Hindawi
Discrete Dynamics in Nature and Society
Volume 2017, Article ID 5193013, 22 pages
https://doi.org/10.1155/2017/5193013

Research Article

Multispecies Coevolution Particle Swarm Optimization Based on Previous Search History

Danping Wang,[1,2,3] Kunyuan Hu,[1] Lianbo Ma,[1,2] Maowei He,[1,2,4] and Hanning Chen[4]

[1] *Department of Information Service & Intelligent Control, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China*
[2] *University of Chinese Academy of Sciences, Beijing 100039, China*
[3] *Shenyang University, Shenyang 110044, China*
[4] *School of Computer Science and Software, Tianjin Polytechnic University, Tianjin 300387, China*

Correspondence should be addressed to Maowei He; hemaowei@hotmail.com and Hanning Chen; perfect_chn@hotmail.com

Received 4 November 2016; Revised 11 April 2017; Accepted 19 April 2017; Published 27 June 2017

Academic Editor: Seenith Sivasundaram

A hybrid coevolution particle swarm optimization algorithm with dynamic multispecies strategy based on K-means clustering and nonrevisit strategy based on Binary Space Partitioning fitness tree (called MCPSO-PSH) is proposed. Previous search history memorized into the Binary Space Partitioning fitness tree can effectively restrain the individuals' revisit phenomenon. The whole population is partitioned into several subspecies and cooperative coevolution is realized by an information communication mechanism between subspecies, which can enhance the global search ability of particles and avoid premature convergence to local optimum. To demonstrate the power of the method, comparisons between the proposed algorithm and state-of-the-art algorithms are grouped into two categories: 10 basic benchmark functions (10-dimensional and 30-dimensional), 10 CEC2005 benchmark functions (30-dimensional), and a real-world problem (multilevel image segmentation problems). Experimental results show that MCPSO-PSH displays a competitive performance compared to the other swarm-based or evolutionary algorithms in terms of solution accuracy and statistical tests.

1. Introduction

Solving numerical optimization problems is a challenging research endeavor in many scientific areas. Many optimization techniques and search algorithms have drawn their motivation from evolution and social behavior. These include ant colony optimization [1], genetic algorithms [2], differential evolution [3], particle swarm optimization [4], and artificial immune systems [5]. In swarm intelligence (SI) systems, there are many simple individuals who can interact locally with one another and with their environments. Although such systems are decentralized, local interactions between individuals lead to the emergence of global behavior and properties. All of them have many variants, which have excellent performance. These variants are based on various improvement strategies.

In the last few years, several heuristics have been developed to improve the performance and set up suitable parameters for the PSO algorithm. van den Bergh [6] analyzed the trajectory of particles under different inertia weights and acceleration coefficients. The original structure of PSO model reflects an intuitive idea that a particle takes any position on which fitness value is better than where it currently is as the reference input. However, many experiments have shown that the basic PSO algorithm easily falls into local optima when solving complex multimodal problems with a huge number of local optima.

To overcome this problem, researchers conducted lots of studies. Inspired by the coevolution phenomenon in nature, in Cooperative Particle Swarm Optimization (CPSO) [7], multiple swarms are partitioned uniformly to optimize different components of the solution vector cooperatively. Inspired by this work, a CCPSO integrating the random grouping and adaptive weighting schemes was developed and demonstrated great promise in scaling up PSO on high-dimensional nonseparable problems [8]. In [9], a competitive and cooperative coevolutionary PSO (CCPSO) has considerable

potential for solving complex optimization problems by explicitly modeling the coevolution of competing and cooperating species. In Multiswarm Self-Adaptive and Cooperative Particle Swarm Optimization (MSCPSO) [10], particles in each subswarms share the only global historical best optimum to enhance the cooperative capability. In [11], Adaptive Cooperative Particle Swarm Optimizer (ACPSO) facilitates cooperation technique through the usage of the Learning Automata (LA) algorithm. Coevolutionary particle swarm optimization (PSO) algorithm associated with the artificial immune principle (ICPSO) [12] adopts an improved immune-clonal-selection operator for optimizing the elite subpopulation and a migration scheme for the information exchange between elite subpopulation and normal subpopulations.

Different from the above works with static-swarms strategy, some cooperative PSO' variants introduce dynamic multiswarms strategy to enhance the ability of exploring local optima. With the aim of maintaining multiple swarms on different peaks, a clustering PSO (CPSO) proposed in [13] employs a nearest neighbor learning strategy to train particles and a hierarchical clustering method to locate and track multiple optima. In hierarchical cluster-based multispecies particle swarm optimization (HCMSPSO) algorithm [14], a swarm is clustered into multiple species at an upper hierarchical level, and each species is further clustered into multiple subspecies at a lower hierarchical level. And in the lower layer, subspecies within the same species are formed adaptively in each iteration during the particle update. In [15], a dynamic multiswarm particle swarm optimizer (DMS-PSO) merges the harmony search (HS) algorithm into each subswarm. These subswarms are regrouped frequently and information is exchanged among the particles in the whole swarm. In a novel optimizer using Adaptive Heterogeneous Particle Swarms (AHPS^2) [16], to best take advantage of the heterogeneous learning strategies, an adaptive competition strategy is designed for dynamically adjusting the population size of an independent swarm with comprehensive learning and another one with subgradient learning based on their group performance. Roles of learning swarm and learnt swarm in particle swarm optimization with interswarm interactive learning strategy (IILPSO) [17] swap to maintain the diversity when interswarm interactive learning (IIL) behavior is triggered to adjust sizes of these two swarms.

In order to avoid the different individuals to repetitively exploit the same researched regions or excessive individuals to prematurely search a narrow region, researchers conducted a series of studies. Inspired by biology, niche [18] and speciation [19] techniques are introduced into PSO to prevent the swarm from crowding too closely and to locate as many optimal solutions as possible. Additionally, PSO topological structures are also widely adopted. The ring topology employed in [20] operating as a niching algorithm can drive the particles exploring the search space more broadly. Further, in [21] the multilayer strategy with multiple topologies is adopted to decrease the amount of noneffective exploiting tries. Inspired by ecological cohabitation, chaotic multiswarm particle swarm optimization (CMS-PSO) modifies the generic PSO with the help of the chaotic sequence for multidimension unknown parameter estimation and optimization by forming multiple cooperating swarms [22]. Historical memory strategy in HMPSO [23], which estimates and preserves distribution information of particles' historical promising, is helpful in preserving the information of optimum solution space and making a comprehensive learning.

Although all the above works improve the performance of particle swarm optimization, there is shortage of research into the performance of the hybrid algorithm with the dynamic multiple swarms strategy and nonrepetition search approach.

In this paper, we propose a new coevolution particle swarm optimization algorithm with dynamic multispecies strategy based on K-means clustering and nonrevisit strategy based on Binary Space Partitioning fitness tree (called MCPSO-PSH). It is shown that when a Binary Space Partitioning tree archive stores the positions and the fitness values of all evaluated solutions, this archive can be treated as an approximation of the fitness function, which can avoid the individuals' revisit phenomenon and improve the search efficiency. Moreover, K-means clustering method is introduced to partition whole population into subpopulations frequently. Information communication among subpopulations is implemented in the dynamic repartition process. Therefore, MCPSO-PSH algorithm can accommodate a considerable potential for solving complex problems. To comprehensively evaluate the performance of the proposed algorithm, MCPSO-PSH is compared with other state-of-the-art algorithms on three categories of experiments: (1) ten 10-dimensional and 30-dimensional benchmark problems with various properties, such as unimodal and multimodal functions, are employed to test MCPSO-PSO's performance for a diverse set of problems; (2) a set of CEC2005 30-dimensional tests including 10 benchmark functions are used to justify MCPSO-PSH's scalability for complex problems; (3) a real-world problem, the multilevel thresholds based on Otsu method for image segmentation, is employed to benchmark MCPSO-PSH's applicability for practical problems. The numerical results demonstrate that MCPSO-PSH significantly improves the performance of PSO and outperforms most of the comparison algorithms during the experiments. The main contributions of the proposed methods lie in the following aspects.

(1) A new way of information communication among multiple swarms is designed. Each particle's position is updated according to not only its personal history information and global information of its locating species but also other global information of other species.

(2) A dynamic K-means cluster is designed. In the process of decreasing the number of clusters and repartitioning the population into species, the search feature general achieves the transition from global search in the earlier evolution period to local search in the later evolution period.

(3) Binary Space Partitioning fitness tree memorizing the previous search history can prevent the individuals from revisiting the unpromising landscapes, which

can improve the search efficiency and avoid trapping in local optimum.

The rest of this paper is organized as follows: Section 2 first gives a review of the related works; Section 3 gives a brief explanation of the proposed coevolution particle swarm optimization algorithm; Section 4 presents the experimental studies of the proposed MCPSO-PSH; in Section 5, the performance of MCPSO-PSH based multilevel thresholding for image segmentation is evaluated. Finally, Section 6 concludes the paper.

2. Related Works

2.1. Particle Swarm Optimization. Particle swarm optimization (PSO) is an evolutionary computation (EC) algorithm paradigm that emulates the swarm behaviors of birds flocking [24]. It is a population-based search algorithm that exploits a population of individuals to probe promising regions of the search space. In this context, the population is called a swarm, and the individuals are called particles. Each particle moves with an adaptable velocity within the search space and retains the best position it ever encountered in its memory. The standard version of PSO is briefly described here [6].

Let s be the swarm size, d be the particle dimension space, and each particle of the swarm have a current position vector X_i, a current velocity vector V_i, and an individual best position vector P_i found by the particle itself so far. The swarm also has the global best position vector P_g found by any particle during all prior iterations in the search space. Assuming that the function $f(X)$ is to be minimized, and describing the following notations in tth iteration, the definitions are as follows:

$$X_i(t) = \left(x_{i,1}(t), x_{i,2}(t), \ldots, x_{i,d}(t)\right), \quad x_{i,j}(t) \in R_j,\ i = 1, 2, \ldots, s, \tag{1}$$

where each dimension of a particle is updated using the following equations:

$$v_{i,d}(t+1) = w \cdot v_{i,d}(t) + c_1 \cdot r_1 \cdot \left(P_{i,d}(t) - x_{i,d}(t)\right) + c_2 \cdot r_2 \cdot \left(P_{g,d}(t) - x_{i,d}(t)\right), \tag{2}$$

$$x_{i,d}(t+1) = x_{i,d}(t) + v_{i,d}(t+1). \tag{3}$$

In (2), c_1 and c_2 denote constant coefficients and r_1 and r_2 are elements from random sequences in the range of (0, 1).

The inertia weight w plays the role of balancing the global and local searches. It can be a positive constant (e.g., 0.9) or even a positive linear or nonlinear function of time [13, 14]. Although sometimes proper fine-tuning of the parameters c_1 and c_2 leads to an improved performance, an extended study of the cognitive and social parameters in [6] suggests $c_1 = c_2 = 2$ as default values.

2.2. K-Means Clustering. K-means clustering is a method of vector quantization, originally from signal processing, that is popular for cluster analysis in data mining. K-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster, as shown in Figure 1. This results in a partitioning of the data space into Voronoi cells [25].

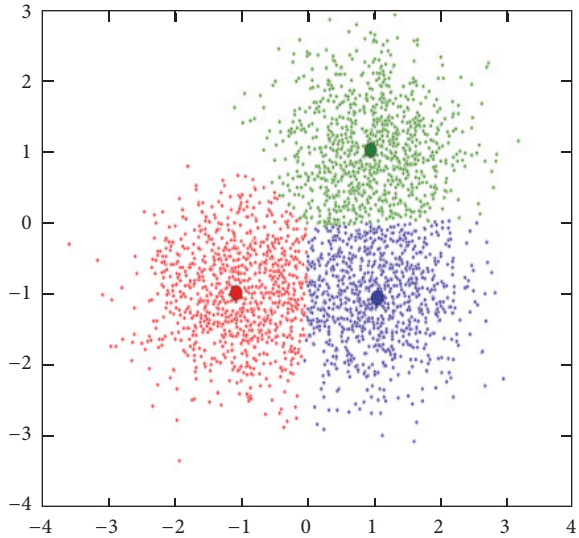


Figure 1: K-means clustering result with $k = 3$.

The cluster centers are substituted for center positions of food sources and the formula of computing the centers is shown in (4). If the ith cluster contains n_i members and the members are denoted as $x_1^i, x_2^i, \ldots, x_{ni}^i$, then the center ($\text{cluster}_i^{\text{center}}$) is determined as

$$\text{cluster}_i^{\text{center}} = \frac{\sum_{i=1}^{n_i} x_i}{n_i}. \tag{4}$$

The radius I of a cluster is defined as the mean distance (Euclidean) of the members from the center of the cluster. Thus, R can be written as follows:

$$R_i = \frac{\sum_{p=1}^{n_i} \left\| x_p - \text{cluster}_i^{\text{center}} \right\|}{n_i}, \tag{5}$$

where x_p is position of the pth member and n_i is the number of members in its corresponding cluster.

2.3. Fitness Tree. Inspired by Binary Space Partitioning (BSP) tree, fitness tree is proposed by Shiu and Chi [26]. The establishment of a fitness tree is aimed at partitioning the whole search space into several subspaces (subregions) according to the history positions of members.

Each node of fitness tree represents a partitioned subspace of the whole search space. And each node has at most two child nodes, left node and right node. The sum of the search spaces of left node and right node is the search space of the parent node. Such fitness tree is constructed by a series of solutions found by the individuals. The process of building fitness tree is random, which means its topology for information communication is constructed stochastically.

Input: Fitness tree and A new solution
Output: Fitness tree
%% A new solution z is inserting into the fitness tree
(1) Initialization: Current node is the root node *Curr_node* = *root*, *RF* = 0
(2) If (*Curr_node* has two child nodes x and y)
(3) Define the comparing dimension j:
(4) $j = \arg \max_{k \in [1,D]} (|x_j - y_j|)$
(5) If ($|x_j - z_j| \leq |x_j - y_j|$)
(6) *Curr_node* = child node x;
(7) Else
(8) *Curr_node* = child node y;
(9) End If and Goto Step (2)
(10) Else
(11) Insert a "virtual" child node to *Curr_node*, create a "real" child node that records z "under" the virtual child node.
(12) End If

Algorithm 1: Pseudocode of fitness tree construction.

The individuals of an evolution algorithm can be regarded as a sequence of solutions $\text{sq} = \{s(1), s(2), \ldots, s(n)\}$, where n is the population size. In each cycle, n parent individuals generate n child individuals. The child individual is inserted into the fitness tree as a leaf node. So fitness tree stores all the search positions $\text{SQ} = \{\text{sq}_1, \text{sq}_2, \ldots\}$. This means that there is no revisit for the moment. For normal evolution algorithms, parent individuals with better solutions withstand higher selection pressure, which results in higher probability to revisit the search space. In the extreme case, an individual trapping into a local optimum may be selected over and over, which causes that after several iterations all the individuals are generated from such individual. The phenomenon of revisit leads to a precipitous loss of population diversity.

The pseudocode of fitness tree construction is given in Algorithm 1. Here, a detailed explanation is given as follows. In the process of inserting a solution into the fitness tree, the current node *Curr_node* represents the survey node. When the survey dimension of current node is equal to the dimension of inserted node *Curr_node* = $s(i)$, the ith solution $s(i)$ in the sequence $\text{sq} = \{s(1), s(2), \ldots\}$ is a revisit.

The core idea of fitness tree is that, under finding out the dimension with max difference, the new individual locates as a leaf node or partitions a parent node (subspace) into two binary nodes (sub-subspace).

3. Algorithm

3.1. Multispecies Coevolution Model for Optimization. Inspired by the mutualism phenomenon, the single population PSO is extended into the interacting multispecies model by enhancing particle dynamics.

In such coevolution model, the information for the particle's position updating includes not only its personal history information and global information of its locating species but also other global information of other species. Information communication simultaneously occurs in and between the local species and neighboring species.

In mathematical terms, the original particle's fly velocity equation and position update equation (please see (2) and (3)) are improved as follows:

$$v_{i,d}^{k}(k+1) = w \cdot v_{i,d}^{k}(t) + c_1 \cdot r_1 \cdot \left(P_{i,d}(t) - x_{i,d}^{k}(t)\right) + c_2 \cdot r_2 \cdot \left(P_{g,d}^{k}(t) - x_{i,d}^{k}(t)\right) + c_3 \cdot r_3 \cdot \left(P_{g,d}^{\theta}(t) - x_{i,d}^{k}(t)\right), \quad (6)$$

$$x_{i,d}^{k}(t+1) = x_{i,d}^{k}(t) + v_{i,d}^{k}(t+1), \quad (7)$$

where the superscript k represents the kth species, $x_{i,d}^{k}$ denotes the position of the ith particle of the kth species, $P_{i,d}$ is the current particle's history best position, $P_{g,d}^{k}$ is the history best position found by its locating species, and $P_{g,d}^{\theta}$ is the history best position found by the whole population. In (6), c_1, c_2, and c_3 are the constant coefficients. And in this paper, c_1, c_2, and c_3 are set to have the same value, $c_1 = c_2 = c_3 = 1.6667$; r_1, r_2, and r_3 are the random values in the range of (0, 1).

In (6), $c_1 \cdot r_1 \cdot (P_{i,d}(t) - x_{i,d}^{k}(t))$ takes into account the individual's own experiences, $c_2 \cdot r_2 \cdot (P_{g,d}^{k}(t) - x_{i,d}^{k}(t))$ means the internal communication within the kth species, and $c_3 \cdot r_3 \cdot (P_{g,d}^{\theta}(t) - x_{i,d}^{k}(t))$ indicates the symbiotic coevolution between dissimilar species.

3.2. Dynamically K-Means Cluster. In order to implement the multispecies coevolution model, the stochastically generated population should be partitioned into n subspecies. The widely adopted K-means cluster method (please see Section 2.2) is employed for achieving the population division. The main drawback of K-means cluster is that the effect of population division depends on the initial distribution of stochastically generated individuals and the selection of initial division points. A relatively effective means is to set a large number of clusters.

Input: Fitness tree and new generated individuals
Output: Fitness tree
(1) All nodes of Fitness tree $S_{\text{tree}} = \{s_{\text{node1}}, s_{\text{node2}}, \ldots\}$ and new generated individuals $S_{\text{new}} = \{s_1, s_2, \ldots\}$
(2) $S = S_{\text{tree}} \cup S_{\text{new}}$
(3) For $j = 1{:}n$
(4) Compute minimum extreme point of each dimension:
(5) $znew_j^{\min} = \min_{s \in Snew} (s_{i,j})$
$z_j^{\min} = \min(z_j^{\min}, znew_j^{\min})$
(6) Compute maximum extreme point of each dimension:
(7) $z_j^{\max} = \max_{s \in Snew} (s_{i,j})$
$z_j^{\max} = \max(z_j^{\max}, znew_j^{\max})$
(8) For $s_j = 1{:}\text{num}(S)$
(9) Normalize each dimension of each individual $z_n(s_{i,j}) = \dfrac{(s_{\text{node},j} - z_j^{\min})}{(z_j^{\max} - z_j^{\min})}$
(10) End For
(11) End For

Algorithm 2: Pseudocode of adaptive normalization of population members.

In our proposed algorithm, the number of clusters is predefined as $C = \{c_1, c_2, \ldots, c_m\}$, where $c_1 > c_2 > \cdots > c_m$. During optimization, each individual in every cluster operates independently as the multispecies coevolution model (please see Section 3.1). After several optimization iterations T_i, the whole population is repartitioned into c_{i+1} clusters; c_i and c_{i+1} are orderly got from $C = \{c_1, c_2, \ldots, c_m\}$. Here, $T_1 + T_2 + \cdots + T_m = T$, where T is the total number of iterations. As c_{i+1} is smaller than c_i, the individuals in small clusters may be redistributed into the relatively larger clusters when the number of the clusters changes.

To balance the exploration and exploitation, the value of T_i is not constant. To enhance the ability of local search especially in early stages of searching, T_i should be larger than T_{i+1}. Therefore, in our proposed algorithm, T_i is calculated as follows:

$$T_i = \frac{m - i + 1}{\sum_{j=1}^{m} j} \cdot T. \tag{8}$$

Moreover, in the process of optimization, the centers of two or more clusters move too close or overlap each other. In order to avoid this phenomenon and to control every cluster to locally search a part of landscape, the relatively small cluster in two or more adjacent clusters decomposes and the individuals in such cluster redistribute into other adjacent clusters. Evaluation criterion is listed as follows:

$$\begin{aligned} &\left(\text{Dis}_{\text{clusters}} - R_{\text{cluster1}} - R_{\text{cluster2}}\right) \\ &\quad < 0.2 \cdot \min\left(R_{\text{cluster1}}, R_{\text{cluster2}}\right), \end{aligned} \tag{9}$$

where $\text{Dis}_{\text{clusters}}$ is the distance between two adjacent clusters and R_{cluster1} and R_{cluster2} are the radiuses of two adjacent clusters (please see (5)); the left term ($\text{Dis}_{\text{clusters}} - R_{\text{cluster1}} - R_{\text{cluster2}}$) represents the gap between two adjacent clusters.

3.3. Adaptive Normalization of Population Members. In the proposed multispecies particle swarm optimization, the idea of fitness tree is also employed to overcome the revisit phenomenon. Here, a modified adaptive normalized fitness tree is proposed.

In the process of inserting a new individual into fitness tree (please see Algorithm 1), there is a potential drawback: when determining the comparing dimension, it depends on the maximum difference among the dimensions of two child nodes.

For overcoming this drawback, the normalization method is employed to determine which dimension has the largest distinguishable difference. First, the extreme points of the new generated population S_{new} and constructed fitness tree S_{tree} are determined by identifying the minimum and maximum values $z_j^{\min}$ and $z_j^{\max}$ for each dimension $j = 1, 2, \ldots, n$ and by constructing the extreme points $z^{\min} = (z_1^{\min}, z_2^{\min}, \ldots, z_n^{\min})$ and $z^{\max} = (z_1^{\max}, z_2^{\max}, \ldots, z_n^{\max})$. Each dimension value of nodes in fitness tree s_{node} is then translated by subtracting the corresponding dimension value of extreme points and dividing the distribution range of corresponding dimension as follows:

$$z_n\left(s_{i,j}\right) = \frac{s_{\text{node},j} - z_j^{\min}}{z_j^{\max} - z_j^{\min}}. \tag{10}$$

The pseudocode of adaptive normalization of population members is listed in Algorithm 2.

3.4. Algorithmic Framework. In this paper, a new multispecies coevolution particle swarm optimization based on previous search history is proposed. The K-means clustering is applied to dynamically partition the whole population into multiple species. Fitness tree is employed to avoid that the individuals revisit the previous search regions.

The pseudocode of the proposed algorithm is listed in Algorithm 3.

Initialization:
(1) Stochastically generate a population of N individuals in search space;
(2) Set the numbers of clusters $C = \{c_1, c_2, \ldots, c_m\}$ and the total number of iteration T;
(3) Calculate fitness values of individuals;
(4) Normalize the dimensions of each individual as Algorithm 2;
(5) Initialize the fitness tree as Algorithm 1;
(6) $t = 1$; $I = 1$;
Loop:
(7) While ($t < T$)
(8) If ($t > \sum t_i$)
(9) Partition the whole population into c_i multiple species based on K-means clustering;
$I = I + 1$;
(10) End If
(11) Update the position of individuals according to Eq. (6) and (7);
(12) Normalize the dimensions of each individual as Algorithm 2;
(13) Insert new individuals into fitness tree as Algorithm 1;
(14) Update some individuals from fitness tree;
(15) Calculate fitness values of individuals;
(16) $t = t + 1$
(17) End

Algorithm 3: Pseudocode of multispecies coevolution particle swarm optimization based on previous search history.

4. Experimental Study

4.1. Experimental Settings. The proposed multispecies coevolution particle swarm optimization based on previous search history (MCPSO-PSH) was executed with four state-of-the-art EA and SI algorithms for comparisons: (1) canonical particle swarm optimization (PSO) [24]; (2) differential evolution (DE) [27]; (3) Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [28]; (4) nonrevisiting genetic algorithm (NrGA) [26].

In all experiments in this section, the values of the common parameters used in each algorithm such as population size and total generation number are chosen to be the same. Population size is set as 100 and the maximum evaluation number is set as 100000. For the continuous testing functions used in this paper, the dimensions are all set as 10D and 30D.

All the control parameters for the EA and SI algorithms are set to be default of their original literatures.

For canonical PSO, the learning rates c_1 and c_2 are both set as 2.0.

For canonical DE, the zoom factor is set as $F = 0.9$ and crossover rate is set as CR = 0.9.

For CMA-ES, the population size λ is chosen by the suggested setting in [28] (i.e., $\lambda = 4 + [3 * \ln D]$).

For NrGA, the crossover rate of 0.8 and the mutation rate of 0.01 are adopted [26].

For our proposed algorithm, the number of clusters is set as $C = \{10, 5, 4, 2, 1\}$, and so T is calculated as $T = \{T_1, T_2, T_3, T_4, T_5\} = \{333, 267, 200, 133, 67\}$. The learning rates c_1, c_2, and c_3 in (6) are all set as $c_1 = c_2 = c_3 = 1.3667$.

4.2. Test and Benchmark Problems. According to the No Free Lunch theorem, "for any algorithm, any elevated performance over one class of problems is exactly paid for in performance over another class" [29]. To fully evaluate the performance of the MCPSO-PSH without a biased conclusion towards some chosen problems, a set of ten basic benchmark functions and ten CEC2005 benchmark functions [30] is employed. The formulas of these functions are presented below.

4.2.1. Basic Benchmark Functions. (1) Rosenbrock function

$$f_1(x) = \sum_{i=1}^{n} 100 \times \left(x_{i+1} - x_i^2\right)^2 + \left(1 - x_i\right)^2. \quad (11)$$

(2) Quadric function

$$f_2(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2. \quad (12)$$

(3) Sum of different powers

$$f_3(x) = \sum_{i=1}^{n} \left|x_i\right|^{i+1}. \quad (13)$$

(4) Axis parallel hyperellipsoid function

$$f_4(x) = \sum_{i=1}^{n} i \cdot x_i^2. \quad (14)$$

(5) Sphere function

$$f_5(x) = \sum_{i=1}^{n} x_i^2. \quad (15)$$

(6) Ackley's function

$$f_6(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right) + 20 + e. \tag{16}$$

(7) Griewank's function

$$f_7(x) = \frac{1}{4000}\sum_{i=1}^{n}x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1. \tag{17}$$

(8) Michalewicz function

$$f_8(x) = -\sum_{i=1}^{n}\sin(x_i)\cdot\left(\sin\left(\frac{i\cdot x_i^2}{\pi}\right)\right)^{2\cdot m}, \quad m = 10. \tag{18}$$

(9) Rastrigin's function

$$f_9(x) = \sum_{i=1}^{n}x_i^2 - 10\cos 2\pi x_i + 10. \tag{19}$$

(10) Schwefel's function

$$f_{10}(x) = \sum_{i=0}^{n}|x_i| + \prod_{i=1}^{n}|x_i|. \tag{20}$$

These ten benchmark functions can be grouped as unimodal continuous functions f_1–f_5 and multimodal continuous functions f_6–f_{10}.

4.2.2. CEC2005 Benchmark Functions. (11) Shifted sphere function

$$f_{11}(x) = \sum_{i=1}^{D}z_i^2 + f\_\text{bias}_{11}, \quad z = x - o,\ x = [x_1, x_2, \ldots, x_D]. \tag{21}$$

(12) Shifted Schwefel's problem 1.2

$$f_{12}(x) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i}z_j\right)^2 + f\_\text{bias}_{12}, \quad z = x - o,\ x = [x_1, x_2, \ldots, x_D]. \tag{22}$$

(13) Shifted rotated high conditioned elliptic function

$$f_{13}(x) = \sum_{i=1}^{D}\left(10^6\right)^{(i-1)/(D-1)}z_i^2 + f\_\text{bias}_{13}, \quad z = (x - o) * M,\ x = [x_1, x_2, \ldots, x_D]. \tag{23}$$

(14) Shifted Schwefel's problem 1.2 with noise in fitness

$$f_{14}(x) = \sum_{i=1}^{D}\left(100\left(z_i^2 - z_{i+1}\right)^2 + \left(z_i - 1\right)^2\right) + f\_\text{bias}_{14}, \quad z = x - o + 1,\ x = [x_1, x_2, \ldots, x_D]. \tag{24}$$

(15) Schwefel's Problem 2.6 on bounds

$$f_{15}(x) = \max\left\{\left|A_i x - B_i\right|\right\} + f\_\text{bias}_{15}, \quad i = 1, \ldots, D,\ x = [x_1, x_2, \ldots, x_D]. \tag{25}$$

(16) Shifted Rosenbrock's function

$$f_{16}(x) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i}z_j\right)^2 * (1 + 0.4\,|N(0,1)|) + f\_\text{bias}_{16}, \quad z = x - o,\ x = [x_1, x_2, \ldots, x_D]. \tag{26}$$

(17) Shifted rotated Griewank's function without bounds

$$f_{17}(x) = \sum_{i=1}^{D}\frac{z_i^2}{4000} - \prod_{i=1}^{D}\cos\left(\frac{z_i}{\sqrt{i}}\right) + 1 + f\_\text{bias}_{17}, \quad z = (x - o) * M,\ x = [x_1, x_2, \ldots, x_D]. \tag{27}$$

(18) Shifted rotated Ackley's function on bounds

$$f_{18}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}z_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\right) + 20 + e + f\_\text{bias}_{18}, \quad z = (x - o) * M. \tag{28}$$

(19) Shifted Rastrigin's function

$$f_{19}(x) = \sum_{i=1}^{D}\left(z_i^2 - 10\cos(2\pi z_i) + 10\right) + f\_\text{bias}_{19}, \quad z = (x - o) * M,\ x = [x_1, x_2, \ldots, x_D]. \tag{29}$$

(20) Schwefel's Problem 2.13

Table 1: Performance of all algorithms on 10D test functions f_1–f_{10}.

Problem		MCPSO-PSH	NrGA	CMA-ES	PSO	DE
f_1	Mean	$4.3136E-02$	$3.9407E-01$	**$9.4574E-05$**	$7.3831E+04$	$3.0851E+13$
	Std	$7.3787E-07$	$3.1066E-05$	**$1.4783E-05$**	$6.3480E+03$	$1.0345E+12$
f_2	Mean	**$8.0378E-33$**	$3.7721E-27$	$8.4991E-18$	$1.0386E-03$	$3.9501E-01$
	Std	**$1.0378E-30$**	$7.0550E-30$	$3.8731E-21$	$2.7801E-03$	$4.0404E+00$
f_3	Mean	**$9.4931E-38$**	$5.7707E-33$	$3.4959E-14$	$1.4301E+47$	$3.5430E-03$
	Std	**$3.9093E-41$**	$7.3900E-37$	$8.0455E-20$	$1.3784E+47$	$5.7791E-04$
f_4	Mean	**$2.9043E-73$**	$3.8844E-36$	$8.3995E-18$	$9.4034E+08$	$3.1893E-43$
	Std	**$4.7898E-99$**	$1.3894E-45$	$2.9393E-20$	$1.0373E+09$	$7.8440E-53$
f_5	Mean	**$4.9550E-64$**	$3.9090E-39$	$4.4139E-43$	$8.0041E-17$	$6.0903E-02$
	Std	**$4.8414E-89$**	$3.4568E-40$	$6.3941E-46$	$9.4389E-18$	$5.9031E-02$
f_6	Mean	**$2.2137E-14$**	$7.3678E-13$	$3.8931E-13$	$7.3491E+01$	$1.3940E+02$
	Std	**$3.7890E-17$**	$2.4654E-14$	$3.9404E-14$	$3.4791E-03$	$4.9448E+01$
f_7	Mean	**0**	**0**	$7.4327E-03$	$3.9051E+00$	$2.6461E-2$
	Std	**0**	**0**	$4.7801E-03$	$9.4803E-01$	$8.4780E-3$
f_8	Mean	**$-9.8452E-01$**	**$-9.8452E-01$**	$9.0377E-01$	$2.0773E-01$	$2.4743E+00$
	Std	**0**	**0**	$6.9010E-01$	$1.3434E-01$	$9.0370E-1$
f_9	Mean	**0**	**0**	**0**	$3.8944E+02$	$1.9640E+03$
	Std	**0**	**0**	**0**	$9.0378E+01$	$8.9901E+01$
f_{10}	Mean	**0**	**0**	**0**	$1.3355E+04$	$7.9140E+04$
	Std	**0**	**0**	**0**	$8.9440E+03$	$4.4708E+04$

$$f_{20}(x) = \sum_{i=1}^{D} \left(A_i - B_i(x)\right) + f\_\text{bias}_{20},$$
$$x = [x_1, x_2, \ldots, x_D],\ A_i = \sum_{j=1}^{D} \left(a_{ij} \sin \alpha_j + b_{ij} \cos \alpha_j\right),\ B_i = \sum_{j=1}^{D} \left(a_{ij} \sin x_j + b_{ij} \cos x_j\right). \tag{30}$$

These ten benchmark functions can be grouped as unimodal continuous functions f_{11}–f_{15} and multimodal continuous functions f_{16}–f_{20}.

4.3. Result and Statistical Analysis

4.3.1. Results for Basic Benchmark Functions. In this experiment, the proposed MCPSO-PSH algorithm is compared with nonrevisiting genetic algorithm (NrGA), Covariance Matrix Adaptation Evolution Strategy (CMA-ES), the canonical particle swarm optimization (PSO) algorithm, and differential evolution (DE) algorithm.

For testing 10D benchmark test functions f_1–f_{10} in 10 runs, the basic statistical results including the mean and standard deviations of the function values are listed in Table 1. From Table 1, it is obvious that the canonical PSO and DE algorithms demonstrate a worse performance than MCPSO-PSH, NrGA, and CMA-ES. For the low-dimensional (10D) unimodal functions f_1–f_5, the uniformity of CMA-ES algorithm is relatively good. For function f_1, CMA-ES gets the best results. And, for functions f_2–f_5, the performance of CMA-ES is rather competitive. Compared with the canonical PSO and other state-of-the-art algorithms, MCPSO-PSH brings in the dynamical multispecies strategy based on K-means to enhance the ability of local search and fitness tree to overcome the revisit phenomenon, which results in a huge improvement of performance. The MCPSO-PSH and NrGA algorithms with nonrevisit strategy have nearly the same results for low-dimensional problems. Benefitting from the dynamic K-means clustering and the adaptive normalization of population members, the improved multispecies PSO has a slightly better performance than NrGA on functions f_1–f_5. For the low-dimensional (10D) multimodal functions f_6–f_{10}, intuitively, the proposed MCPSO-PSH algorithm has rather smaller mean values and standard deviations than other algorithms for function f_6. For functions f_7 and f_8, the MCPSO-PSH and NrGA obtain the global optimum with a high efficiency. It is due to the fact that the nonrevisit strategy based on BSP fitness tree effectively enhances the global search especially when a large number of local optima exist in the search space. For functions f_9 and f_{10}, the CMA-ES algorithm also gets the global optimum because the MCPSO-PSH, NrGA, and CMA-ES own their special mechanisms for jumping out of the local optimum. On the whole, regardless

Table 2: Performance of all algorithms on 30D test functions f_1–f_{10}.

Problem		MCPSO-PSH	NrGA	CMA-ES	PSO	DE
f_1	Mean	**4.8910E − 01**	1.3047E + 01	3.1794E + 00	3.4903E + 12	2.4941E + 21
	Std	**3.3890E − 01**	9.0360E + 00	2.0935E + 00	8.0410E + 11	9.3892E + 19
f_2	Mean	**1.6851E − 14**	1.8440E − 08	5.3882E + 02	3.7840E + 04	9.9038E8
	Std	**9.0376E − 16**	9.5040E − 10	3.0318E + 02	9.9011E + 03	1.3931E7
f_3	Mean	**5.9990E − 37**	6.3890E + 01	1.4890E − 12	3.8964E + 308	7.3280E + 223
	Std	**9.5490E − 41**	1.4378E + 01	2.1809E − 13	1.9430E + 310	1.9653E + 224
f_4	Mean	**1.0099E − 16**	8.3701E − 08	1.0454E − 23	7.9016E + 13	3.7890E + 08
	Std	**8.4098E − 18**	1.0871E − 07	9.5487E − 25	1.0346E + 14	1.3944E + 08
f_5	Mean	**1.3933E − 61**	2.0031E − 11	2.9890E − 16	2.3498E + 02	5.3903E + 05
	Std	**9.0809E − 75**	9.6036E − 13	7.3892E − 23	7.3954E + 01	1.2309E + 05
f_6	Mean	**1.3343E − 14**	1.3408E − 12	6.3378E − 06	8.4723E + 02	2.5044E + 02
	Std	**6.6066E − 15**	8.0651E − 14	1.6999E − 06	3.1704E + 02	7.3909E + 01
f_7	Mean	**0**	**0**	3.0966E − 2	2.0330E − 01	8.9036E − 01
	Std	**0**	**0**	7.8012E − 3	3.3891E − 01	6.7009E − 01
f_8	Mean	**−2.8743E + 01**	−2.7945E + 01	1.7640E + 01	6.2307E − 01	−3.3883E + 00
	Std	**1.664E + 00**	2.0874E + 00	7.3370E + 00	3.1700E − 01	6.0310E − 01
f_9	Mean	**3.3880E − 12**	8.5801E + 00	2.1388E + 00	4.3880E + 04	8.4891E + 03
	Std	**7.3038E − 13**	5.8817E + 00	2.9863E + 00	8.3909E + 03	2.7450E + 03
f_{10}	Mean	7.9063E − 01	**3.6421E − 04**	2.4578E + 03	2.3841E + 04	4.4093E + 05
	Std	4.1939E − 01	**1.3940E − 04**	2.7309E + 03	1.7301E + 04	1.0938E + 06

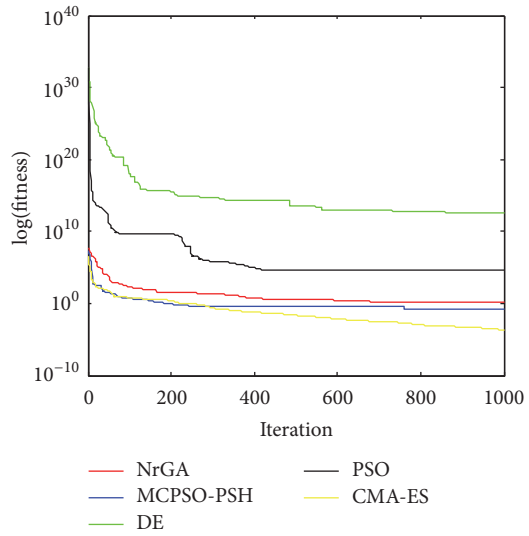
of the unimodal functions or the multimodal functions, the performance of the proposed MCPSO-PSH is uniform.

For testing 30D benchmark test functions f_1–f_{10} in 10 runs, the basic statistical results including the mean and standard deviations of the function values are listed in Table 2. With the increase of the dimensions, the convergence rate becomes slow. The results of Table 2 show great disparities in convergence accuracy. For the high-dimensional unimodal problems f_1–f_5, the performance of MCPSO-PSH algorithm is more outstanding than the other four algorithms because MCPSO-PSH with dynamic clustering strategy can accelerate the convergence rate especially in the later evolution period. The performance of NrGA algorithm becomes slightly poor although it also has the nonrevisit strategy for improving the search efficiency. Only for functions f_1–f_5 does CMA-ES outperform NrGA on convergence rate and accuracy. For solving the high-dimensional multimodal problems f_6–f_{10}, the MCPSO-PSH has a good performance. With the dynamic multispecies strategy based on K-means clustering and nonrevisit strategy based on BSP fitness tree, the MCSPSO-PSH algorithm can well avoid premature convergence to local optimum in the process of solving the high-dimensional multimodal problems. From the basic statistical results, NrGA also owns the most outstanding performance on testing functions f_6 and f_{10}, which means the nonrevisit strategy based on BSP fitness tree plays an important role in guiding search direction. On the whole, the performance of all algorithms tested here is ordered as MCPSO-PSH > NrGA > CMA-ES > PSO > DE.

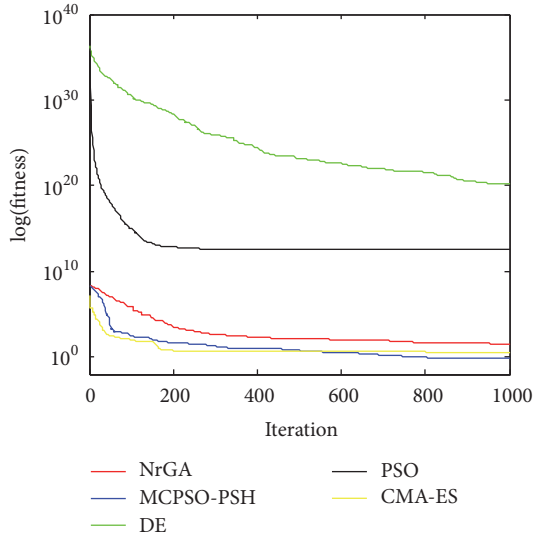
The convergence curves of the MCPSO-PSH algorithms and other EAs against test functions f_1–f_{10} in 10D and 30D are shown in Figure 2. It can be observed that, with the increase of the dimensions, the convergence rate of the proposed MCPSO-PSH algorithm becomes faster. Predictably, with a further increase of the dimensions of the test functions, the MCPSO-PSH algorithm will have an overwhelming superiority in solving the high-dimensional problems.

4.3.2. Results for CEC2005 Benchmark Functions. The statistical results in Table 3 show that MCPSO-PSH has significantly outstanding performance for all these cases except f_{14} and f_{20}. For f_{14}, although all the algorithms provide similar results (except DE), the solutions of MCPSO-PSH are still rather competitive and accessible. For f_{20} MCPSO-PSH achieved the second best solution just behind NrGA. The remarkable performance of MCPSO-PSH for f_{11}–f_{15} indicates that MCPSO-PSH also applies to unimodal functions though it is primarily designed for multimodal functions. For the multimodal functions f_{16}–f_{19}, MCPSO-PSH outperforms NrGA, CMA-ES, PSO, and DE. In terms of average obtained results, MCPSO-PSH is better than NrGA on f_{12}, f_{13}, f_{15}, f_{16}, f_{17}, and f_{18}, and is somehow equivalent to it on f_{11} and f_{19}. MCPSO-PSH and NrGA outperform other three algorithms on most cases, which demonstrates great promise of nonrevisiting strategy for maintaining population diversity and improving search efficiency synchronously. Obviously, MCPSO-PSH is superior to NrGA which benefits from its unique dynamic multispecies strategy based on K-means clustering.

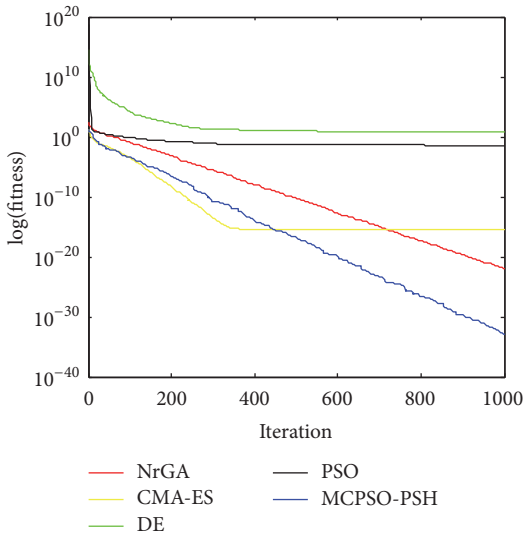
In order to intuitively depict the search effect, the box plots are employed to demonstrate the statistical performance of five different algorithms, as shown in Figure 3. Although



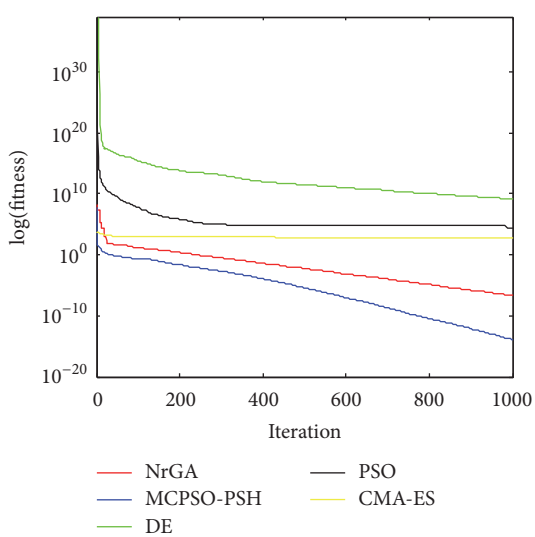
(a) Rosenbrock-10D



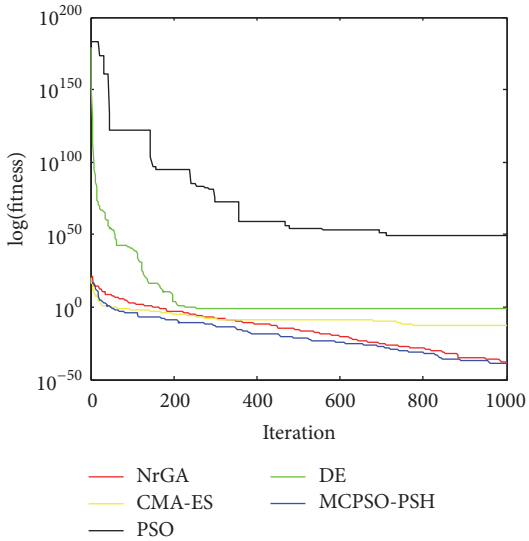
(a′) Rosenbrock-30D



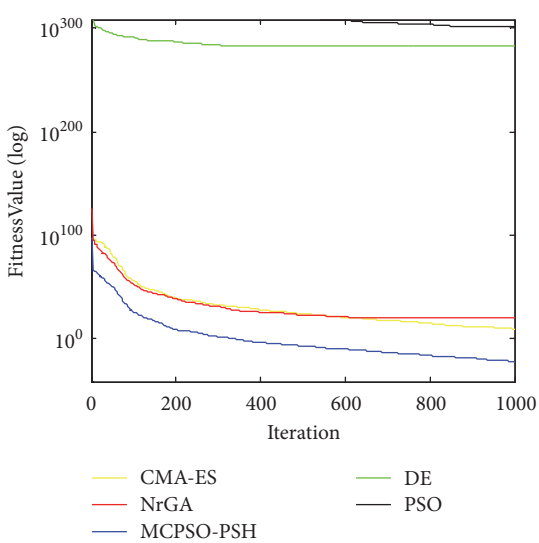
(b) Quadric-10D



(b′) Quadric-30D



(c) Sum-10D



(c′) Sum-30D

Figure 2: Continued.

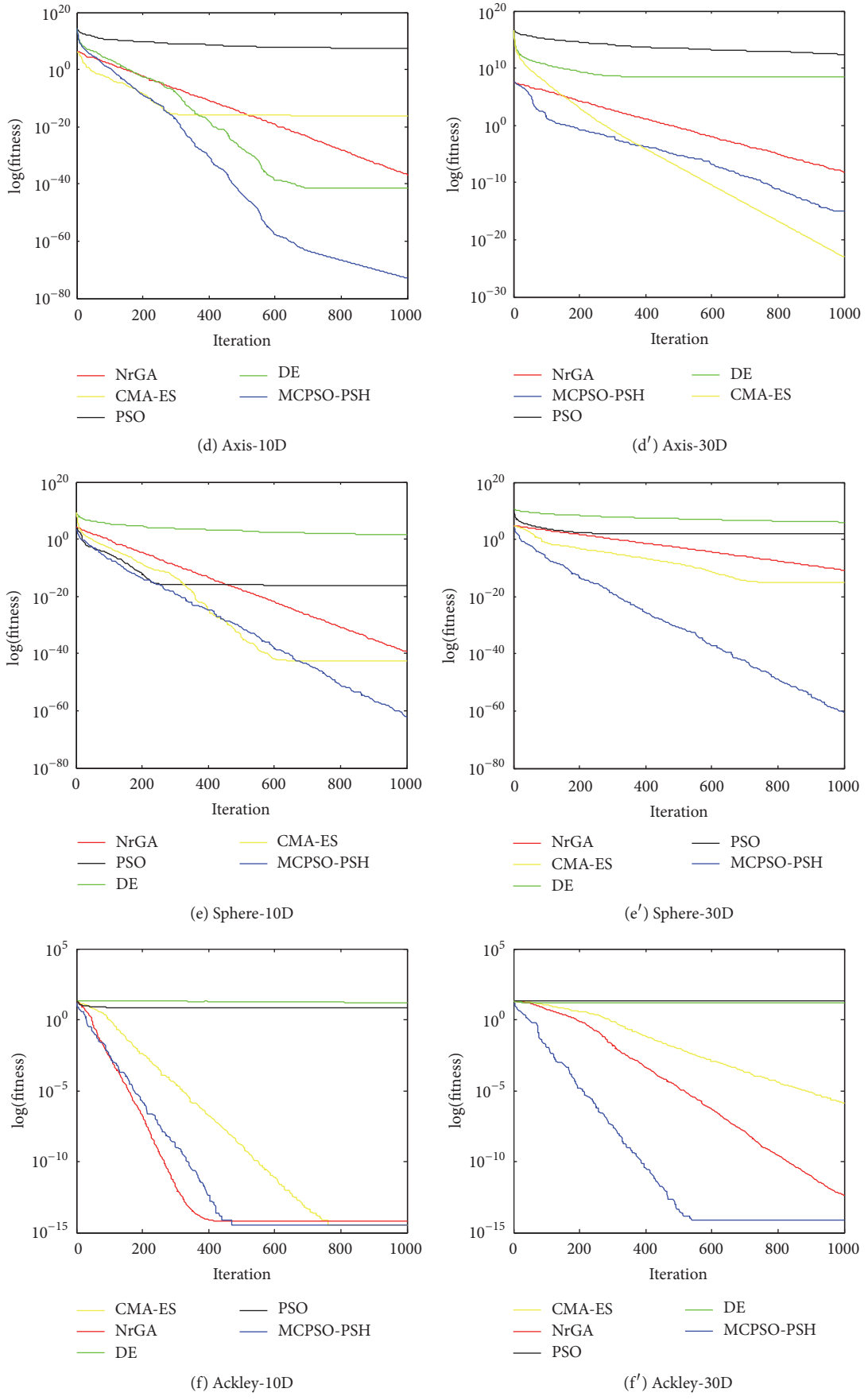
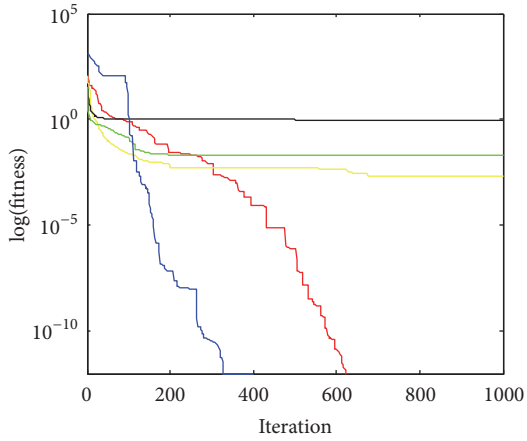
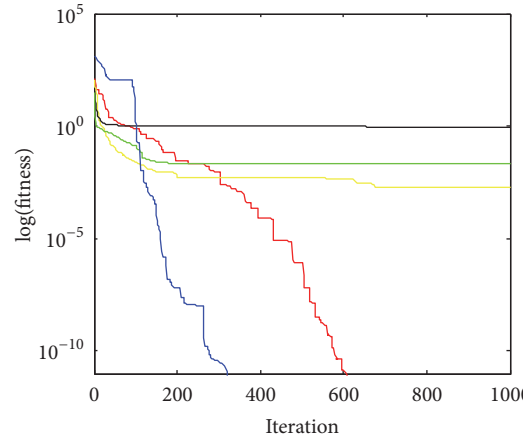


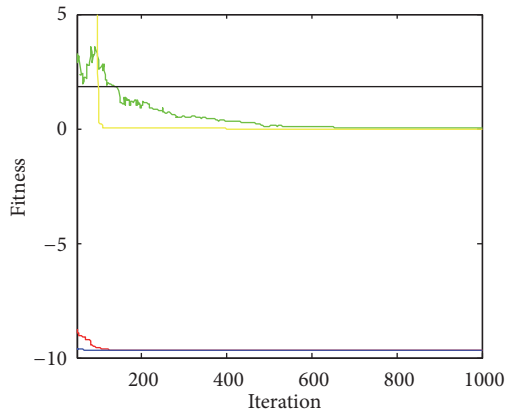
Figure 2: Continued.



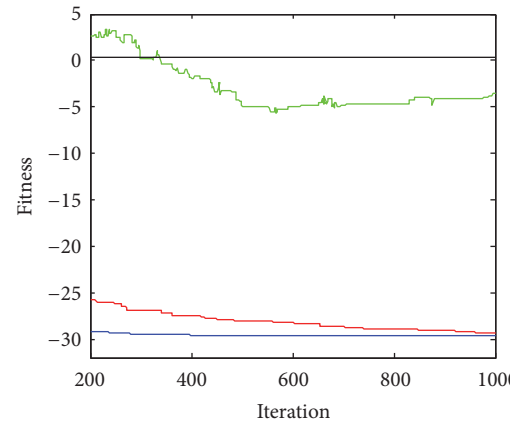
(g) Griewank-10D



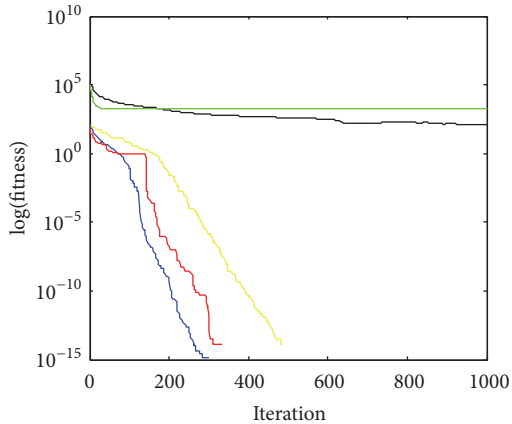
(g′) Griewank-30D



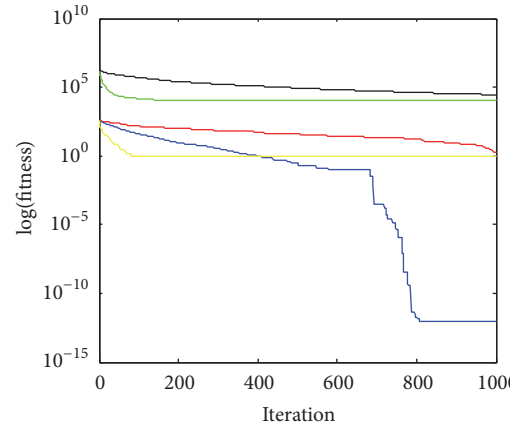
(h) Michalewicz -10D



(h′) Michalewicz -30D



(i) Rastrigin-10D



(i′) Rastrigin-30D

Figure 2: Continued.

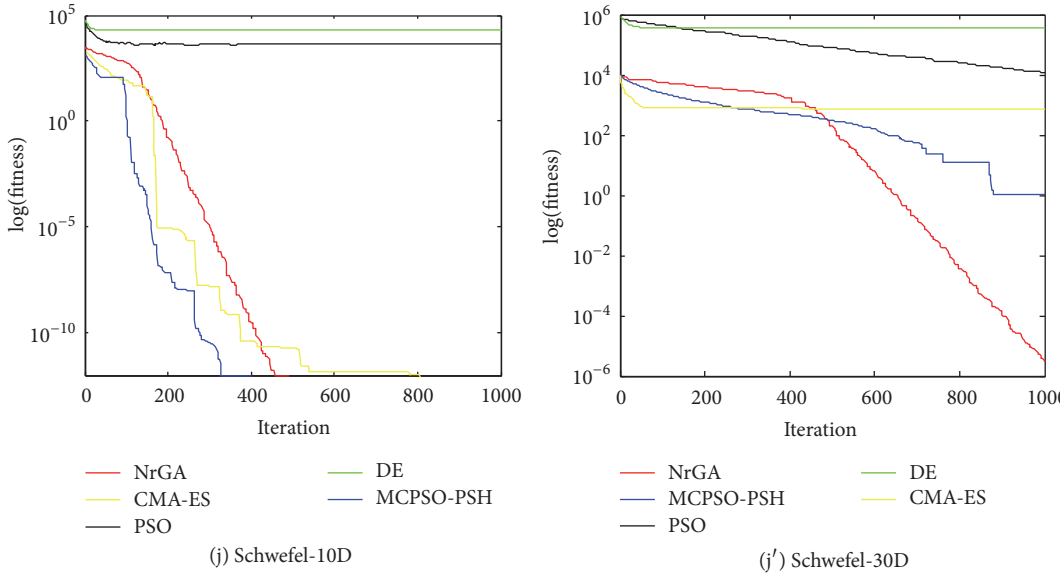


Figure 2: Convergence curves of MCPSO-PSH, NrGA, CMA-ES, PSO, and DE against test functions f_1–f_{10} in low dimension, 10D, and high dimension, 30D, respectively.

Table 3: Basic statistics of f_{11}–f_{20} CEC2005 benchmark test functions obtained by MCPSO-PSH, NrGA, CMA-ES, PSO, and DE.

Problem		MCPSO-PSH	NrGA	CMA-ES	PSO	DE
f_{11}	Mean	**−4.5000E + 02**	**−4.5000E + 02**	−3.1857E + 02	−4.3912E + 02	−4.4313E + 02
	Std	**0**	**0**	1.8576E + 02	2.3441E + 01	9.7454E + 00
f_{12}	Mean	**−4.5000E + 02**	3.1554E + 03	−2.2058E + 02	1.2360E + 03	−4.4506E + 02
	Std	**0**	1.9571E + 03	8.3398E + 01	3.2960E + 02	3.6262E − 09
f_{13}	Mean	**−4.5000E + 02**	6.9857E + 06	2.4164E + 05	2.5967E + 07	3.5243E + 07
	Std	**0**	2.5378E + 06	3.0876E + 04	4.3459E + 06	6.5900E + 07
f_{14}	Mean	3.8043E + 02	**2.3818E + 02**	5.3723E + 02	2.7801E + 02	3.5976E + 03
	Std	3.2991E + 00	**6.8326E + 00**	6.3167E + 01	8.3767E + 00	3.2048E + 03
f_{15}	Mean	**2.3174E + 03**	1.6341E + 04	5.2245E + 03	3.3347E + 03	3.8727E + 03
	Std	**5.0134E + 02**	1.0450E + 03	2.1529E + 03	5.7811E + 02	4.3668E + 03
f_{16}	Mean	**−4.5000E + 02**	2.1850E + 04	−1.8444E + 02	6.5597E + 03	−4.4761E + 02
	Std	**0**	3.1742E + 03	1.2313E + 02	1.3545E + 03	3.4378E − 12
f_{17}	Mean	**−1.8000E + 02**	1.9705E + 03	1.8865E + 04	4.6964E + 03	**−1.8000E + 02**
	Std	**0**	1.9839E + 02	3.9520E + 02	8.4681E + 02	**0**
f_{18}	Mean	**−1.1997E + 02**	−1.1911E + 02	−1.1958E + 02	−1.1909E + 02	−1.1995E + 02
	Std	**8.9147E − 03**	7.8361E − 02	3.7314E − 01	4.2087E − 02	6.0499E − 01
f_{19}	Mean	**−3.3000E + 02**	**−3.300e + 02**	−1.9906E + 02	−3.1776E + 02	−3.2842E + 02
	Std	**0**	**0**	1.6472E + 01	7.5984E + 01	8.5989E + 01
f_{20}	Mean	−4.3294E + 02	**−4.3933E + 02**	−4.2932E + 02	−3.9982E + 02	−4.0363E + 02
	Std	3.1803E + 01	**1.7580E + 01**	2.0839E + 01	6.1074E + 01	3.5741E + 01

nonrevisiting strategy is implemented in both NrGA and MCPSO-PSH, the robustness of NrGA is still undesirable. MCPSO-PSH with dynamic multispecies strategy has great enhancement of exploring ability.

From the above observations, the superiority of MCPSO-PSH over other peer algorithms can be easily presumed.

5. Multilevel Threshold for Image Segmentation

5.1. Otsu Method. The Otsu multithreshold measure [31] proposed by Otsu has been popularly employed in determining whether the optimal threshold method can provide image

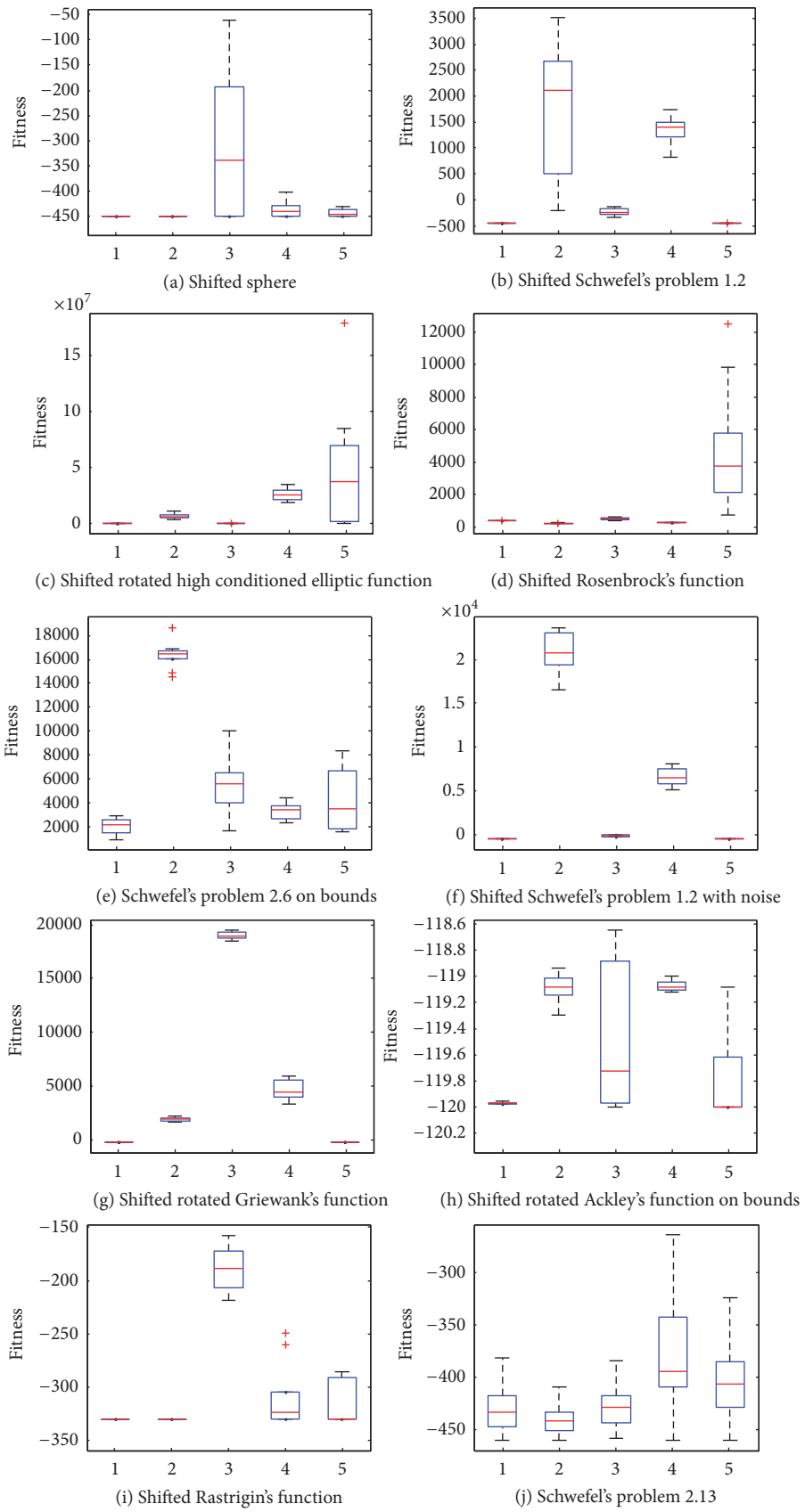


Figure 3: Box plots of results obtained by all algorithms. Here, 1 to 5 in the horizontal axis are indices of MCPSO-PSH, NrGA, CMA-ES, PSO, and DE, respectively (the box has lines at the lower quartile, median, and upper quartile values. The whiskers extend to the most extreme data points not considered outliers. Outliers (denoted by +) are data with values beyond 100 units of interquartile range).

segmentation with satisfactory results. The traditional Otsu method can be described as follows.

Let the grey levels of a given image with N pixels range over $[0, L-1]$ and $h(i)$ denote the number of the ith grey level pixels. $P(i)$ is the probability of i.

$$N = \sum_{i=0}^{L-1} h(i),$$
$$P(i) = \frac{h(i)}{N}, \quad \text{for } 0 \le i \le L-1. \tag{31}$$

Otsu defined the between-class variance as the sum of sigma functions of each class by (22)

$$f(t) = \delta_0 + \delta_1,$$
$$\delta_0 = w_0 \cdot (u_0 - u_T)^2,$$
$$\delta_1 = w_1 (u_1 - u_T)^2, \tag{32}$$
$$\delta_1 = w_1 \cdot (u_1 - u_T)^2,$$

where u_T is the mean intensity of the original image. In bilevel thresholding, the mean level of each class, (u_i), can be calculated as follows:

$$u_0 = \sum_{i=0}^{i=t-1} i \times \frac{P_i}{w_0},$$
$$u_1 = \sum_{i=t}^{i=L} i \times \frac{P_i}{w_0}, \tag{33}$$

where

$$w_0 = \sum_{i=0}^{i=t-1} P_i,$$
$$w_1 = \sum_{i=t}^{i=L} P_i. \tag{34}$$

The optimal threshold can be derived by maximizing the between-class variance function

$$t^* = \arg\max\left(f(t)\right). \tag{35}$$

Bilevel thresholding based on the between-class variance can be extended to multilevel thresholding. The extended between-class variance function is calculated as follows:

$$f(t) = \sum_{i=0}^{M} \delta_i, \tag{36}$$

where M is the number of thresholds. And

$$\delta_0 = w_0 (u_0 - u_T)^2,$$
$$\delta_j = w_j \left(u_j - u_T\right)^2,$$
$$\delta_M = w_M (u_M - u_T)^2,$$
$$u_0 = \sum_{i=0}^{t_i-1} i \times \frac{P_i}{w_i}, \tag{37}$$
$$u_j = \sum_{i=t_j}^{t_{j+1}-1} i \times \frac{P_i}{w_i},$$
$$u_M = \sum_{i=t_M}^{L-1} i \times \frac{P_i}{w_i}.$$

The optimal thresholds are found by maximizing the between-class variance function:

$$[t_1, t_2, \ldots, t_M] = \arg\max\left(f\left([t_1, t_2, \ldots, t_M]\right)\right). \tag{38}$$

Equation (38) is used as the objective function which is to be optimized (maximized). A close look into this equation will show that it is very similar to the expression for uniformity measure.

5.2. Experiment Setup. For evaluating the performance of the proposed algorithm (MCPSO-PSH), we implemented it on a wide variety of images. These images include the popular tested images used in previous studies [32, 33], named avion.ppm, lena.ppm, and hunter.pgm (available in [34]) and 41033.jpg, 62096.jpg, and 145086.jpg provided by the Berkeley Segmentation Database (BSDS) (available in [35]). The sizes of avion.ppm, lena.ppm, and hunter.pgm are $512 * 512$ and the sizes of 41033.jpg, 62096.jpg, and 145086.jpg are $321 * 481$. Each image has a unique grey-level histogram. Most of the images are difficult to segment due to multimodality of the histograms.

The performance of the proposed algorithm is demonstrated by comparing with those of other popular algorithms developed in the literature so far, for example, canonical particle swarm optimization (PSO), differential evolution (DE), Covariance Matrix Adaptation Evolution Strategy (CMA-ES), and nonrevisiting genetic algorithm (NrGA). A comparison between the proposed algorithm and other methods is evaluated based on Otsu, which means (38) is regarded as fitness function to evaluate all involved algorithms. The parameters settings are identical to those introduced in Section 4.1.

The numbers of thresholds M investigated in the experiments were 2, 3, 4, 5, 7, and 9, while all the experiments were repeated 30 times for each image. The population size is 100 and the maximum number of FEs is 2000. The numbers of thresholds $M-1 = 2, 3, 4$ can be realized as low-dimensional problems, which also can be solved through the traditional search method although suffering from the long CPU time. The numbers of thresholds $M - 1 = 5, 7, 9$ are regarded

Table 4: Objective values and thresholds by the Otsu method.

Image	$M = 2$		$M = 3$		$M = 4$	
	Objective values	Optimal thresholds	Objective values	Optimal thresholds	Objective values	Optimal thresholds
Avion	3.4413$E4$	113, 173	3.4489$E4$	93, 145, 191	3.4535$E4$	84, 129, 172, 203
Lena	1.7597$E4$	93, 151	1.7764$E4$	81, 127, 171	1.7827$E4$	75, 114, 145, 180
Hunter	9.2768$E3$	51, 116	9.4261$E3$	36, 86, 135	9.4822$E3$	30, 72, 111, 146
41033.jpg	1.2753$E4$	64, 131	1.2885$E4$	57, 107, 165	1.2945$E4$	49, 79, 122, 171
62096.jpg	2.7469$E4$	102, 196	2.7585$E4$	86, 156, 218	2.7631$E4$	71, 121, 176, 225
145086.jpg	1.5754$E4$	89, 175	1.5982$E4$	69, 122, 186	1.6085$E4$	41, 85, 130, 189
Mean CPU time (s)	1.4427		61.421		2644.543	

as medium levels for image segmentation, which means the high-dimensional problem for image segmentation cannot be solved by the traditional search method.

In all experiments, to ensure that the initial values of each random algorithm are equal, we used the MATLAB command $\text{rand}(X, Y) * (\max L - \min L) + \min L$. $\max L$ and $\min L$ are the maximum and minimum greys of the tested images. Figure 4 presents six original images and their histograms.

5.3. Experimental Results of Multilevel Threshold

Case I: Multilevel Threshold Results with $M - 1 = 2, 3, 4$. We employ (38) as the fitness function to provide a comparison of performances. Since the classical Otsu method is an exhaustive searching algorithm, we regarded its results as the "standard." Table 4 shows the fitness function values (with $M - 1 = 2, 3, 4$) attained by Otsu methods. In Table 4, the mean computation time and corresponding optimal thresholds are listed. For $M - 1 > 4$, owing to the unbearable consumption of CPU time, we do not list the correlative values in our experiment. These "standard" results will be used to compare with the results attained by optimization methods.

It is well known that the EA-based Otsu method for multilevel thresholding segmentation can only accelerate the computation velocity. The attained results will just get incredibly close to the optimal results but fail to improve the optimal results. Using the proposed MCPSO-PSH and other evolution algorithms, the mean fitness function values and their standard deviations obtained by the test algorithms (with $M - 1 = 2, 3, 4$) are shown in Table 5. The bigger mean values and smaller standard deviation listed in Table 5 mean that the algorithm gets the better results.

According to the mean CPU times shown in Table 6, there is no obvious difference between the EA-based Otsu methods. All the EA-based Otsu methods are suitably efficient and practical in terms of time complexity for high-dimensional image segmentation problems. Therefore, in comparison with the traditional Otsu algorithm, all the EA-based Otsu methods effectively shorten the computation time.

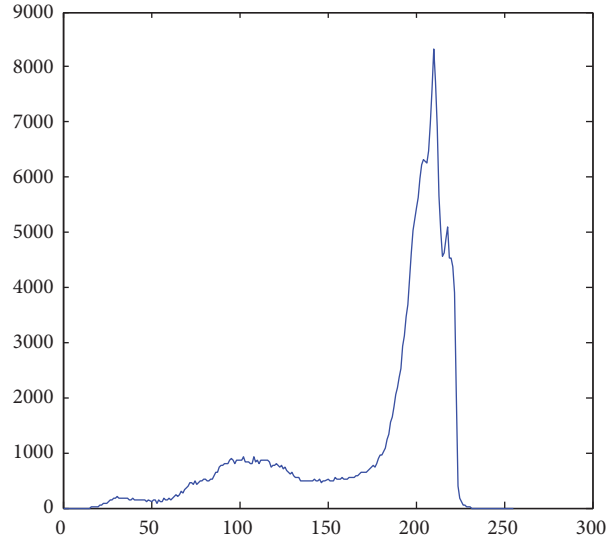
Among the EA-based Otsu methods tested listed in Table 5, MCPSO-PSH obtains results equal to or close to that obtained in the traditional methods when $M - 1 = 2, 3, 4$. Moreover, the standard deviations obtained by MCPSO-PSH are considerably small among the results obtained by the EA-based Otsu methods. This is attributed to the fact that the MCPSO-PSH has a good balance between the global and local search abilities. Therefore, the MCPSO-PSH-based Otsu method is suitable for the low-dimensional segmentation problems ($M - 1 = 2, 3, 4$).

Case II: Multilevel Threshold Results with $M - 1 = 5, 7, 9$. To further investigate the search performance of each population-based method on high-dimensional segmentation problems, we conduct these algorithms on image segment with $M - 1 = 5, 7, 9$. Due to the unacceptable computation time for the traditional Otsu method, it does not need to be assessed in this test. The average fitness value and standard deviation obtained by each of the EA-based Otsu methods are listed in Table 7, where the correlative results with the larger values and smaller standard deviations indicate the better achievements.

From Table 7, it is obvious that, with the growth in dimension, there is statistically significant gap among experiments using these EA-based Otsu methods, in terms of both efficiency (fitness values) and stability (standard deviation). Depending on the nonrevisit strategy based on Binary Space Partitioning fitness tree and dynamic multispecies strategy based on K-means clustering, MCPSO-PSH owns the best performance and stability on a high dimension, which is more efficient than the canonical PSO and other classical EAs for image segmentation with Otsu method. Moreover, NrGA, which used nonrevisit strategy, also gets more acceptable results than the other algorithms, except MCPSO-PSH. With the increase of dimensions, the fitness values and standard deviation demonstrate that the performance of CMA-ES-based Otsu method has greatly improved. The results shown in Table 7 prove that the MCPSO-PSH-based Otsu method is more suitable for resolving multilevel image segmentation problems.

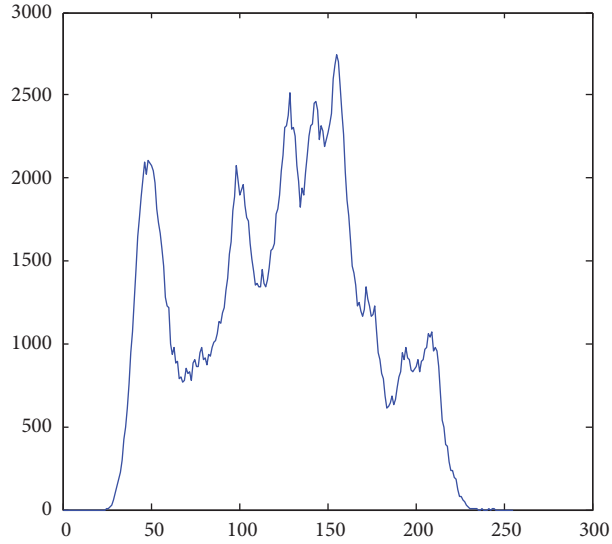
As mentioned above, the EA-based Otsu methods adopt different strategies to conduct global and local searches. An elaborate optimizer should have a better balance between exploration and exploitation search abilities. The above tests show that the MCPSO-PSH-based Otsu method outperforms the other EA-based Otsu method for multilevel image segmentation, especially on high-dimensional thresholding (with $M - 1 = 5, 7, 9$). Moreover, compared with the traditional Otsu method, this method shortens the computational time.

(a) Avion.ppm



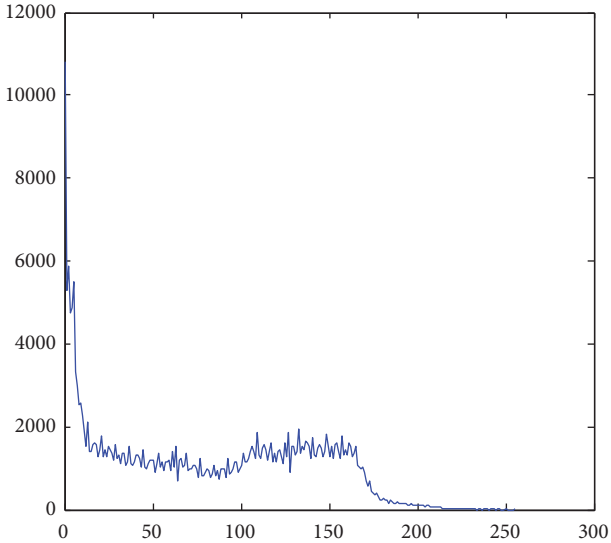
(a′) Histogram of avion.ppm

(b) Lena.ppm



(b′) Histogram of lena.ppm

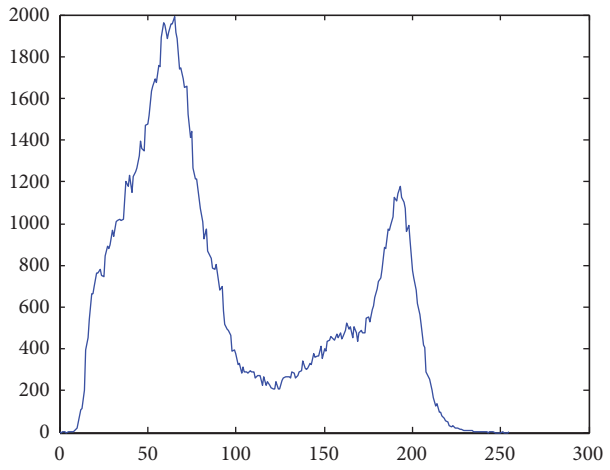
(c) Hunter.pgm



(c′) Histogram of hunter.pgm

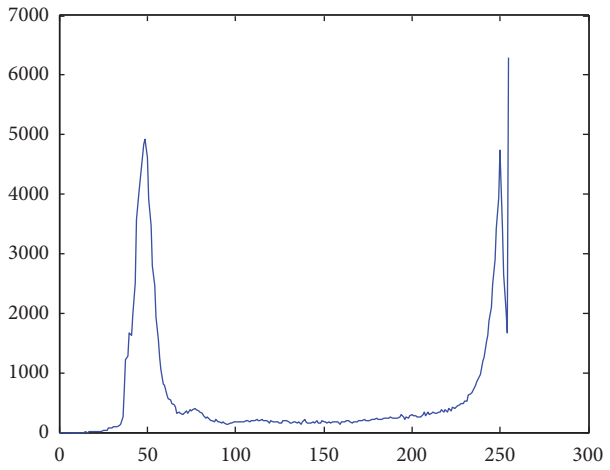
Figure 4: Continued.

(d) 41033.jpg



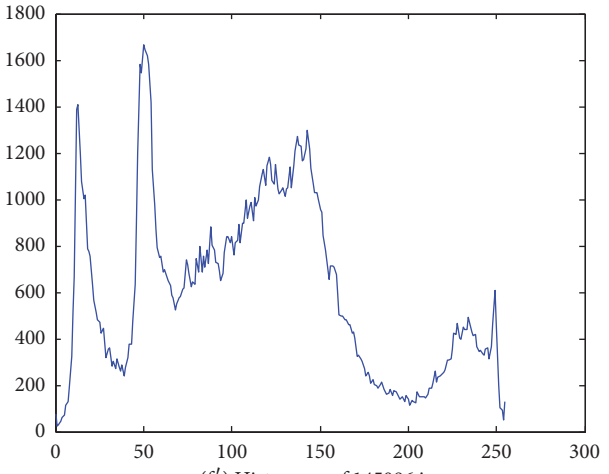
(d′) Histogram of 41033.jpg

(e) 62096.jpg



(e′) Histogram of 62096.jpg

(f) 145086.jpg



(f′) Histogram of 145086.jpg

Figure 4: Test images and their histograms.

6. Conclusions

In this paper, we have presented a new coevolution particle swarm optimization algorithm with dynamic multispecies strategy based on K-means clustering and nonrevisit strategy based on Binary Space Partitioning fitness tree (called MCPSO-PSH). With the help of storing and analyzing the positions and the fitness values of all evaluated solutions in Binary Space Partitioning tree archive, MCPSO-PSH can prevent its individuals from revisiting the same researched regions and improve the search efficiency. The proposed algorithm adopts K-means clustering method to

Table 5: Objective value and standard deviation by the compared EA-based methods on Otsu algorithm.

Image	M	Objective values (standard deviation)				
		MCPSO-PSH	NrGA	CMA-ES	PSO	DE
Avion	2	$3.4413E4$	$3.4413E4$	$3.4054E4$	$3.4411E4$	$3.4054E4$
		0.1390	$7.66953E-12$	$6.633E-2$	2.9091	$7.6695E-12$
	3	$3.4489E4$	$3.4489E4$	$3.4129E4$	$3.4485E4$	$3.4130E4$
		0.2708	$1.9659E-2$	0.2735	4.4573	0.1892
	4	$3.4534E4$	$3.4535E4$	$3.4172E4$	$3.4507E4$	$3.4174E4$
		0.5044	0	1.5801	28.4693	0.4677
Lena	2	$1.7597E4$	$1.7597E4$	$1.7348E4$	$1.7590E4$	$1.7348E4$
		$3.8348E-12$	$3.8347E-12$	0.1153	10.9993	$3.8348E-12$
	3	$1.7764E4$	$1.7764E4$	$1.7513E4$	$1.7723E4$	$1.7515E4$
		$6.3873E-3$	$3.83477E-1$	1.1303	65.6575	$5.6065E-2$
	4	$1.7827E4$	$1.7828E4$	$1.7573E4$	$1.7797E4$	$1.7578E4$
		0.4322	$1.19275E-4$	1.9239	31.2048	0.7919
Hunter	2	$9.2768E3$	$9.2769E3$	$9.1196E3$	$9.1641E3$	$9.1200E3$
		0	0	0.4918	$1.4411E2$	0
	3	$9.4261E3$	$9.4261E3$	$9.2670E3$	$9.3724E3$	$9.2686E3$
		$1.3909E-2$	$1.9174E-12$	1.2988	65.2063	0.6674
	4	$9.4821E3$	$9.4821E3$	$9.3215E3$	$9.4147E3$	$9.3214E3$
		0.1261	0.1262	1.7285	47.8078	3.2285
41033	2	$1.2753E4$	$1.2753E4$	$1.2753E4$	$1.2746E4$	$1.2753E4$
		0	$4.0595E-3$	0.1169	14.7680	0
	3	$1.2885E4$	$1.2885E4$	$1.2884E4$	$1.2866E4$	$1.2885E4$
		0	0	1.0243	42.8287	0.1079
	4	$1.2945E4$	$1.2945E5$	$1.2940E4$	$1.2911E4$	$1.2944E4$
		$1.398E-3$	$1.3245E-3$	2.0414	31.5746	2.2078
62096	2	$2.7469E4$	$2.7469E4$	$2.7469E4$	$2.7457E4$	$2.7469E4$
		0	$3.8348E-12$	0.1693	15.2460	0
	3	$2.7585E4$	$2.7585E4$	$2.7584E4$	$2.7568E4$	$2.7585E4$
		0.5133	0	0.8584	35.4894	0.1933
	4	$2.7631E4$	$2.7631E4$	$2.7629E4$	$2.7596E4$	$2.7629E4$
		0.2255	$15222E-2$	1.1273	33.2765	1.0023
145086	2	$1.5754E4$	$1.5754E4$	$1.5753E4$	$1.5745E4$	$1.5754E4$
		0	$3.8348E-12$	0.5892	24.8969	$3.8348E-12$
	3	$1.5982E4$	$1.5982E4$	$1.5980E4$	$1.5955E4$	$1.5982E4$
		$2.3331E-2$	$1.9174E-12$	1.3726	31.9229	0.1242
	4	$1.6085E4$	$1.6085E4$	$1.6078E4$	$1.6054E4$	$1.6083E4$
		0	$1.9174E-12$	3.2000	19.4620	1.8526

Table 6: The mean CPU time of the compared EA-based methods on Otsu algorithm.

Dim	Algorithms				
	MCPSO-PSH	NrGA	CMA-ES	PSO	DE
2 dim	0.3398	0.3088	0.2802	0.2553	0.2478
3 dim	0.3588	0.3193	0.2206	0.2497	0.3170
4 dim	0.3221	0.3897	0.2264	0.2520	0.3418
5 dim	0.3087	0.3903	0.1991	0.2355	0.3481
7 dim	0.3021	0.3818	0.2068	0.2311	0.3583
9 dim	0.3406	0.3949	0.1932	0.2308	0.3390

Table 7: Objective value and standard deviation by the compared EA-based methods on Otsu algorithm.

Image	M	Objective values (standard deviation)				
		MCPSO-PSH	NrGA	CMA-ES	PSO	DE
Avion	5	3.4561E4	3.4561E4	3.4197E4	3.4547E4	3.4198E4
		2.2681$E-2$	6.6013$E-2$	2.2062	10.5393	1.2606
	7	3.4585E4	3.4585E4	3.4222E4	3.4576E4	3.4218E4
		0.1227	0.4133	1.3292	7.2486	3.1975
	9	3.4598E4	3.4597E4	3.4233E4	3.4580E4	3.4230E4
		0.2757	1.0760	1.1419	11.0739	2.1899
Lena	5	1.7853E4	1.7853E4	1.7598E4	1.7829E4	1.7603E4
		1.1762$E-2$	0.5437	1.3751	26.0899	1.9589
	7	1.7885E4	1.7885E4	1.7626E4	1.7857E4	1.7631E4
		0.5759	0.1317	1.9112	23.0333	4.2468
	9	1.7898E4	1.7898E4	1.7641E4	1.7882E4	1.7642E4
		0.7920	0.1250	2.3463	6.4354	3.2478
Hunter	5	9.5208E3	9.3860E3	9.3567E3	9.4591E3	9.3581E3
		1.3509	0.9022	3.0022	46.7651	3.9539
	7	9.5559E3	9.4292E3	9.3892E3	9.5000E3	9.3895E3
		0.132607	1.6085	3.6985	61.6729	3.6627
	9	9.5736E3	9.4400E3	9.4060E3	9.5248E3	9.4041E3
		0.661602	0.8640	2.6400	24.2906	3.0252
41033	5	1.2974E4	1.2974E4	1.2967E4	1.2938E4	1.2970E4
		1.7191$E-2$	0.1151	2.5816	19.4046	2.1793
	7	1.3006E4	1.3006E4	1.2995E4	1.2968E4	1.2995E4
		0.220026	0.1466	1.9642	12.1857	3.9443
	9	13021.1	1.3020E4	1.3013E4	1.3004E4	1.3012E4
		0.883857	1.2077	2.4069	5.8058	2.4286
62096	5	2.7658E4	2.7658E4	2.7654E4	2.7650E4	2.7654E4
		3.5718$E-2$	0.2520	1.0799	5.7383	2.2228
	7	2.7682E4	2.7682E4	2.7677E4	2.7669E4	2.7677E4
		0.1319	6.7590$E-2$	1.1261	5.7592	2.1289
	9	2.7695E4	2.7695E4	2.7689E4	2.7682E4	2.7689E4
		0.545201	0.4215	1.6069	9.1974	1.1693
145086	5	1.6143E4	1.6143E4	1.6134E4	1.6108E4	1.6139E4
		2.7021$E-2$	3.0505$E-2$	4.0455	43.2172	3.4949
	7	1.6195E4	1.6195E4	1.6184E4	1.6160E4	1.6184E4
		0.1298	5.3333$E-2$	3.5967	23.9921	2.7644
	9	1.6217E4	1.6217E4	1.6205E4	1.6192E4	1.6205E4
		0.4759	0.2663	2.7680	15.9752	2.4381

dynamically partition the population into many clusters and the number of clusters is changing frequently to implement information exchange among the different clusters. The proposed algorithm keeps a good balance between exploration and exploitation search abilities. As a result, the proposed algorithm showed promising results among the compared algorithms. Finally, we applied the proposed algorithm in the multilevel image segmentation problem. The proposed algorithm gets favorable results compared to the other compared methods. The use of the MCPSO-PSH algorithm in other real-world problems merits further research in our future work.

While this research is promising, there is a large margin of improvement for future research:

(1) The number of clusters C and the repartitioned cycle T are preset in this work. However, according to our preliminary experiments, different values of C and T will impact the results of several functions. C and T determine the range and rate of information exchange among species. For getting more suitable results of most test cases, we preset these two values. Therefore, how to adaptively adjust the values of C and T with regard to various functions is to be explored in the next stage.

(2) A nonrevisit strategy based on Binary Space Partitioning fitness tree is employed to partition the whole search space into several subregions according to

the history positions of members. According to our preliminary observation of particles' trajectory, most of the global best positions are local under a parent node or a small patch of leaf nodes in the fitness tree, especially in the later evolution period. Most of the nodes in fitness tree have a lack of vitality. Therefore, in the future work, we may modify the static fitness tree into a dynamic one with a node age or a lifecycle feature.

(3) We would also like to introduce the proposed dynamic multiswarm and nonrevisit strategies to other swarm-based algorithms. At the same time, applying MCPSO-PSH to address more real-world complex problems will also be studied in the future work.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the manuscript.

Acknowledgments

This research is partially supported by Liaoning Province Education Administration (L2015360, L2014473, and L2015372) and Tianjin Province Science and Technology Projects (16ZLZDZF00150) and National Natural Science Foundation of China (61603261, 61503373, 61602343, and 51607122).

References

[1] O. Castillo, H. Neyoy, J. Soria, P. Melin, and F. Valdez, "A new approach for dynamic fuzzy logic parameter tuning in ant colony optimization and its application in fuzzy control of a mobile robot," *Applied Soft Computing Journal*, vol. 28, pp. 150–159, 2015.

[2] M. Qiu, Z. Ming, J. Li, K. Gai, and Z. Zong, "Phase-change memory optimization for green cloud with genetic algorithm," *Institute of Electrical and Electronics Engineers. Transactions on Computers*, vol. 64, no. 12, pp. 3528–3540, 2015.

[3] S.-M. Guo and C.-C. Yang, "Enhancing differential evolution utilizing eigenvector-based crossover operator," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 1, pp. 31–49, 2015.

[4] A. A. A. Esmin, R. A. Coelho, and S. Matwin, "A review on particle swarm optimization algorithm and its variants to clustering high-dimensional data," *Artificial Intelligence Review*, vol. 44, no. 1, pp. 23–45, 2013.

[5] Z. Zhang, S. Su, Y. Lin, X. Cheng, K. Shuang, and P. Xu, "Adaptive multi-objective artificial immune system based virtual network embedding," *Journal of Network and Computer Applications*, vol. 53, pp. 140–155, 2015.

[6] F. van den Bergh, *An Analysis of Particle Swarm Optimizers*, University of Pretoria, Pretoria, South Africa, 2001.

[7] F. van den Bergh and A. P. Engelbrecht, "A cooperative approach to participle swam optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 225–239, 2004.

[8] X. Li and X. Yao, "Tackling high dimensional nonseparable optimization problems by cooperatively coevolving particle swarms," in *Proceedings of the Congress on Evolutionary Computation, CEC '9*, pp. 1546–1553, IEEE Computational Intelligence Magazine, Trondheim, Norway, May 2009.

[9] C. K. Goh, K. C. Tan, D. S. Liu, and S. C. Chiam, "A competitive and cooperative co-evolutionary approach to multi-objective particle swarm optimization algorithm design," *European Journal of Operational Research*, vol. 202, no. 1, pp. 42–54, 2010.

[10] J. Zhang and X. Ding, "A multi-swarm self-adaptive and cooperative particle swarm optimization," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 6, pp. 958–967, 2011.

[11] M. Hasanzadeh, M. R. Meybodi, and M. M. Ebadzadeh, "Adaptive cooperative particle swarm optimizer," *Applied Intelligence*, vol. 39, no. 2, pp. 397–420, 2013.

[12] Z.-H. Liu, J. Zhang, S.-W. Zhou, X.-H. Li, and K. Liu, "Coevolutionary particle swarm optimization using AIS and its application in multiparameter estimation of PMSM," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 1921–1935, 2013.

[13] S. X. Yang and C. H. Li, "A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 6, pp. 959–974, 2010.

[14] C.-F. Juang, C.-M. Hsiao, and C.-H. Hsu, "Hierarchical cluster-based multispecies particle-swarm optimization for fuzzy-system optimization," *IEEE Transactions on Fuzzy Systems*, vol. 18, no. 1, pp. 14–26, 2010.

[15] S.-Z. Zhao, P. N. Suganthan, Q.-K. Pan, and M. F. Tasgetiren, "Dynamic multi-swarm particle swarm optimizer with harmony search," *Expert Systems with Applications*, vol. 38, no. 4, pp. 3735–3742, 2011.

[16] X. Chu, M. Hu, T. Wu, J. . Weir, and Q. Lu, "AHPS2: an optimizer using adaptive heterogeneous particle swarms," *Information Sciences*, vol. 280, pp. 26–52, 2014.

[17] Q. Qin and S. Cheng, "Particle swarm optimization based semi-supervised learning on Chinese text categorization," *IEEE Transactions on Cybernetics*, vol. 46, no. 10, 2016.

[18] R. Brits, A. P. Engelbrecht, and F. van den Bergh, "Locating multiple optima using particle swarm optimization," *Applied Mathematics and Computation*, vol. 189, no. 2, pp. 1859–1883, 2007.

[19] D. Parrott and X. D. Li, "Locating and tracking multiple dynamic optima by a particle swarm model using speciation," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 4, pp. 440–458, 2006.

[20] X. Li, "Niching without niching parameters: particle swarm optimization using a ring topology," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 150–169, 2010.

[21] L. Wang, B. Yang, and Y. Chen, "Improving particle swarm optimization using multi-layer searching strategy," *Information Sciences*, vol. 274, pp. 70–94, 2014.

[22] M. Sumona and B. Santo, "Global optimization of an optical chaotic system by Chaotic Multi Swarm Particle Swarm Optimization," *Expert Systems with Applications*, vol. 39, no. 1, pp. 917–924, 2012.

[23] J. Li, J. Zhang, C. Jiang, and M. Zhou, "Composite particle swarm optimizer with historical memory for function optimization," *IEEE Transactions on Cybernetics*, vol. 45, no. 10, pp. 2350–2363, 2015.

[24] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the International Conference on Neural Networks (ICNN '95)*, vol. 4, pp. 1942–1948, IEEE, Perth, Western Australia, December 1995.

[25] J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley*

Symposium on Mathematical Statistics and Probability, pp. 281–297, University of California Press, Berkeley, Calif, USA, 1967.

[26] Y. Y. Shiu and K. C. Chi, "A genetic algorithm that adaptively mutates and never revisits," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 454–472, 2009.

[27] 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.

[28] C. Igel, N. Hansen, and S. Roth, "Covariance matrix adaptation for multi-objective optimization," *Evolutionary Computation*, vol. 15, no. 1, pp. 1–28, 2007.

[29] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[30] P. N. Suganthan, N. Hansen, J. J. Liang et al., "Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization," Technical Report 2005005, Nanyang Technological University, Singapore and KanGAL, (Kanpur Genetic Algorithms Laboratory, IIT Kanpur), 2005, pp. 1–50.

[31] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.

[32] S. Ouadfel and A. Taleb-Ahmed, "Social spiders optimization and flower pollination algorithm for multilevel image thresholding: a performance study," *Expert Systems with Applications*, vol. 55, pp. 566–584, 2016.

[33] X. Zhao, M. Turk, W. Li, K.-C. Lien, and G. Wang, "A multilevel image thresholding segmentation algorithm based on two-dimensional K–L divergence and modified particle swarm optimization," *Applied Soft Computing Journal*, vol. 48, pp. 151–159, 2016.

[34] http://decsai.ugr.es/cvg/dbimagenes/.

[35] http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/BSDS300/html/dataset/images.html.