) is the formulation of heating source fixed costs, and (8) is the formulation of heating source variable costs.

2.4.2. The Heating Network Cost Model. The heating network cost model is aimed to resolve the calculation problem of t_{ikjr}, which is a part of C_{ikjr} in the objective function (1), and also to optimize the direction of heating network and the pipe diameter. Heating network (heat consuming installation included) cost consists of the heating network operation cost and heat consuming installation costs. Heating network operation cost consists of major repair depreciation discounted cost, power consumption discounted cost, pipe network heat loss discounted cost, and labor discounted cost. By dividing the discounted cost within life cycle of pipe segment allocation to the total heat load bearded by itself directly and evenly, the transport discounted cost within life cycle per specific heat load can be obtained, which is supplied by heat source i to heat consuming installation j. Consider

$$t_{ikjr} = t_{rw}(j) + t_{rz}(j), \tag{9}$$

$$t_{rw}(j) = t_{rw}(j-1) + C\left(L_j\right),$$

$$C\left(L_j\right) = \frac{C_{zd}\left(L_j\right) + C_{sr}\left(L_j\right)}{Q\left(L_j\right)} + C_{dl}\left(L_j\right), \tag{10}$$

$$t_{rw}(0) = 0,$$

$$C_{tz}\left(L_j\right) = a\left[d\left(L_j\right)\right]^b l\left(L_j\right), \tag{11}$$

$$C_{zd}\left(L_j\right) = \gamma\rho C_{tz}\left(L_j\right)u, \tag{12}$$

$$C_{dl}\left(L_j\right) = \frac{2\omega R\left[l\left(L_j\right) + l_{dl}\left(L_j\right)\right]uH_{gl}P_d}{\eta_{xb}\left(t_g - t_h\right)}, \tag{13}$$

$$C_{sr}\left(L_j\right) = \xi k\pi d\left(L_j\right)l\left(L_j\right)(1+\varepsilon) \times H_{gl}P_{sr}u\left(t_{g,pj} + t_{h,pj} - 2t_{hj,pj}\right). \tag{14}$$

Equation (9) is the heating network transportation cost model. The pipe segment cost model is composed of (10)–(14), where (10) is the pipe network discounted cost per specific heat load supplied by heat source i to heat consuming installation j; (11) is the investment cost of pipe segment L_j; (12) is the major repair depreciation discounted cost of pipe segment L_j; (13) is the power consumption discounted cost of pipe segment L_j per specific heat load; and (14) is the heat loss discounted cost of pipe segment L_j. Consider

$$t_{rz}(j) = \frac{C_{zd}(j)}{Q_{ij}} + C_{dl}(j) + C_{rg}(j), \tag{15}$$

$$C_{tz}(j) = c_1 + \alpha c_2 Q_{ij}, \tag{16}$$

$$C_{zd}(j) = \gamma\rho C_{tz}(j)u, \tag{17}$$

$$C_{dl}(j) = \frac{\mu\Delta P_j}{\eta_{xb}\left(t_g - t_h\right)}H_{gl}P_d u, \tag{18}$$

$$C_{rg}(j) = S_{gz}n_{yg}u\Omega. \tag{19}$$

The heat consuming installation cost model is composed between (15) and (19), where (15) is the transport discounted cost within life cycle per specific heat load supplied by heat consuming installation j; (16) is the investment cost of heat consuming installation j; (17) is the major repair depreciation discounted cost of heat consuming installation j; (18) is the power consumption discounted cost of heat consuming installation j per specific heat load; and (19) is the labor discounted cost of heat consuming installation j.

3. PSO and Its Improvement

3.1. PSO Algorithm. The PSO is proposed by Kennedy and Eberhart [31, 32] in 1995, and the motivation for the development of this algorithm was studied based on the simulation of simplified animal social behaviors, such as fish schooling and bird flocking. Similar to other population-based optimization methods such as genetic algorithms, the particle swarm algorithm starts with the random initialization of a population of particles in the search space [33]. However, unlike in other evolutionary optimization methods, in PSO there is no direct recombination of genetic material between individuals during the search. The PSO algorithm works on the social behavior of particles in the swarm. Therefore, it provides the global best solution by simply adjusting the trajectory of each individual toward its own best location and toward the best particle of the entire swarm at each time step (generation) [31, 34, 35]. The PSO method is becoming very popular due to its simplicity of implementation and ability to quickly converge to a reasonably good solution.

3.2. Formulation of General PSO. Specifically, PSO algorithm maintains a population of particles, each of which represents a potential solution to an optimization problem. The position of the particle denotes a feasible, if not the best, solution to the problem. The optimum progress is required to move the particle position in order to improve the value of objective function. The convergence condition always requires setting up the move iteration number of particle.

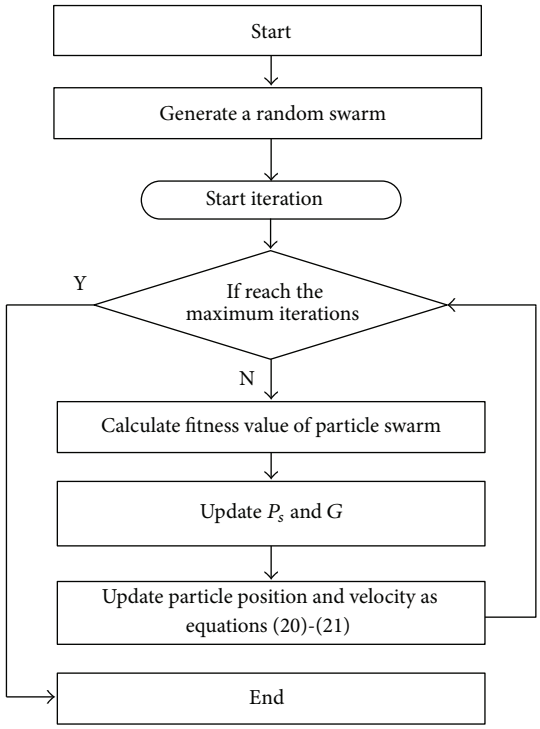


Figure 1: Flow chart of general PSO.

The position of particle move rule is shown as follows:

$$V_s(t+1) = wV_s(t) + C_1 r_1 \left(P_s - X_s(t)\right) + C_2 r_2 \left(G - X_s(t)\right), \tag{20}$$

$$X_s(t+1) = X_s(t) + V_s(t+1), \tag{21}$$

where $V_s(t)$ represents the velocity vector of particle s in t time; $X_s(t)$ represents the position vector of particle s in t time; P_s is the personal best position of particle s; G is the best position of the particle found at present; w represents inertia weight; C_1, C_2 are two acceleration constants, called cognitive and social parameters, respectively; and r_1 and r_2 are two random functions in the range $[0, 1]$.

The flow chart of general PSO is shown in Figure 1.

3.3. Improvement of Particle Swarm Optimization (IPSO) for HSP Problem. For HSP problem and its model in this paper, the value of LCC depends mostly on the distance between heating source and heat consuming installation, and the number of heating source i. It is necessary to make corresponding improvements on PSO, in order to solve this problem more accurately and effectively.

The evolution of the solution set begins with an initial solution set in the PSO; initial solution set is composed of initial particles. Each solution location is represented by an i-dimensional vector; i represents the number of variables of each solution, and it represents the number of heating sources in particularly in this paper.

The position coordinate of heating source (p) has two components, which is represented by two i-dimensional vectors, where x direction coordinates are represented by vector px, and y direction coordinates are represented by vector py. Therefore, x direction component for the position vector of particle s in t time can be represented by $px_s(t)$, and the rest can be done in the same manner.

In the same way, the velocity for location change of heating source (Vp) has two components, which is represented by two i-dimensional vectors, where x direction component for the velocity vector is represented by vector Vpx, and y direction component for the velocity vector is represented by vector Vpy. Therefore, x direction component for the velocity vector of particle s in t time can be represented by $Vpx_s(t)$, and the rest can be done in the same manner.

Thus, the update rule of velocity for each particle is indicated by (22)-(23), and the update rule of position for each particle is indicated by (24)-(25). Consider

$$Vpx_s(t+1) = wVpx_s(t) + C_1 r_1 \left(P_s - px_s(t)\right) + C_2 r_2 \left(G - px_s(t)\right), \tag{22}$$

$$Vpy_s(t+1) = wVpy_s(t) + C_1 r_1 \left(P_s - py_s(t)\right) + C_2 r_2 \left(G - py_s(t)\right), \tag{23}$$

$$px_s(t+1) = px_s(t) + Vpx_s(t+1), \tag{24}$$

$$py_s(t+1) = py_s(t) + Vpy_s(t+1). \tag{25}$$

The meanings of parameters are consistent with previous description.

3.4. Calculated Flow of IPSO. The calculated flow of proposed IPSO is described as follows.

3.4.1. Initial Solution. The initial solution for HSP problem is obtained by random initial position of each heat source; a matrix is employed in recording the coordinates and the heat load-bearing information of heat source, and the calculated flow of initial solution is as follows.

(1) Set up the number of heat source i, and generate an empty matrix for the initial position of heat source.

(2) Based on randomly and evenly distributed manner, generate the position coordinates of heat sources, into the matrix.

(3) Call the decoding function; calculate the heat load-bearing and the cost for each heat source, into the matrix.

(4) Calculate the LCC, the fitness value of the initial particle.

3.4.2. Decoding Function. In this paper, decoding function will call the matrix for current position and heat load of heat consuming installation, and then according to the matrix for the position of heat source, which is represented by current particle, divide the heating range of each heat source, and calculate the LCC.

Information matrix of heat consuming installation (*heat_point*) is a j-line four-column matrix; the first column

represents the serial number of heat consuming installation, the second column represents the x coordinate of heat consuming installation, the third column represents the y coordinate of heat consuming installation, and the heat load of heat consuming installation is represented by the fourth column. The calculated flow of initial solution is as follows.

(1) Read matrix *heat_point*, and let $j = j + 1$.

(2) Calculate the distance to all heat source from the heat consuming installation j, into the vector l_j.

(3) By substituting l_j into (10)–(14), calculate the cost of the heat consuming installation j connected with each heat source.

(4) Find out the minimum cost, and the heat consuming installation j connected with the corresponding heat source.

(5) If j is the last heat consuming installation then stop; otherwise, go to Step 1.

3.4.3. The Evolution of Particle Swarm. After one generation of particles, a new generation is evolved as follows.

(1) Call the decoding function; calculate the fitness value of the particle swarm.

(2) Update the individual optimal solution P_s and the global optimal solution G.

(3) Update the speed vector, by using (22)-(23).

(4) Update the speed vector, by using (24)-(25).

3.4.4. Improvement Approach. The PSO's convergence is fast, so it is liable to fall into local optimal solution. In order to improve the optimizing capability, we add modular arithmetic of velocity vector into each iterative operation. If the norm of velocity vector V is less than the predetermined minimum value $V_{\min}$, then generate a random velocity, let the current particle swarm out of local convergence region, and search other solution spaces. However, after it falls into local optimal solution, the norm of velocity tends to 0 in the general PSO algorithm, the solution stabilized near the local optimal solution, and it cannot explore search space furthermore.

The flow chart of IPSO is shown as Figure 2.

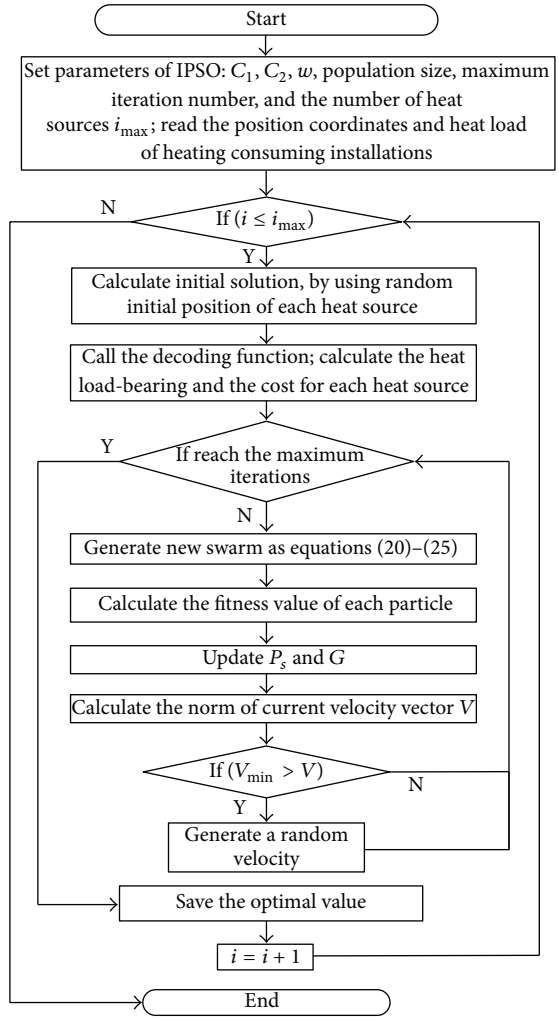


Figure 2: Flow chart of IPSO.

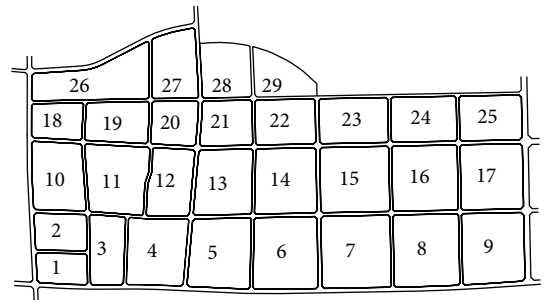


Figure 3: Site location plan of 29 heating districts.

4. Case Study

4.1. Basic Information of Case. This is a heating plan for a new area in China covering the area of 3.346 million square meters, and heat load is 167.3 MW in total. Based on the road network, the new area is divided into 29 heating districts (Figure 3), and the heating load of each district (Figure 4) is supplied by their small gas-fired boiler.

4.2. The Parameters of Algorithms. The role of the inertia weight w, in (20), (22), and (23), is considered critical for the PSO's convergence behaviour. The inertia weight is employed to control the impact of the previous history of velocities on the current one. Accordingly, the parameter w regulates the trade-off between the global and local exploration abilities of the swarm. A large inertia weight facilitates global exploration, while a small one tends to facilitate local exploration. A suitable value for the inertia weight w usually provides balance between global and local exploration abilities resulting in a reduction of the number of iterations required to locate the optimum solution. Initially, the inertia

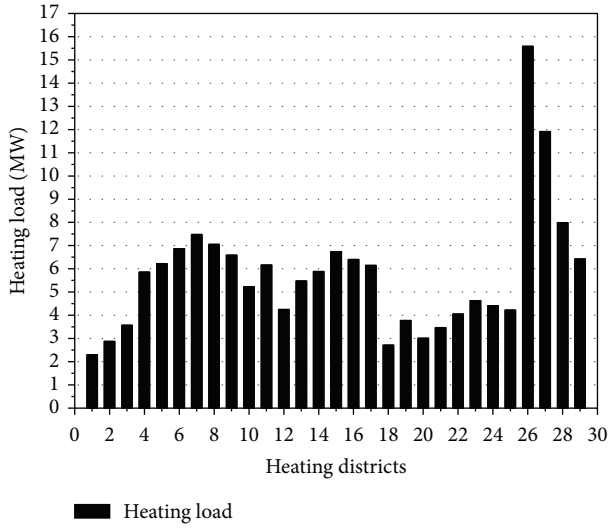


Figure 4: Heating load of 29 heating districts.

weight was constant. However, experimental results indicated that it is better to initially set the inertia to a large value, in order to promote global exploration of the search space, and gradually decrease it to get more refined solutions [32, 36]. Thus, an initial value around 1.2 and a gradual decline towards 0 can be considered as a good choice for w. The parameters C_1 and C_2, in (20), (22), and (23), are not critical for PSO's convergence. However, proper fine-tuning may result in faster convergence and alleviation of local minima. A further study of the acceleration parameter in the first version of PSO is given in [37]. As default values, $C_1 = C_2 = 2$ were proposed, but experimental results indicate that $C_1 = C_2 = 0.5$ might provide even better results. Some work reports that it might be even better to choose a larger cognitive parameter, C_1, than a social parameter, C_2, and $C_1+C_2 \le 4$ [38, 39], but $(C_1+C_2)/2 = 1.494$ was suggested by [35]; the strategy of acceleration parameter linear changing with iterations was proposed by Ratnaweera et al. [40], but acceleration parameter is the nonlinear function of the ratio G-to-P_s, which was proposed by Arumugam et al. [41]; Jie et al. [42] suggested to adjust the acceleration coefficient by measuring diversity.

But so far, the research on the most appropriate values for w, C_1, and C_2 has no unified conclusion. And how the variable values impact the solution to HSP problem is unknown. For HSP problem on kinds of values is unknown. So we set the w, C_1, and C_2 to common values in this study.

4.3. Analysis of Results. By applying PSO and IPSO algorithm, respectively, we solved the HSP problem in this paper. The parameters of PSO and IPSO are summarized in Table 1.

In this study, 29 kinds of schemes of heating (from one heat source to twenty-nine heat sources) were calculated for 10 times through reading initial conditions from the excel file successively, which contains the coordinates and heat load of heat consuming installation, preset maximum number of heat source. The results of LCC and the D-value for the optimum and the average between PSO and IPSO at the same number of heat source are shown as Figures 5 and 6 and Table 2.

Table 1: PSO and IPSO parameters.

Variable	Symbol	Value	
		PSO	IPSO
Population size	—	100	100
Maximum iteration number	—	1000	1000
Inertia weight	w	0.7	0.7
Acceleration constant	C_1	2	2
	C_2	2	2

Table 2: Algorithm calculation results comparison.

Heat source	LCC (billion Yuan)					
	Optimum value			Average value		
	PSO	IPSO	D-value	PSO	IPSO	D-value
1	1.5320	1.5320	0.0000	1.5320	1.5320	0.0000
2	1.5035	1.5035	0.0000	1.5035	1.5035	0.0000
3	1.4940	1.4940	0.0000	1.4947	1.4945	0.0002
4	1.4881	1.4880	0.0001	1.4893	1.4890	0.0003
5	1.4866	1.4864	0.0002	1.4873	1.4872	0.0001
6	1.4857	1.4848	0.0009	1.4867	1.4863	0.0004
7	1.4857	1.4842	0.0015	1.4865	1.4852	0.0013
8	1.4850	1.4828	0.0022	1.4865	1.4854	0.0011
9	1.4849	1.4832	0.0017	1.4872	1.4857	0.0015
10	1.4847	1.4843	0.0004	1.4875	1.4866	0.0009
11	1.4855	1.4852	0.0003	1.4889	1.4874	0.0015
12	1.4864	1.4856	0.0008	1.4894	1.4880	0.0014
13	1.4883	1.4865	0.0018	1.4915	1.4887	0.0028
14	1.4893	1.4873	0.0020	1.4920	1.4910	0.0010
15	1.4910	1.4903	0.0007	1.4933	1.4917	0.0016
16	1.4931	1.4908	0.0023	1.4952	1.4944	0.0008
17	1.4939	1.4933	0.0006	1.4966	1.4954	0.0012
18	1.4966	1.4944	0.0022	1.5002	1.4971	0.0031
19	1.4987	1.4974	0.0013	1.5025	1.4993	0.0032
20	1.4990	1.4985	0.0005	1.5010	1.5007	0.0003
21	1.5018	1.5001	0.0017	1.5031	1.5027	0.0004
22	1.5032	1.5018	0.0014	1.5044	1.5040	0.0004
23	1.5033	1.5031	0.0002	1.5065	1.5056	0.0009
24	1.5062	1.5050	0.0012	1.5080	1.5075	0.0005
25	1.5075	1.5065	0.0010	1.5100	1.5095	0.0005
26	1.5106	1.5102	0.0004	1.5131	1.5120	0.0011
27	1.5121	1.5110	0.0011	1.5132	1.5130	0.0002
28	1.5135	1.5133	0.0002	1.5152	1.5150	0.0002
29	1.5169	1.5145	0.0024	1.5195	1.5170	0.0025

From analyzing the results, we can draw the following conclusion about the HSP problem.

(1) The original plan (the heating load of each district is supplied by its small gas-fired boiler) is not an economic and reasonable plan for the case, and the LCC is the second highest in 29 schemes, which is

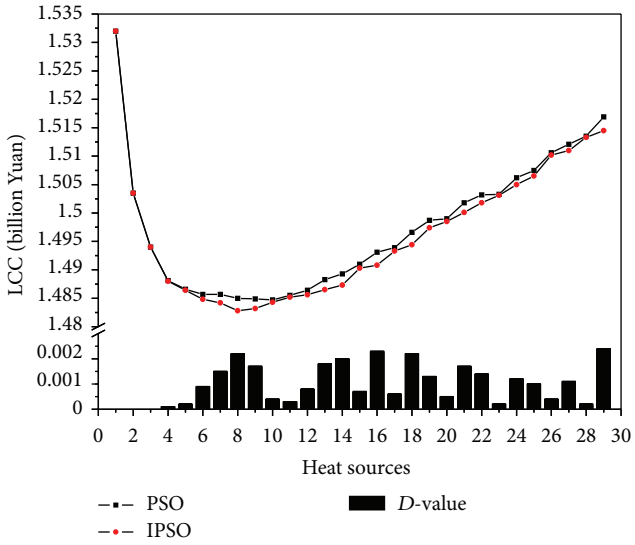


Figure 5: Algorithm calculation results comparison (optimum value).

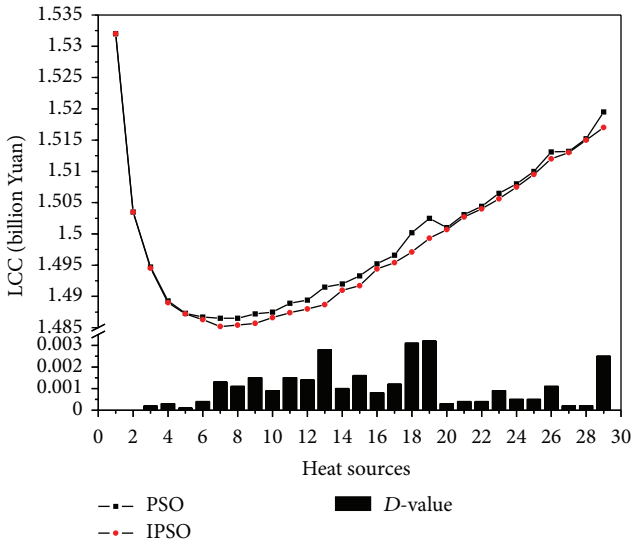


Figure 6: Algorithm calculation results comparison (average value).

only better than the scheme which plans to set up one heat source only.

(2) From one heat source to twenty-nine heat sources, LCC is monotone decreasing until a minimum value first, then monotone increasing.

(3) Only one minimum value of LCC that appeared throughout the change process, which is 1.4828 billion Yuan, the scheme of which plans to set up 8 heat sources, is the best choice for the case. (The detailed calculation results of this scheme are shown in Table 3.)

By observing the algorithms, the following is also concluded.

Table 3: The detailed results of 8 heat sources scheme.

Heating source	Coordinate	Supply heat load (MW)	Heat consuming installation
1	(395, 555)	24.38	1, 2, 3, 10, 11, 12
2	(320, 1050)	22.07	18, 19, 26
3	(1030, 315)	24.43	4, 5, 6, 13
4	(760, 1070)	14.92	20, 27
5	(1745, 555)	24.73	7, 14, 15, 23
6	(1040, 1090)	21.94	21, 22, 28, 29
7	(2380, 850)	14.78	17, 24, 25
8	(2145, 340)	20.05	8, 9, 16

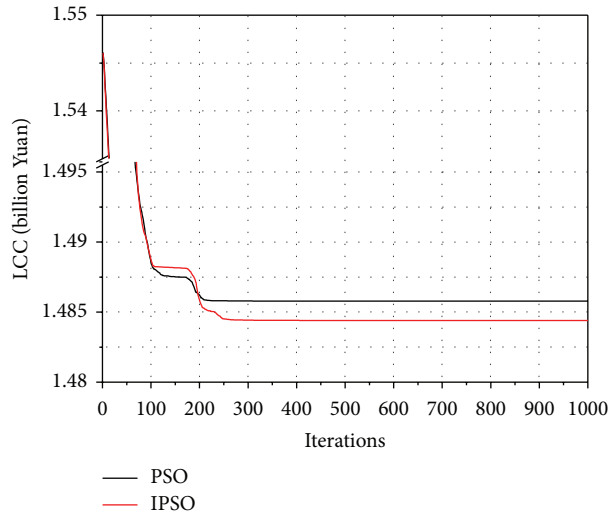


Figure 7: Algorithm calculation comparison (7 heat sources).

(1) The optimal solution of IPSO is better than PSO. The optimum LCC which calculated by IPSO is not larger than PSO for all 29 schemes. The maximum D-value is 2.4 million Yuan in the scheme which plans to set up 29 heat sources.

(2) The real minimum LCC was not calculated by PSO. The minimum LCC calculated by PSO is 1.9 million Yuan larger than the minimum LCC calculated by IPSO.

Figure 7 compares the LCC convergence curves of two algorithms in three kinds of schemes, respectively. When the population size and the iteration number of PSO are same as those of IPSO during the HSP optimization process, although the PSO algorithm is faster for giving the optimization results, but the optimal results by IPSO are better than the searcher values by PSO. The main reason for current performance is that IPSO can avoid local optimal solution and then further expand the search space so as to find a better solution.

Hence, it can be concluded that the improvement approach is effective, and the proposed method IPSO has better significance in solving the HSP problem and competitive to PSO algorithm.

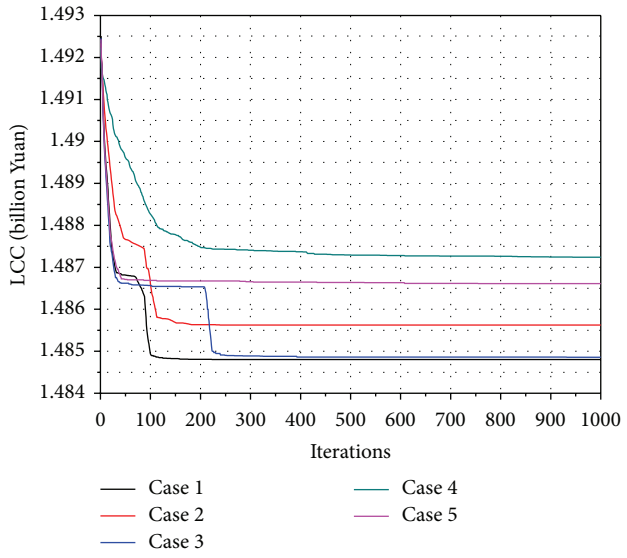


Figure 8: Algorithm calculation in comparison with different parameters (6 heat sources).

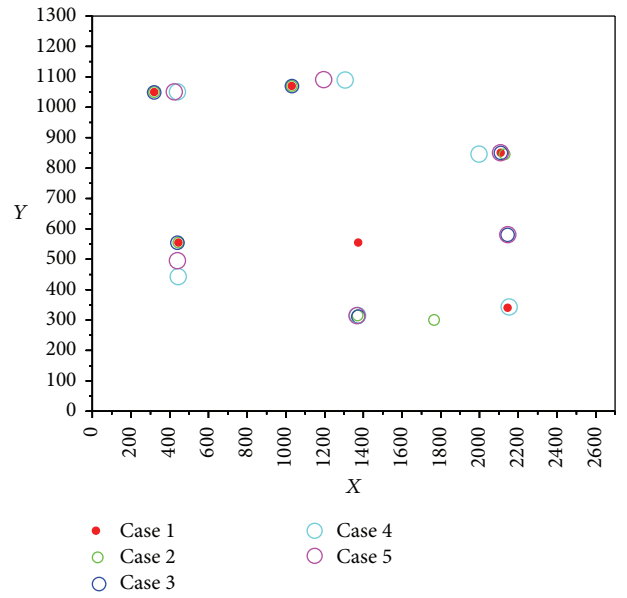


Figure 9: Coordinates of heating sources in comparison with different parameters (6 heat sources).

5. Discussion

Section 4.2 referred to the values of w, C_1, and C_2 which may influence the computational results. In Figures 8 and 9, the results obtained by IPSO were also proved. Algorithm calculation comparison with different parameters is shown in Figures 8 and 9, which is a visual display of the coordinates of heating sources with different parameters.

The parameters of each case in the figure are summarized in Table 4.

The influence aspect of the algorithm is worth further study, but because of the major goal of the present study, the more details were not presented here and will be discussed in a separate paper.

Table 4: The parameters of each case in Figures 8 and 9.

Variable	Symbol	Value				
		Case 1	Case 2	Case 3	Case 4	Case 5
Population size	—	100	100	100	100	100
Maximum iteration number	—	1000	1000	1000	1000	1000
Inertia weight	w	0.7	0.4	0.9	0.7	0.7
Acceleration constant	C_1	2	2	2	0.2	3.8
	C_2	2	2	2	0.2	3.8

6. Conclusions and Prospects

(1) In this paper, we presented an integral mathematical model for solving the heating system planning (HSP) problem taking into account minimizing the cost of heating system for a given life cycle time.

(2) According to the particularity of HSP problem, the particle swarm optimization (PSO) algorithm was introduced and improved, the new definition and update rule of velocity and position vector were proposed, and the improvement approach about generating a random velocity was adopted to avoid particle swarm into local optimal solution. Then an actual case study was calculated to check its feasibility in practical use. The results show that the IPSO algorithm can more preferably solve the HSP problem than PSO algorithm.

(3) Although there is no more discussion about the influence of computational results by changing the values of algorithm parameters (w, C_1, and C_2), but the results of the case study still show the potential to provide useful information when making decisions in the practical planning process. Thus, it is believed that if this approach is applied correctly and in combination with other elements, such as the accurate prediction of heating load, the running efficiency of equipment, and the real operation situation, it can become a powerful and effective optimization tool for HSP problem.

References

[1] V. Arroyo, *Agenda for Climate Action*, Pew Centre on Global Climate Change, Arlington, Va, USA, 2006.

[2] Tsinghua University Building Energy Research Center, *2012 Annual Report on China Building Energy Efficiency*, China Architecture & Building Press, 2012.

[3] M. Kleinpeter, *Energy Planning and Policy*, John Wiley & Sons, New York, NY, USA, 1995.

[4] C. Cormio, M. Dicorato, A. Minoia, and M. Trovato, "A regional energy planning methodology including renewable energy sources and environmental constraints," *Renewable and Sustainable Energy Reviews*, vol. 7, no. 2, pp. 99–130, 2003.

[5] O. Helmer, "Kommentar zur Delphi methode," *Science Journal*, no. 10, pp. 49–53, 1967.

[6] Y.-T. Shen and J.-Y. He, *The Optimization of Thermo-Dynamic System and Equipment*, China Machine Press, 1985.

[7] Z.-G. Wang, X.-Y. Xiang, L. Zhou et al., "Optimization design of central heating system in Liaohe oilfield Zhenxing district," *Oil-Gasfield Surface Engineering*, vol. 16, no. 1, pp. 32–36, 1997.

[8] Z.-G. Wang, Y.-T. Ma, W. Lu et al., "Optimization programming for heating system," *Heating Ventilating & Air Conditioning*, vol. 33, no. 1, pp. 2–4, 2003.

[9] Z.-G. Wang, J. Chen, Y.-C. Song, W.-Z. Yang, and X.-Y. Xiang, "The study on the desing method of "duct-station" layout and pipe networt optimization for central heating system," *Journal of Engineering Thermophysics*, vol. 33, no. 2, pp. 302–304, 2012.

[10] K.-X. Xie, L.-X. Han, and Y.-L. Lin, *Optimization Method*, Tianjin University Press, 1998.

[11] Y. Sawaragi, H. Nakayama, and T. Tanino, "Theory of multiobjective optimization," *Mathematics in Science and Engineering*, vol. 176, pp. 1–296, 1985.

[12] S. M. Lee, *Goal Programming for Decision Analysis*, Auer Bach, Philadelphia, Pa, USA, 1972.

[13] Z.-Y. Shi and H.-J. Li, "Application of genetic algorithms in location selection of heating system with multi-heat sources," *District Heating*, no. 2, pp. 11–14, 1998.

[14] Y.-J. Shi and W. Yuan, "Optimal planning of district heating system of water boiler," *Journal of Hebei Institute of Architectural Engineering*, no. 3, pp. 5–10, 1997.

[15] Y.-J. Shi and L.-L. Liu, "Application of LCC to optimization of heating systems," *Heating Ventilating & Air Conditioning*, vol. 29, no. 5, pp. 65–66, 1999.

[16] Q.-Y. Mu, L.-Q. Yue, and F.-S. Zhou, "Applied LCC's concept in optimum planning of heating system," *Journal of Hebei Institute of Architectural Engineering*, vol. 19, no. 2, pp. 46–47, 2001.

[17] Y. S. Sherif and W. J. Kolarik, "Life cycle costing: concept and practice," *Omega*, vol. 9, no. 3, pp. 287–296, 1981.

[18] International Electro technical Commission (IEC), "Dependability management. Part 3-3. Application guide—Life cycle costing," International standard, IEC 60300-3-3, Pronorm AS, 2004.

[19] M. Rausand and A. Høyland, *System Reliability Theory. Models, Statistical Methods, and Applications*, Wiley Series in Probability and Statistics, Wiley-Interscience, 2004.

[20] J. Lutz, A. Lekov, P. Chan, C. D. Whitehead, S. Meyers, and J. McMahon, "Life-cycle cost analysis of energy efficiency design options for residential furnaces and boilers," *Energy*, vol. 31, no. 2-3, pp. 311–329, 2006.

[21] Norsok Standard, "Life cycle cost for systems and equipment," Common requirements, O-CR-001, 1996.

[22] D. Ravemark, "LCC/LCA Experience. Developing and working with LCC tools," DANTES—EU Life Environment Program, 2004.

[23] K. E. Parsopoulos and M. N. Vrahatis, "Recent approaches to global optimization problems through particle swarm optimization," *Natural Computing*, vol. 1, no. 2-3, pp. 235–306, 2002.

[24] B.-Z. Wang, Z.-R. Liang, H.-F. Su, and Y.-Z. Liu, "Application of improved PSO algorithm in location selection of substations," *Electric Power Science and Engineering*, vol. 25, no. 10, pp. 4–7, 2009.

[25] M.-H. Yang, H. Liu, C.-S. Wang, S.-Y. Ge, and T. Zeng, "Optimal substation locating and sizing based on cultural algorithm of particle swarm," *Journal of Tianjin University*, vol. 45, no. 9, pp. 785–790, 2012.

[26] C.-S. Wu, "Multi-objective distributed generation planning based on improved particle swarm optimization algorithm," *Guangdong Electric Power*, vol. 25, no. 1, pp. 54–58, 2012.

[27] H.-F. Su, J.-H. Zhang, Z.-R. Liang, S. Zhang, and S.-S. Niu, "Substation LCC planning based on refined mean clustering random particle swarm algorithm," *Transactions of China Electrotechnical Society*, vol. 27, no. 4, pp. 209–215, 2012.

[28] B.-K. Gao, Y. Li, and M.-Z. Xu, "Application of particle swarm optimization algorithm in the heating load combination forecasting," *Information and Electronic Engineering*, vol. 9, no. 5, pp. 655–659, 2011.

[29] J. Liu, Y. Yang, and Q.-G. Qiu, "Research on heat burden prediction and control of substation based on PSO algorithm," *Energy Conservation*, no. 6, pp. 27–30, 2008.

[30] W. J. Fabrycky and B. S. Blanchard, *Life-Cycle Cost and Economic Analysis*, Prentice Hall, Upper Saddle River, NJ, USA, 1991.

[31] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Network*, pp. 1942–1948, December 1995.

[32] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proceedings of IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 69–73, May 1998.

[33] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

[34] M. Clerc, "The swarm and the queen: toward a deterministic and adaptive particle swarm optimization," in *Proceedings of the IEEE International Joint Conference on Evolutionary Computation*, vol. 3, pp. 1951–1957, 1999.

[35] 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.

[36] Y. Shi and R. Eberhart, "Parameter selection in particle swarm optimization," in *Evolutionary Programming VII*, vol. 1447 of *Lecture Notes in Computer Science*, pp. 591–600, 1998.

[37] J. Kennedy, "The behavior of particles," in *Evolutionary Programming VII*, vol. 1447 of *Lecture Notes in Computer Science*, pp. 579–589, 1998.

[38] A. Carlisle and G. Dozier, "An off-the-shelf PSO," in *Proceedings of the Particle Swarm Optimization Workshop*, pp. 1–6, 2001.

[39] Y.-X. Shen, G.-Y. Wang, and C.-H. Zeng, "Correlative particle swarm optimization model," *Journal of Software*, vol. 22, no. 4, pp. 695–708, 2011.

[40] 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.

[41] M. S. Arumugam, M. V. C. Rao, and A. W. C. Tan, "A novel and effective particle swarm optimization like algorithm with extrapolation technique," *Applied Soft Computing Journal*, vol. 9, no. 1, pp. 308–320, 2009.

[42] J. Jie, J.-C. Zeng, and C.-Z. Han, "Self-organized particle swarm optimization based on feedback control of diversity," *Computer Research and Development*, vol. 45, no. 3, pp. 464–471, 2008.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 196823, 7 pages
http://dx.doi.org/10.1155/2013/196823

Research Article

Research on an Infectious Disease Transmission by Flocking Birds

Mingsheng Tang,[1,2] Xinjun Mao,[1] and Zahia Guessoum[2]

[1] *College of Computer, National University of Defense Technology, Changsha 410073, China*
[2] *LIP 6, Université Pierre et Marie Curie, 75006 Paris, France*

Correspondence should be addressed to Mingsheng Tang; tms110145@gmail.com

Received 24 April 2013; Accepted 10 June 2013

Academic Editors: P. Agarwal, S. Balochian, and Y. Zhang

The swarm intelligence is becoming a hot topic. The flocking of birds is a natural phenomenon, which is formed and organized without central or external controls for some benefits (e.g., reduction of energy consummation). However, the flocking also has some negative effects on the human, as the infectious disease H7N9 will easily be transmited from the denser flocking birds to the human. Zombie-city model has been proposed to help analyzing and modeling the flocking birds and the artificial society. This paper focuses on the H7N9 virus transmission in the flocking birds and from the flocking birds to the human. And some interesting results have been shown: (1) only some simple rules could result in an emergence such as the flocking; (2) the minimum distance between birds could affect H7N9 virus transmission in the flocking birds and even affect the virus transmissions from the flocking birds to the human.

1. Introduction

Swarm intelligence is becoming a hot research topic, which could be presented in natural, social, or artificial systems. It is often inspired by natural systems, especially biological systems. The flocking of birds is a classical phenomenon of swarm intelligence without central control [1]. How could birds fly in a flocking? Each bird only confirms the simple rules of alignment, cohesion, and separation, and then these birds could fly in a flocking. Why do birds fly in a flocking? Because it could reduce the power requirements of birds and could bring other benefits for these birds [2–4].

However, it also brings some problems, for example infectious diseases could spread quickly among the flocking birds. Moreover, these infectious diseases may be transmitted to the human such as the infectious disease H7N9 which is spreading in China [5]. This paper will study the wild flocking birds and its effect on the infectious disease spread from the viewpoint of the transmission of the infectious disease H7N9. In order to facilitate modeling the flocking and the spread of H7N9 between birds and human, this paper proposes a general model, zombie-city model, which contains five core concepts: agent, role, environment, social network, and rule. Based on this model, we could construct and analyze the artificial flocking birds and also could model the artificial society. We will study the spread of H7N9 in the flocking birds and H7N9 transmission from birds to the human.

This paper is organized as follows: Section 2 introduces the zombie-city model. Section 3 models the flocking and artificial society with the zombie city and then presents the experiments results. Section 4 summarizes the work.

2. Zombie-City Model

2.1. Meat-Model of Zombie-City Model. In the zombie-city model [6], there are mainly five concepts, including agent, environment, social networks, roles, and rules. Rules could constrain agents, environments, and social network; that is, the agents, the environment, and the social network must conform to these rules. Besides, agents have their own capabilities (e.g., movement), and agents may be infected with

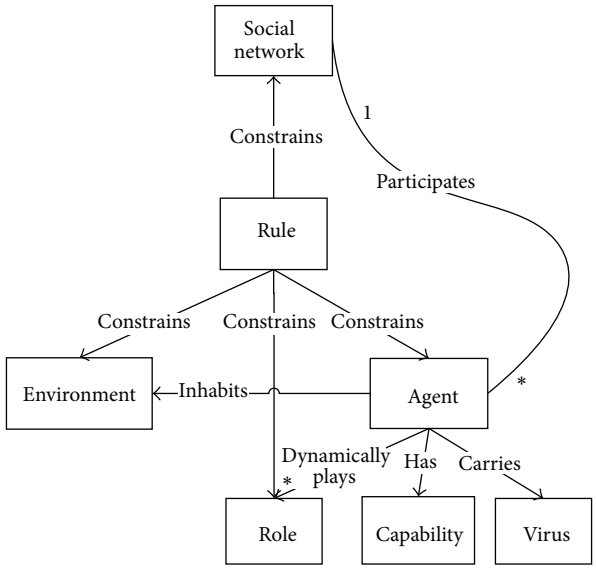


Figure 1: Metamodel of the zombie-city model.

viruses so that they will carry viruses to play the zombie roles. Figure 1 presents the metamodel of the zombie-city model.

(i) *Agent*. It is a proactive, automatic, and self-adaptive entity. It could interact with other agents and has some capabilities such as interaction and sensing the social relationships with other agents. And agents could dynamically play different roles to adapt to the environmental, social, or their own changes.

(ii) *Social Network*. It is made up of social relationships between agents, which construct the social structure. Meanwhile, it must conform to the rules such as scale-free network.

(iii) *Environment*. It is the space where agents inhabit, which is made up of grids. The environment also must conform to the rules.

(iv) *Rule*. It is constraining agents, social network, the environment and roles, and all of them do not act against the discipline or the norm.

(v) *Role*. It is the abstract of agents. In some emergent situations, we could define various special roles, for example, agents infected with viruses could be seen as the role of patient. Agents could dynamically play different roles according to various situations. Besides, roles should also be constrained by rules.

(vi) *Capability*. It is the ability of agents such as movement and interaction. Agents could own some natural capabilities, including self-adaptive capability though dynamically playing different roles according to various situations.

(vii) *Virus*. It is the entity that has not behavioral features and could be transmitted among agents by interactions or other ways. It could be considered as a property of agents to denote or indicate whether agents carry the virus. It is a very important concept for emergency management.

2.2. Formal Specification of the Zombie-City Model. A zombie-city model could be described by 5 tuples, as $Model ::= \langle AGENT, SN, EN, ROLE, RULE\rangle$, where

(i) $AGENT = \{a_1, a_2, \ldots, a_n\}$, for any $a_i (1 \le i \le n)$, where a_i is an agent and $AGENT$ is a finite set of agents.

(ii) $SN = \{l_k \mid 1 \le k \le (n-1)^2\}$, where l_k is a link of the social network, SN is a set of links, and social network is constructed by links between these agents.

(iii) $EN = \{g_1, g_2, \ldots, g_l\}$, where $g_i (1 \le i \le l)$ is a grid and EN is a finite set of grids.

(iv) $ROLE = \{b_1, b_2, \ldots, b_m\}$, for any $b_i (1 \le i \le m)$, where b_i is a role and $ROLE$ is a finite set of roles.

(v) $RULE = \{r_1, r_2, \ldots, r_k\}$, for any $r_i (1 \le i \le k)$, where r_i is a rule. $RULE$ is a finite rules set of rules. This set could be divided into four subsets: R_A, R_S, R_E, and R_R. R_A is a finite rule set for agents, R_S is a finite rule set for the social network, R_E is a finite rule set for the environment, and R_R is a finite rule set for roles.

We use CID to express the set of all the identifications, and cid is a specific one, that is, cid $\in$ CID.

(i) *Agent*. For any $a \in AGENT$, $a ::= \langle cid, AT, AC, N_l, N_r\rangle$, where

(a) cid is the identification of the agent;
(b) AT is the attributes of the agent;
(c) AC is the actions of the agent that the agent could do;
(d) N_l is a finite set of identifications of links that the agent is participating;
(e) N_r means a set of identifications of roles that the agent is playing.

(ii) *Role*. For any $r \in ROLE$, $r ::= \langle cid, AT, AC\rangle$, where

(a) cid is the identification of the role;
(b) AT is the attributes of the role;
(c) AC is the actions of the role that the role could do.

(iii) *Environment*. For any $e \in EN$, $e ::= \langle cid, AT, AC\rangle$, where

(a) cid is the identification of the environment;
(b) AT is the attributes of the environment;
(c) AC is the actions of the environment that the environment could do.

(iv) *Social Network*. For any $l \in SN$, $l ::= \langle cid, AT, (a_i, a_j)\rangle$

(a) cid is the identification of the link;

(b) AT is the attributes of the link;

(c) $(a_i, a_j) \in 2^{AGENT\times AGENT}$ denotes a link from agent a_i to agent a_j. For an undirected graph, $(a_i, a_j) = (a_j, a_i)$, and for directed graph, $(a_i, a_j) \neq (a_j, a_i)$.

(v) *Rule*. For any $u \in R_A \bigcup R_R \bigcup R_E \bigcup R_S$, $u ::= Condition \mid Event \rightarrow (AT \mid AC)$, where

(a) *Condition* is the internal state or attribute of the agent, role, environment, or link of the social network in the system;

(b) *Event* is the set of events that happens in the system;

(c) AT is the set of attributes of the agent, role, environment, or link of the social network in the system;

(d) AC is the set of actions of the agent, role, environment, or other actions in the system.

2.3. Social Network. For an undirected graph, $a \prec_t b\,(a, b \in AGENT)$ means that agent a links with agent b at moment t, which is equal to $b \prec_t a$. $LINK(a, t)$ denotes the set of agents in which agent a connects with at moment t, that is, $LINK(a, t) = \{b \mid a \prec_t b \wedge b \in AGENT\}$.

For a directed graph, let $a \prec_t b\,(a, b \in AGENT)$ denote that agent a links to b, and agent a is the source of this link. $LINK^S(a, t)$ denotes the set of agents that agent a connects, and agent a is the source of these links, that is, $LINK^S(a, t) = \{b \mid a \prec_t b \wedge b \in AGENT\}$. Let $LINK^T(a, t)$ indicate the set of agents that links to agent a at the moment t, and agent a is the target of these links, that is, $LINK^T(a, t) = \{b \mid b \prec_t a \wedge b \in AGENT\}$. Therefore, the set of agents that agent a connects at moment t contains the agents that agent a links to and the agents that link to agent a, that is, $LINK(a, t) = LINK^S(a, t) \bigcup LINK^T(a, t)$.

For describing the initialization process of a static social network and the growing process of a dynamic social network, we have defined two primitives: *Create*(), *Delete*(). $SN(t)$ denotes the set of social links in an artificial society at the moment t.

(i) $Create(a_i, a_j)$. For the undirected graph, $a_i \not\prec_t a_j$, that is, $a_j \notin LINK(a_i, t)$ and $a_i \notin LINK(a_j, t)$, $LINK(a_i, t+1) = LINK(a_i, t) \bigcup \{a_j\}$ and $LINK(a_j, t+1) = LINK(a_j, t) \bigcup \{a_i\}$; for the directed graph, $a_i \not\prec_t a_j$, $LINK^S(a_i, t+1) = LINK^S(a_i, t) \bigcup \{a_j\}$ and $LINK^T(a_i, t+1) = LINK^T(a_i, t)$, $LINK^S(a_j, t+1) = LINK^S(a_j, t)$, and $LINK^T(a_j, t+1) = LINK^T(a_j, t) \bigcup \{a_i\}$. For both two graphs, the link $l = (a_i, a_j)$ and $SN(t+1) = SN(t) \bigcup \{l\}$.

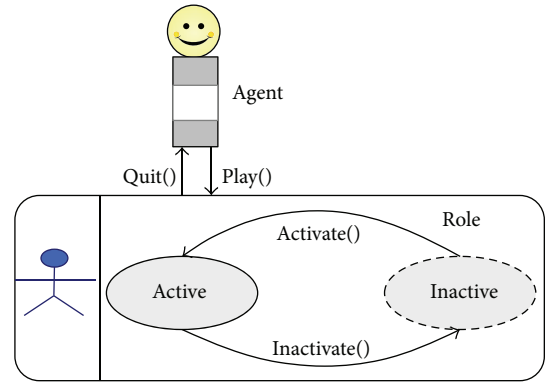


Figure 2: Mechanism of dynamically playing role.

(ii) $Delete(a_i, a_j)$. For the undirected graph, $a_j \in LINK(a_i, t)$ and $a_i \in LINK(a_j, t)$, $LINK(a_i, t+1) = LINK(a_i, t) \setminus \{a_j\}$ and $LINK(a_j, t+1) = LINK(a_j, t) \setminus \{a_i\}$; for the directed graph, $a_j \in LINK^S(a_i, t)$ and $a_i \in LINK^T(a_j, t)$, $LINK^S(a_i, t+1) = LINK^S(a_i, t) \setminus \{a_j\}$ and $LINK^S(a_j, t+1) = LINK^S(a_j, t)$, $LINK^T(a_i, t+1) = LINK^T(a_i, t)$ and $LINK^T(a_j, t+1) = LINK^T(a_j, t) \setminus \{a_i\}$. For both two graphs, the link $l = (a_i, a_j)$ and $SN(t+1) = SN(t) \setminus \{l\}$.

2.4. Self-Adaptive Mechanism of Agents. Agents have the capabilities of self-adaption though dynamically playing roles. Figure 2 presents the self-adaptive mechanism of dynamically playing roles. For example, if agent a is a student (i.e., agent a plays *student* role) and has been infected by an infectious disease, then agent a should adapt the change to transforming its role into *patient* role and go to a hospital. That means agent a will play the *patient* role, and the *student* role of agent a will be *inactive*. After agent a recovers, agent a will not play the *patient* role, that is, quit the *patient* role. Agent a will go back to play the *student* role, which means to *activate* the *student* role. The self-adaptive mechanism could be accurately described by formalization.

Let $a \in_t c$ denote that agent a plays c role at time t. Use $ROLE(a, t)$ to denote the set of roles that agent a plays at moment t, that is, $ROLE(a, t) = \{c \mid a \in_t c\}$. In order to describe the mechanism of dynamically playing roles, let $ROLE^A(a, t)$ indicate the set of active roles that agent a plays at moment t, and let $ROLE^I(a, t)$ denote the set of inactive roles that agent a plays at moment t. Therefore, the set of roles that agent a plays at moment t includes active roles and inactive roles, that is, $ROLE(a, t) = ROLE^A(a, t) \bigcup ROLE^I(a, t)$. In order to describe the process of dynamically playing roles, we define four primitives, including *Play*(), *Quit*(), *Activate*(), and *Inactivate*().

Play(c). $c \notin ROLE(a, t)$, $ROLE^A(a, t+1) = ROLE^A(a, t) \bigcup \{c\}$, and $ROLE^I(a, t+1) = ROLE^I(a, t)$. That means that the agent will play and join the role and the role is active.

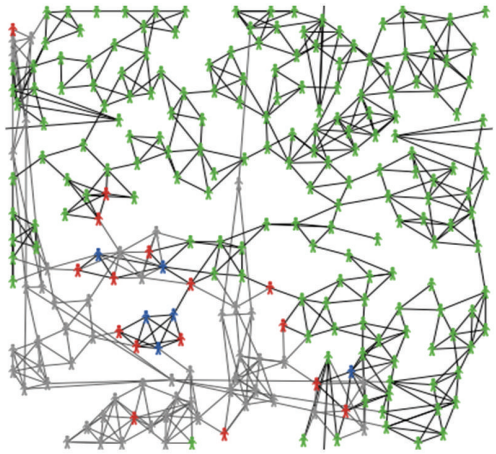


Figure 3: A snapshot of agents playing different roles (different colours) in the zombie-city model.

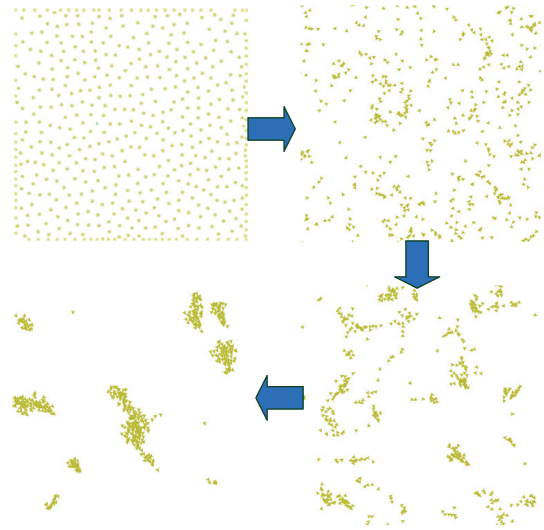


Figure 4: The processes of the flocking.

Quit(c). $c \in ROLE^A(a,t)$, $ROLE^A(a,t+1) = ROLE^A(a,t) \setminus \{c\}$, and $ROLE^I(a,t+1) = ROLE^I(a,t) \bigcup \{c\}$. That means that the agent will not play the role and the role will quit from the set of roles that the agent plays.

Activate(c). $c \in ROLE^I(a,t)$, $ROLE^I(a,t+1) = ROLE^I(a,t) \setminus \{c\}$, and $ROLE^A(a,t+1) = ROLE^A(a,t) \bigcup \{c\}$. That means that the agent will make the role active and behaviors of the agent will be affected by the role.

Inactivate(c). $c \in ROLE^A(a,t)$, $ROLE^A(a,t+1) = ROLE^A(a,t) \setminus \{c\}$, and $ROLE^I(a,t+1) = ROLE^I(a,t) \bigcup \{c\}$. That means that the agent will make the role inactive and behaviors of the agent will not be affected by the role.

As presented in Figure 3, in zombie-city model, agents play different roles (indicated by different colors) and the same agent could dynamically play various roles for adapting to different situations.

3. H7N9 Spreading Based on the Flocking Birds

3.1. Modeling with the Zombie-City Model. The processes of the spread of H7N9 include the infectious disease transmitting among these flocking birds, and then the dense infected birds may transmit H7N9 to the human. In this case, there are mainly four roles: *susceptible_bird, infected_bird, susceptible_person*, and *infected_person*.

The phenomenon of flocking birds is based on some simple rules without external or central controls. This process is a self-organized process. These simple rules could be constructed as follows.

(1) *Rule_Cohesion*. If the bird is far away from its neighbors (flockmates) (*max_distance* is the threshold of the max distance from its neighbors), then this bird will turn towards its neighbors. The parameter *average_flockmates* means the average direction of the nearest neighbors, and *max_cohesion* means the adjusting degree of the direction of this bird. This rule could be described as follows: *Distance(self, nearest-neighbor) > max_distance → turn_towards(average_flockmates, max_cohesion).*

(2) *Rule_Separation*. If the bird is too close to the nearest neighbor (*min_separation* is the threshold of the min distance from the nearest neighbor), then turn away from this neighbor. This rule could be described as follows: *Distance(self, nearest-neighbor) < min_separation → turn_away(nearest_neighbor, max_separation).*

(3) *Rule_Alignment*. Keep the direction of the bird with the flockmates, and then turn towards its neighbors. The parameter *max_ alignment* means the adjusting degree of the direction of this bird. This rule could be described as follows: *Distance(self, nearest-neighbor) < max_distance || Distance(self, nearest-neighbor) > min_separation → turn_towards(average_flockmates, max_alignment).*

As shown in Figure 4, based on these three rules, the flocking will appear.

As we know, H7N9 is highly pathogenic for birds; that is, the infectious disease H7N9 could easily infect the birds. If some birds interact with the bird infected with H7N9 and the distance between these birds is much closed, then all of these birds will be infected with H7N9. So the rule of the disease spreading between birds could be described as follows.

(4) *Rule_Birds_Infected*. If the infected bird is closed with other birds and the distance from the infected bird (*Infected_bird*) to other birds without infection (*Susceptible_bird*) is less than a threshold (*min_distance*), then all these birds without infection will be infected with H7N9, that is, $\forall a \in_t$ *Infected_bird* $\forall b \in_t$ *Susceptible_bird*

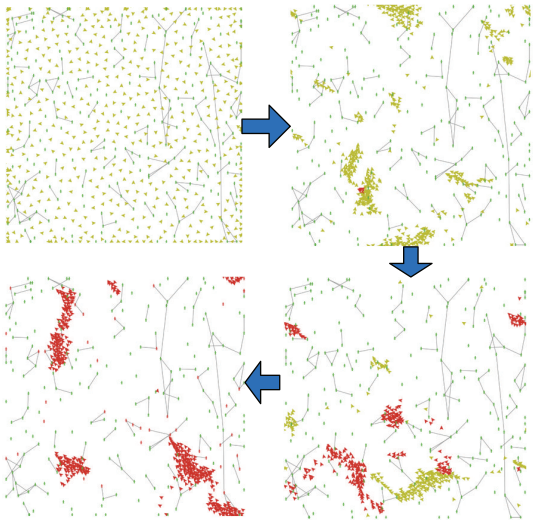


Figure 5: The processes of H7N9 transmission.

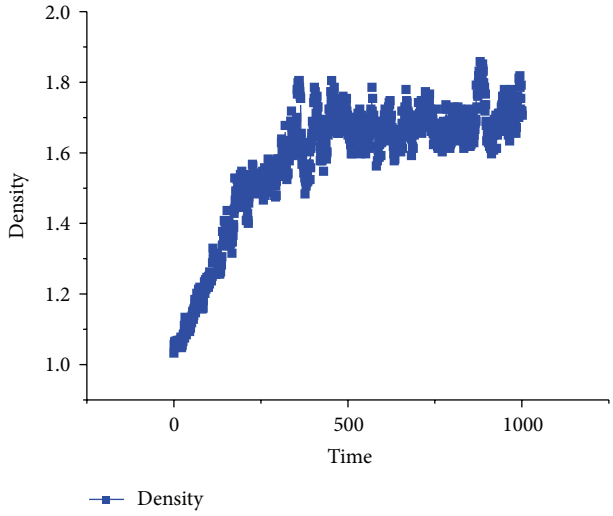


Figure 6: Density of the flocking birds.

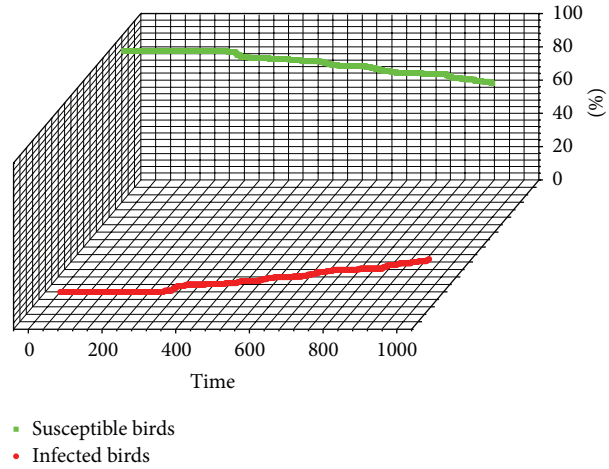


Figure 7: The epidemic status of the flocking birds.

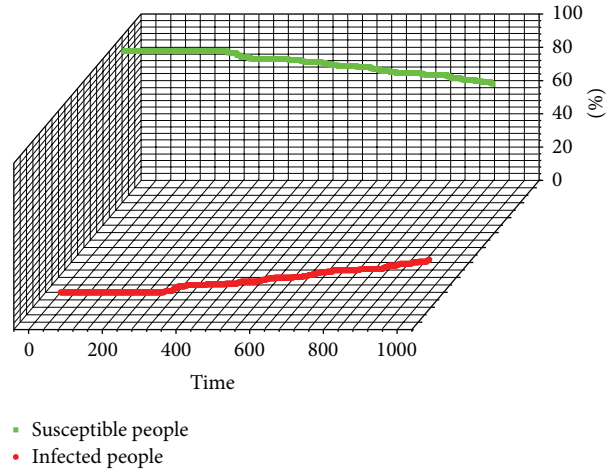


Figure 8: The epidemic status of the human.

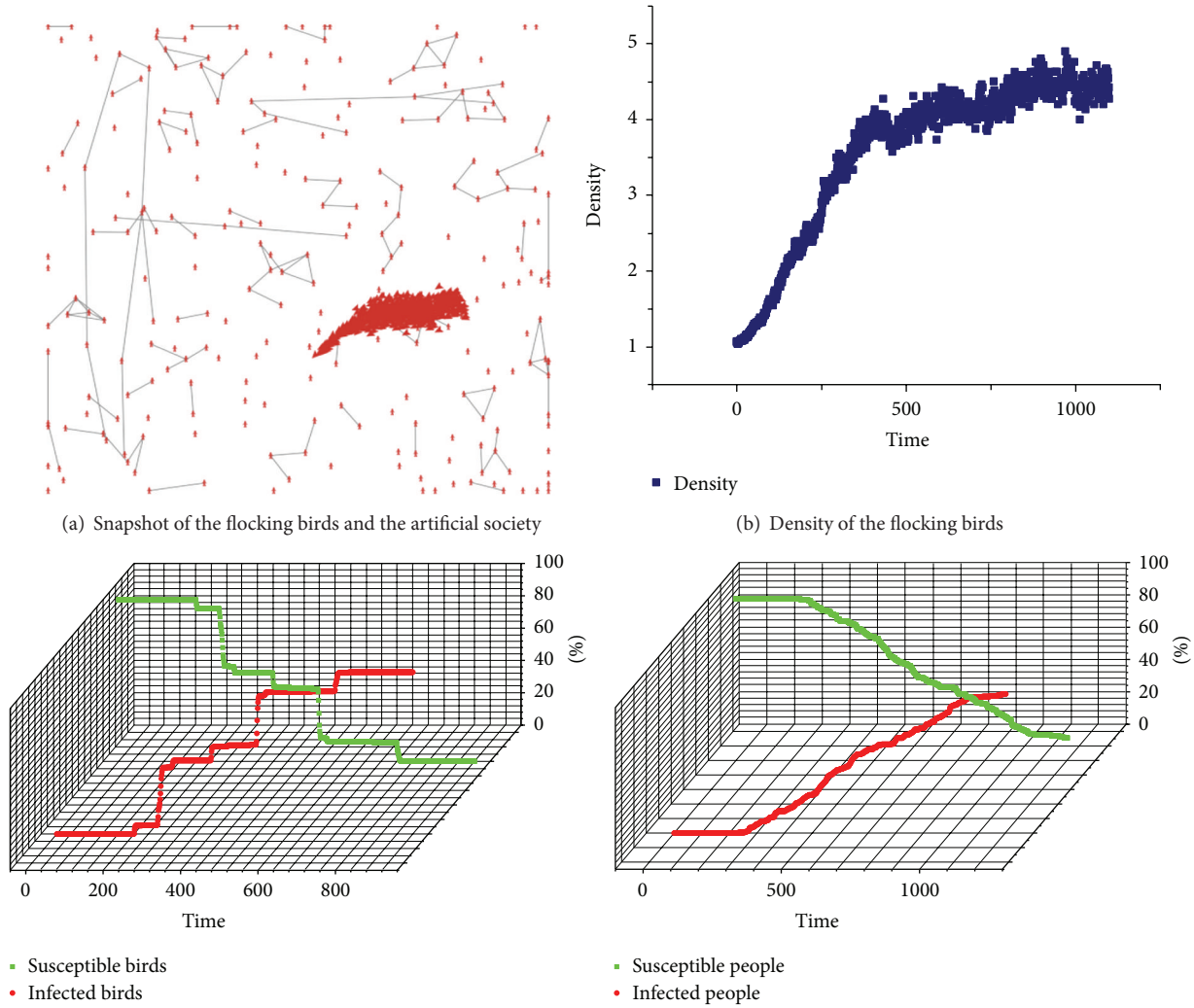
Distance(a,b) < *min_distance* → *b.Quit(Susceptible_bird)* ∧ *b.Play(Infected_bird)*.

As the recent research shows, the infectious disease H7N9 has almost spreaded from the wild birds to the human, and there is no evidence of ongoing human-to-human infection [5, 7, 8]. It could be assumed that human-to-human transmission is only happening between intimate people and the probability of infection is very small. For people, the number of strong social relationships of a person is also very small, and in this case we assume the number to be 1. Usually, these intimate people live nearby. So the rule of this social network of artificial society could be described as follows.

(5) *Rule_Social_Network*. This social network is an undirected graph, and an agent connects with the nearest agent which has not connected with it, that is, *Average_degree* < *1* → $(\forall a, c \in_t$ *(Infected_person* ∨ *Susceptible_person)* $\exists b \in_t$ *(Infected_person* ∨ *Susceptible_person)* $(a,c) \notin LINK^t(a,t) \wedge (a,b) \notin LINK(a,t) \wedge$ *Distance(a,b)* < *Distance(a,c)* → *Create(a,b))*.

Compared with the flocking birds, the positions of people in the environment could be seen as static. The wild flocking birds with H7N9 viruses will infect the human, but there are some conditions: in the unit grid of a person living, there are a number of infected birds (the number is *threshold_infected_human*).

(6) *Rule_Birds_Human_Infected*. If the agent without infection is habiting in a gird and in the gird there are enough infected birds (*threshold_infected_human*), then the agent will be infected, that is, $\forall a \in_t$ *Susceptible_person* $\exists b \in$ *Environment (Habit(a,b)* ∧ *Count(Infected_birds,b)* ≥ *threshold_infected_human)* → *a.Quit(Susceptible_person)* ∧ *a.Play(Infected_person)*.



(a) Snapshot of the flocking birds and the artificial society

(b) Density of the flocking birds

(c) The epidemic status of the flocking birds

(d) The epidemic status of the human

FIGURE 9: The simulation results with the parameter *min_separation* = 0.25.

Although there is no evidence of ongoing human-to-human transmission, we could not exclude the small probability of human-to-human transmission through the intimate social network.

(7) *Rule_Human_to_Human*. If an infected agent a has connected with a susceptible agent b and the randomly produced number is less than the threshold (*infected_chance*), then this agent b will be infected with H7N9, that is, $\forall a \in_t$ *Infected_person* $\forall b \in_t$ *Susceptible_person* $((a,b) \in LINK(a,t) \wedge$ *Random()* < *infected_chance*$) \rightarrow$ *b.Quit(Susceptible_person)* $\wedge$ *b.Play(Infected_person)*.

3.2. Experiments. Based on these defined rules, we could do simulations and see the situations of the spread of this infectious disease H7N9.

(i) *Agents*. The population of agents is 750. The number of the humans is 250, and the number of birds is 500.

(ii) *Environments*. 32 × 32 grids.

(iii) *Role*. *Susceptible_bird (yellow color)*, *susceptible_person (green color)*, *infected_bird (red color)*, and *infected_person (red color)*.

We define the parameters as follows: *min_separation* = *max_distance* = *0.5*, *max_cohesion* = *3*, *max_alignment* = *5*, *max_separation* = *1.5*, *infected_chance* = *0.1*, *threshold_infected_human* = *5*, and the average of the social network of the human is 1. Figure 5 shows the processes of H7N9 transmission between the flocking birds and the human.

As shown in Figure 6, the density of the flocking birds firstly linearly increases, and then it fluctuates around 1.7. This density is related to the parameter *min_separation*.

Figure 7 shows the status of the H7N9 spreading between the flocking birds. At the ticks (200), a bird was infected with H7N9, and then the viruses transmitted among these birds quickly. At the ticks (400), all of these birds were infected with H7N9.

After about 80 ticks of the first bird infection with H7N9, the first person was infected with H7N9 by the dense flocking birds. The number of the infected people is increasing slowly. The birds-to-human transmission is the main way for H7N9 transmitting from birds to the human, and the epidemic status of the human could be shown in Figure 8.

How does the parameter *min_separation* affect the flocking birds and even the transmission of H7N9? We could adjust the parameter *min_separation* to 0.25. It means that the flocking birds will fly closed and the density of birds will increase. As shown in Figure 9, (a) presents the screenshot of the simulation; (b) shows the density of the flocking birds, and the density linearly increased and fluctuated around 4.5; (c) and (d) show the epidemic statuses of the flocking birds and the human, respectively.

As shown in Figure 9, we could clearly see that after the first bird was infected with H7N9 at the 200 ticks and at the 755 ticks, all of the birds were infected with H7N9. Because a small number of birds were fling separatly and very closly, the birds in some small flocking were not infected with H7N9 until these birds met with the bigger flocking. Moreover, in human, more people were infected with H7N9 for the bigger density of birds or the denser flocking birds. During a same time, the number of infected people with *min_separation* = 0.25 is four times that of the number of infected people with *min_separation* = 0.5. Above all, we could know that the density of the flocking birds is related to the H7N9 epidemic in the human.

4. Conclusion

The phenomenon of the flocking is a natural beautiful scene. This phenomenon is organized by the birds without external and central controls. However, the flocking also has some problems, for example, the infectious disease H7N9 transmission. In order to aid analysis and modeling of the flocking and the disease transmission, this paper proposes a new artificial society model, zombie-city model, which includes five key concepts: agent, role, environment, social network, and rule. Based on this model, three simple rules of the flocking and other rules have been described. Then, this paper has studied the infectious disease H7N9 transmission among the flocking birds and the human. The simulation results show that a larger density of the flocking could accelerate the spread of the infectious disease H7N9 transmission from the flocking to the human and could slow down the spread of the H7N9 in the flocking birds.

The next works will include the following. (1) We will continue to perfect this zombie-city model, especially the description of an emergence. (2) We will continue with the flocking algorithm based on the zombie-city model and its effects on the human.

Acknowledgments

The National Nature and Science Foundation of China under Grants nos. 91024030, 61133001, and 61070034, the Program for New Century Excellent Talents in University, and the China Scholarship Council (CSC) support this work. The authors also gratefully express their sincere thanks to Professor Xiaogang Qiu, Dr. Bin Chen, and Dr. Yuanzheng Ge in College of Information System and Management, the National University of Defense Technology.

References

[1] G. Spedding, "The cost of flight in flocks," *Nature*, vol. 474, no. 7352, pp. 458–459, 2011.

[2] A. Bellaachia and A. Bari, "Flock by leader: a novel machine learning biologically inspired clustering algorithm," in *Advances in Swarm Intelligence*, vol. 7332 of *Lecture Notes in Computer Science*, pp. 117–126, Springer, Berlin, Germany, 2012.

[3] M. Nagy, Z. Ákos, D. Biro, and T. Vicsek, "Hierarchical group dynamics in pigeon flocks," *Nature*, vol. 464, no. 7290, pp. 890–893, 2010.

[4] J. R. Usherwood, M. Stavrou, J. C. Lowe, K. Roskilly, and A. M. Wilson, "Flying in a flock comes at a cost in pigeons," *Nature*, vol. 474, no. 7352, pp. 494–497, 2011.

[5] WHO, "Human infection with avian influenza A(H7N9) virus in China—update," http://www.who.int/csr/don/2013_04_19/en/index.html.

[6] M. Tang, X. Mao, and H. Zhou, "Zombie-city: a new artificial soceity model," *Journal of Computational Information Systems*, vol. 9, no. 12, pp. 4989–4996, 2013.

[7] R. Gao, B. Cao, Y. Hu et al., "Human infection with a novel avian-origin influenza A, (H7N9) virus," *The New England Journal of Medicine*, vol. 368, no. 20, pp. 1888–1897, 2013.

[8] T. M. Uyeki and N. J. Cox, "Global concerns regarding novel influenza A (H7N9) virus infections," *The New England Journal of Medicine*, vol. 368, no. 20, pp. 1862–1864, 2013.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 793013, 13 pages
http://dx.doi.org/10.1155/2013/793013

Research Article

Memory-Based Multiagent Coevolution Modeling for Robust Moving Object Tracking

Yanjiang Wang, Yujuan Qi, and Yongping Li

College of Information and Control Engineering, China University of Petroleum, No. 66, Changjiang West Road, Economic and Technological Development Zone, Qingdao 266580, China

Correspondence should be addressed to Yujuan Qi; qiyj@upc.edu.cn

Received 28 March 2013; Accepted 28 April 2013

Academic Editors: P. Agarwal, S. Balochian, V. Bhatnagar, J. Yan, and Y. Zhang

The three-stage human brain memory model is incorporated into a multiagent coevolutionary process for finding the best match of the appearance of an object, and a memory-based multiagent coevolution algorithm for robust tracking the moving objects is presented in this paper. Each agent can remember, retrieve, or forget the appearance of the object through its own memory system by its own experience. A number of such memory-based agents are randomly distributed nearby the located object region and then mapped onto a 2D lattice-like environment for predicting the new location of the object by their coevolutionary behaviors, such as competition, recombination, and migration. Experimental results show that the proposed method can deal with large appearance changes and heavy occlusions when tracking a moving object. It can locate the correct object after the appearance changed or the occlusion recovered and outperforms the traditional particle filter-based tracking methods.

1. Introduction

The problem of object tracking is often posed as that of estimating the trajectory of objects in an image plane as objects move in a scene [1]. Although considerable efforts have been made in establishing a robust tracking framework in the research literature, the problem still remains challenging when appearance abrupt changes or occlusions occur. To address these challenges, in the literature tremendous attempts have been made in characterizing appearance models which are able to handle appearance changes. In this context, most of the extant methods tend to apply a total model updating mechanism for template updating in which the initial template model is updated gradually based on the estimated information, for example particle filters (PF). However, if an object is heavily occluded or its appearance changes abruptly, the total model updating based PF (TMU-PF) will gradually deviate from the target.

Recently, a lot of modifications have been made for improving the performance of particle filters. For example, Zhou et al. [2] presented an approach that incorporated appearance-adaptive models to stabilize the tracker. They made three extensions: (a) an observation model arising from an adaptive appearance model, (b) an adaptive velocity motion model with adaptive noise variance, and (c) an adaptive number of particles. Li et al. [3] proposed a robust observation model to address appearance changes. Wang et al. [4, 5] developed an SMOG appearance model and an SMOG-based similarity measure to deal with appearance variations. Zhang et al. [6] embedded an adaptive appearance model into a particle filter to address the appearance changes and proposed an occlusion handling scheme to deal with occlusion situations. On the other hand, some researchers have incorporated other optimization algorithms into particle filer to enhance the performance. For example, in [7], CamShift was used into the probabilistic framework of particle filter as an optimization scheme for proposal distribution such that both the tracking robustness and computational efficiency are improved. Shan et al. [8] incorporated the mean-shift (MS) optimization algorithm into a particle filter framework to improve the sampling efficiency. Zhou et al. [9] presented a scale invariant feature transform (SIFT) based mean shift algorithm for object tracking, which improved the tracking performance of the classical mean shift and SIFT tracking algorithms in complicated real scenarios. Zhao and Li [10] applied particle swarm optimization (PSO) to find

high likelihood areas where the particles could be distributed even though the dynamic model of the object could not be obtained. Zhou et al. [11] combined multiband Generalized Cross Correlation, KF, and Weighted Probabilistic Data Association within the particle filtering framework, which improves the performance of the algorithm in noisy scenarios. Most of the above methods applied a total model updating mechanism for template updating in which the initial template model is updated gradually based on the estimated information by particle filters. However, if an object is heavily occluded or its appearance changes abruptly, the total model updating based PF (TMU-PF) will gradually deviate from the target.

To tackle the drawback of the TMU-PF, Montemayor et al. [12] introduced memory strategies into PF to store the states of particles, which can deal with some occlusion situations. Mikami et al. [13] proposed a memory-based particle filter (MPF) to handle facial pose variation by predicting the prior distribution of the target state in future time steps. However, both of the methods are neither biologically motivated nor cognitively inspired. They just apply memory to store the states of particles and could not cope with situations with sudden changes.

It is well known that humans can track and recognize an object with little difficulty in the case of appearance changes and partial occlusions. This capability of human beings benefits from the human's memory system. When humans perceive something, the related information which is stored in their memory can be recalled. As a function of information retention organs in the brain, the mechanism of memory system has been extensively studied in neural science, biopsychology, cognitive science, and cognitive informatics [14, 15].

Inspired by the way humans perceive the environment, in this paper, we present a memory-based multiagent coevolution model for tracking the moving objects. The three-stage human brain memory mechanism is incorporated into a multiagent coevolutionary process for finding a best match of the appearance of the object. Each agent can remember, retrieve, or forget the appearance of the object through its memory system by its own experience. A number of such memory-based agents are randomly distributed nearby the located object region and then mapped onto a 2D lattice-like environment for predicting the new location of the object by their coevolutionary behaviors, such as competition, recombination, and migration. Experimental results show that the proposed method can deal with large appearance changes and heavy occlusions when tracking a moving object. It can locate the correct object after the appearance changed or the occlusion recovered.

The remainder of this paper is organized as follows. In Section 2, we will first propose the memory-based multiagent coevolution model including the definitions of each behavior involved. Section 3 gives the detailed description of the memory modeling of an agent and the object appearance template updating process for each agent. Then the color object modeling and the proposed tracking algorithm are described in Section 4. Finally, the performance of our tracking algorithm is verified on different standard video sequences and some conclusions are summarized in Sections 5 and 6.

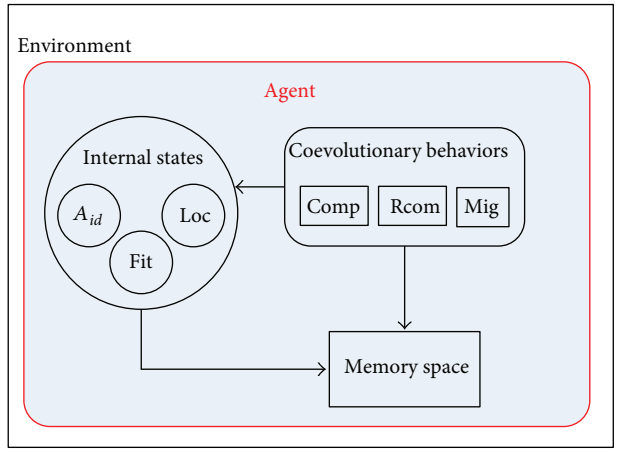


Figure 1: Memory-based agent model.

2. Memory-Based Multiagent Coevolution Modeling

2.1. Memory-Based Multiagent Model. According to [16], an agent can be defined as an intelligent entity that resides in an environment and can act autonomously and collaboratively. It is driven by certain purposes and has some reactive behaviors. Based on this idea, many agent-based applications are reported during past years, such as image feature extraction [17], image segmentation [18], and optimization problems [19–24]. In our previous work [25, 26], we also proposed an evolutionary agent model for color-based face detection and location.

In this paper, we will present a memory-based multiagent model (MMAM) for moving object tracking. Each agent represents a candidate target region in a video frame; it lives in a lattice-like environment, and its main task is to compete or cooperate with its neighbor agents to continuously improve its own fitness by exhibiting its behaviors.

The schematic diagram of the proposed MMMA is shown in Figure 1.

More specifically, the memory-based multiagent model (MMAM) for object tracking can be defined as a 7-tuples: $\langle A_{id}, \text{Loc}, \text{Fit}, \text{MS}, \text{Comp}, \text{Rcom}, \text{Mig}\rangle$. Where A_{id} denotes the identity of an agent; Loc represents the position of an agent in the image, that is, the center of a candidate target; Fit symbolizes its fitness, which is defined by the similarity between the candidate target and the object template; and MS = {USTMS, STMS, LTMS} is a set of hominine memory spaces of an agent for information storage, where USTMS, STMS, and LTMS stand for the ultrashort-term memory space, short-term memory space, and long-term memory space, respectively.

The above 4 parameters describe the internal states of an agent. While Comp, Rcom, and Mig describe the external coevolutionary behaviors of an agent, where Comp represents

the competition behavior, Rcom denotes the recombination behavior, while Mig refers to the migration behavior.

Suppose all the agents inhabit in a lattice-like environment, A, which is called an agent lattice, as shown in Figure 2. Each agent is fixed on a lattice point and it can only interact with its 4 neighbors. The size of A is $N \times N$ and the agent located at (i, j) is denoted by $A_{i,j}$, $i, j = 1, 2, \ldots, N$. Each agent can compete or cooperate with its 4 neighbors in order to improve its fitness.

The mapping process is described as follows.

First, it randomly generates $N \times N$ agents near the located object region at begining. The first generated agent is placed at $A_{1,1}$, the second agent is placed at $A_{1,2}, \ldots$, the Nth agent is placed at $A_{1,N}$, the $(N+1)$th agent is placed at $A_{2,1}, \ldots$, and the final agent $(N \times N)$th is placed at $A_{N,N}$. The neighbors of agent $A_{i,j}$ are defined as $Nb_{i,j} = \{A_{i-1,j}, A_{i+1,j}, A_{i,j-1}, A_{i,j+1}\}$. For the agents at the four edges of the lattice, we define

$$\begin{aligned} A_{0,j} &= A_{N,j}, \qquad A_{i,0} = A_{i,N}, \\ A_{N+1,j} &= A_{1,j}, \qquad A_{i,N+1} = A_{i,1}. \end{aligned} \tag{1}$$

According to the above definition, the neighbors of an agent on the lattice are not its real neighbors in the video image. Because each agent is generated randomly and can only evolve with its neighbors on the lattice-like environment, the mapping process can also be thought as a natural selection before their coevolution.

2.2. Multiagent Coevolutionary Behaviors. There are three coevolutionary behaviors for each agent, that is, competition, recombination, and migration. The three behaviors are defined as follows.

Definition 1 (Comp (competition behavior)). Comp means that an agent will contend with other agents for its survival.

For each agent $A_{i,j}$, if $\mathrm{Fit}(A_{i,j}) < \mathrm{Fit}(Nb\max_{i,j})$, where $Nb\max_{i,j}$ is the agent with maximum fitness among its 4 neighbors, then $A_{i,j}$ will be replaced by the following:

$$A^l_{i,j} = \begin{cases} \underline{l}, & \left(Nb\max^l_{i,j} + U(-1,1)\right) \\ & \times \left(Nb\max^l_{i,j} - A^l_{i,j}\right)\Big) < \underline{l}, \\ \bar{l}, & \left(Nb\max^l_{i,j} + U(-1,1)\right) \\ & \times \left(Nb\max^l_{i,j} - A^l_{i,j}\right)\Big) > \bar{l}, \\ Nb\max^l_{i,j} + U(-1,1) & \\ \times \left(Nb\max^l_{i,j} - A^l_{i,j}\right), & \text{otherwise}, \end{cases} \tag{2}$$

where $U(-1,1)$ is a uniform random number in $[-1,1]$, l denotes the location of agent $A_{i,j}$ in the video frame, $l = (x, y)$, $L = [\underline{l}, \bar{l}]$ represents the whole searching space, that is, the video size, $\underline{l} = [\underline{x}, \underline{y}]$, $\bar{l} = [\overline{x}, \overline{y}]$.

Definition 2 (Rcom (recombination behavior)). Rcom means that an agent may exchange the x or y coordinate with other agents. It is similar to the crossover operator in genetic algorithms.

For each agent $A_{i,j}$, given a recombination probability P_r, if $U(0,1) < P_r$, exchange the x or y coordinate of $A_{i,j}$ and $Nb\max_{i,j}$, a new agent will be created, $Ar_{i,j}$. If $\mathrm{Fit}(A_{i,j}) > \mathrm{Fit}(Ar_{i,j})$, $A_{i,j}$ will continue to exist in the lattice; otherwise it will be replaced by the following:

$$Ar_{i,j} = \begin{cases} \left(A^x_{i,j}, Nb\max^y_{i,j}\right), & U(0,1) < 0.5, \\ \left(Nb\max^x_{i,j}, A^y_{i,j}\right), & \text{else}. \end{cases} \tag{3}$$

Definition 3 (Mig (migration behavior)). Mig means that an agent can move to another location by some random steps in the image other than the lattice it locates at. It is similar to the mutation operator in genetic algorithms.

For each agent $A_{i,j}$, the migration behavior will occur according to a migration probability P_m. if $U(0,1) < P_m$, $A_{i,j}$ will be replaced by the following:

$$Am^l_{i,j} = \begin{cases} \underline{l}, & A^l_{i,j} + U^l(-10,10) < \underline{l}, \\ \bar{l}, & A^l_{i,j} + U^l(-10,10) > \bar{l}, \\ A^l_{i,j} + U^l(-10,10), & \text{otherwise}, \end{cases} \tag{4}$$

where $U(-10,10)$ is a uniform random number in $[-10,10]$; that is, the migration steps are randomly generated within $(-10,10)$ pixels for i and j, respectively.

3. Memory Modeling for an Agent

3.1. Three-Stage Human Brain Memory Modeling for Appearance Updating. As a faculty of information retention organs in the brain, memory has been intensively studied in psychology, neural science, and cognitive science, and several memory models have been proposed since the late 19th century. In 1890, James first divided the human memory into three components: after-image memory, the primary memory, and the secondary memory [27]. Atkinson and Shiffrin modeled the human memory as a sequence of three stages: the sensory memory, short-term memory, and long-term memory [28] (also known as the multistore model). Baddeley and Hitch proposed a multicomponent model of working memory where a central executive responsible for control processes and two slave systems providing modality-specific buffer storage [29]. Recently, Wang proposed a logical architecture of memories in the brain which includes four parts: (a) the sensory buffer memory; (b) the short-term memory; (c) the long-term memory; and (d) the action buffer memory [15, 30]. According to contemporary cognitive psychology, the popular model of a basic human brain memory includes three stages: ultrashort-term memory (USTM), short-term memory (STM), and long-term memory (LTM), as shown in Figure 3 [31].

Each stage includes three processes: (a) encoding, (b) storage, and (c) retrieval. "Encoding" (also referred to as registration) is the process of forwarding physical sensory input into one's memory. It is considered as the first step in memory information processing. "Storage" is the process of retaining information whether in the sensory memory,

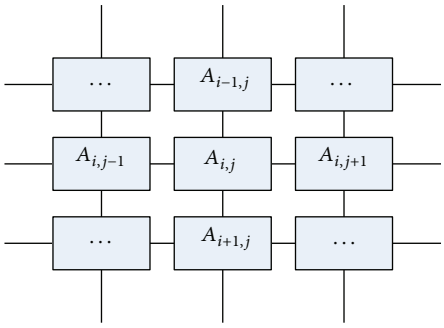


Figure 2: Model of the agent lattice.

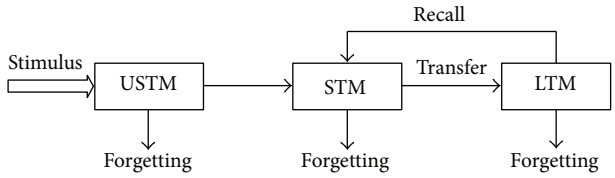


Figure 3: Three-stage human brain memory model.

the short-term memory, or the more permanent long-term memory. "Retrieval" (also referred to as "recall") is to call back the stored information in response to some cues for use in a process or activity.

The memorization process can be described as follows.

(1) USTM is used to store the basic cognitive information.

(2) STM, which in the recent literature has been referred to as working memory, is used to make decision. The information stored in STM includes the new information from USTM, the information processed in STM, or the information recalled from LTM. Therefore, STM can be considered as a complicated system for information storing and processing.

(3) LTM is a library used to store experienced knowledge which can inspire the individual to recall every thing that had happened, cognize all kinds of models, and solve problems (e.g., tracking problems in our work).

(4) Forgetting is a special function of memory which helps the information either not always recalled or not commonly used to be lost from memory.

According to the above three-stage human memory model, the appearance template updating model of an agent can be described as shown in Figure 4,

where the input of the model is the candidate template estimated by the Loc of an agent in the current video frame while the output is the updated template for prediction in the next frame. USTMS, STMS, and LTMS represent the three-stage memories, respectively. They are defined as follows.

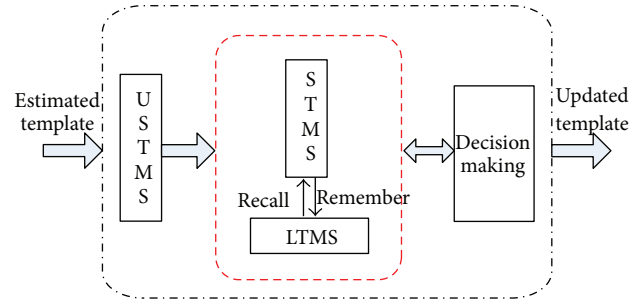


Figure 4: Three-stage memory model for appearance template updating.

Definition 4 (memory space (MS)). A 3-tuple which is used to store the current estimated appearance template and the past templates. Each element in MS is a memory space:

$$\mathrm{MS} = \{\mathrm{USTMS}, \mathrm{STMS}, \mathrm{LTMS}\}. \quad (5)$$

Definition 5 (USTMS). A one-element set for storing the estimated model p in the current video frame, which simulates the stage of ultrashort-term memory of human brain:

$$\mathrm{USTMS} = \{p\}. \quad (6)$$

Definition 6 (STMS). A set of K_s temporary templates, which imitates the stage of short-term memory of human brain. Let q_i denote the ith template in STMS; then

$$\mathrm{STMS} = \{q_i,\ i = 1, 2, \ldots, K_s\}. \quad (7)$$

Definition 7 (LTMS). A set of K_l remembered templates, which simulates the dynamic stage of the long-term memory of human brain. Let q_{Mj} stand for the jth remembered template in LTMS:

$$\mathrm{LTMS} = \left\{q_{Mj},\ j = 1, 2, \ldots, K_l\right\}. \quad (8)$$

The templates stored in STMS include the estimated template transferred from USTMS, the updated templates in STMS, or the templates recalled from LTMS.

According to the theory of cognitive psychology, only the information which is stimulated repeatedly can be stored into LTMS. Therefore, we define a parameter β for each template in STMS to determine whether the templates in STMS can be stored into LTMS or not, where β is a counter indicating the number of successful matches. The bigger β is, the more probably the template can be stored into LTMS.

More specifically, for all $q_i \in \mathrm{STMS}$, $i = 1, 2, \ldots, K_s$, If $q_i.\beta > T_M$ (a predefined threshold), the template will be remembered and stored into LTMS.

The process of template updating can be briefly described as follows.

First, the estimated template of the current frame is stored into USTMS and checked against the current template in STMS (the first one). If they are matched, update the template; otherwise check against the remaining templates in STMS and then LTMS in turn for a match. If a match exists, it will

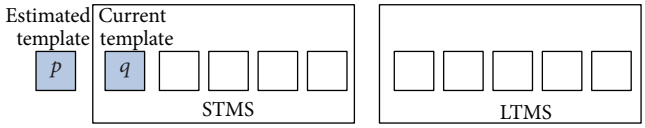


Figure 5: Initialization step.

be selected for the new template. Meanwhile the STMS and LTMS are updated by some behaviors, such as remembering, recall, and forgetting. These behaviors are defined as follows.

Definition 8 (remembering). An action that a template is stored into LTMS.

If there is no match in STMS and LTMS, and the STMS is full and the last template in STMS (denoted by q_{K_s}) is satisfied with $q_{K_s}.\beta > T_M$, then q_{K_s} will be remembered into LTMS and replaced by q_{K_s-1}. In such a circumstance, the estimated template will be reserved for the next estimation.

Definition 9 (recall). An action that a matched template is loaded from LTMS.

If a match is found in LTMS, the matched template will be extracted and used as the current object template.

Definition 10 (forgetting). An action that a template is removed from either of STMS or LTMS.

If the LTMS is full and $q_{K_s}.\beta > T_M$, the oldest template in LTMS will be forgotten in order to remember q_{K_s}.

3.2. Detailed Description of Memory-Based Appearance Updating. According to the above model, the memory-based appearance template updating algorithm can be described as follows.

Step 1 (Initialization). For each agent, store the estimated template (candidate object) p into the USTMS and the current template q into the STMS; set $q.\beta = 1$ and the LTMS to be empty, where p and q are determined by the initial target region, as shown in Figure 5. It is worth mentioning that the STMS and LTMS will be filled up gradually after several time steps during tracking.

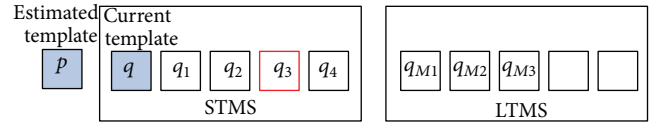
Step 2. Calculate the similarity coefficient $\rho = \rho[p, q]$, if $\rho > T_{dc}$, update the current object template by the following:

$$q = (1 - \alpha)\, q + \alpha \cdot p,$$
$$q.\beta = q.\beta + 1, \tag{9}$$

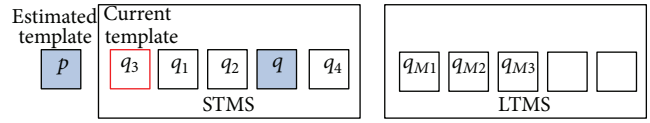
where T_{dc} is a predefined threshold for current template matching and α is the updating rate.

Step 3. If $\rho \le T_{dc}$, check against the remaining templates in STMS for a match, if

$$\rho\left[p, q_i\right] > T_{ds}, \quad i = 1, \ldots, K_s - 1, \tag{10}$$

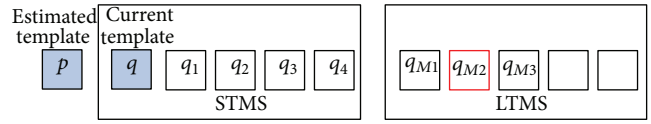


(a) A match is found in STMS

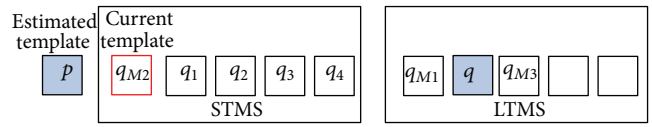


(b) Updating STMS

Figure 6: Illustration of updating process in STMS.



(a) A match is found in LTMS



(b) Updating STMS and LTMS

Figure 7: Illustration of recalling and remembering.

update the matched template by the following:

$$q_i = (1 - \alpha) \cdot q_i + \alpha \cdot p,$$
$$q_i.\beta = q_i.\beta + 1, \tag{11}$$

where T_{ds} is the threshold for template-matching in STMS.

Then, exchange the current template and the matched one, as shown in Figure 6.

For example, if q_3 is a matched template found in STMS (as shown in Figure 6(a)), then it will be moved to the top location in STMS and used as the current template, while the previous current template q will be moved to the original location of q_3 as shown in Figure 6(b).

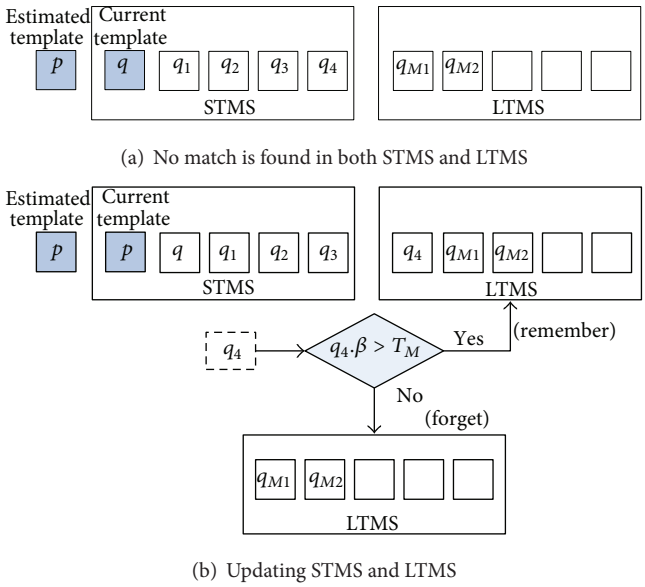
Step 4. If $\rho[p, q_i] \le T_{ds}$, check in LTMS for a match, if

$$\rho\left[p, q_{Mj}\right] > T_{dl}, \quad j = 1, \ldots, K_l, \tag{12}$$

where T_{dl} is the threshold for template-matching in LTMS. Then update the matched template by the following:

$$q_{Mj} = (1 - \alpha)\, q_{Mj} + \alpha \cdot p,$$
$$q_{Mj}.\beta = q_{Mj}.\beta + 1, \tag{13}$$

and then recall the matched one to use as the new object template and remember the current template q, as shown in Figure 7.



(a) No match is found in both STMS and LTMS

(b) Updating STMS and LTMS

FIGURE 8: Illustration of updating STMS and LTMS when no match is found in both memory spaces.

Step 5. If $\rho[p, q_{Mj}] \leq T_{dl}$, it means that there is no any match in STMS and LTMS. The estimated template p is stored into STMS and used as the new object template (set $p.\beta = 1$), as seen in Figure 8. Meanwhile, if the STMS reaches its maximum capacity, remember or forget the oldest template in STMS (i.e., q_{K_s-1}) by the following substeps.

(1) If $q_{K_s-1}.\beta > T_M$ and the LTMS is full, forget the oldest template in LTMS (i.e., q_{MK_l}) and remember q_{K_s-1}.

(2) If $q_{K_s-1}.\beta \leq T_M$, forget q_{K_s-1}.

As shown in Figure 8, when no match is found in both memory spaces, the current estimated template p is stored into STMS, while q_4 (i.e., $K_s - 1 = 4$) is either remembered ($q_4.\beta > T_M$) or forgotten ($q_4.\beta \leq T_M$).

Note that the templates in STMS and LTMS are stored in chronological order; that is, if a template is stored into STMS or LTMS earlier, it will move to the subsequent locations in order to make rooms for the newly reached templates.

4. Moving Object Tracking by MMAM

4.1. Object Detection and Modeling. To detect a color object, it is very important to obtain an effective color model to accurately represent and identify the object under various illumination conditions. In this paper, we use a histogram-based nonparametric modeling technique in YCbCr color space to model an object [32], which is much robust to lighting variations.

Giving the distribution of colors in an object region, let $px_{i,j}$, be a pixel location inside the object region with the origin at the center of the object region, the non-parametric distribution of the object, Q, can be represented by the following [32]:

$$Q = \{q_u; u = 1, 2, \ldots, m\}, \tag{14}$$

where

$$q_u = C \sum_{i=1,j=1}^{x,y} k\left(\left\|px_{i,j}\right\|^2\right) \delta\left[b\left(px_{i,j}\right) - u\right], \tag{15}$$

where k is the Epanechnikov kernel function, δ is the Kronecker delta function, and the function $b : R^2 \rightarrow \{1, \ldots, m\}$ associates the pixel at location $px_{i,j}$ with its color's index $b(px_{i,j})$ in the histogram. The normalization constant C is derived by imposing the condition $\sum_{u=1}^{m} q_u = 1$.

Suppose P_y is the non-parametric distribution of the candidate object at position y in the image, then the similarity or Bhattacharyya coefficient can be decided by the following [32]:

$$\rho(y) = \rho\left[P_y, Q\right] = \sum_{u=1}^{m} \sqrt{p_u(y)\, q_u}. \tag{16}$$

For tracking by agents, $\rho(y)$ can be used to compute the fitness of an agent and the similarity coefficient between two appearance templates.

4.2. Implementation of the Tracking Algorithm. The memory-based multiagent model for object tracking can be described as follows.

Step 1. First locate the object in a video scene and then build the object appearance model by (14).

Step 2. Randomly generate $N \times N$ agents near the located object region by adding a 2D Gaussian distribution $G_{x,y}(0, 10)$, as shown in Figure 9(a), and then map the agents onto the 2D lattice-like environment.

Step 3. For each agent on the lattice, first retrieve the appearance template from its memory spaces, then compute the fitness of the agent, and then perform the competition, recombination, and migration behaviors when the object moves. A snapshot of multiagent coevolution is shown in Figure 9(b).

Step 4. Compute the final target by weighted averaging of all the agents on the lattice, and the tacking result after the end of coevolution is shown in Figure 9(c).

5. Experimental Results and Discussions

In this section, we aim to experimentally verify the efficacy of the proposed object tracking method. We compare the performance of the proposed method with the total model updating PF (TMU-PF) in practical tracking problems. We use some standard video sequences [33, 34] as testing dataset and the experiments are conducted on a computer with a P4 3.0 G Processor.

(a) Randomly distributed agents (yellow points)

(b) A snapshot of multiagent coevolution

(c) Tacking result after the end of coevolution

Figure 9: Object tracking by multiagent coevolution.

It is worth noting that the parameters for the algorithms are set initially as follows in our experiments:

(a) m is the number of the bins for modeling the object using histogram and is set as $m = 16 \times 16$;

(b) T_{dc} is used to measure the similarity between the estimated template and the current object template and is set as $T_{dc} = 0.9$;

(c) T_{ds} and T_{dl} are the thresholds used to find a match in STMS and LTMS, respectively, and are set as $T_{ds} = T_{dl} = 0.8$;

(d) K_s and K_l are the capacity of the STMS and LTMS, respectively, and are set as $K_l = K_s = 5$;

(e) T_M is a predefined threshold used to decide whether the template in STMS can be stored into LTMS or not and is initially set as $T_M = 1$;

(f) the total number of agents is 49; that is, the size of the lattice A is 7×7, the recombination probability P_r is 0.6, and the migration probability P_m is 0.05.

(g) the number of the particles used in particle filter-based tracking is set as 50 (almost equal to the number of agents used).

5.1. Tracking a Person with Large Appearance Change. The first sets of experiments are to track a person with abrupt appearance changes. The video used in this experiment is clipped from the standard sequence "seq_dk" (The video sequences can be downloaded from http://www.ces.clemson.edu/~stb/research/headtracker/seq/) [33]. The tracking results of the man by traditional PF, TMU-PF, and the proposed method at frames 21, 58, 82, 83, 87, and 96 are shown in Figures 10(a), 10(b), and 10(c), respectively (the template is initialized manually). The human appearance changes very abruptly from frame 82 to frame 83. The results show that when the appearance is far from the initialized template, PF and TMU-PF deviate from the target gradually, while the original templates are remembered by the proposed method and when the appearance changes abruptly the relevant template can be recalled from the memory space of an agent.

Figure 11 displays experiments to track a person whose pose changes continuously in Head Pose Image Database (The video sequences can be downloaded from http://www-prima.inrialpes.fr/perso/Gourier/Faces/HPDatabase.html) [34]. Experimental results show that our proposed method can track more precisely than the other two methods.

5.2. Tracking a Person with Heavy Occlusions by Others. The second set of experiments aims at tracking persons who are occasionally occluded by another object.

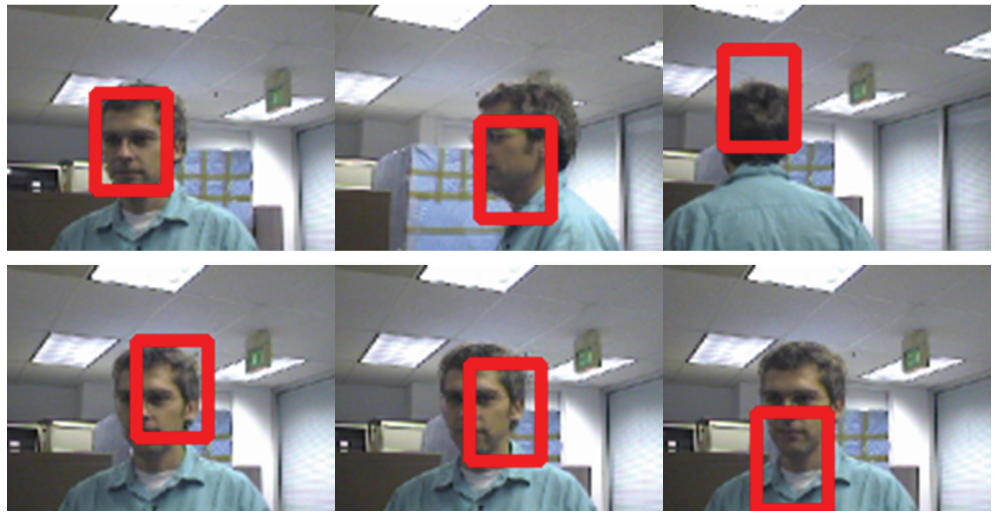
The sequence used in the first experiment is also a standard sequence "seq_jd" [33]. In this sequence, the man is occluded twice by another person. The tracking results by PF, TMU-PF and the proposed MMAM are shown in Figures 12(a), 12(b), and 12(c), respectively (the template is initialized manually). It is worth noting that the man is totally occluded at frame 52 and frame 253. The results show that the proposed MMAM can still track the person correctly after recovered from the occlusion at frame 55 and frame 256.

Figure 13 shows the results of tracking a face which is fully occluded by another person (The templates are initialized manually).

Finally, unlike the particle filter-based tracking method, the proposed approach has no restrictions to the face moving direction and speed. The face will be located and tracked at any time.

6. Conclusions

In this paper, we propose a different approach for visual tracking inspired by the way human perceive the environment. A number of memory-based agents are distributed nearby the located object region and then mapped onto a 2D lattice-like environment for predicting the new location of the object by their coevolutionary behaviors, such as competition, recombination, and migration, which imitate the process when many people search for a target in real world. The three-stage of human brain memory model is incorporated into a multiagent coevolutionary process for finding a best match of the appearance of the object. Each agent can remember, retrieve, or forget the appearance of the object through its memory system by its own experience. Experimental results show that the proposed method can deal with large appearance changes and heavy occlusions when tracking a moving object. It can locate the correct object

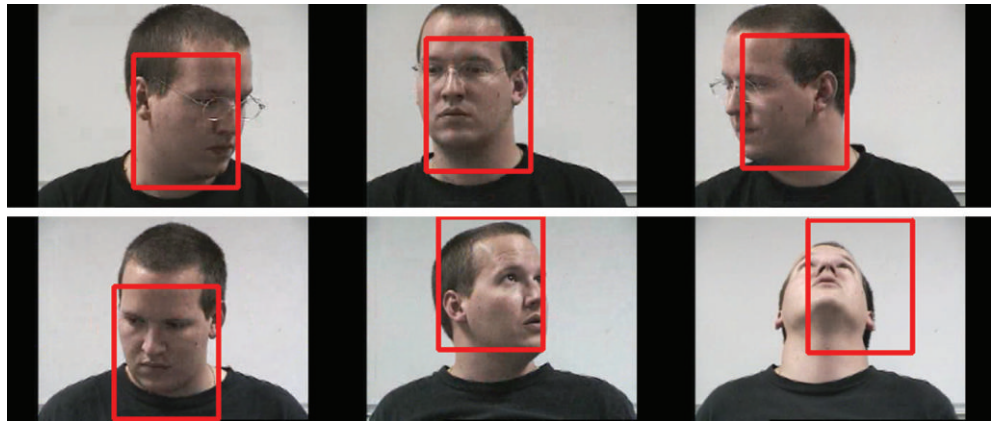


(a) Tracking results by PF

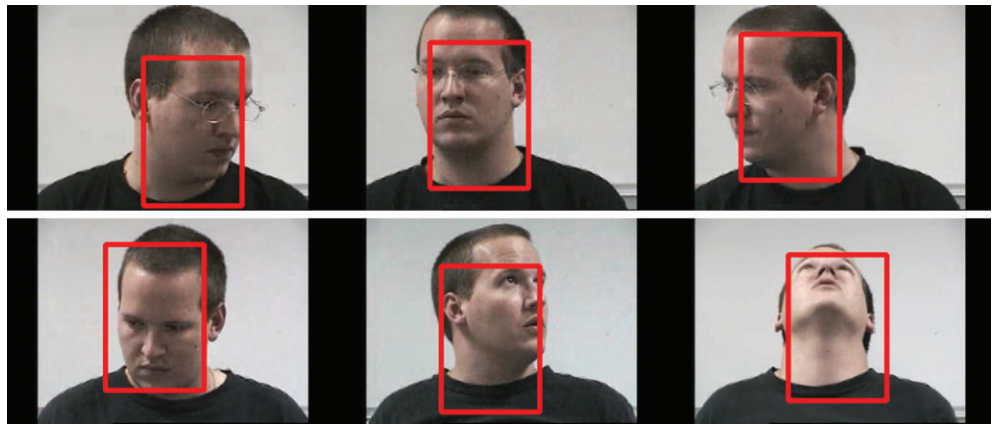
(b) Tracking results by TMU-PF

(c) Tracking results by MMAM

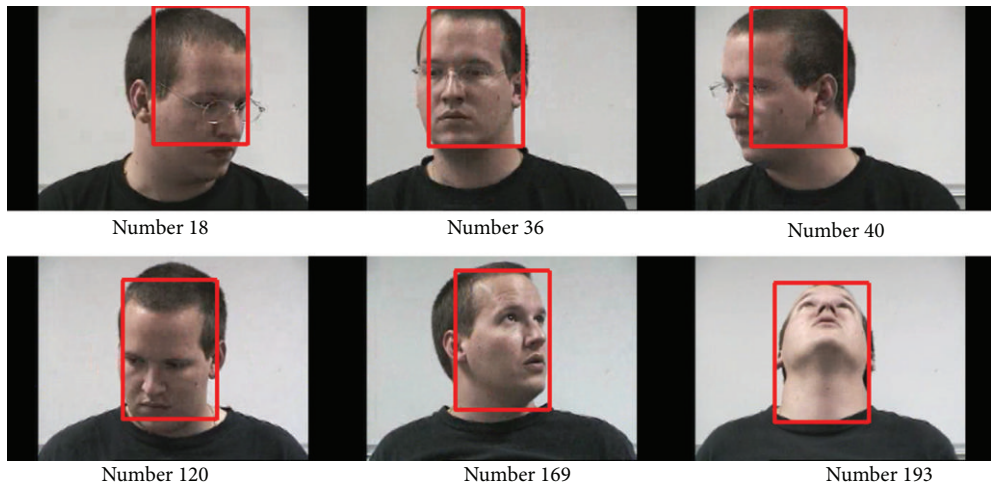
Figure 10: Tracking results of "seq_dk" sequence.



(a) Tracking results by PF

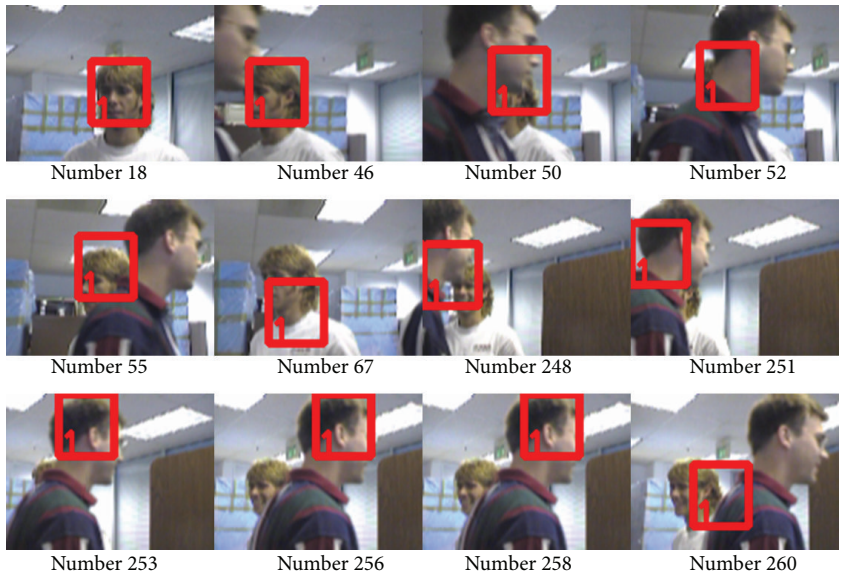


(b) Tracking results by TMU-PF

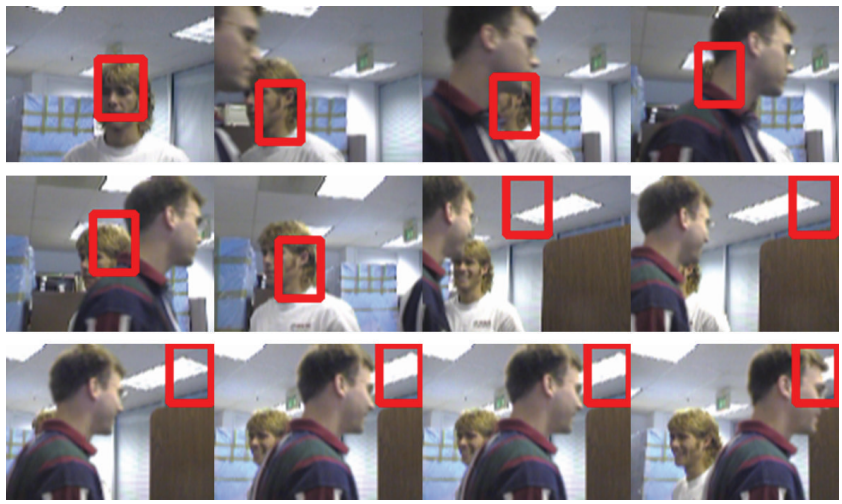


(c) Tracking results by MMAM

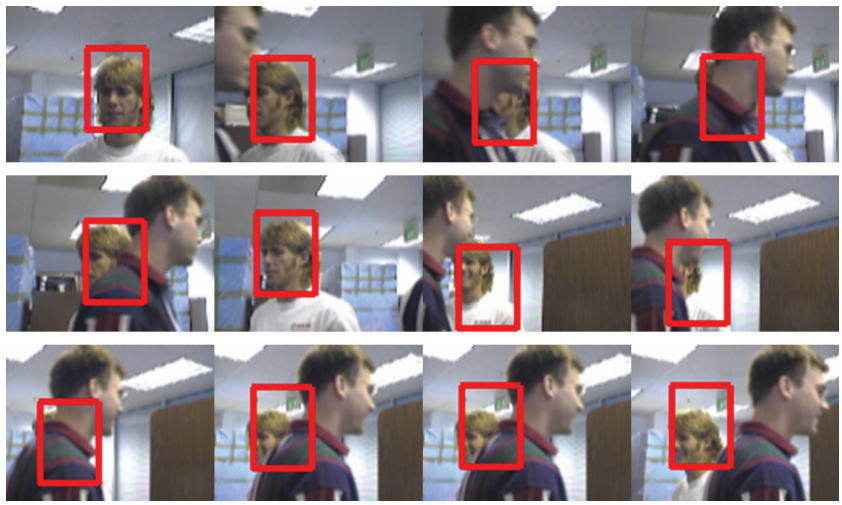
Figure 11: Tracking a person with pose changes.



(a) Tracking results by PF

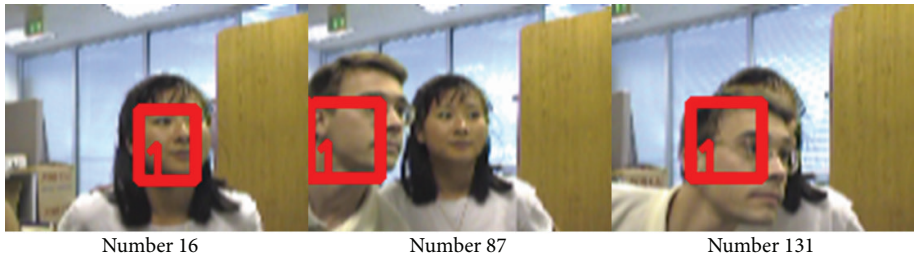


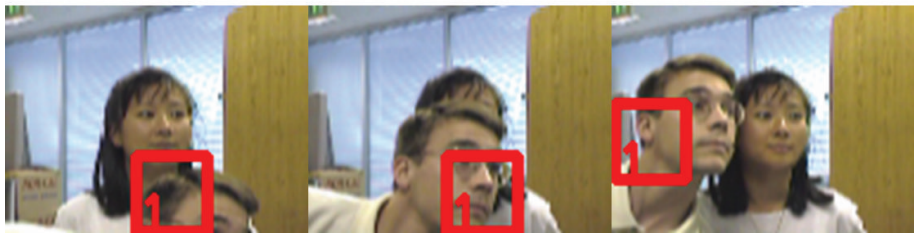
(b) Tracking results by TMU-PF



(c) Tracking results by MMAM

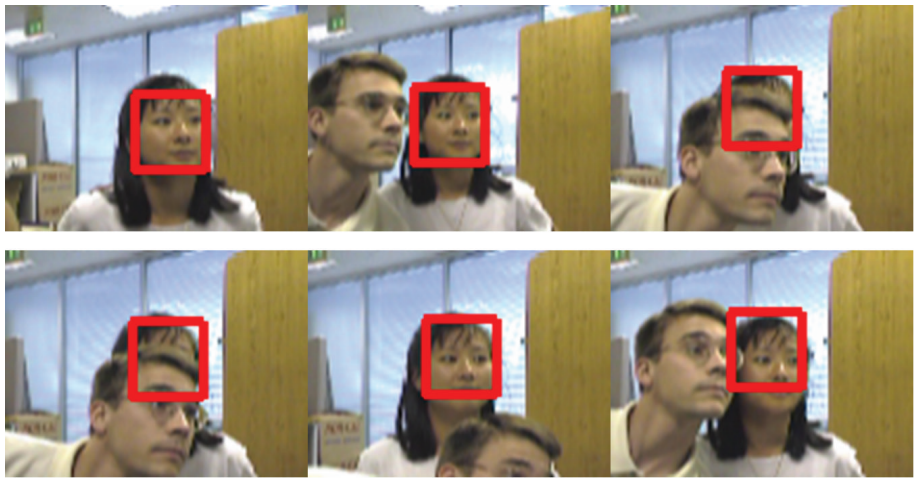
Figure 12: Tracking a fully occluded person.





(a) Tracking results by PF

(b) Tracking results by MTU-PF



(c) Tracking results by MMAM

Figure 13: Tracking a fully occluded face.

after the appearance changed or the occlusion recovered and outperforms the traditional particle filter based tracking.

Acknowledgments

The paper is funded by the National Natural Science Foundation of China (no. 60873163, no. 61271407) and the Fundamental Research Funds for the Central Universities (no. 27R1105019A, no. R1405008A).

References

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: a survey," *ACM Computing Surveys*, vol. 38, no. 4, pp. 1–45, 2006.

[2] S. K. Zhou, R. Chellappa, and B. Moghaddam, "Visual tracking and recognition using appearance-adaptive models in particle filters," *IEEE Transactions on Image Processing*, vol. 13, no. 11, pp. 1491–1506, 2004.

[3] A. Li, Z. Jing, and S. Hu, "Robust observation model for visual tracking in particle filter," *International Journal of Electronics and Communications*, vol. 61, no. 3, pp. 186–194, 2007.

[4] H. Wang, D. Suter, and K. Schindler, "Effective appearance model and similarity measure for particle filtering and visual tracking," in *Proceedings of the 9th European Conference on Computer Vision, Part III (ECCV '06)*, vol. 3953 of *Lecture Notes in Computer Science*, pp. 606–618, Graz, Austria, May, 2006.

[5] H. Wang, D. Suter, K. Schindler, and C. Shen, "Adaptive object tracking based on an effective appearance filter," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 9, pp. 1661–1667, 2007.

[6] B. Zhang, W. Tian, and Z. Jin, "Robust appearance-guided particle filter for object tracking with occlusion analysis," *International Journal of Electronics and Communications*, vol. 62, no. 1, pp. 24–32, 2008.

[7] Z. Wang, X. Yang, Y. Xu, and S. Yu, "CamShift guided particle filter for visual tracking," *Pattern Recognition Letters*, vol. 30, no. 4, pp. 407–413, 2009.

[8] C. Shan, T. Tan, and Y. Wei, "Real-time hand tracking using a mean shift embedded particle filter," *Pattern Recognition*, vol. 40, no. 7, pp. 1958–1970, 2007.

[9] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using SIFT features and mean shift," *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 345–352, 2009.

[10] J. Zhao and Z. Li, "Particle filter based on particle swarm optimization resampling for vision tracking," *Expert Systems with Applications*, vol. 37, no. 12, pp. 8910–8914, 2010.

[11] H. Zhou, M. Taj, and A. Cavallaro, "Target detection and tracking with heterogeneous sensors," *IEEE Journal on Selected Topics in Signal Processing*, vol. 2, no. 4, pp. 503–513, 2008.

[12] A. S. Montemayor, J. J. Pantrigo, and J. Hernamdez, "A memory-based particle filter for visual tracking through occlusion," in *Proceedings of the International Work-Conference on the Interplay Between Natural and Artificial Computation, Part II (IWINAC '09)*, vol. 5602 of *Lecture Notes in Computer Science*, pp. 274–283, 2009.

[13] D. Mikami, K. Otsuka, and J. Yamato, "Memory-based particle filter for face pose tracking robust under complex dynamics," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 999–1006, Miami, FL, USA, 2009.

[14] Y. Wang and V. Chiew, "On the cognitive process of human problem solving," *Cognitive Systems Research*, vol. 11, no. 1, pp. 81–92, 2010.

[15] Y. X. Wang, "Formal description of the cognitive process of memorization," *Transactions on Computational Intelligence*, vol. 1, no. 3, pp. 1–15, 2009.

[16] M. Wooldridge and N. R. Jennings, "Intelligent agents: theory and practice," *The Knowledge Engineering Review*, vol. 10, no. 2, pp. 115–152, 1995.

[17] J. Liu, Y. Y. Tang, and Y. C. Cao, "An evolutionary autonomous agents approach to image feature extraction," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 2, pp. 141–158, 1997.

[18] E. G. P. Bovenkamp, J. Dijkstra, J. G. Bosch, and J. H. C. Reiber, "Multi-agent segmentation of IVUS images," *Pattern Recognition*, vol. 37, no. 4, pp. 647–663, 2004.

[19] J. Liu, H. Jing, and Y. Y. Tang, "Multi-agent oriented constraint satisfaction," *Artificial Intelligence*, vol. 136, no. 1, pp. 101–144, 2002.

[20] J. Liu, X. Jin, and K. C. Tsui, "Autonomy-oriented computing (AOC): formulating computational systems with autonomous components," *IEEE Transactions on Systems, Man, and Cybernetics A*, vol. 35, no. 6, pp. 879–902, 2005.

[21] K. C. Tsui and J. Liu, "An evolutionary multiagent diffusion approach to optimization," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 16, no. 6, pp. 715–733, 2002.

[22] W. Zhong, J. Liu, M. Xue, and L. Jiao, "A multiagent genetic algorithm for global numerical optimization," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 34, no. 2, pp. 1128–1141, 2004.

[23] J. Liu, W. Zhong, and L. Jiao, "A multiagent evolutionary algorithm for constraint satisfaction problems," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 36, no. 1, pp. 54–73, 2006.

[24] J. Liu, W. Zhong, and L. Jiao, "An organizational evolutionary algorithm for numerical optimization," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 37, no. 4, pp. 1052–1064, 2007.

[25] Y. Wang and B. Yuan, "A novel approach for human face detection from color images under complex background," *Pattern Recognition*, vol. 34, no. 10, pp. 1983–1992, 2001.

[26] Y. Wang and B. Yuan, "Fast method for face location and tracking by distributed behaviour-based agents," *IEE Proceedings*, vol. 149, no. 3, pp. 173–178, 2002.

[27] W. James, *Principles of Psychology*, Holt, New York, NY, USA, 1890.

[28] R. C. Atkinson and R. M. Shiffrin, "Human memory: a proposed system and its control processes," in *The Psychology of Learning and Motivation*, K. W. Spence, Ed., vol. 2, pp. 89–195, Academic Press, New York, NY, USA, 1968.

[29] A. D. Baddeley and G. J. Hitch, "Working memory," in *The Psychology of Learning and Motivation*, G. H. Bower, Ed., vol. 8, pp. 47–89, 1974.

[30] Y. X. Wang and Y. Wang, "Cognitive informatics models of the brain," *IEEE Transactions on Systems, Man and Cybernetics C*, vol. 36, no. 2, pp. 203–207, 2006.

[31] M. W. Eysenck and M. T. Keane, *Cognitive Psychology: A Student's Handbook*, Psychology Press, New York, NY, USA, 6th edition, 2010.

[32] C. Lerdsudwichai, M. Abdel-Mottaleb, and A. Ansari, "Tracking multiple people with recovery from partial and total occlusion," *Pattern Recognition*, vol. 38, no. 7, pp. 1059–1070, 2005.

[33] S. Birchfield, "Elliptical head tracking using intensity gradients and color histograms," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 232–237, Santa Barbara, Calif, USA, June 1998.

[34] N. Gourier, D. Hall, and J. L. Crowley, "Estimating face orientation from robust detection of salient facial features," in *Proceedings of the Pointing International Workshop on Visual Observation of Deictic Gestures*, Cambridge, UK, 2004.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 409167, 8 pages
http://dx.doi.org/10.1155/2013/409167

Research Article

A New Logistic Dynamic Particle Swarm Optimization Algorithm Based on Random Topology

Qingjian Ni[1,2] and Jianming Deng[1]

[1] *School of Computer Science and Engineering, Southeast University, Nanjing 211189, China*
[2] *Provincial Key Laboratory for Computer Information Processing Technology, Soochow University, Suzhou 215006, China*

Correspondence should be addressed to Qingjian Ni; niqingjian@gmail.com

Received 17 April 2013; Accepted 19 May 2013

Academic Editors: P. Agarwal, S. Balochian, and V. Bhatnagar

Population topology of particle swarm optimization (PSO) will directly affect the dissemination of optimal information during the evolutionary process and will have a significant impact on the performance of PSO. Classic static population topologies are usually used in PSO, such as fully connected topology, ring topology, star topology, and square topology. In this paper, the performance of PSO with the proposed random topologies is analyzed, and the relationship between population topology and the performance of PSO is also explored from the perspective of graph theory characteristics in population topologies. Further, in a relatively new PSO variant which named logistic dynamic particle optimization, an extensive simulation study is presented to discuss the effectiveness of the random topology and the design strategies of population topology. Finally, the experimental data are analyzed and discussed. And about the design and use of population topology on PSO, some useful conclusions are proposed which can provide a basis for further discussion and research.

1. Introduction

Particle swarm optimization (briefed as PSO) is a kind of bionic evolutionary algorithm which rooted in imitation of behavioral mechanisms in populations such as birds and fish stocks and has been widely used in engineering field as optimization method [1–3].

In the PSO algorithms, the particles evolve according to their own experience and the experience of the neighborhood particles. During the evolutionary process, particles identify their own neighborhood according to the population topology and then learn from each other and update the positions of the particles. Therefore, population topology determines the form of information sharing among particles and thus has a very important impact on the solving performance of PSO algorithms. Therefore, it is important to explore the population topologies of PSO algorithms. This will produce a deep understanding of the working mechanism of PSO algorithms and thus improve the solving performance.

In the PSO algorithms, the most common used static population topologies are the fully connected topology (Gbest model) and the ring topology (lbest model) which are also first proposed [4]. Since then, researchers have proposed different population topologies in succession. Kennedy carried out a preliminary analysis of four static population topologies [5]. Suganthan adjusted the neighborhood structure of particles through calculating distances between particles in the evolutionary process [6]. Mendes et al. detailly analyzed the relationship between the population topology and a class of PSO variant [7–9]. Clerc initially attempted to adopt the random topology [10]. However, these studies concerned population topologies paid more attention to the static classic population topology, and research of random population topologies is relatively small. As the population topology directly affect the exchange of information between the particles, it is necessary to design the suitable population topology according to the characteristics of different types of applications. Therefore, it is necessary to explore the population topologies in depth from a theoretical and experimental point of view.

In this paper, in a relatively new PSO variant which named logistic dynamic particle optimization, we analyze the

linkages between population topologies and the performance of PSO algorithms from graph theory and experimental point of view. The rest of the paper is organized as follows. In Section 2, we describes the PSO variant which are used in the paper. Section 3 describes the classic population topologies and introduces the proposed random population topologies. In Section 4, we have presented the experimental analysis and comparative performance between the classic and the proposed random population topologies. Section 5 concludes the paper.

2. The PSO Variants

PSO is a population-based method which is similar to other evolutionary computation methods. The individual in the PSO population is called the particle, and particles generally have the speed and position in most PSO variants.

2.1. The Canonical PSO. Based on the earlier version of PSO, Clerc developed the PSO with the compression factor [11]. This PSO variants have been widely used in practical applications, and the velocity and position update of particles in this variant are as follows:

$$v_{id} = \chi * \left[v_{id} + c_1 * \text{rand}( ) * \left(p_{id} - x_{id} \right) + c_2 * \text{Rand}( ) * \left(p_{gd} - x_{id} \right) \right], \tag{1}$$

$$\chi = \frac{2}{\left| 2 - \varphi - \sqrt{\varphi^2 - 4\varphi} \right|}, \tag{2}$$

$$x_{id} = x_{id} + v_{id}. \tag{3}$$

In (1), v_{id} is the dth dimensional component of particle is velocity attribute, c_1 and c_2 are two positive acceleration factors, rand() and Rand() are random number generating functions between 0 and 1, x_{id} is the dth dimensional component of the particle is position property, p_{id} is the dth dimensional component of the best position that particle i obtained, and p_{gd} is the dth dimensional component of the best position that the whole population obtained. In actual use, usually χ in (2) is set to 0.729, and φ is often set to 4.1.

The right part of the equation1 can be understood as particle's memory, cognitive and social cognition. Velocity of particles is precisely through this three-part interaction effects thereby, position of particles is updated.

2.2. The Dynamic Probabilistic Particle Swarm Optimization. In the previous PSO variants, particles usually have both velocity attribute and position attribute. Kennedy first proposed a new PSO variant without velocity attribute, which named Gaussian dynamic particle swarm optimization [12]. Ni and Deng carried out a further study on the PSO variants without velocity [13]. This type of algorithm variants can be called dynamic probabilistic particle swarm optimization (briefed as DPPSO). In the DPPSO algorithms, particles have no velocity attributes, and the update of particles' positions is reorganized as follows:

$$X_i(t+1) = X_i(t) + \alpha * \left(X_i(t) - X_i(t-1) \right) + \beta * \text{CT}_i(t) + \gamma * \text{Gen}( ) * \text{OT}_i(t), \tag{4}$$

$$\text{CT}_{id}(t) = \sum_{k=1}^{K} \frac{P_{kd}}{K - X_{id}(t)}, \tag{5}$$

$$\text{OT}_{id}(t) = \sum_{k=1}^{K} \frac{\left| P_{id} - P_{kd} \right|}{K}. \tag{6}$$

In (4), (5), and (6), t represents the current evolution generation of particle, i is the index number of particle, k is the index number of particle's neighborhood, K represents the number of particle's neighborhood particles, P_k is the optimum position of the particle's neighborhood that numbered k, and d is the number of the particles is dimension. $\text{CT}_i(t)$ is an abbreviation of centralized tendency, which is a D-dimensional vector and is determined by the particle's current location and neighborhood particles' optimal positions. $\text{OT}_i(t)$ is an abbreviation of outlier trend, which is also a D-dimensional vector and is determined by the particle's current location and neighborhood particles' optimal positions. In (4), α, β, and γ are generally preferable to positive constants. Gen() is a dynamic probabilistic evolutionary operator which is the random number generator function which satisfies a specific distribution, and this particular distribution may be provided by a Gaussian distribution or a logistic distribution and so forth.

In (4), the calculation of particle's position in new generation is decided jointly by the right four parts. The first part is the memory of particles on the self-position. The second part is the trend of the particles along the previous direction of movement of the "flying." The third part means the influence of neighborhood particles' experience to the new generation position, this part of the calculations needs neighborhood particles' experience, and this part determines the degree of influence of the neighborhood particles' experience. The part IV reflects the impact of differences in best position between particles on the next generation position.

The performance of DPPSO variants is different when using different dynamic probabilistic evolutionary operator Gen() [13]. DPPSO-Gaussian (briefed as GDPS) has faster convergence speed in the early evolution. DPPSO-Cauchy may get better solutions on certain issues, but the performance is unstable. DPPSO-Logistic (logistic dynamic particle swarm optimization, briefed as LDPSO) still shows good exploration ability in the later evolution. For DPPSO variants, the calculation of $\text{CT}_i(t)$ and $\text{OT}_i(t)$ will use the experience of each neighborhood particles which can also be seen from (5) and (6). The optimal information of neighborhood particles could be fully utilized, so the research of population topology in DPPSO variants is more important.

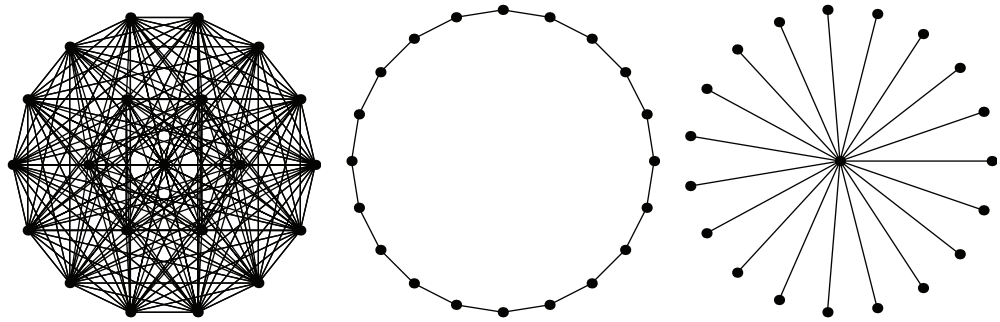


Figure 1: (a) Fully connected topology; (b) ring topology; (c) star topology.

3. Population Topologies

3.1. The Classic Population Topologies. Figure 1 shows the fully connected topology, ring topology, and star topology. Fully connected topology and ring topology are two commonly used topologies which are also called Gbest model and Lbest model. In the fully connected topology, particle's neighborhood contains all particles in the population. And in the evolutionary process, only the particle that obtains the optimal position is considered in the entire population. PSO algorithms with this topology converge very fast but easy to fall into local optimum.

For a ring topology, typically particle's neighborhood includes the particles on both sides of one or a few particles. In this topology, the exchange of information is relatively slow within the population, but once a particle searched for an optimal location, the information eventually will slowly spread to the entire population.

For a star topology, one particle is connected with all the other particles, and other particles only connect with the particle. In addition to a central particle, other particles are independent of each other, the dissemination of information must be passed through the central particle.

However, the information dissemination mechanism of the social groups is not static throughout the whole evolutionary process, and tends to have a certain degree of randomness and dynamic characteristics. Mendes studied random population topology and confirmed that population topology directly affects the execution performance of PSO algorithms [9].

3.2. Graph Theory Characteristics of Population Topology. The population topology of the PSO algorithm can be abstracted into a connected undirected graph, represented by the symbol $G(V, E)$, where V is the set of vertices, E is the set of edges, and the number of vertices is denoted by n. For any two points u and v in G, $d(u, v)$ denotes the distance from u to v, that is, the length of the shortest path between two points.

Definition 1 (average degree). The degree of the vertex v is the number of its adjacent vertices, denoted by k_v. Average degree k of undirected connected graph is calculated by

$$K = \sum_{v \in V} \frac{k_v}{n}. \tag{7}$$

Definition 2 (average clustering coefficient). The local clustering coefficient $c(v)$ of vertex v, equals to the number of edges which can be connected between vertices associated with vertex v, divided by the maximum number of edges between these vertices. The average clustering coefficient c of a graph is the arithmetic mean of the local clustering coefficient of all vertices, which can be calculated by

$$c = \sum_{v \in V} \frac{c(v)}{n}. \tag{8}$$

The average degree of population topology means the average number of particles' neighborhood particles; it represents the degree of socialization of population. A small number of neighborhood particles means that the particle is both difficult to obtain information from the population and difficult to influence other particles. On the contrary, a particle which have large number of neighborhood particles can get a lot of information available in the population, and such a particle has a greater influence in the population. In the common used population topologies, fully connected topology has the largest average degree which is equal to the population size minus 1. The ring topology has the minimum average degree; the average degree of a ring topology such as in Figure 1 is equal to 2. The local clustering coefficient is the ratio of the number of connections between the actual existence and the possible existence, and the average clustering coefficient represents the degree of aggregation of the vertices in a graph which is the average of the local clustering coefficient of all vertices. In this paper, we will analyze the role of the population topology based on the previous graph theory characteristics.

3.3. Random Population Topologies. Clerc initially attempted to proposed a method of random population topology [10]. The basic idea is to generate a random topology by selecting the neighborhood particles randomly for each particle. The concrete steps could be described as Algorithm 1.

In the resulting matrix L of Algorithm 1, $L(u, v) = 1$ means that the particles u and v are connected. By this

(1) For the population that the size is S, build a $S * S$ matrix, and let $L(i, i) = 1$;
(2) Select a K value, for each row i of the matrix L, generate a uniformly distributed random number m from $\{1, \ldots, m\}$ (m may be selected repeatedly), let $L(i, m) = 1$;
(3) For the matrix L, If $L(i, m) = 1$, then let $L(m, i) = 1$;

Algorithm 1: Clerc's generating method of random population topology.

(1) For the population that the size is S, build a $S * S$ matrix, and let $L(i, i) = 1$;
(2) Select a K value, for each row i of the matrix L, generate a uniformly distributed random number m ($m \neq i$) from $\{1, \ldots, m\}$ (m may be selected repeatedly), let $L(i, m) = L(m, i) = 1$;
(3) Use the Dijkstra algorithm to calculate the distance between the particles and stored in the $S * S$ matrix D;
(4) **While** *The graph corresponding to the generating random topology is not connected* **do**
(5) Scan the matrix D;
(6) IF two particles u and v are not connected;
(7) Let $L(u, v) = 1$;

Algorithm 2: The proposed generating method of random population topology.

method, a random topology could be generated with an average degree slightly larger than K. In the random topology by this method, the distribution of degree is the sum of $S - 1$ independent Bernoulli random variables which is described in [10]

$$\text{prob}(Y = n) = C_{S-1}^{n-1}\left(\frac{K}{S}\right)^{n-1}\left(1 - \frac{K}{S}\right)^{s-n}. \quad (9)$$

In this paper, in order to ensure the connectivity of undirected graph which is corresponding to the generating random topology, we proposed a new generating method of random topology based on Clerc's method. The basic idea is: in selecting the neighborhood particles for each particle if the selection is itself, reselect in order to reduce the probability of particles isolated; after the random population is generated, use the Dijkstra algorithm to compute the distance between the particles if there exist the unconnected particles in the generated topology, then add an edge between these unconnected particles, and retest. The improved random population topology generating method is as Algorithm 2.

4. Experiment and Analysis

4.1. Experiment Setting. Two sets of experiments were conducted which used the canonical PSO (briefed as CPSO) and the DPPSO-Logistic, respectively, that are described in Section 2. For CPSO, set $c_1 = c_2 = 2.05$, $\chi = 0.7298$. For DPPSO-Logistic, set $\alpha = 0.729$, $\beta = 2.187$, and $\gamma = 0.5$.

The algorithms were used to solve five benchmark functions, which is defined in Table 1. These functions consist of Ackley, Schwefel, Schaffer's F6, Rastrigin, and Sphere. Table 2 shows the settings of these functions.

In the experiment, the population size is set to 20, in addition to the the Schaffer's F6 function of the dimension 2, the remaining functions are carried out in the case of 30-dimensional test, and the frequency of repeated experiments is 50.

The performance of the algorithms will be evaluated by the following aspects:

(i) in the case of a certain number of iterations, compare the accuracy (briefed as Perform.) of the optimal fitness value in each case. These values reflect the quality of the optimal solution obtained in the last;

(ii) in the case of a certain number of iterations, compare the success rates (briefed as Prop.) which means that the algorithms achieve the accuracy (accepted error) that is defined in Table 2. These data reflect the stability of the algorithms;

(iii) in the case of a certain number of iterations, compare the evolutionary trends of various algorithms, these figures reflect the evolution of the optimal solution in the evolutionary process.

4.2. Comparison between Random Topologies and Different Average Degrees. For the random topology, the first set of experiments used the canonical PSO algorithm to solve the five benchmark functions. By changing the K value, we generated random topologies with different average degrees for comparison and evaluation. Experimental results are shown in Table 3.

As can be seen from Table 3, with the increase in the value of K, the indicators of Perform. and Prop. have improved. For multimodal functions such as Schwefel, Schaffer's F6, Rastrigin, and Ackley function, with the increase in the value of K, when the K value is from 3 to 4 (i.e., the average degree of the random topology is between 5 and 7, substantially in about 6), the PSO algorithm could get better performance. And when the K value is larger, the performance is usually poor. For multimodal functions, when a particle has found a local optimal solution, if the interaction between particles is more, the dissemination of information will be very quickly, and the entire population is susceptible to rapid convergence to the local optimal solution. Therefore, too large average

Table 1: Definition of benchmark functions.

Benchmark function	Formula
Sphere	$F(\vec{x}) = \sum_{i=1}^{n} x_i^2$
Schaffer's F6	$F(\vec{x}) = 0.5 + \frac{\left(\sin^2\sqrt{x^2 + y^2}\right) - 0.5}{\left(1.0 + 0.001\left(x^2 + y^2\right)\right)^2}$
Rastrigin	$F(\vec{x}) = 10 * n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i))$
Schwefel	$F(\vec{x}) = 418.9829 * n + \sum_{i=1}^{n} x_i \sin(\sqrt{\lvert x_i \rvert})$
Ackley	$F(\vec{x}) = -20 \cdot \exp\left(-0.2\sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n} \cdot \sum_{i=1}^{n} \cos\left(2\pi x_i\right)\right) + 20 + \exp(1)$

Table 2: Settings of benchmark functions.

Benchmark function	Dimension	Optimal value	Optimal solution	Range	Accepted error
Sphere	30	0	(0, 0, 0, . . ., 0)	(−100, 100)	0.01
Schaffer's F6	2	0	(0, 0)	(−100, 100)	0.00001
Rastrigin	30	0	(0, 0, 0, . . ., 0)	(−5.12, 5.12)	100
Schwefel	30	0	(0, 0, 0, . . ., 0)	(−500, 500)	6000
Ackley	30	0	(0, 0, 0, . . ., 0)	(−30, 30)	5

degree of population topology is not conducive to find the global optimal solution for a multimodal function.

Taking these factors together, when the value of K is at about 4, the PSO algorithm has a more satisfactory performance for most benchmark functions. Accordingly, the following experiments will generate random topologies which the K value is 4, and compare these topologies with other classic static population topologies.

4.3. Comparison between Random Topologies and Classic Topologies. The second set of experiments used the DPPSO-Logistic algorithm to solve the five benchmark functions on the fully connected topology, ring topology, star topology, and random topology, wherein K value is set to 4 to generate random topologies. Comparison and evaluation are conducted by the evolutionary trends of algorithms with various population topologies. In the figures of evolutionary trends, different line types expressed different DPPSO-Logistic algorithms with various population topologies.

For Sphere function (Figure 2), the performance of ring topology and star topology is poor, and the fully connected topology and random topology show better search ability. For Schaffer's F6 function (Figure 2), random topology is significantly better than the three classic neighborhood topologies.

For Rastrigin function (Figure 3), random topology is superior to the three classic population topologies; in the early stages of evolution, the convergence speed of the random topology is almost the same as the fully connected topology; however, in the later stage of evolution, random topology shows a larger advantage, and the end result is better than the fully connected topology.

For Schwefel function (Figure 3), the performance of the ring topology is poor; the fully connected topology convergence fast in the early stage of evolution, but the end result is poor; the star topology has achieved good results; the convergence speed of random topology is second only to the fully connected topology in the early evolution stage, and the final result of random topology is better than the other classic topologies.

For Ackley function (Figure 4), the fully connected topology and the star topology show poor performance; the ring topology performs better; the random topology shows obvious advantage in both the convergence speed and the final result.

Overall, according to the convergence speed, random topology is relatively stable in the early stages of evolution, faster in the midstages of evolution, and shows a distinct advantage in the late stages of evolution. From the view of final result, the PSO algorithms using random topology demonstrate remarkable performance.

For unimodal function (such as Sphere function), because there is no problem of falling into a local optimum, the close ties between particles can make faster convergence and achieve better results. Therefore, the performance of the fully connected topology and random topology with a relatively high average degree is ideal. In the case of random topology for unimodal function, the convergence speed and the solution will be better with the greater average degree of population topology.

For multimodal function, it can be seen that the convergence speed will be faster when the average degree of population topology is increasing. If the average degree of population topology is too high, it is easy to fall into local optimum. On the average degree after 6, the optimal solution

Table 3: Comparison between random topologies with different average degrees.

Benchmark function	K	Average degree	Clustering coefficient	Perform.	Prop.
Sphere	1	1.958	0.753	$1.38E-15$	0.768507
	2	3.700	0.541	$1.53E-30$	0.817693
	3	5.244	0.498	$3.70E-32$	0.82624
	4	6.640	0.510	$5.92E-33$	0.833993
	5	7.928	0.534	$4.15E-33$	0.836973
	6	9.040	0.568	$8.09E-34$	0.84178
	7	10.116	0.606	$1.00E-33$	0.84702
	8	11.092	0.641	$1.68E-34$	0.84278
	9	11.814	0.671	$9.18E-34$	0.84446
Schaffer's F6	1	1.942	0.753	0.002915	0.497393
	2	3.700	0.545	0.001757	0.63916
	3	5.228	0.497	0.001749	0.681847
	4	6.626	0.508	0.00136	0.716053
	5	7.926	0.530	0.001943	0.6303
	6	8.998	0.563	0.003303	0.558613
	7	10.070	0.602	0.00272	0.59538
	8	11.078	0.643	0.003313	0.542707
	9	11.854	0.673	0.002915	0.558667
Rastrigin	1	1.956	0.751	63.65224	0.868587
	2	3.688	0.546	58.10571	0.923933
	3	5.198	0.500	58.64275	0.915993
	4	6.694	0.507	62.68228	0.934207
	5	7.940	0.532	58.80197	0.94144
	6	9.018	0.565	62.54299	0.93782
	7	10.118	0.608	59.10044	0.944673
	8	10.982	0.638	61.90622	0.908687
	9	11.916	0.674	63.51805	0.925493
Schwefel	1	1.956	0.752	4773.649	0.91388
	2	3.694	0.548	4366.848	0.942427
	3	5.234	0.500	4388.532	0.9436
	4	6.616	0.502	4399.952	0.943533
	5	7.922	0.532	4286.961	0.950633
	6	9.046	0.571	4397.724	0.950987
	7	10.074	0.602	4412.291	0.932327
	8	11.016	0.641	4420.19	0.951007
	9	11.734	0.667	4397.566	0.933313
Ackley	1	1.948	0.751	1.224299	0.960753
	2	3.690	0.549	1.030908	0.969733
	3	5.282	0.493	0.985613	0.972333
	4	6.672	0.507	1.256866	0.97304
	5	7.872	0.539	1.238158	0.973947
	6	9.028	0.568	1.593962	0.974087
	7	10.134	0.611	1.755415	0.97432
	8	10.922	0.640	1.919407	0.974847
	9	11.876	0.671	1.962931	0.97564

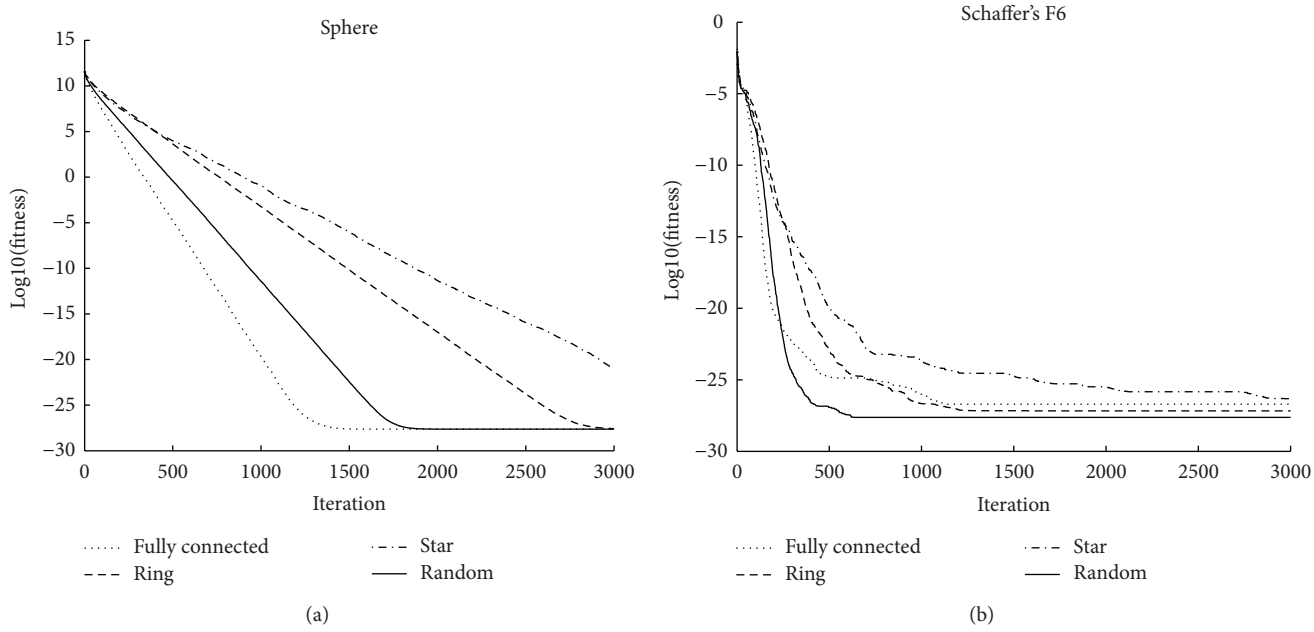


Figure 2: Comparison of evolutionary trend between four topologies (Sphere and Schaffer's F6).

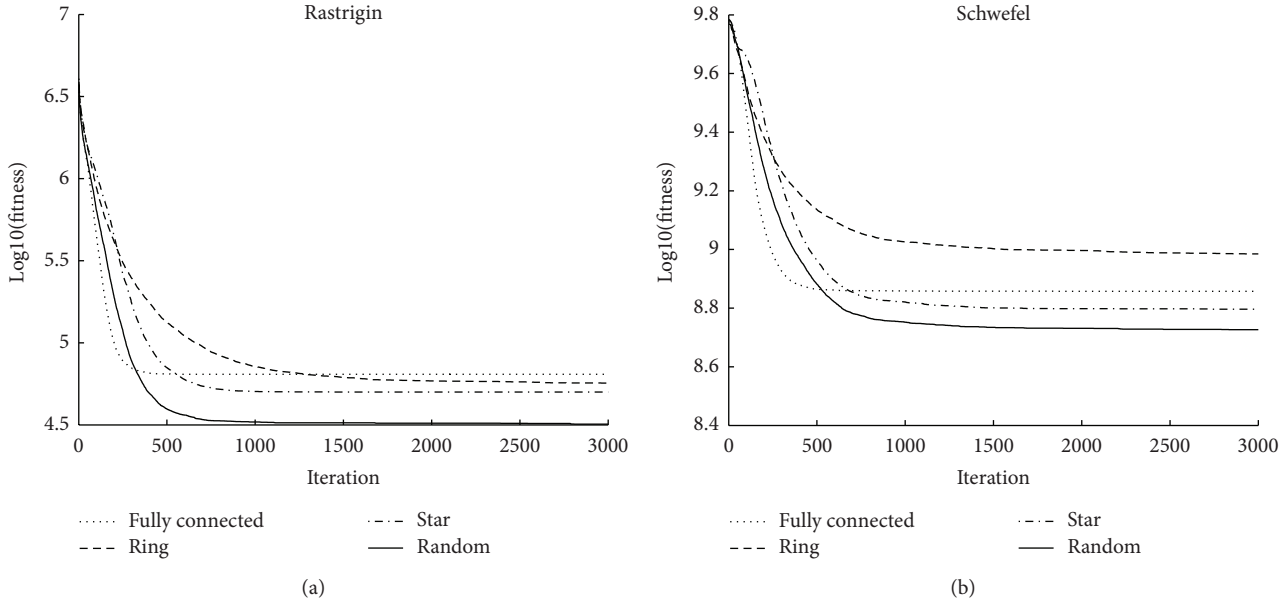


Figure 3: Comparison of evolutionary trend between four topologies (Rastrigin and Schwefel).

quality of most algorithms begins to decrease. When the average degree is between 5 and 7 (K is set to 4), the performance of algorithms is usually ideal.

5. Conclusion

In this paper, we propose an improved method of generating random topology based on previous research. And we carry out in-depth research of the performance of the algorithms using random topology based on the canonical PSO and DPPSO-Logistic, respectively. Combined with experimental results, we conduct the analysis and interpretation of the performance of the algorithms from the perspective of graph theory. And empirical laws in generating random topologies are given according to our experimental results and theoretical analysis.

On the whole, relative to the three classic population topologies (fully connected topology, ring topology, and star topology), the algorithm has obvious advantages which is using the proposed random topology. Further work will include the theoretical analysis of different population topologies, as well as dynamic population topology strategy

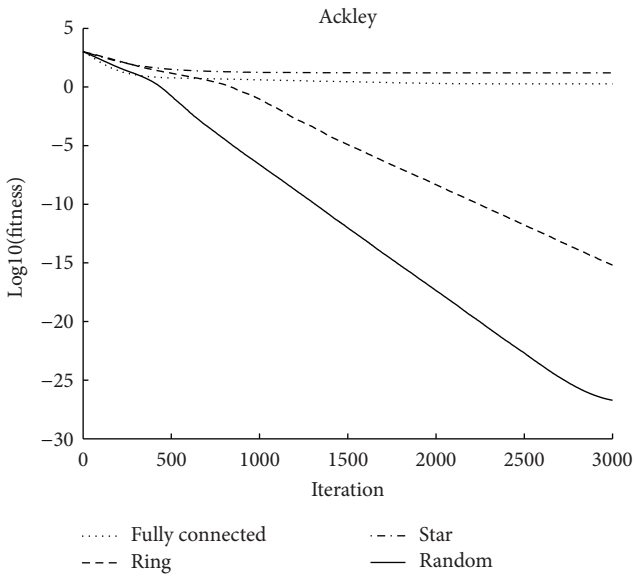


Figure 4: Comparison of evolutionary trend between four topologies (Ackley).

which is designed in accordance with the conclusions of this paper.

Acknowledgments

This paper is supported by Provincial Key Laboratory for Computer Information Processing Technology, Soochow University, Suzhou, China, and NSFC (Grant no. 61170164).

References

[1] M. R. AlRashidi and M. E. El-Hawary, "A survey of particle swarm optimization applications in electric power systems," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 913–918, 2009.

[2] A. A. Esmin, R. A. Coelho, and S. Matwin, "A review on particle swarm optimization algorithm and its variants to clustering highdimensional data," *Artificial Intelligence Review*, 2013.

[3] R. V. Kulkarni and G. K. Venayagamoorthy, "Particle swarm optimization in wireless-sensor networks: a brief survey," *IEEE Transactions on Systems, Man and Cybernetics C*, vol. 41, no. 2, pp. 262–267, 2011.

[4] 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.

[5] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance," in *Proceedings of the Congress on Evolutionary Computation (CEC '99)*, vol. 3, IEEE, 1999.

[6] P. N. Suganthan, "Particle swarm optimiser with neighbourhood operator," in *Proceedings of the Congress on Evolutionary Computation (CEC '99)*, vol. 3, IEEE, 1999.

[7] 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.

[8] M. A. M. de Oca and T. Stützle, "Convergence behavior of the fully informed particle swarm optimization algorithm," in *Proceedings of the 10th Annual Genetic and Evolutionary Computation Conference (GECCO '08)*, pp. 71–78, ACM, July 2008.

[9] R. Mendes, *Population topologies and their influence in particle swarm performance [Ph.D. dissertation]*, Universidade do Minho, 2004.

[10] M. Clerc, "Back to random topology," Tech. Rep. 2007, 2007.

[11] 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.

[12] J. Kennedy, "Dynamic-probabilistic particle swarms," in *Proceedings of the Conference on Genetic and Evolutionary Computation Conference*, pp. 201–207, ACM, June 2005.

[13] Q. Ni and J. Deng, "Two improvement strategies for logistic dynamic particle swarm optimization," in *Adaptive and Natural Computing Algorithms*, pp. 320–329, Springer, 2011.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 597803, 6 pages
http://dx.doi.org/10.1155/2013/597803

Research Article

A Novel Complex Valued Cuckoo Search Algorithm

Yongquan Zhou[1,2] and Hongqing Zheng[1]

[1] *College of Information Science and Engineering, Guangxi University for Nationalities, Nanning 530006, China*
[2] *Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis, Nanning 530006, China*

Correspondence should be addressed to Yongquan Zhou; yongquanzhou@126.com

Received 3 March 2013; Accepted 8 May 2013

Academic Editors: P. Agarwal, V. Bhatnagar, and Y. Zhang

To expand the information of nest individuals, the idea of complex-valued encoding is used in cuckoo search (PCS); the gene of individuals is denoted by plurality, so a diploid swarm is structured by a sequence plurality. The value of independent variables for objective function is determined by modules, and a sign of them is determined by angles. The position of nest is divided into two parts, namely, real part gene and imaginary gene. The updating relation of complex-valued swarm is presented. Six typical functions are tested. The results are compared with cuckoo search based on real-valued encoding; the usefulness of the proposed algorithm is verified.

1. Introduction

Recently, a new metaheuristic search algorithm, called Cuckoo Search (CS) [1], has been developed by Yang and Deb (2009). The algorithm is inspired by the reproduction strategy of cuckoos. Because of this method is simple, efficient and optimal random search paths, and successfully applied to practical engineering optimization problems [2]. In term of cuckoo search algorithm, there are many methods to improve its performance; some people study the parameters of cuckoo search algorithm. But these methods are using binary and decimal to encode the bird's nest, individual's information capacity is very limited.

Complex-valued encoding method is already used to express neural network weights [3] and individual genes of evolutionary algorithm [4, 5]; it uses diploid in the expression of individual genes and greatly expands the individual's information capacity. From individual coding method, this paper studies the plural coding performance improvement of cuckoo search algorithm. The value of independent variables for objective function is determined by modules, and the sign of them is determined by angles. The two variables of real and imaginary parts to represent an independent variable, thus nest groups, can enhance the information and tap the individual diversity of the population, reducing the local convergence. We provide a new way for the Cuckoo search algorithm to solve practical problems.

2. Cuckoo Search Algorithm

2.1. Original CS. CS is a heuristic search algorithm which has been proposed recently by Yang and Deb [1]. The algorithm is inspired by the reproduction strategy of cuckoos. At the most basic level, cuckoos lay their eggs in the nests of other host birds, which may be of different species. The host bird may discover that the eggs are not its own and either destroy the egg or abandon the nest all together. This has resulted in the evolution of cuckoo eggs which mimic the eggs of local host birds. For simplicity in describing the Cuckoo Search, we now use the following three idealized rules:

(1) Each Cuckoo lays one egg, which represents a set of solution coordinates, at a time, and dumps it in a random nest.

(2) A fraction of the nests containing the best eggs, or solutions, will be carried over to the next generation.

(3) The number of nests is fixed and there is a probability that a host can discover an alien egg. If this happens,

Cuckoo search via Lévy flight algorithm:

Begin
Objective function $f(x), x = (x_1, x_2, \ldots, x_d)^T$
Generate initial population of n host nests x_i $(i = 1, 2, \ldots, n)$
While ($t <$ Max Generation) or (stop criterion)
Get a cuckoo randomly by Lévy flight
Evaluate its quality/fitness F_i
Choose a nest among n (say, j) randomly
If ($F_i > F_j$),
replace j by the new solution;
End
A fraction (p_a) of worse nests are abandoned and new ones are built;
Keep the best solutions (or nests with quality solutions);
Rank the solutions and find the current best
End while
Post process results and visualization
End

Algorithm 1: Pseudo code of cuckoo search via Lévy flight algorithm.

the host can either discard the egg or the nest and this results in building a new nest in a new location.

Based on these three rules, the basic steps of the Cuckoo Search (CS) can be summarized as the pseudo code shown in Algorithm 1.

When generating new solution $x^{(t+1)}$ for, say, cuckoo i, a Lévy flight is performed

$$x_i^{(t+1)} = x_i^{(t)} + \partial \oplus \text{Lévy}(\beta), \tag{1}$$

where $\partial > 0$ is the step size which should be related to the scales of the problem of interests. In most cases, we can use $\partial = 1$.

The product $\oplus$ means entry-wise walk during multiplications. Lévy flights essentially provide a random walk while their random steps are drawn from a Lévy Distribution for large steps

$$\text{Lévy} \sim u = t^{-1-\beta} \quad (0 < \beta < 2). \tag{2}$$

This has an infinite variance with an infinite mean. Here the consecutive jumps/steps of a cuckoo essentially form a random walk process which obeys a power-law step-length distribution with a heavy tail. In addition, a fraction p_a of the worst nests can be abandoned so that new nests can be built at new locations by random walks and mixing. The mixing of the eggs/solutions can be performed by random permutation according to the similarity/difference to the host eggs.

Obviously, the generation of step size s samples is not trivial using Lévy flights. A simple scheme discussed in detail by Yang can be summarized as

$$x_i^{(t+1)} = x_i^{(t)} + \partial \oplus \text{Lévy}(\beta) \sim 0.01 \frac{u}{|v|^{1/\beta}} \left(x_j^{(t)} - x_i^{(t)}\right), \tag{3}$$

where u and v are drawn from normal distributions. That is

$$u \sim N\left(0, \sigma_u^2\right), \qquad v \sim N\left(0, \sigma_v^2\right). \tag{4}$$

Table 1: Nest chromosome structure shown.

(R_{p1}, I_{p1})	(R_{p2}, I_{p2})	...	(R_{pM}, I_{pM})

With $\sigma_u = \{(\Gamma(1+\beta)\sin(\pi\beta/2))/(\Gamma[(1+\beta)/2]\beta 2^{(\beta-1)/2})\}^{1/\beta}$, $\sigma_v = 1$. Here Γ is the standard Gamma function [6].

2.2. Cuckoo Search Based on Complex-Valued Encoding. Containing the M-variable function optimization problem, with M complex, corresponding to the M complex nest location is recorded as

$$x_p = R_p + I_p j \quad p = 1, 2, \ldots, M. \tag{5}$$

The gene of the nest can be expressed as the diploid and is recorded as (R_p, I_p); R_p, I_p express, respectively, the real and imaginary parts of the variable in (5). So the ith nest can be expressed as shown in Table 1.

2.2.1. Initialize the Nest. Assume that the variable interval of function is $[A_L, B_L]$, $L = 1, 2, \ldots, M$. Of course, since the interval of the variable is open or half open, half closed, it would not affect the feasibility of the algorithm, such that only for writing convenience. Randomly generating M-modules and M-angles, the vector of the module and the angle made the following relationship:

$$\rho_L = \left[0, \frac{B_L - A_L}{2}\right], \quad \theta_L = [-2\pi, 2\pi], \quad L = 1, 2, \ldots, M,$$
$$R_L + jI_L = \rho_L(\cos\theta_L + j\sin\theta_L), \quad L = 1, 2, \ldots, M, \tag{6}$$

where M real and imaginary parts as shown in Table 2 are assigned to the Bird's Nest, resulting in an initial nest.

Complex-valued Cuckoo search via Lévy flight algorithm:

Begin

Objective function $f(x), x = (x_1, x_2, \ldots, x_d)^T$

Generate initial population of n plurality host nests x_i $(i = 1, 2, \ldots, n)$ according to (6)

Setting the Max Generation, and find the best modules, angles and fmin.

While ($t <$ Max Generation) or (stop criterion)

Keep the best module, angle for the SAN

Get a new module and angle randomly by Lévy flight according to (7) and (8)

Plurality nest be transformed into real nest according to (9)

Evaluate its quality/fitness F_i

Choose a nest among n (say, j) randomly

If $(F_i > F_j)$,

Replace j by the new solution;

$nest_j \leftarrow nest_i$

$\rho_j \leftarrow \rho_i$

$\theta_j \leftarrow \theta_i$

End

A fraction (p_a) of worse modules, angles are abandoned and new ones are built;

Compare rand with p_a, obtain a new module and angle.

Evaluate its quality/fitness F_i, and keep the best nest, module and angle

Rank the solutions and find the current best nest, module and angle.

End while

Post process results and visualization

End

Algorithm 2: Pseudo code of the plurality cuckoo search (PCS).

2.2.2. The Method of Nest Update

(1) The method of module update is as follows:

$$\rho_i^{(t+1)} = \rho_i^{(t)} + \partial \oplus L(\lambda), \quad i = 1, 2, 3, \ldots, n, \tag{7}$$

where $\rho_i^{(t)}$ expresses the tth generation value in the ith module. The product $\oplus$ means entry-wise multiplications where $\partial > 0$ is the step size which should be related to the scales of the problem of interest. In most cases, we can use $\partial = 1$. $L(\lambda) \sim u = t^{-\lambda}$, $(1 < \lambda \leq 3)$. Module vector is updated, if r and $> p_a$ then $\rho_i^{(t+1)}$ random can be changed, or not changed. The last to retain a good module vector $\rho_i^{(t+1)}$.

(2) The method of angle update is as follows:

$$\theta_i^{(t+1)} = \theta_i^{(t)} + \partial \oplus L(\lambda), \quad i = 1, 2, 3, \ldots, n, \tag{8}$$

where $\theta_i^{(t)}$ expresses the tth generation value in the ith angle. The product $\oplus$ means entry-wise multiplications, where $\partial > 0$ is the step size which should be related to the scales of the problem of interest. In most cases, we can use $\partial = 1$. $L(\lambda) \sim u = t^{-\lambda}$, $(1 < \lambda \leq 3)$. Angle vector is updated, if r and $> p_a$ then $\theta_i^{(t+1)}$ random can be changed, or not changed. The last to retain a good angle vector $\theta_i^{(t+1)}$.

2.2.3. Fitness Calculation. In order to solve the fitness function, plural Bird's Nest must be changed into a real number; the real value of objective function is determined by modules, and sign of them is determined by amplitude angle, specific practices are as follows:

$$\rho_n = \sqrt{X_{Rn}^2 + X_{In}^2}, \quad n = 1, 2, \ldots, M$$

$$\mathrm{RV}_n = \rho_n \operatorname{sgn}\left(\sin\left(\frac{X_{In}}{\rho_n}\right)\right) + \frac{B_L + A_L}{2}, \quad n = 1, 2, \ldots, M, \tag{9}$$

where ρ_n denotes the nth dimension module, X_{Rn}, X_{In} denote the real part and imaginary part of the nth dimension; respectively, RV_n is converted real variable.

3. The Basic Steps of PCS

Based on above analysis, the basic steps of complex-valued encoding (PCS) can be summarized as the pseudo code shown in Algorithm 2.

4. Simulation Experiment

4.1. Design of Experiment. In this section, the performance of the PCS algorithm is extensively investigated by a large number of benchmark optimization problems. All computational experiments are conducted with Matlab7.0 and run on CPU T3100, 1.90 GHZ with 2 GB memory capacity.

Algorithm parameters are set as follows: because the updates of bird's nest locations are divided into two steps in the complex-valued encoding, there are two update calculations; when the two types of encoding have the same sizes

Table 2: Test the improved algorithm's benchmark functions.

Functions	Dim	Domain	Theoretical value
$f(x) = \sum_{i=1}^{d-1} \left[(1 - x_i)^2 + 100(x_{i+1} - x_i^2)^2 \right]$	10	$[-100, 100]$	0
$f(x) = \sum_{i=1}^{d} x_i^2$	15	$[5.12, 5.12]$	0
$f(x) = 10d + \sum_{i=1}^{d} \left[x_i^2 - 10\cos(2\pi x_i) \right]$	20	$[5.12, 5.12]$	0
$f(x) = -20\exp\left[-0.2\sqrt{(1/d)\sum_{i=1}^{d} x_i^2} \right] - \exp\left[(1/d)\sum_{i=1}^{d} \cos(2\pi x_i) \right] + 20 + e$	30	$[-32.768, 32.768]$	0
$f(x) = -\cos(x)\cos(y)\exp\left[-(x-\pi)^2 - (y-\pi)^2 \right]$	2	$[100, 100]$	-1
$f(x) = 1 + (1/4000)\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d} \cos(x_i/\sqrt{i})$	10	$[-600, 600]$	0

Table 3: The results of experiment in running 20 times.

Functions	Algorithm	Best	Worst	Mean	Variance
Rosenbrock	CS	671.2474	4.8400e + 003	2.1090e + 003	1.0339e + 006
	PCS	**7.9206**	**8.9835**	**8.3874**	**0.1028**
Sphere	CS	0.0648	0.1723	0.1141	0.0012
	PCS	**0**	**0.0088**	**8.9793e − 004**	**4.6582e − 006**
Rastrigin	CS	112.8084	128.5006	120.8996	20.0181
	PCS	**100**	**104.0008**	**100.8662**	**1.6589**
Ackley	CS	10.7296	13.7988	12.3351	0.5920
	PCS	**0.0052**	**13.0287**	**7.7304**	**13.5185**
Easom	CS	−1	−0.9992	−0.9998	4.1112e − 008
	PCS	**−1**	**−0.9998**	**−1**	**3.1784e − 009**
Griewank	CS	0.4814	0.9043	0.7230	0.0111
	PCS	**0**	**0.1129**	**0.0256**	**0.0017**

of population, the computational complexity of the complex-valued encoding is approximately two times the one of the real encoding. To compare with the performance of the two methods, the size of plurality population nest is half of the one of real population nest. The size of complex-valued encoding nest is 20, the size of real encoding nest is 40, and the maximum iteration times is 200, $p_a = 0.25$.

4.2. Experimental Results and Analysis. In this section, we test on six different functions to verify that the algorithm proposed in this paper is feasible and effective. 20 independent runs are made for the PCS algorithms, and the results obtained by the PCS algorithms are presented in Table 3. From Table 3 we can find that complex-valued encoding method can achieve better fitness than real number coding method. In terms of Rosenbrock function, either best fitness or average fitness, we can see that the precision of PCS is improved 10^2 and 10^3 higher than CS, respectively. As far as Sphere function are concerned, the optimal value of PCS reaches the theoretical value; the average fitness is improved 10^3 higher than CS. About Rastrigin, Ackley function, PCS average value and the optimal ratio of CS are improved, but not obvious. With respect to Easom function, the optimal value of PCS reaches the theoretical value, but CS does not. In terms of Griewank function, the optimal value of PCS can also reach the theoretical value. Figure 1 shows that this method is better than that real-coded in the convergence rate and convergence precision; this can be explained from the average fitness evolution curves. From the average fitness figures, we can see that the change of average fitness obtained by the complex-valued encoding method is much greater than the one obtained by the real number encoding method, especially in the early evolution. Because the average fitness of the change is bigger, that individual is scattered, not concentrated in one or a few local points. In the iterative process, there is a trend that these points are close to a better location, but this is easy to make the population into the local convergence; therefore maintaining the diversity of the population is very important. Average fitness changes greatly; to some extent, the diversity of the population is better, thus not easy to fall into the local convergence.

5. Implement PCS in Determining PID Controller Parameter

In industry process, people generally implement Ziegler-Nichlos rule in determining PID controller parameters; the control effect is generally difficult to meet the requirements of the control system. In this section, we implement PCS in determining PID controller parameters and compare the results with CS and PSO results. The parameters of controller are mapped bird's nest, then optimizing them by the PCS method. In the previous work [7, 8], authors have implemented transfer function from industry

$$G(s) = \frac{2}{s^2 + 1.5s + 2} e^{-0.2s} \tag{10}$$

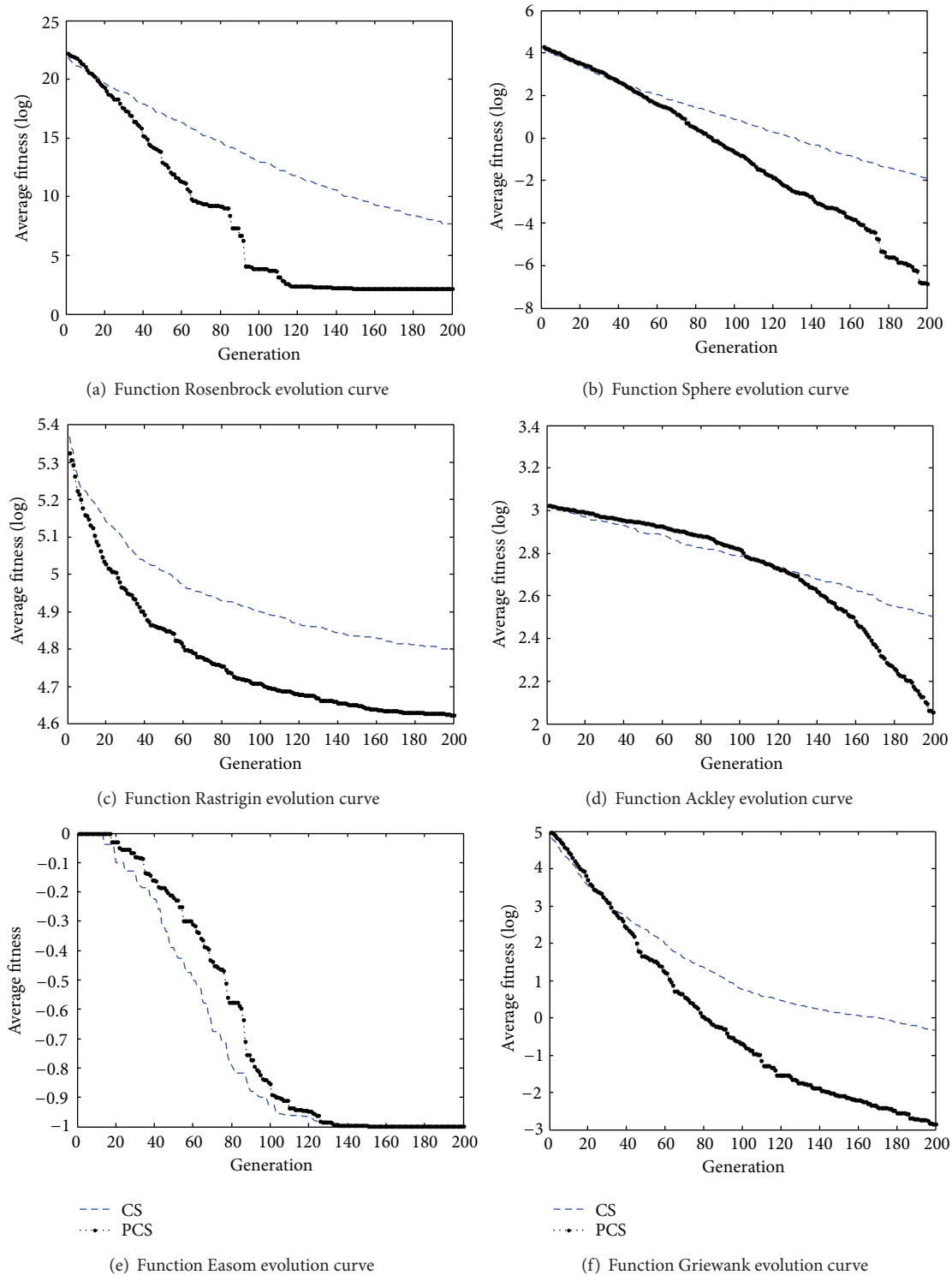


Figure 1: The evolution graph of the average fitness in 20 trails.

PID Controller can be described as

$$G_c(s) = k_p + \frac{k_i}{s} + k_d s. \tag{11}$$

Through adjusting the three parameters, the system satisfies the required performance indicators. The bird's nest is in the three dimensional space encoded; the parameters are set as follows: $k_p \in [0.01, 20]$, $k_i \in [0.01, 2]$, $k_d \in [0.01, 2]$.

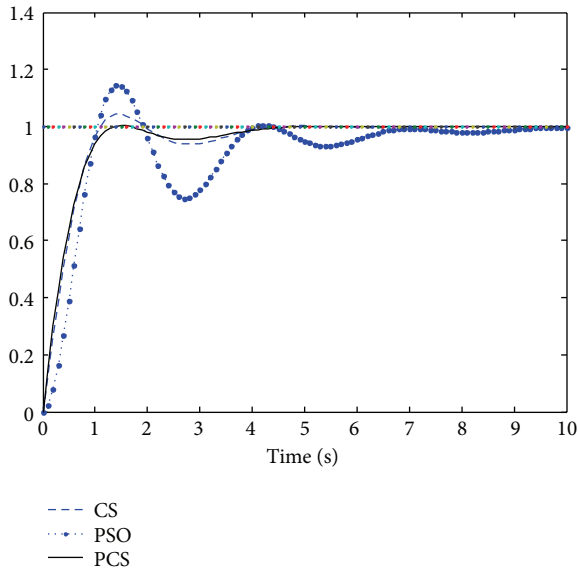


Figure 2: The comparison of unit step response curve.

Table 4: PID Controller tuning parameters.

Tuning method	k_p	k_i	k_d
PSO	0.0100	2.0000	1.0518
CS	0.6491	2.0000	2.0000
PCS	0.8096	1.9738	1.9995

The number of nest is 20; the maximum iteration time; is 100. The most crucial step in applying PCS to choose the objective functions J that are used to evaluate fitness of each nest; the performance indices are defined as follow [8]:

$$J = \alpha * \int_0^{\infty} |e(t)|\, td_t + \beta * ts, \tag{12}$$

where $e(t)$ is the error signal in time domain, ts is the tuning time, α, β is weight, which is set as 0.1 and 0.9, respectively.

In the experiments, we implement PSO, CS, and PCS in determining PID controller parameters. The results obtained by the three algorithms are presented in Table 4. Figure 2 is the unit step response curve of control object.

In Figure 2, we can see that the settling time is not so different among all methods; the settling time from short to long is followed by PCS, CS, and PSO. In addition, the overshoot of the three algorithms is (in descending order): PSO, CS, and PCS.

6. Conclusions

This paper proposes Cuckoo search based on complex-valued encoding; the individual of Bird's Nest is denoted by plurality, so a diploid swarm is structured by a sequence plurality; the Bird's Nest can express the space dimension much more than the real-coded one. Compared with traditional real-coded, the Bird's Nest also has to contain more information, so that the algorithm improves the search capabilities of the global optimum. To verify the proposed PCS algorithm, also a number of benchmark optimization problems and PID controller parameter tuning are solved using this concept and quite satisfactory results are obtained. CS is the algorithm proposed in the last two years; the theoretical analysis and other applications require further study.

Acknowledgments

This work is supported by National Science Foundation of China under Grant no. 61165015, Key Project of Guangxi Science Foundation under Grant no. 2012GXNSFDA053028, and Key Project of Guangxi High School Science Foundation under Grant no. 20121ZD008 and funded by Open Research Fund Program of Key Lab of Intelligent Perception and Image Understanding of Ministry of Education of China under Grant no. IPIU01201100.

References

[1] X. S. Yang and S. Deb, "Cuckoo search via Lévy flights," in *Proceedings of the World Congress on Nature and Biologically Inspired Computing (NABIC '09)*, pp. 210–214, IEEE, December 2009.

[2] X. S. Yang and S. Deb, "Engineering optimization by cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimization*, vol. 4, pp. 330–343, 2010.

[3] D. Casasent and S. Natarajan, "A classifier neural net with complex-valued weights and square-law nonlinearities," *Neural Networks*, vol. 8, no. 6, pp. 989–998, 1995.

[4] Z. H. Zheng, Y. Zhang, and Y. H. Qiu, "Genetic algorithm based on complex-valued encoding," *Control Theory & Applications*, vol. 20, no. 1, pp. 97–100, 2003.

[5] D. B. Chen, H. J. Li, and Z. Li, "Particle swarm optimization based on complex-valued encoding and application in function optimization," *Computer and Applications*, vol. 45, no. 10, pp. 59–61, 2009.

[6] X. S. Yang and S. Deb, "Multiobjective cuckoo search for design optimization," *Computer ' Operations Research*, vol. 40, no. 6, pp. 1616–1624, 2013.

[7] X. Liu, "Improved particle swarm optimization and its application in PID parameters optimization," *Electronic Design Engineering*, vol. 19, no. 9, pp. 79–82, 2011.

[8] C. L. Zhang, J. L. Jiang, S. H. Jiang, and Q. Li, "Adaptive hybrid particle swarm optimization algorithm and application," *Application Research of Computers*, vol. 28, no. 5, pp. 1696–1698, 2011.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 951475, 11 pages
http://dx.doi.org/10.1155/2013/951475

Research Article

Hybrid Support Vector Regression and Autoregressive Integrated Moving Average Models Improved by Particle Swarm Optimization for Property Crime Rates Forecasting with Economic Indicators

Razana Alwee, Siti Mariyam Hj Shamsuddin, and Roselina Sallehuddin

Soft Computing Research Group, Faculty of Computing, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia

Correspondence should be addressed to Razana Alwee; razana@utm.my

Received 2 April 2013; Accepted 8 May 2013

Academic Editors: P. Agarwal, S. Balochian, V. Bhatnagar, and J. Yan

Crimes forecasting is an important area in the field of criminology. Linear models, such as regression and econometric models, are commonly applied in crime forecasting. However, in real crimes data, it is common that the data consists of both linear and nonlinear components. A single model may not be sufficient to identify all the characteristics of the data. The purpose of this study is to introduce a hybrid model that combines support vector regression (SVR) and autoregressive integrated moving average (ARIMA) to be applied in crime rates forecasting. SVR is very robust with small training data and high-dimensional problem. Meanwhile, ARIMA has the ability to model several types of time series. However, the accuracy of the SVR model depends on values of its parameters, while ARIMA is not robust to be applied to small data sets. Therefore, to overcome this problem, particle swarm optimization is used to estimate the parameters of the SVR and ARIMA models. The proposed hybrid model is used to forecast the property crime rates of the United State based on economic indicators. The experimental results show that the proposed hybrid model is able to produce more accurate forecasting results as compared to the individual models.

1. Introduction

Quantitative forecasting methods are classified into causal and time series models. The causal models are based on the relationship between the variable to be forecasted and independent variables. A linear relationship is typically used in the causal models. Regression, econometric models, and input-output models are examples of some of the causal models. The time series models are models that use historical data to estimate the future which can be categorized into linear and nonlinear models. Most of the linear models are statistical models such as exponential smoothing, moving average, and autoregressive integrated moving average (ARIMA). However, the nonlinear models consist of statistical models such as bilinear models, the threshold autoregressive (TAR) models, and autoregressive conditional heteroscedastic (ARCH) as well as nonstatistical models such as artificial neural networks (ANN) and support vector regression (SVR).

The causal models such as regression and econometric models are commonly used in crime rates forecasting. The causal models can describe the causal relationship between the crime variable and other explanatory variables. However, the development of the causal models is quite complex and requires theoretical assumptions about the relationship between the explanatory variables. Therefore, the time series model has been considered as a promising alternative tool for crime rates forecasting. The application of time series models for crime rates forecasting is still scarce. Standard time series models usually require a substantial number of observations. However, insufficient crime data makes the standard time series models less suitable for crime rates forecasting. Therefore, a new model that suits small data set is needed to improve the crime rates forecasting performance.

There are two types of time series models, namely, linear and nonlinear models. The linear models can only model the linear relationship, while the nonlinear models only

model the nonlinear relationship. In the literature, there is no single model that can predict well in all conditions. Therefore, many researchers have used a hybridization of linear model with nonlinear model as an approach to time series forecasting [1]. The hybrid linear and nonlinear models are not only capable of modeling the linear and nonlinear relationships, but are also more robust to changes in time series patterns [2]. Artificial neural networks (ANNs) and support vector regression (SVR) are two nonlinear models usually being employed, while ARIMA, seasonal autoregressive integrated moving average (SARIMA), autoregression (AR), exponential smoothing, moving average, and multiple linear regression are usually used to represent linear model in hybridization of linear and nonlinear model. Several examples of hybrid time series models that have been proposed in the literature are ARIMA and ANN [1–16], ARIMA and SVR [17–23], seasonal autoregressive integrated moving average (SARIMA) and SVR [24, 25], autoregression (AR) and ANN [26], exponential smoothing and ANN [27], ARIMA and genetic programming (GP) [28], exponential smoothing, ARIMA and ANN [29], and multiple linear regression (MLR) and ANN [30].

A hybridization of ARIMA and ANN models as linear and nonlinear model is extensively studied by researchers since it produces promising results. However, this hybridization requires sufficient data to produce a good model. Furthermore, ANN models suffer from several problems such as the need for controlling numerous parameters, uncertainty in solution (network weights), and the danger of over fitting. Support vector regression (SVR) was proposed by Vapnik [31] in order to overcome the drawback of ANN. SVR is a nonlinear model to solve regression problems and has been used by researchers as an alternative model to ANN [17–23]. A hybridization of ARIMA and SVR has been successfully applied in time series forecasting such as stock market [17, 20], electricity price [22], and power load [18, 19]. There are four factors that contributed to the success of SVR which are good generalization, global optimal solution, the ability to handle nonlinear problems, and the sparseness of the solution. This has made SVR a robust model to work with small training data, nonlinear, and high-dimensional problems [32]. Despite the advantages, SVR also has some limitations. For example, SVR model parameters must be set correctly as it can affect the regression accuracy. Inappropriate parameters may lead to overfitting or underfitting [33]. Genetic algorithm (GA) and particle swarm optimization (PSO) are among the approaches that have been used by researchers to estimate the SVR parameters. However, PSO is easier to implement as compared to GA, because it does not require evolution operators such as crossover and mutation [34]. As for linear model, ARIMA is the preferred choice by researchers for hybridization with nonlinear model due to its ability to model various smoothing models such as simple autoregressive (AR), a simple moving average (MA), and a combination of AR and MA (ARMA model) [23]. ARIMA has high forecasting performance when the data set is large and linear. However, it is not robust for small data sets and nonlinear data. Therefore, several improvements have been proposed to improve the performance of ARIMA [35]. Asadi et al. [36] suggested the use of PSO to estimate the parameters of ARIMA model for small data sets.

Most hybrid models use a sequence of linear and nonlinear models, but there are also hybrid models that use a sequence of nonlinear and linear models. It depends on which component is more dominant, either linear or nonlinear. The dominant component needs to be modeled first. However, the linear and nonlinear components in the data can have interaction. It is quite difficult to determine which component is more dominant. Modeling linear patterns using linear model will change the nonlinear patterns, and vice versa. However, in comparison with ANN model, the SVR is able to keep the linear components undamaged [22]. A hybridization of SVR and ARIMA model with a sequence of nonlinear and linear models has been found to outperform the existing neural network approaches, traditional ARIMA models, and other hybrid models, such as ARIMA and ANN (with a sequence of linear and nonlinear models and a sequence of nonlinear and linear models), as well as ARIMA and SVR (with a sequence of linear and nonlinear models) [22]. Therefore, this study uses a sequence of nonlinear and linear models by combining SVR with ARIMA model.

The hybrid linear and nonlinear model has never been used for crime rates forecasting. Most of the models used for crime rates forecasting are linear. As for real-world data, crime rates may have linear and nonlinear components, where the use of linear model may not be adequate to forecast the crime rates. Therefore, this study attempts to propose a hybrid time series model that can work well with limited data that consist of both linear and nonlinear components. SVR is used as nonlinear model and ARIMA is employed as linear model. First, the SVR is used to model the nonlinear component. After that, the remaining from the SVR model, which represents the linear component, is modeled by using ARIMA. In order to overcome the drawbacks of the SVR and ARIMA model, particle swarm optimization (PSO) is used to estimate the parameters of SVR and ARIMA models. PSO has the ability to escape from local optima, easy to implement, and has fewer parameters to be adjusted [34]. There are several factors that influence the property crime rates, among these are economic indicators. Therefore, this study uses three economic indicators, namely, unemployment rate, gross domestic product, and consumer price index as inputs to the proposed hybrid model.

The remainder of this study is organized as follows. Related work on crimes and economic indicators is first discussed in Section 2. In Section 3, brief explanations on the support vector regression, ARIMA, particle swarm optimization, and the proposed hybrid model are described. Section 4 describes the data set and model evaluation employed in this study. The determination parameters of model and the analysis of the results are presented in Sections 5 and 6, respectively. Finally, a brief conclusion is drawn in Section 7.

2. Related Work on Crimes and Economic Conditions

Economic conditions are often considered to be related to crimes, especially property crimes. In the literature, many

studies have been done by researchers in order to relate the economic conditions with property crimes. The unemployment rate is often selected by the researchers in their studies to represent the economic conditions [37]. A study using a country level panel data set from Europe found that unemployment has a positive influence on property crimes [38]. Meanwhile, another study based on UK annual regional data has discovered that unemployment is an important explanatory variable for crimes motivated by economic gain [39]. Results produced by some other studies also found significant relationship between the unemployment and property crimes. Among the findings is that motor vehicle theft is significantly associated with the unemployment rate [40] and is also cointegrated with male youth unemployment [41]. Another finding shows that unemployment has a positive effect on burglary, car theft, and bike theft [42].

Unemployment, especially among youth and young adults, is also found to influence crimes. According to a study on the United States arrest data, unemployment has a positive relationship with theft crimes among youth and young adults (16–24 years) [43]. Another study investigated the relationship between crime with male adult (26–64 years) and youth (16–25 years) unemployment in Britain [44]. The results indicate that youth unemployment and adult unemployment are both significantly and positively related to burglary, theft, fraud, and forgery as well as total crime rates.

In addition to unemployment, other economic indicators such as consumer price index, gross domestic product, and consumer sentiment index were also studied by the researchers to examine the relationship between economic conditions with crimes. Several researchers used the consumer price index to measure the inflation [45, 46]. Inflation reduces the purchasing power and increases the cost of living. A study on the impact of inflation rate on crime in the United States using the modified Wald causality test found that the crime rate is co integrated with inflation and unemployment rates [45]. Further, another study which examined the linkages between inflation, unemployment, and crime rates in Malaysia revealed that inflation and unemployment are positively related to the crime rate [46]. Meanwhile, for gross domestic product, a study to explain the relationship between national crime rates with social and economic variables has found that robbery and homicide have significant negative relationship with gross domestic product [47].

As a conclusion, the economic conditions do have an influence on the property crime rates. Therefore, this study attempts to employ the economic conditions to forecast the property crime rates. The economic conditions will be represented by three economic indicators, namely, unemployment rate, consumer price index, and gross domestic product. These economic indicators are used as input to the proposed hybrid model for forecasting property crime rates.

3. Methodology

In this section, explanations on support vector regression, ARIMA, and particle swarm optimization are summarized as a basis to describe the proposed hybrid model.

3.1. Support Vector Regression (SVR). Support vector regression (SVR) is a nonlinear model to solve regression problems. SVR training process is similar to solving the linearly constrained quadratic programming problems that provides a unique optimal value and there is no local minimum problem. The solution is sparse, as only essential data are used to solve the regression function. Lagrangian multipliers are introduced to solve the problem. The SVR model is given by formula [48]

$$f(\mathbf{x}) = (\mathbf{z} \cdot \phi(\mathbf{x})) + b, \tag{1}$$

where $\mathbf{z}$ is weight vector, b is a bias value, and $\phi(\mathbf{x})$ is a kernel function. SVR used ε-insensitivity loss function which can be expressed as formula

$$L_\varepsilon(f(\mathbf{x}) - y) = \begin{cases} |f(\mathbf{x}) - y| - \varepsilon, & \text{if } |f(\mathbf{x}) - y| \geq \varepsilon, \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where ε is the region for ε-insensitivity. Loss is accounted only if the predicted value falls out of the band area. The SVR model can be constructed to minimize the following quadratic programming problem:

$$\begin{aligned} \text{min:} \quad & \frac{1}{2}\mathbf{z}^T\mathbf{z} + C\sum_i (\xi_i + \xi_i^*), \\ \text{subjected to} \quad & y_i - \mathbf{z}^T\mathbf{x}_i - b \leq \varepsilon + \xi_i, \\ & \mathbf{z}^T\mathbf{x}_i + b - y_i \leq \varepsilon + \xi_i^*, \\ & \xi_i, \xi_i^* \geq 0, \end{aligned} \tag{3}$$

where $i = 1, 2, \ldots, n$ is the number of training data, $(\xi_i + \xi_i^*)$ is the empirical risk, $(1/2)\mathbf{z}^T\mathbf{z}$ is the structure risk preventing overlearning and lack of applied universality, and C is the regularization parameter. After selecting proper regularization parameter (C), width of band area (ε) and kernel function (K), the optimum of each parameter can be resolved through Lagrange function. The commonly used kernels are linear kernel, polynomial kernel, radial basis function (RBF), or Gaussian kernel and sigmoid kernel. Formulas (4), (5), (6), and (7) are the equation for linear kernel, polynomial kernel, RBF kernel [49], and sigmoid kernel [50], respectively. Consider

linear kernel,

$$K(x_i, x_j) = x_i^T x_j, \tag{4}$$

polynomial kernel,

$$K(x_i, x_j) = (1 + x_i \cdot x_j)^d, \tag{5}$$

RBF kernel,

$$K(x_i, x_j) = \exp\left(-\gamma \left\| x_i - x_j \right\|^2\right), \tag{6}$$

sigmoid kernel,

$$K\left(x_i, x_j\right) = \tanh\left[v\left(x_i, x_j\right) + \alpha\right]. \quad (7)$$

The type of kernel function influences the parameters of SVR kernel. The kernel function and parameters of SVR kernel function should be set properly because it can affect the regression accuracy. Inappropriate parameters may lead to over-fitting or under-fitting [33]. This study uses the RBF kernel function because it suits most forecasting problems [51]. The RBF kernel is also effective and has fast training process [52]. For the RBF kernel function, there are three important parameters to be determined [53].

(i) Regularization parameter C: C is parameter for determining the tradeoff cost between minimizing training error and minimizing model complexity.

(ii) Kernel parameter (γ): γ represents the parameter of the RBF kernel function.

(iii) The tube size of e-insensitive loss function (ε): ε is the approximation accuracy placed on the training data points.

These parameters must be set correctly, in order to produce accurate estimation model. In this study, these parameters are determined through particle swarm optimization (PSO). The explanation on how PSO is used to estimate the parameters of SVR is given in Section 3.3.1.

3.2. Autoregressive Integrated Moving Average (ARIMA). Autoregressive integrated moving average (ARIMA) model was introduced by Box and Jenkins and has become one of the most popular models in forecasting [17]. The ARIMA model is a stochastic model for time series forecasting where the future value of a variable is a linear function of past observations and random errors, expressed as

$$\begin{aligned} y_t &= \theta_0 + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \varepsilon_t \\ &\quad - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_q \varepsilon_{t-q}, \end{aligned} \quad (8)$$

where y_t is the actual value and ε_t is the random error at time t, and ϕ_i ($i = 1, 2, \ldots, p$) and θ_j ($j = 0, 1, 2, \ldots, q$) are model parameters. Integers, p and q are referred to as order of the model and random errors, ε_t, are assumed to be independently and identically distributed with a mean of zero and a constant variance of σ^2 [2].

ARIMA model is developed using Box-Jenkins methodology that involves the following three iterative steps [2].

(i) Model Identification. At this step, data transformation should be done if necessary, to produce a stationary time series. Stationary time series is needed because the ARIMA model is developed with the assumption that the time series is stationary. Mean and autocorrelation structure are constant over time for stationary time series. Therefore, for a time series that exhibit trends and heteroscedasticity, differentiation and power transformation are necessary to change the time series to be stationary. Next, autocorrelation (ACF) and partial autocorrelation (PACF) are calculated from the data and compared to theoretical autocorrelation and partial autocorrelation for the various ARIMA models to identify the appropriate model form. The selected model is considered as a tentative model. Steps (ii) and (iii) in turn will determine whether the model is adequate [54].

(ii) Parameter Estimation. Once the tentative model is identified, parameters in ARIMA model can be estimated using the nonlinear least square procedure.

(iii) Diagnostic Checking. The last step in model development is to check whether the model is adequate. Model assumptions about the errors must be met. Several diagnostic statistics and plots of the residual can be done to check the goodness of fit of the tentative model to the historical data. Among plots that can be very useful are histogram, normal probability plot, and time sequence plot. Residual autocorrelations should be small where chi-square test can be used to test the overall model adequacy. However, if the model is considered inadequate, a new tentative model should be identified and steps (ii) and (iii) will be repeated again.

Once a satisfactory model is produced, the three-step development process is no longer repeated and selected model will be used for forecasting purposes. In this study particle swarm optimization (PSO) as suggested by Asadi et al. [36] is used to estimate the parameters of ARIMA model. The explanation on how PSO is used to estimate the parameters of ARIMA model is given in Section 3.3.2.

3.3. Particle Swarm Optimization (PSO). Particle swarm optimization (PSO) is one of stochastic optimization methods introduced by Kennedy and Eberhart [55]. This method is based on the natural evolution process which uses swarming strategies in bird flocking and fish schooling. PSO is a population-based which consists of particles. Initially, the particles are randomly generated. Each particle has a position and velocity, which represents a potential solution to a problem in D-dimensional space. The position and velocity of ith particle are denoted by $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$ and $V_i = (v_{i1}, v_{i2}, \ldots, v_{iD})$, respectively. While solving the search problem, each particle explores the search space by moving in previous direction, its previous best particle (*pbest*), and the best solution for the entire population (*gbest*). The velocity and position of each particle are updated by using the following [56]:

$$\begin{aligned} v_{ij}(t+1) &= w \cdot v_{ij}(t) + c_1 \cdot \text{rand1}_{ij} \cdot \left(pbest_{ij}(t) - x_{ij}(t)\right) \\ &\quad + c_2 \cdot \text{rand2}_{ij} \cdot \left(gbest_j(t) - x_{ij}(t)\right), \end{aligned} \quad (9)$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1), \quad (10)$$

where $v_{ij}(t)$ is the velocity of ith particle at iteration t, $x_{ij}(t)$ is the position of ith particle at iteration t, $j = 1, 2, \ldots, D$, is the dimension of the search space, w is the˜inertia weight

to balance the global and local search abilities of particles, $\text{rand}1_{ij}$ and $\text{rand}2_{ij}$ are two uniform random numbers generated independently within the range of $[0, 1]$, c_1 and c_2 are two learning factors which control the influence of the social and cognitive components, $pbest_{ij}(t)$ is the best previous position yielding the best fitness value for ith particle at iteration t, and $gbest_j$ is the global best particle by all particles at iteration t. After changing the position of the particle, the particle's fitness value is evaluated. The *pbest* and *gbest* are updated based on the current position of the particles. As this process is repeated, the whole population evolves towards the optimum solution.

The following are the steps in PSO implementation [57].

Step 1. Initialize the positions and velocities of all the particles randomly in the D-dimensional search space by uniform probability distribution function.

Step 2. Evaluate the fitness values of the particles.

Step 3. Update *pbest* for each particle; if the current fitness value of the particle is better than its *pbest* value, set the *pbest* equal to the current position of the particle.

Step 4. Update *gbest*; if the current fitness value of the particle is better than the *gbest* value, then set *gbest* equal to the current position of the particle.

Step 5. Update the velocity and position of each particle using (9) and (10), respectively.

Step 6. Repeat Steps 2 to 5, until stopping criteria are met, such as a sufficient good fitness value or a maximum number of iterations.

The explanations on how PSO is used to estimate the parameters of SVR and ARIMA models are given in Sections 3.3.1 and 3.3.2, respectively.

3.3.1. PSO for SVR Parameters Estimation (PSOSVR). Since there are three parameters to be estimated, the ith particle is represented by the three-dimensional vectors, $X_i = (x_{i1}, x_{i2}, x_{i3})$ and $V_i = (v_{i1}, v_{i2}, v_{i3})$, where the first, second, and third dimensions of the vectors refer to C, γ, and ε, respectively. In this study, the fitness is defined by k-fold cross-validation, where $k = 5$. In k-fold cross-validation, the training data set is divided into k subsets of equal size. One subset is used for validation. The regression function is built with a given set of parameters (C, γ, ε) using the remaining $k-1$ subsets. The performance of the parameter set is measured by the root mean square error (RMSE) on the validation set. Each subset is used once for validation and the process is repeated k times. The average of RMSE on the validation set from 5 trials is used as a measure of fitness. The RMSE is defined as

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \widehat{y}_t)^2}, \tag{11}$$

where n is the number of validation data; y_t is the actual value and $\widehat{y}_t$ is the predicted value.

3.3.2. PSO for ARIMA Parameters Estimation (PSOARIMA). A hybrid model of PSO and ARIMA was proposed by Asadi et al. [36] to estimate the parameters of ARIMA model. This method is efficient for cases where inadequate historical data is available. The implementation of this method involves two main steps. First, an ARIMA model is generated by applying the Box and Jenkins method. Second, the PSO model is used to estimate the ARIMA parameters. The data set is divided into training and testing data set. The training data set is used to estimate the ARIMA model. However, testing data set is used to evaluate the estimation results. In this study, the fitness is defined by sum square error (SSE) as follows:

$$\text{SSE} = \sum_{t=1}^{n}(y_t - \widehat{y}_t)^2, \tag{12}$$

where n is the number of training data; y_t is the actual value and $\widehat{y}_t$ is the predicted value.

3.4. The Proposed Model. The proposed hybrid model consists of a nonlinear model, SVR, and a linear model, ARIMA. According to Zhang [2], it is reasonable to consider a time series as the composition of a linear autocorrelation structure and a nonlinear component, as

$$y_t = N_t + L_t, \tag{13}$$

where N_t denotes the nonlinear component and L_t denotes the linear component. These two components are estimated from data using the following two steps. First, SVR model is used to model nonlinear components in the data. Second, the residual from the nonlinear model is modeled using linear model, ARIMA. Let r_t denote the residual which is represented by

$$r_t = y_t - \widehat{N}_t. \tag{14}$$

The residual represents linear components that cannot be modeled by SVR model. The SVR and ARIMA parameters are estimated by applying PSO, as described previously. Forecasting results from SVR and ARIMA models will be combined to represent the forecasting results of the proposed hybrid model. The combined forecast is shown by the formula (15)

$$\widehat{y}_t = \widehat{N}_t + \widehat{L}_t. \tag{15}$$

Figure 1 shows the flowchart for the proposed hybrid model, PSOSVR_PSOARIMA.

4. Data Set and Model Evaluation

This section describes the data set used and the model evaluation carried out in this study.

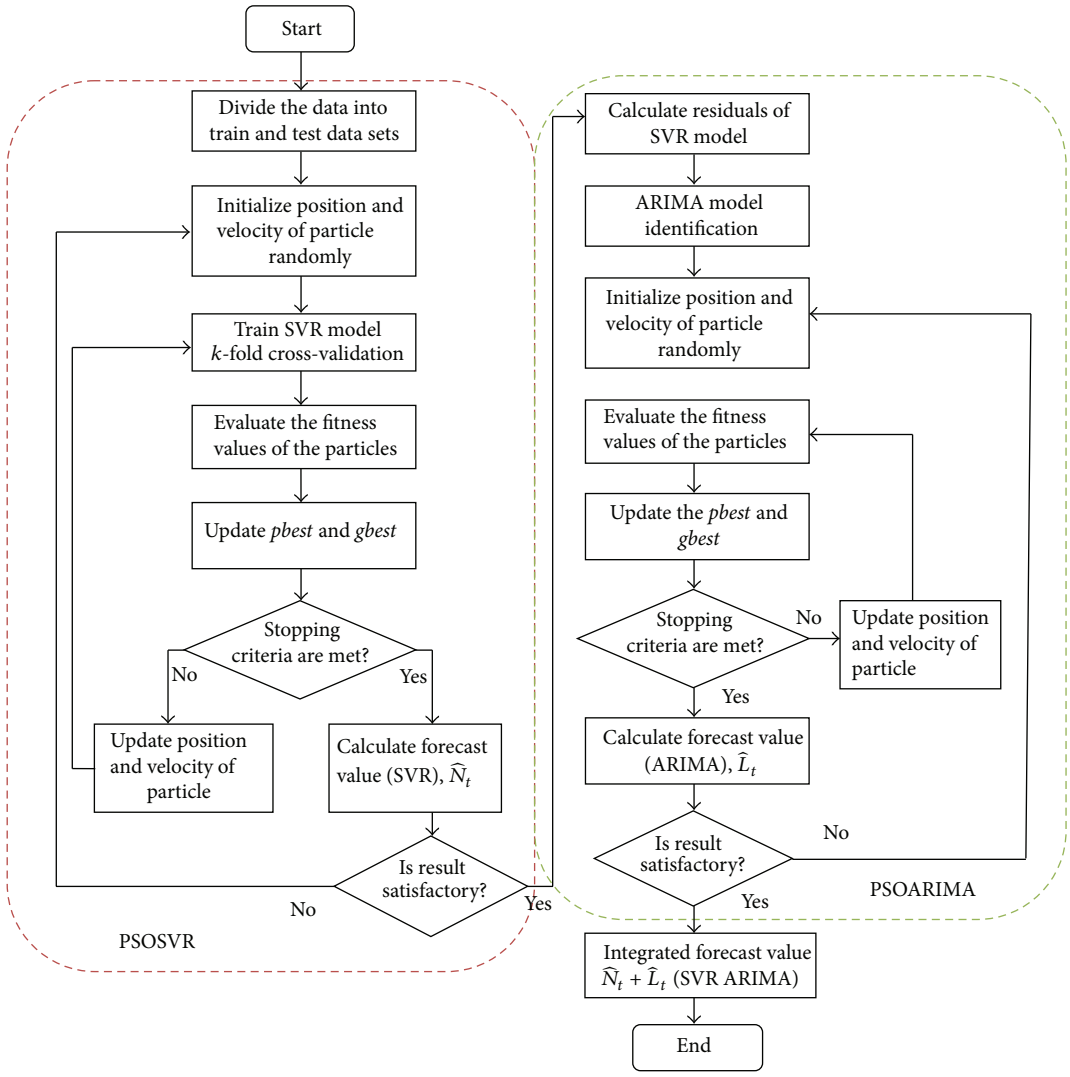


Figure 1: The Flow chart for the proposed hybrid PSOSVR and PSOARIMA model.

4.1. Data Set. This study uses annual data of property crime rates, consumer price index for all urban consumers (Apparel), gross domestic product in United States natural log of billions of chained 2005 US Dollars, and unemployment rate (20 to 24 years) from 1960 to 2009 in United States. The crime rates are obtained from the Uniform Crime Reporting Statistics website (http://www.ucrdatatool.gov/), while economic indicators data are available on Economic Research Federal Reserve Bank of St. Louis website (http://research.stlouisfed.org/). Property crime data comprised property crime rate, vehicle rate, larceny-theft rate, and burglary rate. In addition to economic indicators, two crime indicators are also used in the model, which are one-year lagged property crime rate (PCR) and one-year lagged robbery rate. Data is divided into training and test data sets. The training data set is used to develop the models while the test data set is used to evaluate the forecasting performance of the models. In this study, 90 percent of the data will be used as training (1961 to 2004) and 10 percent is as test data set (2005 to 2009).

4.2. Model Evaluation. The performance of the proposed hybrid model is evaluated using the test data set. The forecasting performance of the proposed hybrid model is evaluated using four types of evaluations.

(i) Descriptive Statistics. Graph of actual values and forecasting of testing data set is plotted in order to see the pattern of model predictions compared with the actual data patterns. A box plot diagram is used to check the error values. Box plot is used to see the dispersion of error values such as the position of median whether it is close to zero, and to ensure that there are no extreme values in error.

(ii) Quantitative Error Measurements. Four types of quantitative error measurements are conducted, namely, root mean

square error (RMSE), mean square error (MSE), mean absolute percentage error (MAPE), and mean absolute deviation (MAD). Formulas (16), (17), (18), and (19) are the equation for RMSE, MSE, MAPE, and MAD, respectively. Consider

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \widehat{y}_t)^2}, \tag{16}$$

$$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n}(y_t - \widehat{y}_t)^2, \tag{17}$$

$$\text{MAPE} = \sum_{t=1}^{n}\left|\frac{y_t - \widehat{y}_t}{y_t}\right| \times \frac{100}{n}, \tag{18}$$

$$\text{MAD} = \sum_{t=1}^{n}\frac{|y_t - \widehat{y}_t|}{n}, \tag{19}$$

where n is the number of test data; y_t is the actual value and $\widehat{y}_t$ is the predicted value.

(iii) Model Comparative Performance. The performance of the proposed model is compared with individual model, PSOSVR, ARIMA, and PSOARIMA. A comparison with the hybrid model PSOSVR_ARIMA is also done to see the effects of PSO on the ARIMA model in the hybridization of SVR and ARIMA model.

(iv) Hypothesis Test. The hypothesis test is performed to prove that there is no significant means difference between the forecasting values and the actual data. Paired sample t-test is used in this study. There are two hypotheses, the null hypothesis (H_0) and the alternative hypothesis (H_1). Let μ_1 be the mean of actual data, μ_2 the mean of forecasting values of the forecasting model, and ($\mu_1 - \mu_2 = \mu_D$) the difference of means. The hypothesis is that

$$\begin{aligned} H_0 &: \mu_D = 0, \\ H_1 &: \mu_D \neq 0. \end{aligned} \tag{20}$$

The test statistic is shown by formula (21), which is t distributed with $n_D - 1$ degrees of freedom. Consider

$$t = \frac{\overline{y}_D - \mu_D}{s_D/\sqrt{n_D}}, \tag{21}$$

where $\overline{y}_D$ is a sample mean difference, s_D is a sample standard deviation of the difference, and n_D is a sample size. The mean of forecasting values is equal to the mean of actual values, if the hypothesis test fails to reject the null hypothesis. It indicates that the model is appropriately used as forecasting model since it represents the real situation.

5. Determination Parameters of Model

In PSO algorithm, population size is set to 5, maximum number of iterations is set to 50 and the value of c_1, c_2 is set to 1. The inertia weight, w, initially is set to 1.4, and its value is decreased along with the iterations according to [58]

$$\text{Weight} = \frac{(\text{Weight} - 0.4) * (\text{Maxiter} - \text{iter})}{\text{Maxiter}} + 0.4, \tag{22}$$

where Maxiter is the maximum iteration and iter is the current iteration. The range values of SVR parameters (C, γ, ε) used in this study are shown in Table 1. Meanwhile, the searching range for ARIMA parameters is between −100 and 100.

Table 1: The range values for the parameters (C, γ, ε).

Parameters	Range values
C	2^{-1} to 2^{7}
γ	2^{-4} to 2^{2}
ε	0.01 to 0.05

The optimum parameter for SVR model obtained by PSO is $(C, \gamma, \varepsilon) = (60.1924, 0.0625, 0.01)$. SVR model was developed using the optimum parameter which is then used to forecast the property crime rates using the training data and test data. After that, the residual which represents the difference between the actual value and prediction value was calculated. The residual of the training data was used to build the ARIMA model. Based on ACF and PACF, the appropriate model for the residuals is ARIMA (2,0,0), represented as

$$\widehat{e}_t = a_0 + a_1 e_{t-1} + a_2 e_{t-2}, \tag{23}$$

where $\widehat{e}_t$ is the forecast for period t, e_{t-1} and e_{t-2} are the residuals at time lags $t-1$ and $t-2$, and a_0, a_1, a_2 are the coefficients to be estimated. Since the data varies around zero, the coefficient a_0 is not required. Therefore, the coefficients to be estimated are a_1 and a_2. PSO was applied to estimate the value of the coefficients. The ith particle is represented by the two-dimensional vectors, $X_i = (x_{i1}, x_{i2})$ and $V_i = (v_{i1}, v_{i2})$, where the first and second dimensions of the vectors refer to a_1 and a_2, respectively. After using the PSO, the coefficients were estimated and the ARIMA model is shown as

$$\widehat{e}_t = 0.3445 e_{t-1} - 0.1916 e_{t-2}. \tag{24}$$

Next, the ARIMA model obtained is used to forecast the residual from the SVR model. The forecast results from SVR and ARIMA models are combined, to represent forecast results of the proposed hybrid model.

6. Analysis of the Results

Table 2 and Figure 2 show the comparison of actual values and forecast values of property crime rates for test data set. PSOSVR_PSOARIMA, PSOSVR, and PSOSVR_ARIMA were found to predict closer to the actual value and have a similar pattern with the actual data. Meanwhile, ARIMA and PSOARIMA show unsatisfactory forecasting performance with the predicted values slightly higher than the actual value. Figure 3 shows box plot of forecast errors for testing data set. Based on the box plot, there is an outlier in the error of

Table 2: Comparison of actual value and forecast value of property crime rates.

Year	Actual value of property crime rates	Forecast value of property crime rates				
		Individual models			Hybrid models	
		ARIMA	PSOARIMA	PSOSVR	PSOSVR ARIMA	PSOSVR PSOARIMA
2005	3431.5	3450.674	3446.673	3432.788	3415.503	3427.515
2006	3334.5	3383.523	3378.434	3346.533	3345.65	3346.024
2007	3263.5	3277.728	3279.746	3255.626	3250.134	3251.728
2008	3211.5	3230.002	3225.111	3188.747	3200.334	3193.765
2009	3036.1	3187.335	3189.071	3030.249	3037.259	3036.579

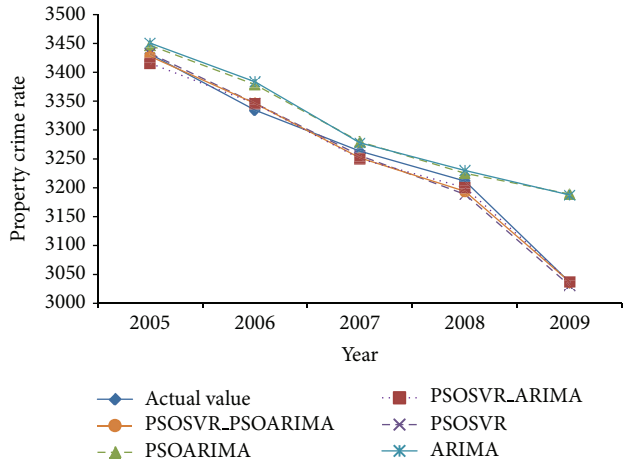


Figure 2: Forecasting of test data set.

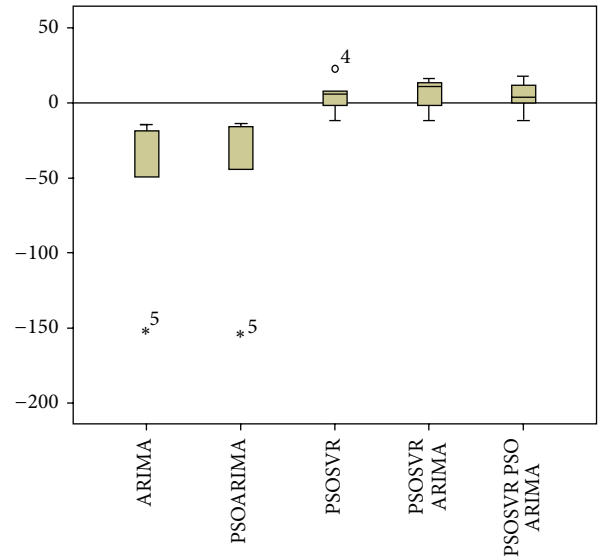


Figure 3: Forecasting Errors of test data set.

the PSOSVR model, while for ARIMA and PSOARIMA models, respectively, there is an extreme error value. The hybrid models, PSOSVR_PSOARIMA and PSOSVR_ARIMA, show better forecasting performance than the individual models, with no outliers or extreme error values. The proposed hybrid model was found to have the least forecast errors with a median closer to zero as compared to other forecasting models.

Table 3 shows the RMSE, MSE, MAPE, and MAD of the proposed hybrid model in comparison to the individual models, ARIMA, PSOARIMA, PSOSVR, and hybrid model, PSOSVR_ARIMA. The linear models, ARIMA and PSOARIMA, show poor performance with a relatively large error values in comparison to nonlinear model, PSOSVR and hybrid models. On the other hand, the nonlinear model, PSOSVR, shows a comparable performance to hybrid models with a slight difference in error values. The proposed hybrid model is found to have smaller errors than the individual models and PSOSVR_ARIMA. In addition, PSO is found to be able to improve the performance of the ARIMA model in the hybridization of SVR and ARIMA model. The results have shown that the use of PSO on ARIMA model is capable of improving the performance of the hybrid model PSOSVR_ARIMA. Therefore, PSOARIMA is suitable to be applied on small data set.

Table 4 shows the results of paired samples t-test. Paired samples t-test is used to compare the actual data with the

Table 3: Comparison of errors.

Model	RMSE	MSE	MAPE	MAD
ARIMA	72.371	5237.568	1.604	50.433
PSOARIMA	72.125	5201.967	1.544	48.387
PSOSVR	12.332	152.077	0.308	9.960
PSOSVR_ARIMA	11.704	136.980	0.319	10.568
PSOSVR_PSOARIMA	10.973	120.408	0.278	9.099

five forecasting models that have been developed, namely, ARIMA, PSOARIMA, PSOSVR, PSOSVR_ARIMA, and PSOSVR_PSOARIMA. The test is to ensure that there is no statistically significant difference of means between the actual data and the forecast values from the forecasting models. The results show that between the actual data and the forecast values from the forecasting models, the P value > 0.05 (0.124, 0.145, 0.463, 0.333, and 0.443). The difference between the upper and lower values for 95% interval is range between negatives and positives values. The results indicate that the hypothesis test fails to reject the null hypothesis. This implies that there is no statistically significant difference in the means between the actual data and the forecasting models. However, PSOSVR_PSOARIMA shows the smallest mean, standard deviation, and standard error mean as compared to other

Table 4: Paired samples test.

	Paired differences					t	df	Sig. (2-tailed)
	Mean	Std. deviation	Std. error mean	95% confidence interval of the difference				
				Lower	Upper			
Pair 1 PCR-ARIMA	−50.43259	58.03148	25.95247	−122.48819	21.62301	−1.943	4	0.124
Pair 2 PCR-PSOARIMA	−48.38690	59.74264	26.74264	−122.63637	25.86257	−1.809	4	0.145
Pair 3 PCR-PSOSVR	4.63141	5.71462	5.71462	−11.23491	20.49773	0.810	4	0.463
Pair 4 PCR-PSOSVR_ARIMA	5.64408	5.12651	5.12651	−8.58939	19.87756	1.101	4	0.333
Pair 5 PCR-PSOSVR_PSOARIMA	4.29777	5.04821	5.04821	−9.71831	18.31385	0.851	4	0.443

Legend: Pair 1: actual data and ARIMA; Pair 2: actual data and PSOARIMA; Pair 3: actual data and PSOSVR; Pair 4: actual data and PSOSVR_ARIMA; Pair 5: actual data and PSOSVR_PSOARIMA.

forecasting models. Therefore, the assumption can be made that PSOSVR_PSOARIMA is able to give better forecasting performance in comparison to other forecasting models.

Based on model evaluation that has been carried out, we can conclude that the individual models, ARIMA, PSOARIMA, and PSOSVR, are not sufficient to model the property crime rates. The hybrid models are more suitable to be employed as forecasting model for property crime rates, where the proposed hybrid model shows the best forecasting performance. These results indicate that there are linear and nonlinear components in property crime rates. Therefore, the use of linear and nonlinear models individually is not sufficient for modeling the property crime rates. However, the optimal parameters are very important for ensuring the accuracy of SVR models. The use of PSO has facilitated the searching process for the optimal parameters of SVR model, thus able to produce more accurate models. Results of this study also demonstrated that the use of PSO in estimating the parameters of ARIMA model was able to improve the accuracy of the ARIMA model for small data sets.

7. Conclusions

This paper proposes a time series model for crime rates forecasting. The proposed model is a hybrid model that combines the nonlinear model, SVR, with linear model, ARIMA. The proposed model was used to predict the property crime rates. The PSO is used to estimate the parameters of SVR and ARIMA models. Economic indicators are used as inputs to the proposed model. The economic indicators used are gross domestic product, unemployment rate, and consumer price index. The experimental results have found that the proposed model, PSOSVR_PSOARIMA, can produce smaller errors as compared to the individual models and hybrid model, PSOSVR_ARIMA. In conclusion, it can be concluded that the proposed hybrid model is an acceptable model to be applied in the crime rates forecasting.

Acknowledgments

This work is supported by Ministry of Higher Education (MOHE) under Research University Grant (03H72). Authors would like to thank Research Management Centre (RMC), Universiti Teknologi Malaysia, for the research activities and *Soft Computing Research Group* (SCRG) for the support and motivation in making this study a success.

References

[1] L. Qu, Y. Chen, and Z. Liu, "Time series forecasting model with error correction by structure adaptive RBF neural network," in *Proceedings of the 6th World Congress on Intelligent Control and Automation (WCICA '06)*, pp. 6831–6835, June 2006.

[2] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, 2003.

[3] D. K. Wedding and K. J. Cios, "Time series forecasting by combining RBF networks, certainty factors, and the Box-Jenkins model," *Neurocomputing*, vol. 10, no. 2, pp. 149–168, 1996.

[4] J. V. Hansen and R. D. Nelson, "Time-series analysis with neural networks and ARIMA-neural network hybrids," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 15, no. 3, pp. 315–330, 2003.

[5] I. Rojas, O. Valenzuela, F. Rojas et al., "Soft-computing techniques and ARMA model for time series prediction," *Neurocomputing*, vol. 71, no. 4–6, pp. 519–537, 2008.

[6] D. Zeng, J. Xu, J. Gu, L. Liu, and G. Xu, "Short term traffic flow prediction using hybrid ARIMA and ANN models," in *Proceedings of the Workshop on Power Electronics and Intelligent Transportation System (PEITS '08)*, pp. 621–625, August 2008.

[7] R. Sallehuddin, S. M. Shamsuddin, and S. Z. M. Hashim, "Hybridization model of linear and nonlinear time series data for forecasting," in *Proceedings of the 2nd Asia International Conference on Modelling and Simulation (AMS '08)*, pp. 597–602, May 2008.

[8] L. A. Díaz-Robles, J. C. Ortega, J. S. Fu et al., "A hybrid ARIMA and artificial neural networks model to forecast particulate matter in urban areas: the case of Temuco, Chile," *Atmospheric Environment*, vol. 42, no. 35, pp. 8331–8340, 2008.

[9] Y. Xiaojian and Z. Jiaping, "A comparison of hybrid ARMA-Elman models with single models for forecasting interest rates," in *Proceedings of the 2nd International Symposium on Intelligent Information Technology Application (IITA '08)*, pp. 985–989, December 2008.

[10] R. Sallehuddin and S. M. H. Shamsuddin, "Hybrid grey relational artificial neural network and auto regressive integrated moving average model for forecasting time-series data," *Applied Artificial Intelligence*, vol. 23, no. 5, pp. 443–486, 2009.

[11] R. Sallehuddin, *Hybridization of nonlinear and linear model for time series forecasting [Ph.D. thesis]*, Universiti Teknologi Malaysia, Johor Bahru, Malaysia, 2010.

[12] C. H. Aladag, E. Egrioglu, and C. Kadilar, "Forecasting nonlinear time series with a hybrid methodology," *Applied Mathematics Letters*, vol. 22, no. 9, pp. 1467–1470, 2009.

[13] D. Ömer Faruk, "A hybrid neural network and ARIMA model for water quality time series prediction," *Engineering Applications of Artificial Intelligence*, vol. 23, no. 4, pp. 586–594, 2010.

[14] M. Shafie-Khah, M. P. Moghaddam, and M. K. Sheikh-El-Eslami, "Price forecasting of day-ahead electricity markets using a hybrid forecast method," *Energy Conversion and Management*, vol. 52, no. 5, pp. 2165–2169, 2011.

[15] R. Askari Moghadam and M. Keshmirpour, "Hybrid ARIMA and neural network model for measurement estimation in energy-efficient wireless sensor networks," in *Proceedings of the International Conference on Informatics Engineering and Information Science(ICIEIS '11)*, pp. 35–48, 2011.

[16] M. Khashei and M. Bijari, "A novel hybridization of artificial neural networks and ARIMA models for time series forecasting," *Applied Soft Computing Journal*, vol. 11, no. 2, pp. 2664–2675, 2011.

[17] P. F. Pai and C. S. Lin, "A hybrid ARIMA and support vector machines model in stock price forecasting," *Omega*, vol. 33, no. 6, pp. 497–505, 2005.

[18] H. Yujun, Z. Youchan, and D. Dongxing, "Research on hybrid ARIMA and support vector machine model in short term load forecasting," in *Proceedings of the 6th International Conference on Intelligent Systems Design and Applications (ISDA '06)*, pp. 804–808, October 2006.

[19] K. Feng and W. Xiaojuan, "Time series forecasting model with error correction by structure adaptive support vector machine," in *Proceedings of the International Conference on Computer Science and Software Engineering (CSSE '08)*, pp. 1067–1070, December 2008.

[20] D. Y. Zhang, H. W. Song, and P. Chen, "Stock market forecasting model based on a hybrid ARMA and support vector machines," in *Proceeding of 15th Annual International Conference on Management Science and Engineering (ICMSE '08)*, pp. 1312–1317, September 2008.

[21] L. Xiang, G. J. Tang, and C. Zhang, "Simulation of time series prediction based on hybrid support vector regression," in *Proceedings of the 4th International Conference on Natural Computation (ICNC '08)*, pp. 167–171, October 2008.

[22] J. Che and J. Wang, "Short-term electricity prices forecasting based on support vector regression and Auto-regressive integrated moving average modeling," *Energy Conversion and Management*, vol. 51, no. 10, pp. 1911–1917, 2010.

[23] J. H. Lo, "A study of applying ARIMA and SVM model to software reliability prediction," in *Proceedings of the International Conference on Uncertainty Reasoning and Knowledge Engineering (URKE '11)*, pp. 141–144, 2011.

[24] K. Y. Chen and C. H. Wang, "A hybrid SARIMA and support vector machines in forecasting the production values of the machinery industry in Taiwan," *Expert Systems with Applications*, vol. 32, no. 1, pp. 254–264, 2007.

[25] Y. Bao, D. Yi, T. Xiong, Z. Hu, and S. Zheng, "A comparative study on hybrid linear and nonlinear modeling framework for air passenger traffic forecasting," *Advances in Information Sciences and Service Sciences*, vol. 3, no. 5, pp. 243–254, 2011.

[26] F. Pan, H. Zhang, and M. Xia, "A hybrid time-series forecasting model using extreme learning machines," in *Proceeding of the 2nd International Conference on Intelligent Computing Technology and Automation (ICICTA '09)*, pp. 933–936, October 2009.

[27] K. K. Lai, L. Yu, S. Wang, and W. Huang, "Hybridizing exponential smoothing and neural network for financial time series predication," in *Proceedings of the 6th International Conference on Computational Science (ICCS '06)*, pp. 493–500, 2006.

[28] Y. S. Lee and L. I. Tong, "Forecasting time series using a methodology based on autoregressive integrated moving average and genetic programming," *Knowledge-Based Systems*, vol. 24, no. 1, pp. 66–72, 2011.

[29] J. J. Wang, J. Z. Wang, Z. G. Zhang, and S. P. Guo, "Stock index forecasting based on a hybrid model," *Omega*, vol. 40, no. 6, pp. 758–766, 2012.

[30] M. Khashei, A. Zeinal Hamadani, and M. Bijari, "A novel hybrid classification model of artificial neural networks and multiple linear regression models," *Expert Systems With Applications*, vol. 39, no. 3, pp. 2606–2620, 2012.

[31] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, NY, USA, 1995.

[32] Z. Ding, "Application of support vector machine regression in stock price forecasting," in *Proceeding of the International Conference on Business, Economics, and Financial Sciences, Management (BEFM '11)*, pp. 359–365, 2012.

[33] J. Wu and L. Jin, "Daily rainfall prediction with SVR using a novel hybrid PSO-SA algorithms," in *Proceeding of the 2nd International Conference on High-Performance Networking, Computing and Communications Systems(ICHCC '11)*, pp. 508–515, 2011.

[34] R. Liao, H. Zheng, S. Grzybowski, and L. Yang, "Particle swarm optimization-least squares support vector regression based forecasting model on dissolved gases in oil-filled power transformers," *Electric Power Systems Research*, vol. 81, no. 12, pp. 2074–2080, 2011.

[35] Y. S. Lee and L. I. Tong, "Forecasting time series using a methodology based on autoregressive integrated moving average and genetic programming," *Knowledge-Based Systems*, vol. 24, no. 1, pp. 66–72, 2011.

[36] S. Asadi, A. Tavakoli, and S. R. Hejazi, "A new hybrid for improvement of auto-regressive integrated moving average models applying particle swarm optimization," *Expert Systems with Applications*, vol. 39, no. 5, pp. 5332–5337, 2012.

[37] R. Rosenfeld and R. Fornango, "The impact of economic conditions on robbery and property crime: the role of consumer sentiment," *Criminology*, vol. 45, no. 4, pp. 735–769, 2007.

[38] D. T. Altindag, "Crime and unemployment: evidence from Europe," *International Review of Law and Economics*, vol. 32, no. 1, pp. 145–157, 2012.

[39] D. Wu and Z. Wu, "Crime, inequality and unemployment in England and Wales," *Applied Economics*, vol. 44, no. 29, pp. 3765–3775, 2012.

[40] D. L. Yearwood and G. Koinis, "Revisiting property crime and economic conditions: an exploratory study to identify predictive indicators beyond unemployment rates," *Social Science Journal*, vol. 48, no. 1, pp. 145–158, 2011.

[41] P. K. Narayan and R. Smyth, "Crime rates, male youth unemployment and real income in Australia: evidence from Granger causality tests," *Applied Economics*, vol. 36, no. 18, pp. 2079–2095, 2004.

[42] K. Edmark, "Unemployment and crime: is there a connection?" *Scandinavian Journal of Economics*, vol. 107, no. 2, pp. 353–373, 2005.

[43] C. L. Britt, "Reconsidering the unemployment and crime relationship: variation by age group and historical period," *Journal of Quantitative Criminology*, vol. 13, no. 4, pp. 405–428, 1997.

[44] F. Carmichael and R. Ward, "Male unemployment and crime in England and Wales," *Economics Letters*, vol. 73, no. 1, pp. 111–115, 2001.

[45] C. F. Tang and H. H. Lean, "Will inflation increase crime rate? New evidence from bounds and modified wald tests," *Global Crime*, vol. 8, no. 4, pp. 311–323, 2007.

[46] C. F. Tang, "The linkages among inflation, unemployment and crime rates in Malaysia," *International Journal of Economics and Management*, vol. 3, no. 1, pp. 50–61, 2009.

[47] P. Fajnzylber, D. Lederman, and N. Loayza, "What causes violent crime?" *European Economic Review*, vol. 46, no. 7, pp. 1323–1357, 2002.

[48] C. J. Lu, C. H. Chang, C. Y. Chen, C. C. Chiu, and T. S. Lee, "Stock index prediction: a comparison of MARS, BPN and SVR in an emerging market," in *Proceeding of the IEEE International Conference on Industrial Engineering and Engineering Management (IEEM '09)*, pp. 2343–2347, December 2009.

[49] I. D. Lins, M. D. C. Moura, E. Zio, and E. L. Droguett, "A particle swarm-optimized support vector machine for reliability prediction," *Quality and Reliability Engineering International*, vol. 28, no. 2, pp. 141–158, 2012.

[50] Y. Hu, C. Wu, and H. Liu, "Prediction of passenger flow on the highway based on the least square support vector machine," *Transport*, vol. 26, no. 2, pp. 197–203, 2011.

[51] J. Wu and E. Chen, "A novel hybrid particle swarm optimization for feature selection and kernel optimization in support vector regression," in *Proceedings of the International Conference on Computational Intelligence and Security (CIS '10)*, pp. 189–194, December 2010.

[52] N. Xin, X. Gu, H. Wu, Y. Hu, and Z. Yang, "Application of genetic algorithm-support vector regression (GA-SVR) for quantitative analysis of herbal medicines," *Journal of Chemometrics*, vol. 26, no. 7, pp. 353–360, 2012.

[53] S. Zhao and L. Wang, "Support vector regression based on particle swarm optimization for rainfall forecasting," in *Proceeding of the 3rd International Joint Conference on Computational Sciences and Optimization (CSO '10)*, pp. 484–487, May 2010.

[54] J. E. Hanke and D. W. Wichern, *Business Forecasting*, Prentice Hall, Englewood Cliffs, NJ, USA, 2009.

[55] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.

[56] H. Wang, H. Sun, C. Li et al., "Diversity enhanced particle swarm optimization with neighborhood search," *Information Sciences*, vol. 223, pp. 119–135, 2013.

[57] G. He and N. J. Huang, "A modified particle swarm optimization algorithm with applications," *Applied Mathematics and Computation*, vol. 219, no. 3, pp. 1053–1060, 2012.

[58] X. Wang, J. Yang, X. Teng, W. Xia, and R. Jensen, "Feature selection based on rough sets and particle swarm optimization," *Pattern Recognition Letters*, vol. 28, no. 4, pp. 459–471, 2007.

Hindawi Publishing Corporation
The Scientific World Journal
Volume 2013, Article ID 510763, 19 pages
http://dx.doi.org/10.1155/2013/510763

Research Article

Improving Vector Evaluated Particle Swarm Optimisation by Incorporating Nondominated Solutions

Kian Sheng Lim,[1] Zuwairie Ibrahim,[2] Salinda Buyamin,[1] Anita Ahmad,[1] Faradila Naim,[2] Kamarul Hawari Ghazali,[2] and Norrima Mokhtar[3]

[1] *Faculty of Electrical Engineering, Universiti Teknologi Malaysia, 81310 Johor Bahru, Malaysia*
[2] *Faculty of Electrical and Electronic Engineering, Universiti Malaysia Pahang, 26600 Pekan, Malaysia*
[3] *Department of Electrical Engineering, Faculty of Engineering, University of Malaya, 50603 Kuala Lumpur, Malaysia*

Correspondence should be addressed to Zuwairie Ibrahim; zuwairie@ump.edu.my

Received 16 January 2013; Accepted 11 March 2013

Academic Editors: P. Agarwal, V. Bhatnagar, and Y. Zhang

The Vector Evaluated Particle Swarm Optimisation algorithm is widely used to solve multiobjective optimisation problems. This algorithm optimises one objective using a swarm of particles where their movements are guided by the best solution found by another swarm. However, the best solution of a swarm is only updated when a newly generated solution has better fitness than the best solution at the objective function optimised by that swarm, yielding poor solutions for the multiobjective optimisation problems. Thus, an improved Vector Evaluated Particle Swarm Optimisation algorithm is introduced by incorporating the nondominated solutions as the guidance for a swarm rather than using the best solution from another swarm. In this paper, the performance of improved Vector Evaluated Particle Swarm Optimisation algorithm is investigated using performance measures such as the number of nondominated solutions found, the generational distance, the spread, and the hypervolume. The results suggest that the improved Vector Evaluated Particle Swarm Optimisation algorithm has impressive performance compared with the conventional Vector Evaluated Particle Swarm Optimisation algorithm.

1. Introduction

In multiobjective optimisation (MOO) problems, multiple objective functions are solved simultaneously by either minimising or maximising the fitness of the functions. These multiple objective functions usually conflict with each other. Therefore, the solution to an MOO problem is a set of multiple tradeoffs, or nondominated solutions, rather than a single solution.

The Vector Evaluated Particle Swarm Optimisation (VEPSO) [1] algorithm introduced by Parsopoulos and Vrahatis has been used to solve various MOO problems, such as the design of radiometer array antennas [2], the design of supersonic ejectors for hydrogen fuel cells [3], the design of composite structures [4], the design of steady-state performance for power systems [5], and the design of multiple machine-scheduling systems [6]. In the VEPSO algorithm, one swarm of particles optimises an objective function using guidance from the best solution found by another swarm.

The nondominated solutions found during the optimisation are usually preferred for effective guidance [7]. As an example, the multiobjective PSO (MOPSO) algorithm [8, 9] divides all nondominated solutions into several groups based on their locations in the objective space. Then, one of the nondominated solutions is randomly selected from the group that has the fewest solutions to be used as the particle guide. Furthermore, the nondominated sorting PSO (NSPSO) algorithm [10] uses the primary mechanism of nondominated sorting genetic algorithm-II [11], in which one nondominated solution is randomly selected to be used as the guide for the particles based on the niche count and nearest-neighbour density estimator. In addition, the optimised MOPSO (OMOPSO) algorithm [12] by Margarita Reyes-Sierra and Carlos Coello Coello uses the crowding

distance mechanism for binary tournaments to select one of the nondominated solutions as the guide for each particle. Abido [13] uses two nondominated solutions, a local set and a global set, to optimise the problem. Each particle is guided by the nondominated solution that has smallest distance between the particle and both nondominated solution sets.

The conventional VEPSO algorithm solves an MOO problem by improving the solutions in a swarm under the guidance of the best solution with respect to a single objective, found by another swarm. However, the nondominated solution which has better fitness with respect to the other objectives may exist, but it was not used to guide the particles in other swarm. The nondominated solutions are always equal or better solutions compared with the best solution used in conventional VEPSO. The superiority of the nondominated solutions motivates the use of nondominated solutions as particle guides for each swarm in improving the VEPSO algorithm. Thus, in this study, the guidance of a swarm is selected by the nondominated solution which has best fitness with respect to a single objective function, optimised by the other swarm.

The paper is organized as follows. In Section 2, we explain some information on MOO problem. Then, in Section 3, we explain the particle swarm optimisation (PSO), the conventional VEPSO, and the improved VEPSO algorithms. In the next section, we demonstrate the simulation experiment which includes several performance measures and benchmark test problem, before we discuss the results. Lastly, we present the conclusion and include some suggestion for future work.

2. Multiobjective Optimisation

Consider a minimisation of a multiobjective problem:

$$\begin{aligned} &\text{minimise the fitness function,} \\ &\vec{F}(\vec{x}) = \left\{ f_m(\vec{x}) \in \mathfrak{R}^M,\ m = 1, 2, \ldots, M \right\} \\ &\text{subject to } g_j(\vec{x}) \le 0, \quad j = 1, 2, \ldots, J, \\ &h_k(\vec{x}) = 0, \quad k = 1, 2, \ldots, K, \end{aligned} \tag{1}$$

where $\vec{x} = \{x_n \in \mathfrak{R}^N,\ n = 1, 2, \ldots, N\}$ is the N-dimensional vector of decision variables that represent the possible solutions, M is the number of objectives, $f_m \in \mathfrak{R}^M$ is the objective function, and $\{g_j, h_k\} \in \mathfrak{R}$ are the inequality and equality constraint functions, respectively.

In explaining the concept of Pareto optimality, consider two vectors $\{\overrightarrow{F^a}, \overrightarrow{F^b}\} \in \mathfrak{R}^M$. $\overrightarrow{F^a}$ *dominates* $\overrightarrow{F^b}$ (denote as $\overrightarrow{F^a} \prec \overrightarrow{F^b}$) if and only if $f_m^a \le f_m^b$ for $m = 1, 2, \ldots, M$ and $f_m^a < f_m^b$ at least once. The dominance relations $\overrightarrow{F^a} \prec \overrightarrow{F^b}$ and $\overrightarrow{F^a} \prec \overrightarrow{F^c}$ for a two-objective problem are indicated by the labelled circles in Figure 1. Hence, a vector of decision variables $\overrightarrow{x^a}$ is a *nondominated solution* if and only if there is no other solution $\overrightarrow{x^b}$ such that $\vec{F}(\overrightarrow{x^a}) \prec \vec{F}(\overrightarrow{x^b})$. The nondominated solution is also known as the Pareto optimal solution. The set of nondominated solutions of an MOO problem is known as the *Pareto Optimal set*, $\mathscr{P}$. The set of objective vectors with respect to $\mathscr{P}$ is known as the *Pareto Front*, $\mathscr{PF} = \{\vec{F}(\vec{x}) \in \mathfrak{R}^M \mid \vec{x} \in \mathscr{P}\}$. The $\mathscr{PF}$ for a two-objective problem is illustrated by the black circles in Figure 1.

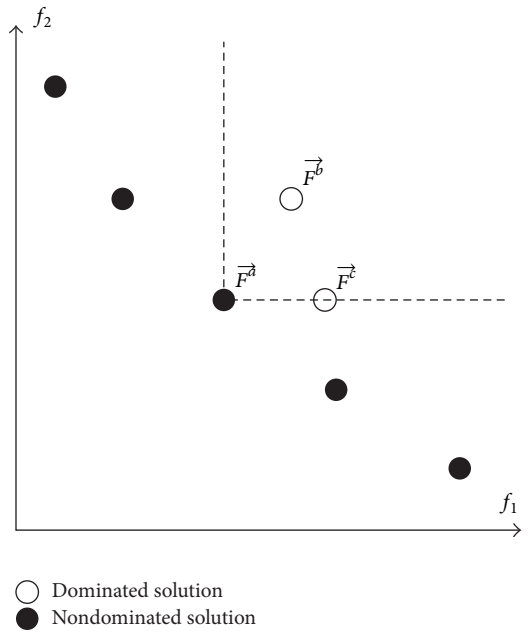


Figure 1: Dominance relation for two objectives problem.

The goal of an MOO algorithm is to find as many nondominated solutions as possible according to the objective functions and constraints. The Pareto front corresponding to the nondominated set should be as close to and well distributed over the true Pareto front as possible. However, it is possible to have different solutions that map to the same fitness value in objective space.

3. Particle Swarm Optimisation

3.1. Original Particle Swarm Optimisation Algorithm. Based on the social behaviour of birds flocking and fish schooling, a population-based stochastic optimisation algorithm named Particle Swarm Optimisation (PSO) was introduced by Kennedy et al. [14, 15]. The PSO algorithm contains individuals referred to as particles that encode the possible solutions to the optimisation problem using their positions. These particles explore the defined search space to look for solutions that better satisfy the objective function of the optimised problem. Each particle collaborates with the others during the search process by comparing its current position with the best position that it and the other particles in the swarm have found [16].

Figure 2 shows the flow chart of the PSO algorithm. For the PSO algorithm, consider the following minimisation problem: there are I-particles flying around in an N-dimensional search space, where their positions, p_n^i (i =

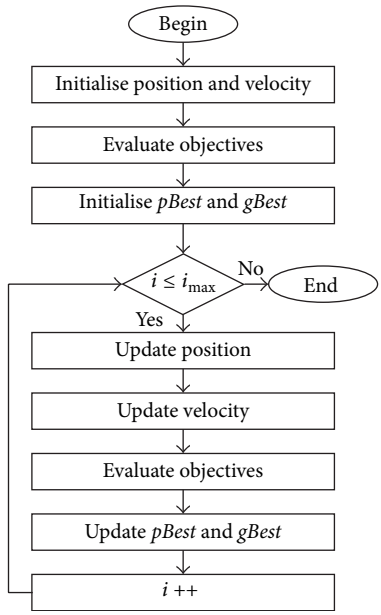


Figure 2: The PSO algorithm.

$1, 2, \ldots, I$; $n = 1, 2, \ldots, N$), represent the possible solutions. Initially, all particles are randomly positioned in the search space and assigned random velocities, $v_n^i(t)$. Then, the objective fitness, $\overrightarrow{F^i}(t)$, for each particle is evaluated by calculating the objective functions with respect to $p^i(t)$. Next, each particle's best position, $pBest^i(t)$, is initialised to its current position. Meanwhile, the best among all $pBest^i(t)$ is set as the swarm's best position, $gBest(t)$, as specified in (2), where S is the swarm of particles:

$$gBest = \left\{pBest^i \in S \mid f\left(pBest^i\right) = \min f\left(\forall pBest^i \in S\right)\right\}. \tag{2}$$

Next, the algorithm iterates until the stopping condition is met; that is, either the maximum number of iterations is exceeded or the minimum error is attained. In each iteration, each particle's velocity and position are updated using (3) and (4), respectively,

$$v_n^i(t+1) = \chi\left[\omega v_n^i(t) + c_1 r_1\left(pBest_n^i - p_n^i(t)\right) + c_2 r_2\left(gBest_n - p_n^i(t)\right)\right], \tag{3}$$

$$p_n^i(t+1) = p_n^i(t) + v_n^i(t+1), \tag{4}$$

where χ is the constriction factor and ω is the inertia weight. c_1 and c_2 are the cognitive and social coefficients, respectively. Meanwhile, r_1 and r_2 are both random values between zero and one. After the velocity and position are updated, the $\overrightarrow{F^i}(t)$ for each particle is evaluated again. Later, $pBest^i(t)$ is updated with the more optimal between the new position of the ith particle or $pBest^i(t)$. Then, the $gBest(t)$ is updated with the most optimal $pBest^i(t)$ among all the particles, as given in (2). Finally, when the stopping condition is met, $gBest(t)$ represents the optimum solution found for the problem optimised using this algorithm.

3.2. Vector Evaluated Particle Swarm Optimisation Algorithm. Parsopóulos and Vrahatis [1] introduced the VEPSO algorithm, which was inspired by the multiswarm concept of the VEGA algorithm [17]. In this multiswarm concept, each objective function is optimised by a swarm of particles using the $gBest(t)$ from another swarm. The $gBest(t)$ for the mth swarm is the $pBest^i(t)$ that has most optimal fitness with respect to the mth objective, among all $pBest^i(t)$ from the mth swarm, as given below:

$$gBest^m = \left\{pBest^i \in S^m \mid f_m\left(pBest^i\right) = \min f_m\left(\forall pBest^i \in S^m\right)\right\}. \tag{5}$$

Generally, the PSO and VEPSO algorithms have similar process flows, except that all processes are repeated for M swarms when optimising problems with M objective functions. Because each swarm optimises using $gBest(t)$ from another swarm, in VEPSO, the velocity is updated using (6). The velocity equation for particles in the mth swarm updates $gBest^k(t)$, where k is given in (7):

$$v_n^m i(t+1) = \chi\left[\omega v_n^{mi}(t) + c_1 r_1\left(pBest_n^{mi} - p_n^{mi}(t)\right) + c_2 r_2\left(gBest_n^k - p_n^{mi}(t)\right)\right], \tag{6}$$

$$k = \begin{cases} M, & m = 1, \\ m-1, & \text{otherwise.} \end{cases} \tag{7}$$

In addition to the difference in the velocity equation, all nondominated solutions found during the optimisation are stored in an archive each time after the objective functions are evaluated. To ensure that the archive contains nondominated solutions only, the fitness $\overrightarrow{F^i}(t)$ of each particle is compared, based on the Pareto optimality criterion, to those of all particles before it is compared to the nondominated solutions in the archive. All nondominated solutions in the archive represent possible solutions to the MOO problem.

3.3. The Improved VEPSO Algorithm. In conventional VEPSO, each particle of a swarm is updated by the $gBest(t)$ from the other swarm that is optimal with respect to the objective function optimised by the other swarm. Consider a two-objective optimisation problem as an example; the $gBest(t)$ of the first swarm is only updated when a newly generated solution has better fitness with respect to the first objective, as specified in (5). Thus, $gBest(t)$ is not updated even if the new solution, nondominated solution, has equal fitness with respect to the first objective and better fitness with respect to the second objective. Hence, as in Figure 3(a), each particle from the second swarm moves under the guidance of the $gBest^1(t)$ but not the better, nondominated solutions.

However, this limitation can be overcome by updating $gBest(t)$ with a new solution, nondominated solution, that

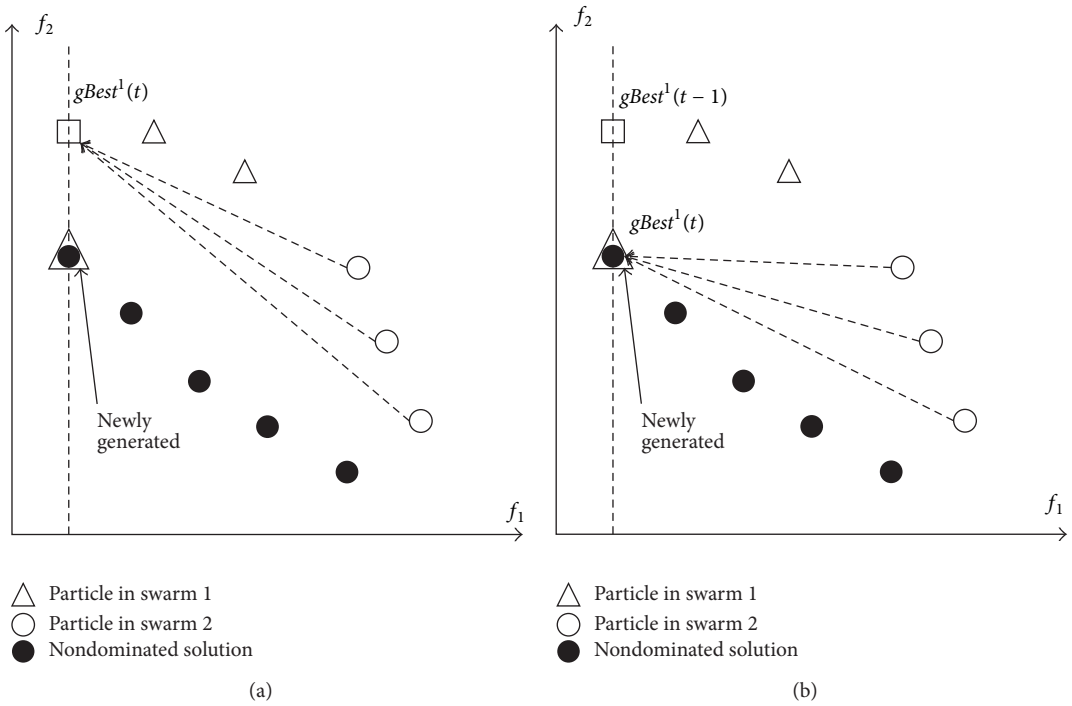


Figure 3: Particles guided by (a) the best solution from the other swarm and (b) a nondominated solution.

has equal fitness with respect to the optimised objective function and better fitness with respect to the other objective. This improved VEPSO algorithm is represented in Figure 3(b), where $gBest^1(t)$ is now a nondominated solution that is best with respect to the first objective function. Thus, each particle from the second swarm will be guided by its own $pBest^2i(t)$ and $gBest^1(t)$, which is a nondominated solution, with the hope that the particle will converge toward the Pareto front faster.

In the improved VEPSO algorithm, the generality of conventional VEPSO is not lost; so the $gBest(t)$ of a swarm is the best nondominated solution with respect to the objective function optimised by the swarm. Therefore, the $gBest(t)$ of the mth swarm is given as following:

$$gBest^m = \{X \in \mathscr{P} \mid f_m(X) = \min f_m \ (\forall X \in \mathscr{P})\}, \quad (8)$$

where X is a nondominated solution and $\mathscr{P}$ is the set of nondominated solutions in the archive. For a two-objective-function problem, the particles from the second swarm are guided by the nondominated solution that is best with respect to the first objective function. Meanwhile, the particles of the first swarm are guided by the nondominated solution that is optimal with respect to the second objective function. Thus, this improved algorithm is called Vector Evaluated Particle Swarm Optimisation incorporate nondominated solutions (VEPSOnds).

In addition, the PSO algorithm has the natural limitation that particles tend to become stuck in locally optimal solutions [18, 19]. Therefore, this improved VEPSO algorithm also includes the polynomial mutation mechanism from nondominated sorting genetic algorithm-II [11]. The polynomial mutation mechanism modifies the particle position with a certain probability such that the particle can mutate out from the locally optimal solution and continue the search for a globally optimal solution. In this work, one of every ten particles is mutated in the improved VEPSO algorithm.

4. Experiment

4.1. Performance Measure. In order to analyse the performance of the VEPSOml algorithm, several quantitative performance mesasures are used. Since MOO problems have different features, for example multilocal optima solution, which could trap the particles from obtaining more nondominated solutions; hence, the number of solution (NS) measure is used to quantify the total number of nondominated solutions found at the end of the computation. Besides, for example when the particles one trapped in a local optima solution, the obtained Pareto front will not be converged close to the true Pareto front which means that the best possible solutions were not found yet. Thus, the generational distance (GD) [20] is used and defined as the average Euclidean distance between the obtained Pareto front, $\mathscr{PF}_o$, and the true Pareto front, $\mathscr{PF}_t$, using (9). A smaller GD value indicates better performance:

$$\mathrm{GD} = \frac{\left(\sum_{q=1}^{\|\mathscr{PF}_o\|} d_q^M\right)^{1/M}}{\|\mathscr{PF}_o\|},$$
$$d_q = \min_{1 \le k \le \|\mathscr{PF}_t\|} \sqrt{\sum_{j=1}^{M} \left(f_j^q - f_j^k\right)^2}. \quad (9)$$

A well-converged Pareto front does not guarantee to have good diversity of nondominated solutions along the Pareto

front. Therefore, the third performance metric used is the spread (SP) [11], which is used to measure the extent of the distribution of the $\mathcal{PF}_o$ along the $\mathcal{PF}_t$. Equations (10), are used to measure SP, and smaller values indicate better performance:

$$\mathrm{SP} = \frac{d_f + d_l + \sum_{q=1}^{\|\mathcal{PF}_o\|-1} \left| d_q - \overline{d} \right|}{d_f + d_l + \left(\|\mathcal{PF}_o\| - 1 \right) \overline{d}},$$
$$\overline{d} = \frac{\sum_{q=1}^{\|\mathcal{PF}_o\|-1} d_q}{\|\mathcal{PF}_o\| - 1}, \tag{10}$$
$$d_q = \sqrt{\left(f_1^q - f_1^{q+1} \right)^2 + \left(f_2^q - f_2^{q+1} \right)^2},$$

where d_f is the Euclidean distance between the first extreme members in $\mathcal{PF}_o$ and $\mathcal{PF}_t$ and d_l is the Euclidean distance between the last extreme members in $\mathcal{PF}_o$ and $\mathcal{PF}_t$. In some cases, the obtained Pareto fronts could be converged well to the true Pareto front but it has poor diversity performance. Hence, it is not fair by comparing different algorithms with the GD and SP measures only. Finally, the hypervolume (HV) [21] is used to measure the total space or area enclosed by the $\mathcal{PF}_o$ and a reference point, R, which is a vector constructed from the worst objective value from the $\mathcal{PF}_t$. Equation (11) is used to evaluate the HV value. The total area for HV is the enclosed area in Figure 4 and is calculated using (11). Larger HV values represent better performance:

$$\mathrm{HV} = \sum_{q=1}^{\|\mathcal{PF}_o\|} v_q, \tag{11}$$

where v_q is the space or area between R and the diagonal corner of qth solution of $\mathcal{PF}_o$.

4.2. Test Problems. Five of the benchmark test problems from ZDT [22] are used to evaluate the performance of the algorithm. Because this study focused on continuous search space problems, the ZDT5 problem is not used as it is for the evaluation of binary problems. All benchmark problems are set up using the parameter values recommended in the paper [22]. For evaluating the performance measure, the true Pareto front for each problem is obtained from the standard database generated by the jMetal (http://jmetal.sourceforge.net/problems.html).

4.3. Evaluation of VEPSO Algorithms. Because the VEPSOnds algorithm includes polynomial mutation, the experiment in this work should analyse a version of VEPSOnds that does not include polynomial mutation. This implementation exists because the polynomial mutation affects the algorithm's performance, and it is necessary to determine whether the change in performance is due to the use of multiple nondominated solutions or the polynomial mutation. Thus, in this work, the VEPSOnds algorithm without mutation is denoted as VEPSOnds1 and the VEPSOnds algorithm with mutation is denoted as VEPSOnds2.

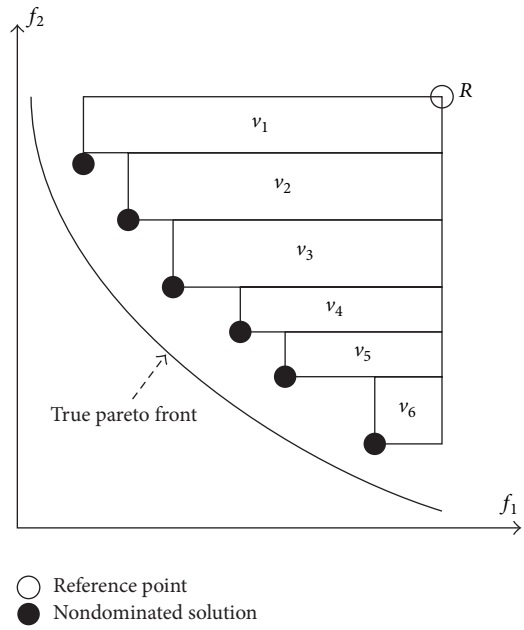


Figure 4: Hypervolume measure with area covered by the nondominated solutions and a reference point.

In this experiment, the total number of particles is fixed to 100 and divided equally among all swarms. The archive size is controlled by removing the nondominated solutions with the smallest crowding distance [11]. In addition, the maximum iteration and archive size are set to 250 and 100, respectively. During the computation, the inertia weight is linearly degraded from 1.0 to 0.4. The cognitive and social constants are both random values between 1.5 and 2.5. Moreover, the distribution index is set to 0.5 for the mutation operation. Each test problem is simulated for 100 runs to enable statistical analysis.

The performance of each algorithm tested on the ZDT1 problem is presented in Table 1. For the average NS measure, the number of nondominated solutions found by both improved VEPSO algorithms was significantly greater for conventional VEPSO. For the GD measure, VEPSOnds1 demonstrated significant improvement compared with the conventional VEPSO algorithm. Meanwhile, the VEPSOnds2 algorithm exhibited an extremely large improvement compared with both conventional VEPSO and VEPSOnds1. Similarly, the SP measures for both improved VEPSO algorithms also indicated significant improvement compared with conventional VEPSO. As expected, the HV performance also improved dramatically when the problem was optimised using multiple nondominated solutions as particle guides.

For better visual comparison, the Pareto fronts with the best GD value returned for each test problem are shown in Figure 5 through Figure 9. Figure 5 shows the plot of nondominated solutions with the best GD measure returned for the ZDT1 problem. The nondominated solutions obtained by VEPSO are clearly located very far away from the true Pareto front, which leads to a large GD value. Moreover, the obtained solutions are unevenly distributed around the

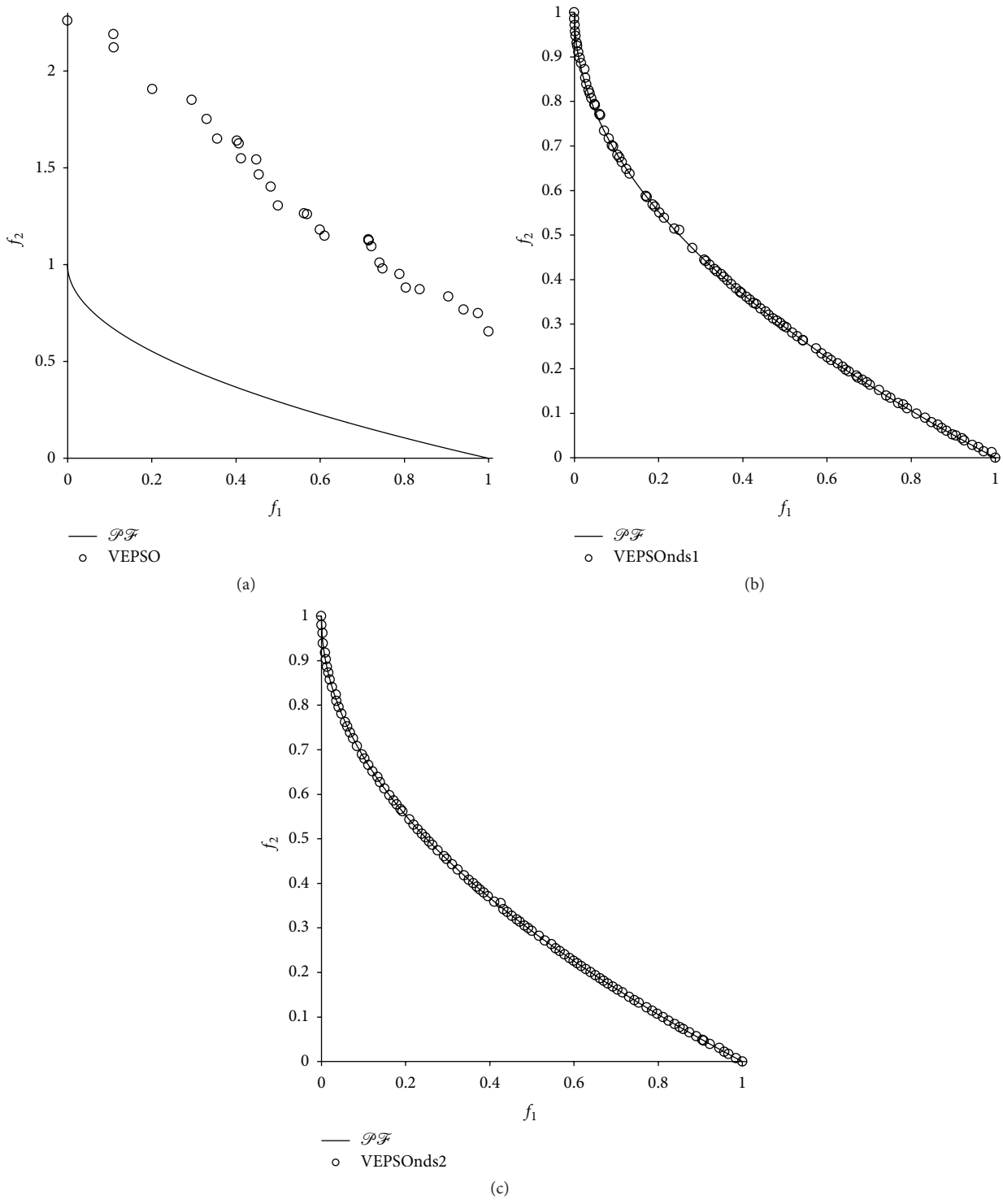


Figure 5: Plot of nondominated solutions returned by each algorithm for the ZDT1 test problem.

objective space, which yields a large SP value. In contrast, the VEPSOnds1 and VEPSOnds2 algorithms generated nondominated solutions close to and evenly distributed over the true the Pareto front. Therefore, the GD and SP values for both improved VEPSO algorithms are significantly smaller than those for conventional VEPSO. However, the VEPSOnds2 has better distribution of nondominated solutions than the VEPSOnds1.

Table 2 presents the performance measures for all algorithms tested on the ZDT2 problem. Again, both the

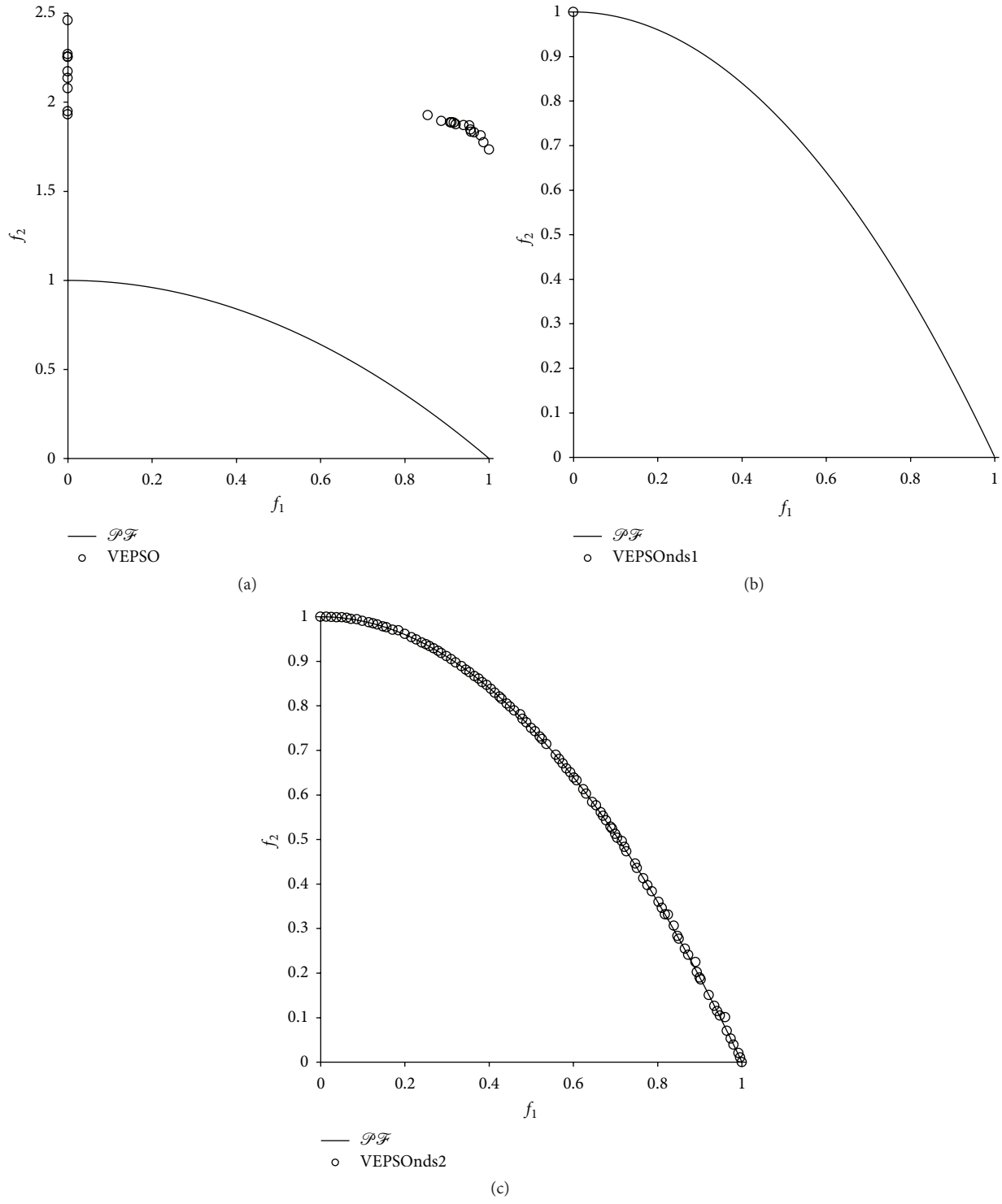


Figure 6: Plot of nondominated solutions returned by each algorithm for the ZDT2 test problem.

improved VEPSO algorithms dramatically improved the ability to obtain a large number of solutions compared with VEPSO, especially VPESOnds2. In addition to the NS performance, the GD and SP performances were also dramatically improved because the nondominated solutions used in the improved VEPSO algorithms are better guides compared with the best solution among each particle, which is used in conventional VEPSO. However, in SP measure, the VEPSOnds1 shows negligible improvement, whereas the VEPSOnds2 shows distinguished improvement over the conventional VEPSO. The conventional VEPSO algorithm was unable to yield a meaningful HV because the obtained

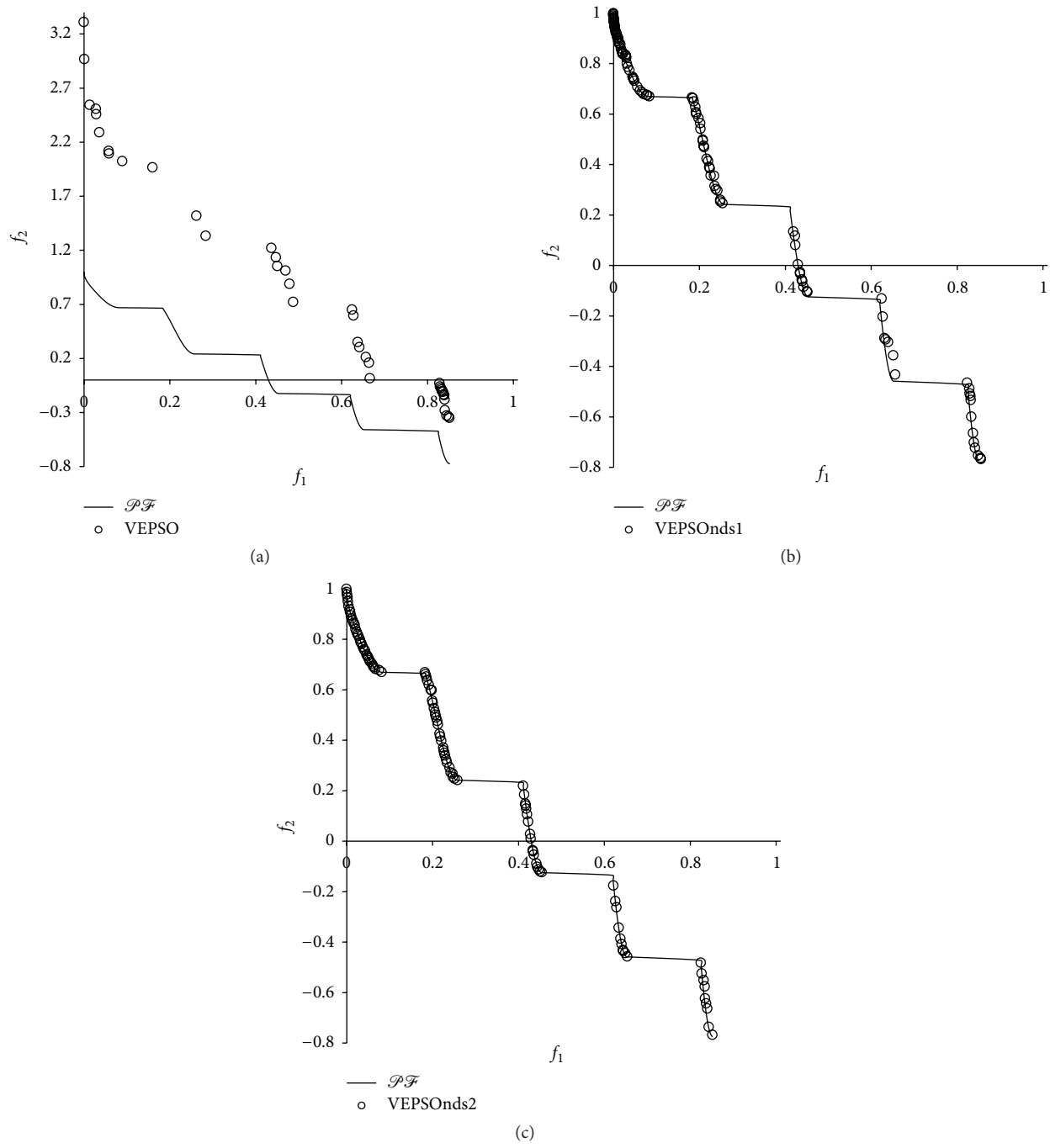


Figure 7: Plot of nondominated solutions returned by each algorithm for the ZDT3 test problem.

nondominated solutions were far worse than the true Pareto front. However, the VEPSOnds1 and VEPSOnds2 algorithms yielded good HV values.

Figure 6 shows the nondominated solutions with the best GD measure returned for the ZDT2 problem. The poor performance of conventional VEPSO is visible because the nondominated solutions found are very distant from the true Pareto front and distributed unevenly in the objective space. Conversely, the VEPSOnds1 algorithm was able to obtain a nondominated solution that is located on the true Pareto front. However, there is only one nondominated solution, which increases the SP value of this algorithm. In contrast, the VEPSOnds2 algorithm successfully found nondominated solutions very close to the true Pareto front, and the nondominated solutions found are distributed evenly over the true Pareto front. Thus, the polynomial mutation preventing

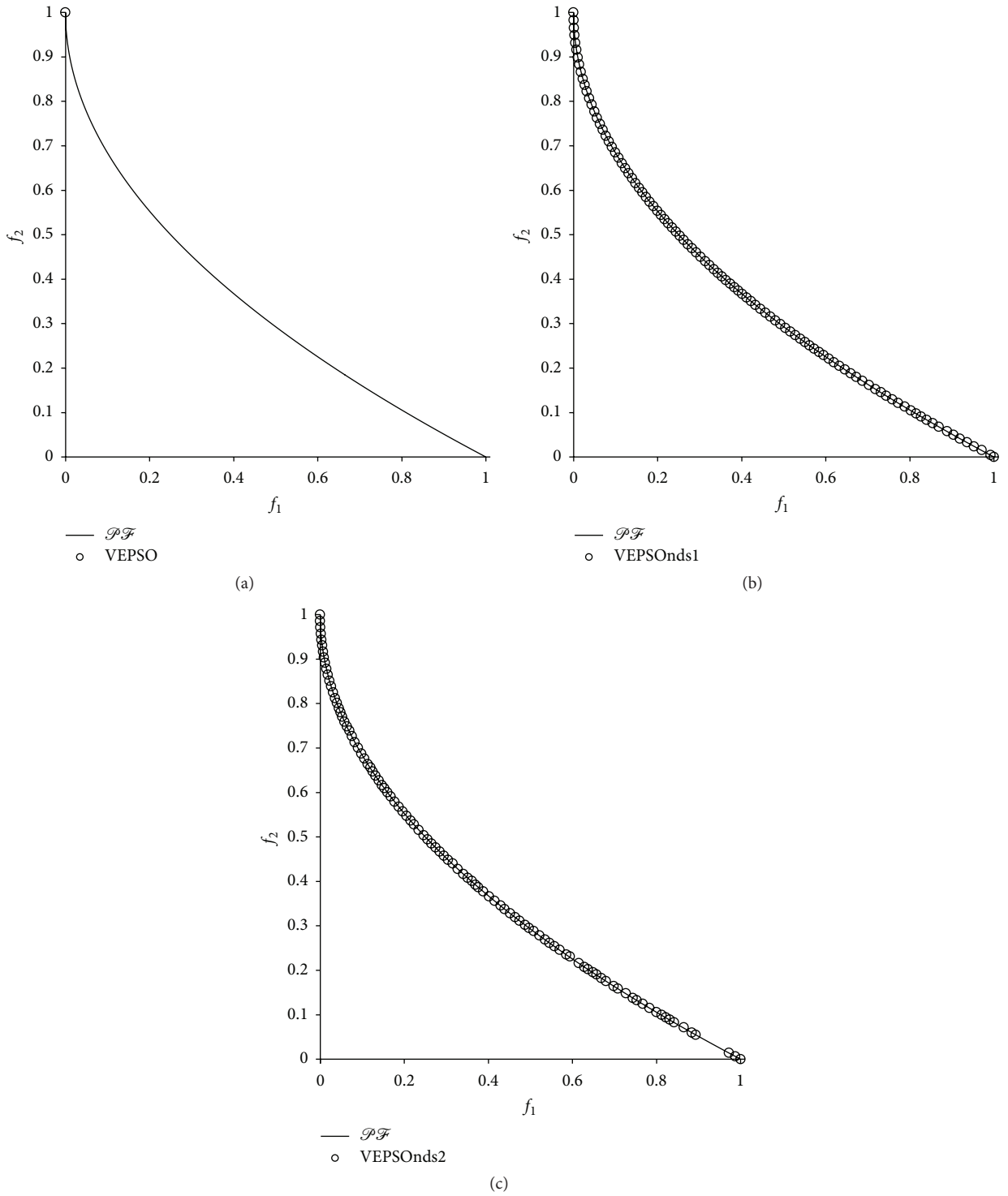


Figure 8: Plot of nondominated solutions returned by each algorithm for the ZDT4 test problem.

the particles from converging too early is an important mechanism in improving the diversity performance of the algorithm.

Table 3 presents the performance measures of all algorithms tested for the ZDT3 problem. Regarding the NS measure, both the improved VEPSO algorithms successfully obtained a large number of nondominated solutions. Moreover, both improved VEPSO algorithms yielded great improvement compared with the conventional VEPSO in terms of convergence. However, the SP value of the solutions

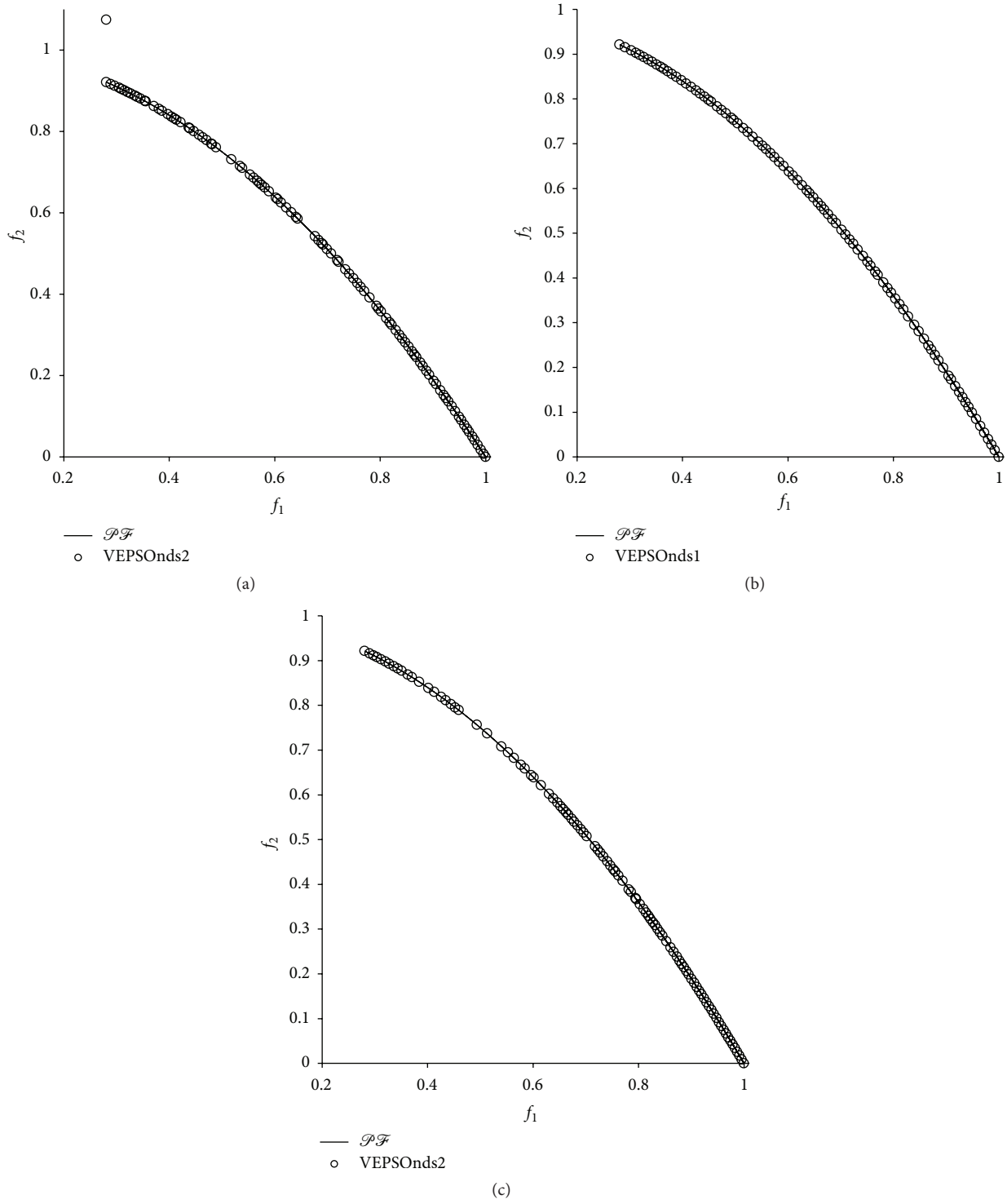


Figure 9: Plot of nondominated solutions returned by each algorithm for the ZDT6 test problem.

obtained by both improved VEPSO was degraded in this test. Even with the degradation in the diversity performance of both improved VEPSO, they still hold the performance advantages with their superior convergence improvement. Besides, both improved VEPSO performances are better as their HV value was also improved when the particles in the algorithm used additional guides during the optimisation.

Figure 7 shows the nondominated solutions with the best GD measure returned for ZDT3 problem. Unavoidably, the nondominated solutions obtained by the conventional

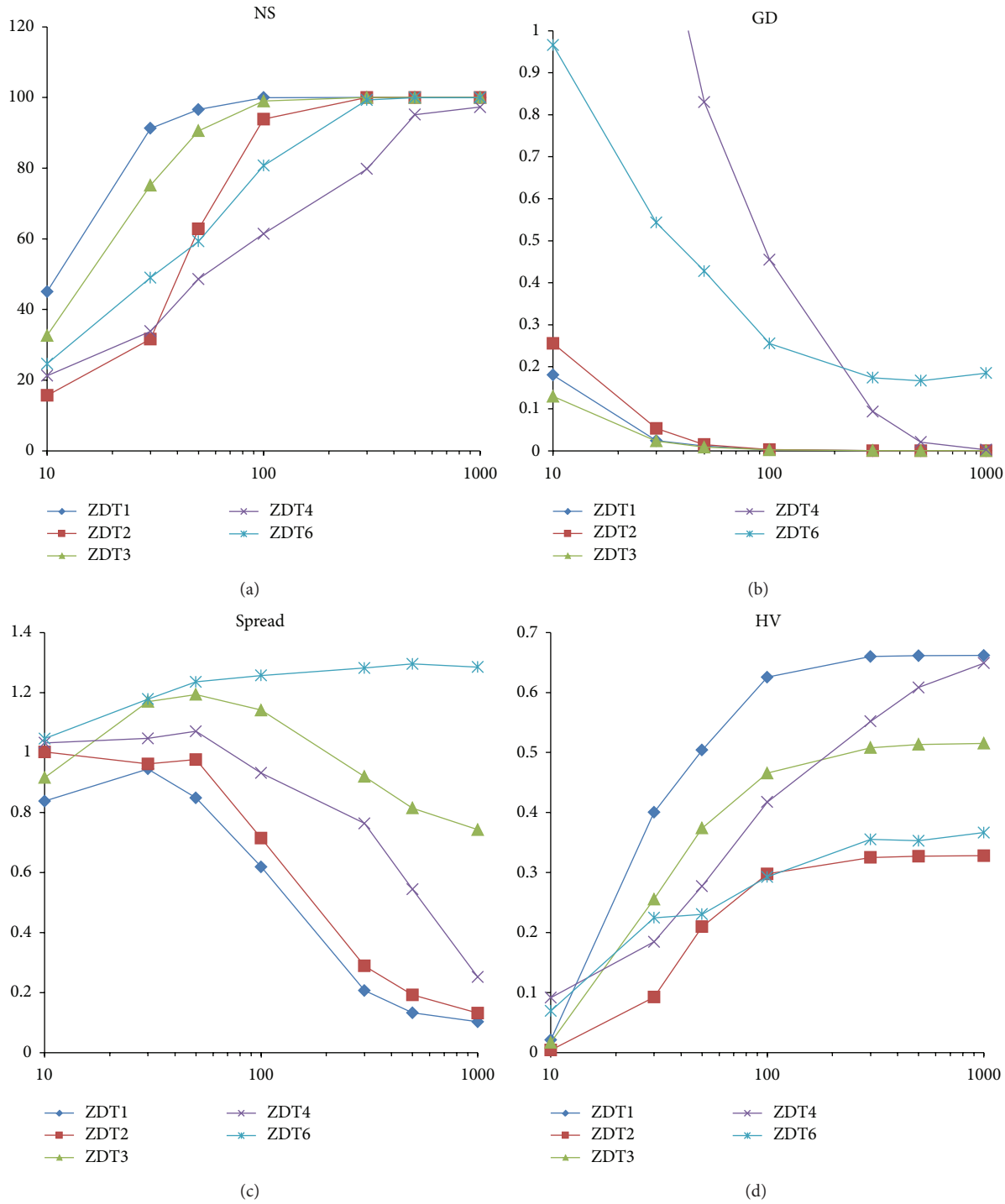


Figure 10: Plots of the performance metrics for various numbers of particles. (a) Number of solutions. (b) Generational distance. (c) Spread. (d) Hypervolume.

VEPSO algorithm are scattered far from the true Pareto front, which leads to poor performance. Conversely, both the improved VEPSO algorithms were able to obtain nondominated solutions that cover the true Pareto front almost perfectly. Hence, both the improved VEPSO algorithms exhibited almost equal improvement, but VEPSOnds1 has weaker diversity performance as there are lesser solutions at the middle of the Pareto front.

Table 4 presents the performance measures for all algorithms tested for the ZDT4 problem. The average number of nondominated solutions found by the conventional VEPSO algorithm is relatively low compared with VEPSOnds1, which found most of the solutions. The conventional VEPSO algorithm had great difficulty escaping from the multiple local optima, which resulted in a very large GD value. However, the improved VEPSO algorithms, in which particles are guided

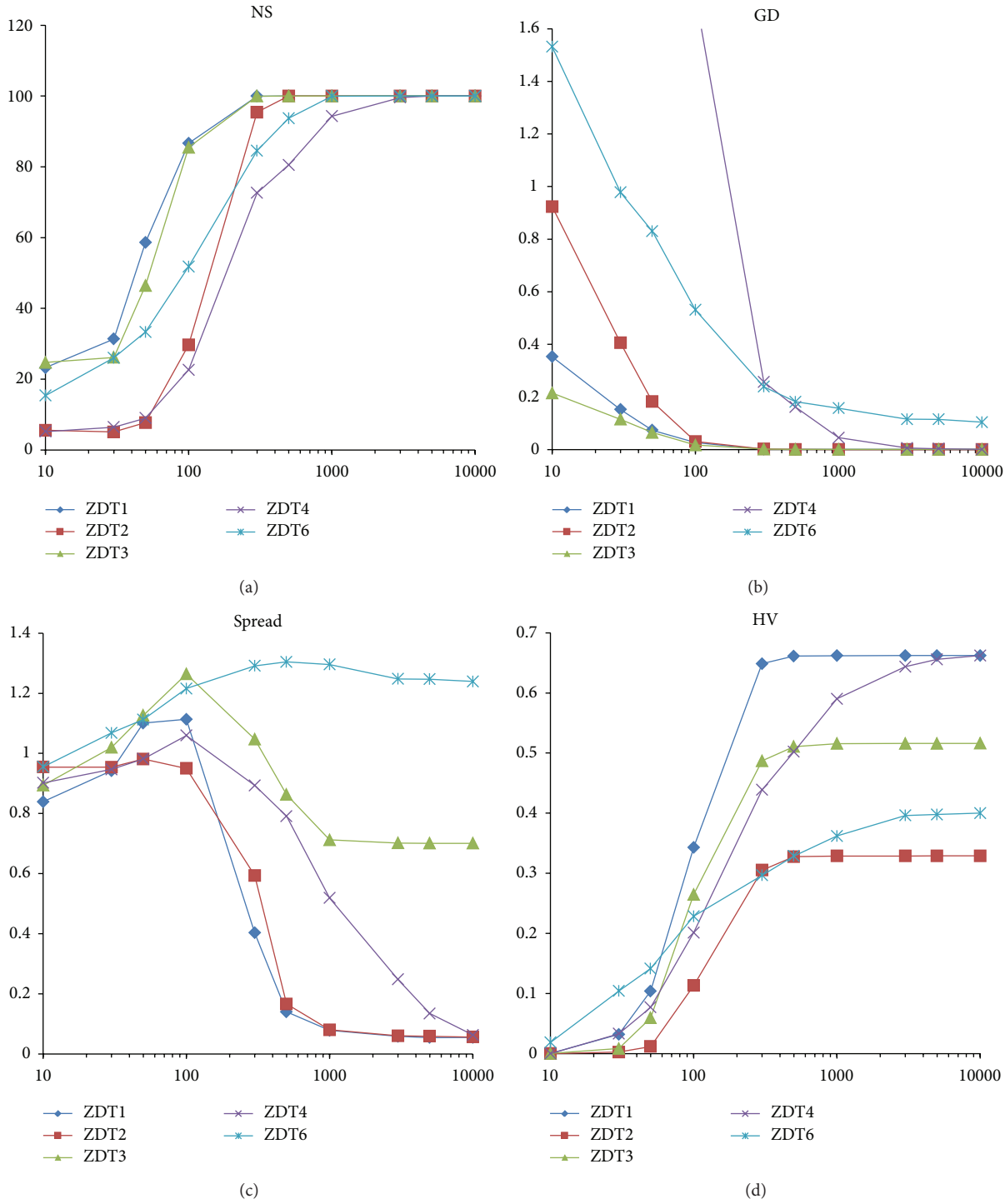


Figure 11: Plots of the performance metrics for various numbers of iterations. (a) Number of solution. (b) Generational distance. (c) Spread. (d) Hypervolume.

by the nondominated solutions, had less chance of being stuck in local optima. Meanwhile, the HV value yielded by the conventional VEPSO algorithm is relatively small compared with that of the improved VEPSO algorithms. Thus, the smaller SP value for conventional VEPSO does not mean it has better performance, as both improved VEPSO still maintain performance advantages with their better GD and HV values.

Figure 8 shows the nondominated solutions with the best GD measure returned for the ZDT4 problem. The conventional VEPSO algorithm, in which particles follow only one guide, was easily stuck in local optima, as shown

Table 1: Algorithm performance tested on ZDT1 problem.

Measure	VEPSO	VEPSOnds1	VEPSOnds2
NS			
Ave.	30.220000	100.000000	99.790000
SD	5.697031	0.000000	1.458483
Min.	16.000000	100.000000	86.000000
Max.	44.000000	100.000000	100.000000
GD			
Ave.	0.295865	0.022637	0.002194
SD	0.051645	0.014201	0.003505
Min.	0.139491	0.000283	0.000169
Max.	0.432478	0.073477	0.019113
SP			
Ave.	0.834481	0.729350	0.571807
SD	0.039111	0.160298	0.248304
Min.	0.705367	0.322322	0.168144
Max.	0.917087	1.219625	1.127141
HV			
Ave.	0.001886	0.428153	0.631216
SD	0.010058	0.113432	0.046091
Min.	—	0.185313	0.438793
Max.	0.087426	0.659603	0.661363

Table 2: Algorithm performance tested on ZDT2 problem.

Measure	VEPSO	VEPSOnds1	VEPSOnds2
NS			
Ave.	8.070000	38.120000	97.490000
SD	6.356822	25.747131	7.832198
Min.	1.000000	1.000000	49.000000
Max.	24.000000	100.000000	100.000000
GD			
Ave.	0.766956	0.039653	0.002003
SD	0.324444	0.063791	0.003483
Min.	0.240509	0.000000	0.000198
Max.	1.679803	0.310345	0.017750
SP			
Ave.	0.944524	0.947356	0.687560
SD	0.065266	0.111963	0.278814
Min.	0.797757	0.695715	0.242474
Max.	1.080351	1.278655	1.460767
HV			
Ave.	—	0.137784	0.296372
SD	—	0.117596	0.053300
Min.	—	—	0.043514
Max.	—	0.311075	0.327309

Table 3: Algorithm performance tested on ZDT3 problem.

Measure	VEPSO	VEPSOnds1	VEPSOnds2
NS			
Ave.	35.150000	99.600000	99.400000
SD	6.853997	3.405284	6.000000
Min.	21.000000	66.000000	40.000000
Max.	53.000000	100.000000	100.000000
GD			
Ave.	0.173060	0.009607	0.002040
SD	0.031253	0.008293	0.002268
Min.	0.079595	0.000433	0.000223
Max.	0.276801	0.039481	0.013231
SP			
Ave.	0.871146	1.109448	1.121149
SD	0.043319	0.086041	0.099980
Min.	0.701884	0.902861	0.858725
Max.	1.001428	1.322024	1.362217
HV			
Ave.	0.004722	0.373133	0.471686
SD	0.021699	0.083015	0.038568
Min.	—	0.112859	0.332399
Max.	0.167359	0.506222	0.514600

Table 4: Algorithm performance tested on ZDT4 problem.

Measure	VEPSO	VEPSOnds1	VEPSOnds2
NS			
Ave.	6.610000	95.250000	64.220000
SD	3.920665	16.518967	38.860949
Min.	1.000000	15.000000	4.000000
Max.	21.000000	100.000000	100.000000
GD			
Ave.	5.062543	0.383646	0.349438
SD	3.167428	0.478535	0.431632
Min.	0.000000	0.000155	0.000165
Max.	13.350278	2.049212	1.923652
SP			
Ave.	0.858655	1.035510	0.962023
SD	0.147255	0.347336	0.367664
Min.	0.483073	0.077112	0.144160
Max.	1.236461	1.419225	1.435101
HV			
Ave.	0.228824	0.399914	0.437755
SD	0.188151	0.159971	0.155761
Min.	—	—	—
Max.	0.573978	0.661941	0.660821

Table 5: Algorithm performance tested on ZDT6 problem.

Measure	VEPSO	VEPSOnds1	VEPSOnds2
NS			
Ave.	76.590000	78.040000	81.030000
SD	32.884891	26.684055	25.075021
Min.	11.000000	22.000000	24.000000
Max.	100.000000	100.000000	100.000000
GD			
Ave.	0.338537	0.260666	0.266259
SD	0.370336	0.158592	0.168404
Min.	0.001746	0.044137	0.035520
Max.	1.552521	0.709692	0.735990
SP			
Ave.	1.201796	1.276529	1.286909
SD	0.146782	0.083293	0.075052
Min.	0.492064	0.987981	1.067748
Max.	1.435395	1.437289	1.410091
HV			
Ave.	0.304584	0.303381	0.281256
SD	0.134813	0.102216	0.119017
Min.	—	0.038143	0.026496
Max.	0.400964	0.400780	0.401005

in the first plot. Thus, the algorithm was able to find only one nondominated solution. However, both the improved VEPSO algorithms, in which additional guides are used, had less difficulty in obtaining a greater number of diverse nondominated solutions.

Table 5 presents the performance measures for all algorithms tested on the ZDT6 problem. Interestingly, all algorithms found approximately the same number of nondominated solutions. Moreover, the SP and HV values for all algorithms are also similar. However, noticeably, both improved VEPSO have outperformed the conventional VEPSO in terms of convergence performance.

Figure 9 shows the nondominated solutions with the best GD measure returned for the ZDT6 problem. As predicted, the plots of nondominated solutions are similar because all algorithms exhibit similar results in terms of convergence and diversity. However, the nondominated solutions for the VEPSOnds2 algorithm were not well distributed over the true Pareto front, middle of the Pareto front in this case, which caused the algorithm to have the largest SP value, as shown in Table 5.

For all test problems, the improved VEPSO algorithms exhibited significant improvement compared with the conventional VEPSO algorithm for most of the performance measures. The performance improvements occurred because the nondominated solutions always provide a better solution than a solution that optimises only a single-objective function. Using a better solution as the leader increases the quality of the result.

4.4. Analysis of the Number of Particles. The performance of the VEPSOnds2 algorithm with various numbers of particles is analysed in this experiment. Most of the parameters are the same as in the previous experiment, except that the particles are equally divided between swarms for a total of 10, 30, 50, 100, 300, 500, and 1000 particles. The performance measurements, taken for each total number of particles and for each benchmark problem, are plotted in Figure 10.

In short, the performance of VEPSOnds2 improves when the number of particles is increased. When VEPSOnds2 is computed with 250 iterations, the algorithm performs well at 300 particles, which is equivalent to 75 000 function evaluations. With a higher number of particles, the algorithm exhibits even better results, but the computational time increases dramatically.

4.5. Analysis of the Number of Iterations. The effect of various numbers of iterations on VEPSOnds2 performance is investigated in this experiment. In this experiment, the number of iterations becomes 10, 30, 50, 100, 300, 500, 1000, 3000, 5000, or 10 000. All parameters are set as in the previous experiments, and the number of particles is set to 100, which is divided equally between swarms. The plot of performance metrics for the various numbers of iterations for each benchmark problem is displayed in Figure 11.

When the number of iterations is increased, the performance of VEPSOnds2 improves. The VEPSOnds2 algorithm performs consistently and acceptably with 100 particles when there are 300 iterations or 30 000 function evaluations. Computation of the algorithm with a higher number of iterations, such as 3000 particles or 300 000 function evaluations, could result in a better performance but is only recommended if a powerful computing platform is used.

4.6. Benchmarking with the State-of-the-Art Multiobjective Optimisation Algorithms. The VEPSOnds2 algorithm performed better than the other algorithms in most test cases. Thus, the performance of this algorithm is compared to four other MOO algorithms which are nondominated sorting genetic algorithm-II (NSGA-II) [11], strength pareto evolutionary algorithm 2 (SPEA2) [23], archive-based hYbrid scatter search (AbYSS) [24], and speed-constrained multiobjective PSO (SMPSO) [25]. For a fair comparison, all algorithms compute 25 000 function evaluations with the archive size set to 100. The NSGA-II, commonly used for performing comparisons, was set to use a population size of 100 for optimisation. This algorithm was set to use the Simulated Binary Crossover (SBX) operator with the crossover probability $p_c = 0.9$ and polynomial mutation [26] operators with the mutation probability $p_m = 1/N$. The distribution index for both operators was set to $\mu_n = \mu_m = 20$. The SPEA2 was set to use the same parameters as in NSGA-II. The AbYSS was set to use a population size of 20. The pairwise combination parameters in AbYSS were set to $\text{RefSet}_1 = 10$ and $\text{RefSet}_2 = 10$. The polynomial mutation parameters were set to similar values as those in NSGA-II and SPEA2. In SMPSO, the population size and maximum iteration were set to 100 and 250, respectively. The

Table 6: Performance comparison based on ZDT1 test problem.

Measure	AbYSS	NSGA-II	SPEA2	SMPSO	VEPSOnds2
NS					
Ave.	100.000000	100.000000	100.000000	100.000000	99.790000
SD	0.000000	0.000000	0.000000	0.000000	1.458483
Min.	100.000000	100.000000	100.000000	100.000000	86.000000
Max.	100.000000	100.000000	100.000000	100.000000	100.000000
GD					
Ave.	0.000185	0.000223	0.000220	0.000117	0.002194
SD	0.000035	0.000038	0.000028	0.000031	0.003505
Min.	0.000125	0.000146	0.000154	0.000053	0.000169
Max.	0.000343	0.000374	0.000400	0.000172	0.019113
SP					
Ave.	0.105387	0.379129	0.148572	0.076608	0.571807
SD	0.012509	0.028973	0.012461	0.009200	0.248304
Min.	0.080690	0.282485	0.116765	0.056009	0.168144
Max.	0.136747	0.441002	0.174986	0.099653	1.127141
HV					
Ave.	0.661366	0.659333	0.659999	0.661801	0.631216
SD	0.000269	0.000301	0.000301	0.000100	0.046091
Min.	0.660267	0.658486	0.659347	0.661372	0.438793
Max.	0.661724	0.659909	0.660629	0.661991	0.661363

Table 7: Performance comparison based on ZDT2 test problem.

Measure	AbYSS	NSGA-II	SPEA2	SMPSO	VEPSOnds2
NS					
Ave.	100.000000	100.000000	100.000000	100.000000	97.490000
SD	0.000000	0.000000	0.000000	0.000000	7.832198
Min.	100.000000	100.000000	100.000000	100.000000	49.000000
Max.	100.000000	100.000000	100.000000	100.000000	100.000000
GD					
Ave.	0.000131	0.000176	0.000182	0.000051	0.002003
SD	0.000067	0.000066	0.000039	0.000003	0.003483
Min.	0.000056	0.000093	0.000090	0.000044	0.000198
Max.	0.000433	0.000707	0.000304	0.000060	0.017750
SP					
Ave.	0.130425	0.378029	0.158187	0.071698	0.687560
SD	0.090712	0.028949	0.027529	0.013981	0.278814
Min.	0.080831	0.311225	0.118114	0.035786	0.242474
Max.	0.833933	0.430516	0.365650	0.106749	1.460767
HV					
Ave.	0.325483	0.326117	0.326252	0.328576	0.296372
SD	0.023209	0.000297	0.000908	0.000077	0.053300
Min.	0.096409	0.325278	0.318785	0.328349	0.043514
Max.	0.328505	0.326696	0.327559	0.328736	0.327309

Table 8: Performance comparison based on ZDT3 test problem.

Measure	AbYSS	NSGA-II	SPEA2	SMPSO	VEPSOnds2
NS					
Ave.	100.000000	100.000000	100.000000	99.900000	99.400000
SD	0.000000	0.000000	0.000000	0.904534	6.000000
Min.	100.000000	100.000000	100.000000	91.000000	40.000000
Max.	100.000000	100.000000	100.000000	100.00000	100.000000
GD					
Ave.	0.000193	0.000211	0.000230	0.000203	0.002040
SD	0.000019	0.000013	0.000019	0.000061	0.002268
Min.	0.000144	0.000180	0.000184	0.000155	0.000223
Max.	0.000264	0.000268	0.000327	0.000717	0.013231
SP					
Ave.	0.707651	0.747853	0.711165	0.717493	1.121149
SD	0.013739	0.015736	0.008840	0.032822	0.099980
Min.	0.696859	0.715199	0.698590	0.697943	0.858725
Max.	0.796404	0.793183	0.775317	0.950901	1.362217
HV					
Ave.	0.512386	0.514813	0.513996	0.514996	0.471686
SD	0.011314	0.000159	0.000675	0.001737	0.038568
Min.	0.463776	0.514449	0.510764	0.500484	0.332399
Max.	0.515960	0.515185	0.514668	0.515818	0.514600

Table 9: Performance comparison based on ZDT4 test problem.

Measure	AbYSS	NSGA-II	SPEA2	SMPSO	VEPSOnds2
NS					
Ave.	99.680000	100.000000	100.000000	100.000000	64.220000
SD	3.100603	0.000000	0.000000	0.000000	38.860949
Min.	69.000000	100.000000	100.000000	100.000000	4.000000
Max.	100.000000	100.000000	100.000000	100.000000	100.000000
GD					
Ave.	0.001231	0.000486	0.000923	0.0001347	0.349438
SD	0.002632	0.000235	0.001428	0.000027	0.431632
Min.	0.000148	0.000163	0.000176	0.000070	0.000165
Max.	0.014472	0.001374	0.012292	0.000187	1.923652
SP					
Ave.	0.159842	0.392885	0.298269	0.092281	0.962023
SD	0.120180	0.037083	0.125809	0.011777	0.367664
Min.	0.078244	0.324860	0.137934	0.067379	0.144160
Max.	1.073669	0.473358	0.884091	0.124253	1.435101
HV					
Ave.	0.646058	0.654655	0.645336	0.661401	0.437755
SD	0.034449	0.003406	0.018773	0.000162	0.155761
Min.	0.472299	0.642177	0.505799	0.660934	0.000000
Max.	0.661594	0.659710	0.658784	0.661726	0.660821

Table 10: Performance comparison based on ZDT6 test problem.

Measure	AbYSS	NSGA-II	SPEA2	SMPSO	VEPSOnds2
NS					
Ave.	100.000000	100.000000	100.000000	100.000000	81.030000
SD	0.000000	0.000000	0.000000	0.000000	25.075021
Min.	100.000000	100.000000	100.000000	100.000000	24.000000
Max.	100.000000	100.000000	100.000000	100.000000	100.000000
GD					
Ave.	0.000549	0.001034	0.001761	0.012853	0.266259
SD	0.000015	0.000102	0.000192	0.024813	0.168404
Min.	0.000510	0.000804	0.001267	0.000502	0.035520
Max.	0.000596	0.001360	0.002207	0.092434	0.735990
SP					
Ave.	0.097740	0.357160	0.226433	0.390481	1.286909
SD	0.013129	0.031711	0.020658	0.497140	0.075052
Min.	0.070455	0.282201	0.179482	0.042666	1.067748
Max.	0.130389	0.441311	0.292897	1.377582	1.410091
HV					
Ave.	0.400346	0.388304	0.378377	0.401280	0.281256
SD	0.000172	0.001604	0.002714	0.000076	0.119017
Min.	0.399821	0.383637	0.371907	0.401081	0.026496
Max.	0.400842	0.392123	0.385626	0.401402	0.401005

terms $r_1 = r_2 = \text{random}[0.1, 0.5]$, and the terms $c_1 = c_2 = \text{random}[1.5, 2.0]$. This algorithm was also set to use polynomial mutation [27] with $p_m = 1/N$ and $\mu_m = 20$.

Table 6 lists the performance of the algorithms on the ZDT1 test problem. The number of solutions found by the VEPSOnds2 is comparable to the other algorithms. However, the average GD value of the VEPSOnds2 is at least 10 times greater than that of the others even though its minimum GD value is close to that of the other algorithms. VEPSOnds2 also has the highest average SP value, but its minimum SP is better than that of NSGA-II. The HV value for VEPSOnds2 is similar to that of the other algorithms.

Table 7 lists the performance of the algorithms on the ZDT2 test problem. VEPSOnds2 was able to obtain a reasonable number of solutions compared to the other algorithms. However, the GD value for VEPSOnds2 is the highest among all algorithms. Additionally, VEPSOnds2 has the greatest average SP value, even though its minimum SP value is better than that of NSGA-II. In the HV measure, the average value returned by VEPSOnds2 is relatively close to the other algorithms and even outperforms the NSGA-II with its maximum value.

Table 8 lists the performance of the algorithms on the ZDT3 test problem. SMPSO and VEPSOnds2 both show poor performance with respect to the maximum number of solutions for all runs. Again, VEPSOnds2 has a 10 times greater GD value compared to the other algorithms. Interestingly, the diversity performance of VEPSOnds2 is very poor, as the average SP value is higher than 1.0. However, the maximum HV value of VEPSOnds2 was not the smallest, and its average is almost as large as the rest.

Table 9 lists the performance of the algorithms on the ZDT4 test problem. The multiple local optima featured in this problem challenged VEPSOnds2 greatly, as the number of solutions obtained is very low. In addition, the convergence and diversity performances were very poor, as the GD and SP values are both very large compared to the other algorithms. The HV value was also poor, as the multiple local optima feature is well known as a natural weakness in PSO-based algorithms [18, 19].

Finally, Table 10 lists the performances of the algorithms on the ZDT6 test problem. On average, VEPSOnds2 does not obtain the highest number of nondominated solutions, but the number is still in an acceptable range. However, the GD value for VEPSOnds2 was too far from the other algorithms. In addition, the SP value for VEPSOnds2 is extremely large compared to the other algorithms, and the average HV value for VEPSOnds2 is smaller than that for the other algorithms. On a positive note, the maximum HV value for VEPSOnds2 improves upon that for AbYSS, NSGA-II, and SPEA2.

The main purpose of this experiment is to present the overall performance of the improved VEPSO algorithm in comparison to state-of-the art algorithms, not to show how it outperforms them. Indeed, the overall performance of the VEPSOnds2 is not better than all the compared algorithms. However, relatively speaking, its performance is still within the acceptable range and is better than some of the other algorithms in certain cases.

5. Conclusions

The conventional VEPSO algorithm uses one swarm to optimise one objective function. The optimisation is guided using only one best solution found by another swarm with respect to the objective function optimised by that swarm. In contrast, recent PSO-based MOO algorithms prefer to use the nondominated solutions as the particle guides. Thus, it is possible to modify the VEPSO algorithm such that the particles are guided by nondominated solutions that are optimal at specific objective function. Five ZDT test problems were used to investigate the performance of the improved VEPSO algorithm based on the measures of the number of nondominated solutions found, the Generational Distance, the Spread, and the Hypervolume.

The experimental results show that the improved algorithms were able to obtain better quality Pareto fronts than conventional VEPSO, especially VEPSOnds2, which consistently returned the best convergence and diversity performance. On the other hand, the introduction of polynomial mutation should reduce the chance for a particle to get stuck in local optima, which features greatly in the ZDT4 test problem. However, VEPSOnds2 did not show much improvement compared to VEPSOnds1. This could possibly be due to the choice of the number of particles that are subject to mutation. Hence, the analysis for proper number of particles subject to mutation should be considered in future work. Even so, VEPSOnds2 is relatively better than VEPSOnds1, as confirmed by most of the performance measurements.

In addition, in VEPSOnds2, the particles of a swarm are guided by the same $gBest(t)$. Thus, there is a greater chance for them to converge prematurely around the $gBest(t)$ that might represent a locally optimal solution. On the other hand, in SMPSO, each particle will select one of the nondominated solutions by binary tournament, using the crowding distance as its guide. This means that in SMPSO, each particle has a different $gBest(t)$ as a guide during optimisation. Thus, the future VEPSOnds2 algorithm should reduce the chances for all particles to follow the same $gBest(t)$, in order to prevent premature convergence.

Acknowledgments

This work is supported by the Research University Grant (VOT 04J99) from Universiti Teknologi Malaysia, High Impact Research—UM.C/HIR/MOHE/ENG/16(D000016-16001), Research Acculturation Grant Scheme (RDU 121403), and MyPhD Scholarship from Ministry of Higher Education of Malaysia.

References

[1] K. E. Parsopóulos and M. N. Vrahatis, "Particle swarm optimization method in multiobjective problems," in *Proceeedings of the ACM Symposium on Applied Computing*, pp. 603–607, ACM, March 2002.

[2] D. Gies and Y. Rahmat-Samii, "Vector evaluated particle swarm optimization (VEPSO): optimization of a radiometer array antenna," in *IEEE Antennas and Propagation Society Symposium*, pp. 2297–2300, June 2004.

[3] S. M. V. Rao and G. Jagadeesh, "Vector evaluated particle swarm optimization (VEPSO) of supersonic ejector for hydrogen fuel cells," *Journal of Fuel Cell Science and Technology*, vol. 7, no. 4, Article ID 0410141, 2010.

[4] S. N. Omkar, D. Mudigere, G. N. Naik, and S. Gopalakrishnan, "Vector evaluated particle swarm optimization (VEPSO) for multi-objective design optimization of composite structures," *Computers and Structures*, vol. 86, no. 1-2, pp. 1–14, 2008.

[5] J. G. Vlachogiannis and K. Y. Lee, "Multi-objective based on parallel vector evaluated particle swarm optimization for optimal steady-state performance of power systems," *Expert Systems with Applications*, vol. 36, no. 8, pp. 10802–10808, 2009.

[6] J. Grobler, *Particle Swarm Optimization and Differential Evolution for Multi Objective Multiple Machine Scheduling [M.S. thesis]*, University of Pretoria, 2009.

[7] M. Reyes-Sierra and C. A. C. Coello, "Multi-objective particle swarm optimizers: a survey of the state-of-the-art," *International Journal of Computational Intelligence Research*, vol. 2, no. 3, 2006.

[8] C. A. Coello Coello and M. S. Lechuga, "MOPSO: a proposal for multiple objective particle swarm optimization," in *Proceedings of the Congress on Evolutionary Computation (CEC '02)*, vol. 2, pp. 1051–1056, 2002.

[9] C. A. Coello Coello, G. T. Pulido, and M. S. Lechuga, "Handling multiple objectives with particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 256–279, 2004.

[10] X. Li, "A non-dominated sorting particle swarm optimizer for multiobjective optimization," in *Genetic and Evolutionary Computation*, E. CantÃž-Paz, J. Foster, K. Deb et al., Eds., vol. 2723 of *Lecture Notes in Computer Science*, pp. 198–198, Springer, Berlin, Germany, 2003.

[11] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[12] M. Reyes-Sierra and C. A. Coello Coello, "Improving PSO-based Multi-Objective optimization using crowding, mutation and ε,-dominance," in *Evolutionary Multi-Criterion Optimization*, C. A. Coello Coello, A. HernÃąndez Aguirre, and E. Zitzler, Eds., vol. 3410 of *Lecture Notes in Computer Science*, pp. 505–519, Springer, Berlin, Berlin, 2005.

[13] M. A. Abido, "Multiobjective particle swarm optimization with nondominated local and global sets," *Natural Computing*, vol. 9, no. 3, pp. 747–766, 2010.

[14] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.

[15] J. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm Intelligence*, The Morgan Kaufmann Series in Evolutionary Computation, Morgan Kaufmann Publishers, San Francisco, Calif, USA, 2001.

[16] H. El-Sayed, M. Belal, A. Almojel, and J. Gaber, "Swarm intelligence," in *Handbook of Bioinspired Algorithms and Applications*, S. Olariu and A. Y. Zomaya, Eds., Chapman AND Hall/CRC computer and information science series, pp. 55–63, Taylor & Francis, Boca Raton, Fla, USA, 1st edition, 2006.

[17] J. D. Schaffer, *Some Experiments in Machine Learning Using Vector Evaluated Genetic Algorithms (Artificial Intelligence, Optimization, Adaptation, Pattern Recognition) [Ph.D. thesis]*, Vanderbilt University, 1984.

[18] E. Ãűzcan and M. Yäślmaz, "Particle swarms for multimodal optimization," in *Adaptive and Natural Computing Algorithms*, B. Beliczynski, A. Dzielinski, M. Iwanowski, and B. Ribeiro, Eds., vol. 4431 of *Lecture Notes in Computer Science*, pp. 366–375, Springer, Berlin, Germany, 2007.

[19] I. Schoeman and A. Engelbrecht, "A parallel vector-based particle swarm optimizer," in *Adaptive and Natural Computing Algorithms*, B. Ribeiro, R. F. Albrecht, A. Dobnikar, D. W. Pearson, and N. C. Steele, Eds., pp. 268–271, Springer, Vienna, Austria, 2005.

[20] D. A. Van Veldhuizen, *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations [Ph.D. thesis]*, Air Force Institute of Technology, Air University, 1999.

[21] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[22] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[23] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: improving the strength pareto evolutionary algorithm for multiobjective optimization," in *Evolutionary Methods for Design, Optimisation and Control with Application To Industrial Problems*, E. ZitlLer, M. Laumanns, and L. Thiele, Eds., pp. 95–100, International Center for Numerical Methods in Engineering (CIMNE), 2002.

[24] A. J. Nebro, F. Luna, E. Alba, B. Dorronsoro, J. J. Durillo, and A. Beham, "AbYSS: adapting scatter search to multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 4, pp. 439–457, 2008.

[25] J. Durillo, J. GarcÃna-Nieto, A. Nebro, C. A. Coello Coello, F. Luna, and E. Alba, "Multi-objective particle swarm optimizers: An experimental comparison," in *Evolutionary Multi-Criterion Optimization*, M. Ehrgott, C. Fonseca, X. Gandibleux, J. K. Hao, and M. Sevaux, Eds., vol. 5467 of *Lecture Notes in Computer Science*, pp. 495–509, Springer, Berlin, Germany, 2009.

[26] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*, vol. 16 of *Systems and Optimization Series*, John Wiley and Sons, Chichester, UK, 2001.

[27] A. J. Nebro, J. J. Durillo, G. Nieto, C. A. C. Coello, F. Luna, and E. Alba, "SMPSO: a new pso-based metaheuristic for multi-objective optimization," in *Proceedings of the IEEE Symposium on Computational Intelligence in Multi-Criteria Decision-Making (MCDM '09)*, pp. 66–73, usa, April 2009.