Neurocomputing 334 (2019) 79–88

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

A multilevel sampling strategy based memetic differential evolution for multimodal optimization

Xi Wang[a], Mengmeng Sheng[a,d], Kangfei Ye[a], Jian Lin[a], Jiafa Mao[a], Shengyong Chen[a,c], Weiguo Sheng[b,*]

[a] School of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, PR China
[b] Department of Computer Science, Hangzhou Normal University, Hangzhou 311121, PR China
[c] College of Computer Science, Tianjin University of Technology, Tianjing 300384, PR China
[d] Zhejiang Police College, Hangzhou 310053, PR China

ARTICLE INFO

Article history:
Received 24 October 2018
Revised 20 December 2018
Accepted 6 January 2019
Available online 11 January 2019

Communicated by Nianyin Zeng

Keywords:
Multimodal optimization
Differential evolution
Niching
Crossover-based local search

ABSTRACT

Multimodal optimization, aiming to locate multiple optima in parallel, is a challenging task. In this paper, a multilevel sampling strategy based memetic differential evolution algorithm is proposed to tackle the problem. In the proposed algorithm, a multilevel sampling strategy is devised to sample a subpopulation for evolution at each generation. In this strategy, the entire population is dynamically divided into multiple levels according to the fitness of individuals at each generation. A subpopulation is then adaptively sampled from the individuals at different levels to undergo a niching based evolution for identifying multiple optima in the search space. Further, a crossover-based local search scheme is designed to fine-tune the seed solutions of niches in the population during evolution. We evaluate the proposed method on 20 benchmark multimodal problems and compare it with state-of-the-art multimodal optimization algorithms. The results show that our proposed algorithm can effectively and accurately locate multiple optima, outperforming related methods to be compared.

1. Introduction

Evolutionary algorithms (EAs) have been widely applied to various optimization problems [1–3] due to its robustness and easy implementation. EAs are typically designed to locate a single optimum in the solution space. However, many real world optimization problems usually involve multiple optima, which are referred to multimodal optimization problems (MMOPs). To address the MMOPs, simultaneously locating multiple optimal solutions is therefore required. Obviously, the traditional implementation of EAs cannot meet such a requirement [4–6]. To alleviate this issue, various schemes such as niching and multi-objective based EAs have been developed to deal with MMOPs [7–12]. Here, we focus on niching based EA scheme. The niching methods [13–15] work by forming multiple niches within a single population, and each subpopulation is used to locate one or multiple optimal solutions in the corresponding subspace, therefore allowing EAs to identify multiple optima simultaneously.

Although niching based EAs have been widely applied for MMOPs, effectively identifying the optima is still a challenging task, especially for the MMOP with a large number of optima. This is partially due to the niching based EAs need to address the contradiction of maintaining the identified optima and locating new optima during the evolution. Generally, maintaining more identified optima will require more computation resources of EAs and reduce their exploration capability to locate new optima. On the other hand, enhancing the exploration capability of EAs by allowing more individuals to explore the search space could lead to the loss of identified optima. To tackle this issue, two approaches are usually employed. The first one is to utilize an external archive to record the identified optima [16–18]. However, identifying and dealing with the optimal solutions in the evolving population is not a trivial task. Another approach is employing a large population. This approach could lead to an expensive evolution in order to recover the optima in the search space.

Apart from locating multiple optima, it is also important to deliver the optima with high accuracy. The traditional niching based EAs are good at global search, however, is not good at local search. This could significantly reduce the solution accuracy of recovered optima. To address such an issue, various local search operations have been devised and incorporated to improve the accuracy of

* Corresponding author.
E-mail address: w.sheng@ieee.org (W. Sheng).

https://doi.org/10.1016/j.neucom.2019.01.006
0925-2312/

solutions, resulting hybrid EAs called memetic algorithms (MAs) [19–22]. These local search operations usually carry with certain drawbacks. For instance, in [20], Vitela and Castanos employed a hill-climbing gradient-based local search algorithm to improve each individual in the population until convergence. This local search operation requires expensive calculation of the gradient of objective function. In [21], a local search is used to improve the personal best by adding a small difference vector, which is obtained based on the difference between the personal best and its closest personal best. Such a local search mechanism could be invalidated in the cases that the personal best and its closest personal best are located far away. In [22], Yang et al. employed a Gaussian sampling search to improve the seed solutions of niches. This local search mechanism does not involve any heuristics and, thus, could lead to an inefficient local search. In [23,24], crossover based local searches have been suggested to improve the search efficiency of EAs. The idea behind these local searches is that crossover operations produce offspring around their parents and thus can be suitably modified to possess certain local search property [25,26]. The idea also motivates us to devise a novel crossover-based local search for MMOPs in this work.

In this paper, we propose a multilevel sampling strategy based memetic differential evolution algorithm for MMOPs. In the proposed method, two mechanisms (i.e., a multilevel sampling strategy and a crossover based local search operation) have been devised with the purpose of effectively and accurately identifying the optima in search space. The multilevel sampling strategy works by, firstly, dynamically dividing the population into multiple levels according to the fitness of individuals at each generation. Then, a subpopulation is adaptively sampled from the individuals at different levels to support a balanced evolution for identifying multiple optima in the search space. While, the crossover based local search operation is designed and employed to fine-tune the seed solutions of niches during evolution. By incorporating these two mechanisms into a differential evolution algorithm along with a niching method as well as a parameter adaptation scheme, a multilevel sampling strategy based memetic differential evolution algorithm is finally proposed. In the experiments, we evaluate the performance of proposed algorithm on CEC'2013 multimodal function set and compare it with state-of-the-art multimodal optimization algorithms. The results clearly show the significance of proposed mechanisms in our algorithm for MMOPs. The results also show that our proposed algorithm is consistently outperforming state-of-the-art multimodal algorithms.

The main contributions of this work are therefore twofold:

- A multilevel sampling strategy, which is able to support a balanced evolution, has been proposed and incorporated into a niching DE to effectively search the space for identifying multiple optima;
- A crossover-based local search scheme, which is used to improve the accuracy of seed solution in the niches, has been devised to efficiently exploit the search space and accurately locate the optima.

2. Related work

2.1. Differential evolution algorithm

The differential evolution (DE) algorithm, proposed by Storn and Price [27], consists of mutation, crossover and selection operations to evolve the population. The population of DE contains *Np D*-dimensional real coded vectors

$$X_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,D}\} \tag{1}$$

where i=1, 2, ... , Np. At the initial stage of evolution, these vectors are initialized as

$$x_{i,j} = x_{\min,j} + rand(0,1) \cdot \left(x_{\max,j} - x_{\min,j}\right) \tag{2}$$

where $x_{\max,j}$ and $x_{\min,j}$ are upper and lower bounds, respectively, of jth gene of the solutions and the function $rand(0,1)$ is used to generate a random real number in the range of 0 to 1.

After initialization, a mutant is obtained by a mutation strategy for each individual X_i. Among the various mutation strategies [28], the DE/rand/1 has been used in this work, which is defined as

$$V_i = X_{r_1} + F \cdot (X_{r_2} - X_{r_3}) \tag{3}$$

where r_1, r_2, r_3 are chosen randomly from the set {1, 2, ..., Np}, such that $r_1 \neq r_2 \neq r_3 \neq i$, and F is a scaling factor.

Following the mutation, a crossover operation is then performed to produce trial vector. Typically, the trial vector U_i can be generated by applying binomial crossover on the X_i and V_i:

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if } (rand(0,1) \le CR) \text{ or } (j = j_{rand}) \\ x_{i,j} & \text{otherwise} \end{cases}, \tag{4}$$

where j_{rand} is a random integer selected from the set {1, 2, ..., Np}, and CR denotes the crossover probability. According to the above formula, at least one of the genes will be different between X_i and U_i.

Finally, a selection operation is performed to select an individual between X_i and U_i. The one with better fitness will be selected and undergo the evolution at next generation. The above evolution process will continue until a specified termination condition is met. The DE will be employed as the base for our proposed method.

2.2. Niching methods

Niching methods, which can be used to search multiple optima in parallel, have been widely incorporated into EAs for MMOPs. Some early works includes crowding [29,30,13], sharing [14,15], clearing [31,32], speciation [7], derating [33] and parallelization [34] etc. These methods generally involve certain niching parameters, which need to be properly configured in order to have a good performance. As the optimal values of these parameters depend the problem at hand and may vary during the evolution of EAs [35], setting them properly is a hard task.

Recently, to reduce the parameter dependency, several new niching strategies [4–6,8,9,22,36] have been proposed. Among the various strategies, the neighborhood-based niching technique is perhaps the most popular. This technique attempts to partition a single population into multiple subpopulations according to a specific neighborhood metric. Then, the mating of individuals is performed in the respective subpopulations to which they belong. The neighborhood metrics could be either index-based or distance-based. The index-based neighborhood strategy forms a neighborhood of an individual by using adjacent indices in the population. For example, Li [4] employed a ring topology structure between particles' personal memories to form a stable network in particle swarm optimization (PSO). In this method, the neighborhood structure is a ring topology based on particle's indices in which every particle move toward the best memory of its neighborhood. In [36], a mutation vector is generated based on the nearest neighbor of the index-based neighborhood of the individual. In [5], the population is divided into multiple equal size subpopulations according to an index-based metric. For the distance-based neighborhood strategy, the neighborhood is formed by using adjacent Euclidean distance between individuals. For instance, in [8], subpopulations are formed based on the Euclidean distances between individuals. Gao et al. [9] proposed a clustering based method for

niching, which partitions the population into multiple disjoint subpopulations and adopted a neighborhood mutation strategy to generate offspring. Specifically, in this method, the best individual is selected from the population as a seed to form a niche by including M-1 individuals, which are closest to the seed. These M individuals are then eliminated from the population. All the niches are formed according to the above process and the value of parameter M is set to be 5 in the method. After the population is completely divided, each niche is subsequently evolved independently based on DE operation. This method has been used in [6] to form niches. In this work, we will also employ it for niching purpose.

2.3. Archive methods for MMOPs

To prevent the loss of identified optima while searching for new optima in the search space, external archive approach has been employed. For example, Lung and Dumitrescu [37] employed an external archive to store potential optima while Epitropakis et al. [16] used a dynamic archive to maintain niches of optimal solutions. In [38], Zhang et al. introduced a generic archive technique, which is composed of subpopulation identification and convergence detection. In this method, when the convergence of a subpopulation is detected, the individuals in this subpopulation will be stored in an external archive and re-initialized. Lacroix et al. [17] designed an alternative external archive to store identified optima. The archive used in this method consists of two collections: the first one is used to store the identified optima while the second one contains a sorted index of regions represented by solutions stored in the first one. The regions listed in the index are prohibited from further searching. In [18], the archive was implemented as a hierarchical tree. The nodes at the top level are niche centers and the nodes below them are solutions belonging to corresponding niche identified during the entire search history. The nodes will be automatically merged based on the specified niching radius. The above methods show that the archive strategy can help prevent the loss of identified optima during the evolutionary search. However, identifying and dealing with optimal solutions in the evolving population is not a trivial task, which render the applicability of such an approach. In this paper, we propose a multilevel sampling strategy, which can drive a large population for simultaneously locating and maintaining optima while requires no archive.

2.4. Local search for MMOPs

It has been widely agreed that traditional niching based EAs have the difficulty to accurately identify the optima in search space. This is due to EAs are not good at local search. To improve the situation, various local search operations have been devised and incorporated into EAs for MMOPs. For example, in [20], a hill-climbing gradient-based local search algorithm is applied to each individual in the population to accelerate convergence to its nearest peak. Ni et al. [39] integrated two local search operators to accomplish different search tasks. Specifically, a chaotic local search is employed to the particle, which cannot improve its personal best by a comprehensive learning strategy to escape the local optima. While, a simulated annealing-based local search combined with the “cognition-only” model is devised to guide elite particles to search for promising areas. Qu et al. [21] introduced a particle’s personal best mutation based operation to enhance the local search capability of a multimodal PSO. Wang et al. [40] proposed an adaptive local search method, in which cognition-based local search (CBLS) [41] and random walk with direction exploitation (RWDE) [42] are employed cooperatively to improve the solutions. The CBLS in this method is used to guide a particle to search towards its personal best while the RWDE executes a random search around the current particle. In [43], Solis and Wets applied the algorithm [44] on new individuals with high fitness. In [45], the naive directed search algorithm [46] is employed as local search to improve the locally best solutions. In [17], a derandomized evolution strategy with covariance matrix adaptation (CMA-ES) [47] is incorporated to fine-tune the best individual in population. Yang et al. [22] proposed an adaptive local search scheme to improve the seeds of niches by performing a sampling search around their surrounding areas. In this paper, we proposed a crossover-based local search heuristic, which is also used to improve the seeds of niches.

Algorithm 1 A multilevel sampling strategy based memetic differential evolution algorithm for multimodal optimization problems.

Step 1. Generate an initial population P with Np solutions.
Step 2. Sampling a subset $subP$ from P using the multilevel sampling strategy (see Section 3.1).
Step 3. Employing a clustering based niching method [9] to partition the $subP$ into niches.
Step 4. For each niche:
  For each individual p in the same niche:
    (a) Generate the values of CR_p and F_p according to Eq. (9).
    (b) Produce a mutation vector v using Eq. (3).
    (c) Perform the crossover operator on the individual p and mutation vector v according to Eq. (4) to generate a new individual c.
    (d) Pair the individual c with the most similar solution in the current niche and replace it if the individual c has a better fitness.
    (e) If c wins the competition, then save values of CR_p and F_p into S_{CR} and S_F, respectively, as well as the absolute fitness difference Δf_p between the offspring c and its paired individual.
Step 5. Apply the crossover-based local search (see Section 3.3) to fine-tune seed solutions of niches.
Step 6. If S_{CR} and S_F are not empty, then update M_{CR} and M_F using Eq. (10).
Step 7. Terminate the evolution when the stopping condition is met. Otherwise go to Step 2.
Step 8. Output the optimal solutions in P.

3. Proposed algorithm

In this section, we propose a multilevel sampling strategy based memetic differential evolution algorithm (MMDE) for MMOPs. The proposed algorithm consists of a multilevel sampling strategy (denoted as MS, see Section 3.1), which is devised and employed to adaptively sample a subset of population at each generation. The selected subpopulation is then undergoing evolution based on a niching method [9]. Further, a crossover-based local search scheme (denoted as XLS, see Section 3.2) has been designed and used to fine-tune the seed solutions of niches during evolution. Additionally, a parameter adaptation mechanism (see Section 3.3) has also been adopted in the proposed method to control the DE parameters. The evolution will continue until the specified termination condition is met. Algorithm 1 shows the procedure of the proposed algorithm.

3.1. Multilevel sampling strategy

Here, we propose a multilevel sampling strategy, which can greatly improve the efficiency of evolution while properly searching the space of MMOPs. In the proposed strategy, the entire population is firstly divided into multiple levels according to the fitness of individuals at each generation. A subpopulation is then adaptively sampled from the individuals at different levels to support a balanced evolution for identifying multiple optima in the search space. The rational behind the proposed strategy is that, during evolution, the exploration and exploitation capability of EAs depends on the composition of population. For a population comprised with many relatively low fitness individuals, the evolutionary search will generally bias towards exploring the space. On the other hand, if most of the individuals in the population have rel-

Algorithm 2 A crossover-based local search.

Step 1. For an individual p to be improved, sampling tp according to a Cauchy distribution.
Step 2. Let the *winner* and *loser* to be the better and worse one of p and tp, respectively.
Step 3. Generate a mutation vector according to Eq. (8).
Step 4. Perform the binomial crossover operator on the *winner* and v to generate a new individual c.
Step 5. If c has a better fitness than the *winner*, then replace it.
Step 6. Output the *winner*.

atively high fitness, the evolutionary search will tend to exploit the space. Generally, to appropriately search the solution space, the evolution should focus more on exploration to discover potential optima in the space at the early stage of evolution. While at the later stage of evolution, the task of search should shift towards exploitation, thus accurately identifying the optima. Based on the above rational, it is therefore desirable to adaptively select a suitable subpopulation at each generation for evolution, thus efficiently and appropriately searching the space.

Specifically, the proposed multilevel sampling strategy works as follows. At each generation of evolution, we first sort the individuals in population in descending order according to their fitness. Then, the sorted population is divided equally into NL levels denoted as L_i ($i=0, 1, ..., NL-1$). The individuals with higher fitness will be assigned to a higher level, which has a smaller index. The subpopulation is finally sampled from these NL level individuals according to the following procedure. At L_i level, the sampling intensity p_i is calculated as follows:

$$p_i = \left\lceil \left(p_{i,begin} + \left(p_{i,end} - p_{i,begin} \right) \cdot \frac{FEs}{FE\_Max} \right) \right\rceil, \tag{5}$$

where FEs and FE_Max are consumed and total function evaluations, respectively. The initial sampling intensity $p_{i,begin}$ is calculated as

$$p_{i,begin} = \frac{i}{NL-1}, \tag{6}$$

while the final intensity $p_{i,end}$ is computed to be

$$p_{i,end} = 1 - p_{i,begin}. \tag{7}$$

It can be seen that, according to the above formulas, the individuals at the lower level will have a high possibility to be sampled at the beginning of evolution, thus encouraging exploration of the space to locate promising areas. While, along the advance of evolution, the individuals at higher level will be sampled with an increasing probability, thus strengthening the exploitation capability to accurately identify the optima at the later stage of evolution.

3.2. Crossover-based local search scheme

To improve the accuracy of recovered optima, here we proposed a crossover-based local search, which is employed to fine-tune the seed solutions of niches. The proposed local search works as follows. For a target seed solution p to be improved, a trial solution tp is sampled from the Cauchy distribution of p. Between p and tp, marking the one with better fitness as *winner* and the other as *loser*. Based these two individual, then generate a mutation vector v according to the following equation:

$$v_j = winner_j + random(0,1) \cdot \left(winner_j - loser_j \right). \tag{8}$$

From the Eq. (8), it can be seen that the local search is biased towards searching away from the side of *loser*. After that, a binomial crossover operator is performed on the *winner* and v to produce a new individual c. Let c to compete with the *winner* and replace the *winner* if c has a better fitness. Algorithm 2 shows the proposed local search procedure.

Table 1
Parameter settings.

Function	Max_FEs	Population size
F1–F5	5.0E+4	80
F6	2.0E+5	100
F7	2.0E+5	300
F8-F9	4.0E+5	300
F10	2.0E+5	100
F11–F13	2.0E+5	200
F14–F20	4.0E+5	200

3.3. DE parameter adaptation

Rather than using a fix parameter values, an adaptation mechanism presented in [48] has been adopted to dynamically set the parameter CR and F of DE in our proposed method. This mechanism maintains a historical memory of H entries for the DE parameters CR and F, denoted as M_{CR} and M_F, respectively. The sizes of both M_{CR} and M_F are set to be 100 (i.e., $H=100$). The values of M_{CR} and M_F are initialized to be 0.5. During the evolution, when generating offspring using the individual p, an index r_p is firstly selected randomly from [1, H], and then the F and CR for the p are calculated as

$$\begin{cases} CR_p = Gaussian\left(M_{CR,r_p}, 0.1\right) \\ F_p = Cauchy\left(M_{F,r_p}, 0.1\right) \end{cases}, \tag{9}$$

where Gaussian(•, 0.1) and Cauchy(•, 0.1) are Gaussian and Cauchy distribution, respectively, with variance of 0.1. The resulting value of CR_p will be truncated to between [0.0, 1.0], and the value of F_p will be regenerated or taken as 1.0 depending on whether it is less than 0.0 or greater than 1.0.

If the offspring generated using the values of CR_p and F_p win the competition against its paired individual, the CR_p and F_p values are recorded in S_{CR} and S_F. At the end of each generation, if both the S_{CR} and S_F are not empty, then the M_{CR} and M_F are updated as

$$\begin{cases} M_{CR,k} = mean_{WA}(S_{CR}) \\ M_{F,k} = mean_{WL}(S_F) \end{cases}, \tag{10}$$

where k ($1 \le k \le H$) is the position index in the M_{CR} and M_F, and is set to be 1 at the initialization stage. The value of k will be incremented by one after the M_{CR} and M_F are updated and will be reset to 1 when k is greater than H. The weighed mean $mean_{WA}(S_{CR})$ and weighted Lehmer mean $mean_{WL}(S_F)$ are calculated as

$$\begin{cases} mean_{WA}(S_{CR}) = \sum_{p=1}^{|S_{CR}|} w_p \cdot S_{CR,p} \\ mean_{WL}(S_F) = \frac{\sum_{p=1}^{|S_F|} w_p \cdot S_{F,p}^2}{\sum_{p=1}^{|S_F|} w_p \cdot S_{F,p}} \end{cases}, \tag{11}$$

where w_p is defined as

$$w_p = \frac{\Delta f_p}{\sum_{i=1}^{|S_{CR}|} \Delta f_i}. \tag{12}$$

Here, Δf_p denotes the absolute fitness difference between the offspring p and its paired individual.

4. Experiments

In this section, we carry out a series of experiments to evaluate the proposed method and compare it with state-of-the-art multimodal optimization algorithms. Before presenting experimental results, benchmark functions used in our experiments are firstly described. Then, the parameter configurations of our algorithm are given. The results are averaged over 100 independent runs of the algorithms.

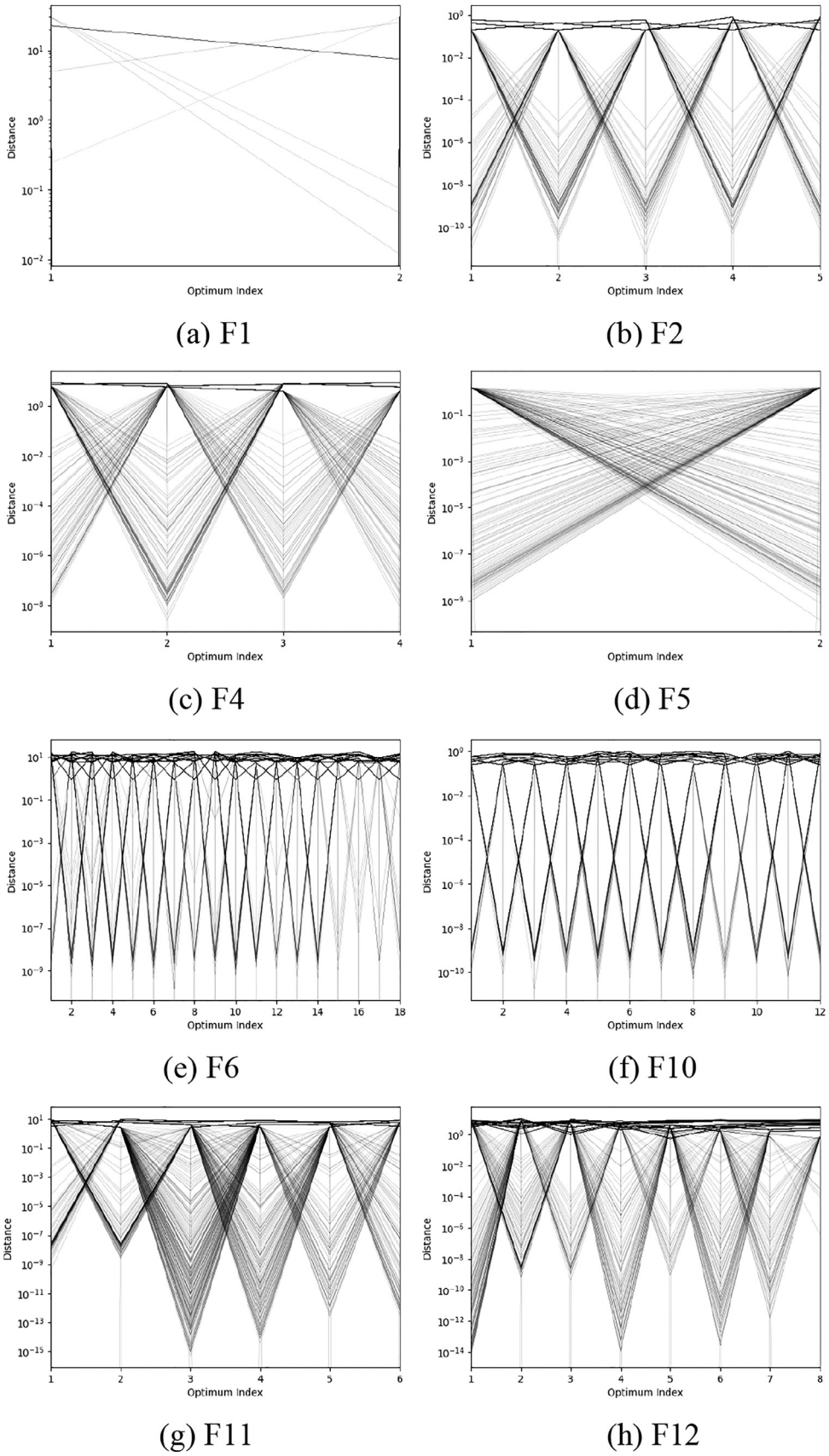


Fig. 1. Solution set obtained by MMDE on F1, F2, F4, F5, F6, F10, F11 and F12.

Table 2
Comparing results in PR among different versions of MMDE with best PR highlighted in bold.

ε	MMDE F1	MMDE_1	MMDE_2	MMDE F2	MMDE_1	MMDE_2	MMDE F3	MMDE_1	MMDE_2	MMDE F4	MMDE_1	MMDE_2
1.0E − 1	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
1.0E − 2	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
1.0E − 3	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
1.0E − 4	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
1.0E − 5	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
ε	F5			F6			F7			F8		
1.0E − 1	**1.000**	**1.000**	**1.000**	**1.000**	0.989	0.889	**0.917**	0.860	0.519	**0.996**	0.986	0.983
1.0E − 2	**1.000**	**1.000**	**1.000**	**1.000**	0.989	0.889	**0.917**	0.860	0.519	**0.995**	0.983	0.983
1.0E − 3	**1.000**	**1.000**	**1.000**	**1.000**	0.989	0.878	**0.917**	0.860	0.519	**0.991**	0.976	0.981
1.0E − 4	**1.000**	**1.000**	**1.000**	**1.000**	0.989	0.878	**0.917**	0.860	0.517	0.978	0.960	**0.980**
1.0E − 5	**1.000**	**1.000**	**1.000**	**1.000**	0.989	0.867	**0.914**	0.860	0.517	0.943	0.920	**0.972**
ε	F9			F10			F11			F12		
1.0E − 1	**0.463**	0.379	0.199	**1.000**	**1.000**	0.979	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.969
1.0E − 2	**0.463**	0.379	0.199	**1.000**	**1.000**	0.979	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.956
1.0E − 3	**0.463**	0.379	0.199	**1.000**	**1.000**	0.979	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	0.944
1.0E − 4	**0.463**	0.379	0.199	**1.000**	**1.000**	0.979	**1.000**	**1.000**	**1.000**	**1.000**	0.988	0.931
1.0E − 5	**0.463**	0.379	0.199	**1.000**	**1.000**	0.979	**1.000**	**1.000**	0.992	**1.000**	0.988	0.925
E	F13			F14			F15			F16		
1.0E − 1	0.700	0.675	**0.758**	**0.800**	0.700	0.717	0.756	0.756	**0.769**	**0.942**	0.892	0.825
1.0E − 2	0.667	0.667	**0.708**	**0.667**	**0.667**	**0.667**	**0.750**	**0.750**	0.744	**0.667**	**0.667**	**0.667**
1.0E − 3	0.667	0.667	**0.708**	**0.667**	**0.667**	**0.667**	**0.750**	**0.750**	0.738	**0.667**	**0.667**	**0.667**
1.0E − 4	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.750**	**0.750**	0.725	**0.667**	**0.667**	0.658
1.0E − 5	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.667**	**0.750**	**0.750**	0.713	**0.667**	**0.667**	0.625
ε	F17			F18			F19			F20		
1.0E − 1	**0.706**	0.656	0.606	**1.000**	**1.000**	0.433	**0.544**	0.519	0.344	**1.000**	**1.000**	0.269
1.0E − 2	**0.694**	0.625	0.500	**0.667**	0.658	0.375	**0.500**	**0.500**	0.294	0.256	**0.263**	0.244
1.0E − 3	**0.675**	0.619	0.456	**0.667**	0.658	0.367	**0.500**	**0.500**	0.263	0.244	**0.263**	0.244
1.0E − 4	**0.638**	0.619	0.425	**0.658**	**0.658**	0.358	**0.500**	**0.500**	0.263	0.013	**0.263**	0.244
1.0E − 5	0.594	**0.613**	0.369	0.650	**0.658**	0.358	**0.500**	**0.500**	0.250	0.006	0.106	**0.244**

Table 3
Comparing results in terms of PR and SR between MMDE and state-of-the-art multimodal methods on 20 functions at accuracy level ε = 1.0E−1. The best PR is highlighted in bold.

F	CDE		NCDE		NSDE		LoICDE		LoISDE		Self_CCDE		Self_CSDE		MMDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F2	**1.000**	1.000	**1.000**	1.000	0.396	0.000	**1.000**	1.000	0.252	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F3	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F4	**1.000**	1.000	**1.000**	1.000	0.250	0.000	**1.000**	1.000	0.250	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F5	**1.000**	1.000	**1.000**	1.000	0.500	0.000	**1.000**	1.000	0.500	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F6	**1.000**	1.000	0.969	0.660	0.056	0.000	0.984	0.780	0.056	0.000	**1.000**	1.000	0.444	0.000	**1.000**	1.000
F7	**0.997**	0.940	0.933	0.170	0.036	0.000	0.935	0.130	0.030	0.000	0.881	0.030	0.452	0.000	0.917	0.000
F8	0.008	0.000	**0.999**	0.950	0.012	0.000	0.838	0.000	0.012	0.000	**0.999**	0.900	0.187	0.000	0.996	0.700
F9	**0.503**	0.000	0.451	0.000	0.005	0.000	0.483	0.000	0.005	0.000	0.462	0.000	0.121	0.000	0.463	0.000
F10	**1.000**	1.000	**1.000**	1.000	0.083	0.000	**1.000**	1.000	0.087	0.000	**1.000**	1.000	0.994	0.930	**1.000**	1.000
F11	0.993	0.960	0.778	0.160	0.168	0.000	**1.000**	1.000	0.173	0.000	0.987	0.920	0.993	0.960	**1.000**	1.000
F12	0.159	0.000	0.706	0.020	0.125	0.000	0.810	0.120	0.125	0.000	0.968	0.790	0.805	0.140	**1.000**	1.000
F13	**0.932**	0.660	0.693	0.000	0.183	0.000	0.708	0.040	0.168	0.000	0.678	0.010	0.847	0.250	0.700	0.000
F14	**0.857**	0.420	0.750	0.140	0.173	0.000	0.722	0.050	0.167	0.000	0.752	0.080	0.673	0.000	0.800	0.250
F15	**0.970**	0.870	0.491	0.000	0.125	0.000	0.563	0.010	0.125	0.000	0.600	0.000	0.523	0.000	0.756	0.000
F16	**0.957**	0.920	0.905	0.570	0.157	0.000	0.928	0.670	0.165	0.000	0.940	0.730	0.605	0.000	0.942	0.700
F17	0.110	0.010	0.308	0.000	0.086	0.000	0.489	0.000	0.078	0.000	0.460	0.000	0.335	0.000	**0.706**	0.000
F18	0.648	0.260	0.960	0.850	0.087	0.000	0.872	0.650	0.148	0.000	0.950	0.790	0.300	0.000	**1.000**	1.000
F19	0.000	0.000	0.313	0.030	0.013	0.000	0.190	0.000	0.015	0.000	0.483	0.010	0.190	0.000	**0.544**	0.000
F20	0.129	0.080	0.281	0.000	0.006	0.000	0.129	0.000	0.074	0.000	0.528	0.010	0.173	0.000	**1.000**	1.000
bprs	13		7		2		7		2		8		5		13	

4.1. Benchmark functions and performance evaluation

The benchmark function set used in experiments is from CEC'2013 [49], which contains 20 functions. The characteristics of these functions can also be found in [49]. We evaluate the algorithms based on two measures, i.e., the peak ration (PR) and success rate (SR), which have been widely used to evaluate the performance of multimodal optimization algorithms. Given a fixed maximum number of function evaluations and a specified level of accuracy, the PR measures average percentage of identified optima out of all known optima in the search space over multiple runs, and the SR measures percentage of successful runs that all known

Table 4
Comparing results in terms of PR and SR between MMDE and state-of-the-art multimodal methods on 20 functions at accuracy level ε = 1.0E−2. The best PR is highlighted in bold.

F	CDE		NCDE		NSDE		LoICDE		LoISDE		Self_CCDE		Self_CSDE		MMDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F2	**1.000**	1.000	**1.000**	1.000	0.396	0.000	**1.000**	1.000	0.252	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F3	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F4	**1.000**	1.000	**1.000**	1.000	0.250	0.000	**1.000**	1.000	0.250	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F5	**1.000**	1.000	**1.000**	1.000	0.500	0.000	**1.000**	1.000	0.500	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F6	0.998	0.960	0.859	0.120	0.056	0.000	0.879	0.140	0.056	0.000	**1.000**	1.000	0.444	0.000	**1.000**	1.000
F7	0.887	0.020	0.869	0.000	0.036	0.000	0.890	0.000	0.030	0.000	0.872	0.020	0.452	0.000	**0.917**	0.000
F8	0.000	0.000	**0.999**	0.950	0.012	0.000	0.620	0.000	0.012	0.000	**0.999**	0.900	0.187	0.000	0.995	0.650
F9	0.475	0.000	0.449	0.000	0.005	0.000	**0.482**	0.000	0.005	0.000	0.451	0.000	0.121	0.000	0.463	0.000
F10	**1.000**	1.000	**1.000**	1.000	0.083	0.000	**1.000**	1.000	0.087	0.000	**1.000**	1.000	0.994	0.930	**1.000**	1.000
F11	0.667	0.000	0.678	0.000	0.168	0.000	**1.000**	1.000	0.173	0.000	0.953	0.740	0.993	0.960	**1.000**	1.000
F12	0.014	0.000	0.401	0.000	0.125	0.000	0.690	0.000	0.125	0.000	0.779	0.040	0.805	0.140	**1.000**	1.000
F13	0.588	0.000	0.667	0.000	0.183	0.000	0.667	0.000	0.168	0.000	0.667	0.000	**0.847**	0.250	0.667	0.000
F14	0.662	0.000	0.667	0.000	0.173	0.000	0.667	0.000	0.167	0.000	0.667	0.000	**0.668**	0.000	0.667	0.000
F15	0.231	0.000	0.333	0.000	0.125	0.000	0.444	0.000	0.125	0.000	0.471	0.000	0.521	0.000	**0.750**	0.000
F16	0.235	0.000	0.665	0.000	0.155	0.000	**0.667**	0.000	0.165	0.000	**0.667**	0.000	0.587	0.000	**0.667**	0.000
F17	0.000	0.000	0.245	0.000	0.085	0.000	0.301	0.000	0.078	0.000	0.260	0.000	0.319	0.000	**0.694**	0.000
F18	0.245	0.000	0.337	0.000	0.085	0.000	0.277	0.000	0.148	0.000	0.428	0.000	0.287	0.000	**0.667**	0.000
F19	0.000	0.000	0.169	0.000	0.008	0.000	0.141	0.000	0.015	0.000	0.320	0.000	0.185	0.000	**0.500**	0.000
F20	0.000	0.000	0.231	0.000	0.004	0.000	0.129	0.000	0.074	0.000	0.245	0.000	0.161	0.000	**0.256**	0.000
bprs	6		7		2		9		2		9		7		16	

Table 5
Comparing results in terms of PR and SR between MMDE and state-of-the-art multimodal methods on 20 functions at accuracy level ε = 1.0E−3. The best PR is highlighted in bold.

F	CDE		NCDE		NSDE		LoICDE		LoISDE		Self_CCDE		Self_CSDE		MMDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F2	**1.000**	1.000	**1.000**	1.000	0.396	0.000	**1.000**	1.000	0.252	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F3	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F4	**1.000**	1.000	**1.000**	1.000	0.250	0.000	**1.000**	1.000	0.250	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F5	**1.000**	1.000	**1.000**	1.000	0.500	0.000	**1.000**	1.000	0.500	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F6	0.862	0.140	0.686	0.000	0.056	0.000	0.669	0.000	0.056	0.000	0.998	0.970	0.444	0.000	**1.000**	1.000
F7	0.843	0.000	0.866	0.000	0.036	0.000	0.889	0.000	0.030	0.000	0.872	0.020	0.452	0.000	**0.917**	0.000
F8	0.000	0.000	**0.999**	0.950	0.012	0.000	0.428	0.000	0.012	0.000	**0.999**	0.900	0.187	0.000	0.991	0.450
F9	0.473	0.000	0.448	0.000	0.005	0.000	**0.482**	0.000	0.005	0.000	0.451	0.000	0.121	0.000	0.463	0.000
F10	**1.000**	1.000	0.998	0.980	0.083	0.000	**1.000**	1.000	0.087	0.000	**1.000**	1.000	0.994	0.930	**1.000**	1.000
F11	0.660	0.000	0.672	0.000	0.168	0.000	0.997	0.980	0.173	0.000	0.882	0.460	0.990	0.940	**1.000**	1.000
F12	0.001	0.000	0.213	0.000	0.125	0.000	0.540	0.000	0.125	0.000	0.633	0.000	0.805	0.140	**1.000**	1.000
F13	0.297	0.000	0.662	0.000	0.183	0.000	0.667	0.000	0.168	0.000	0.667	0.000	**0.847**	0.250	0.667	0.000
F14	0.508	0.000	**0.667**	0.000	0.173	0.000	**0.667**	0.000	0.167	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000
F15	0.088	0.000	0.301	0.000	0.125	0.000	0.411	0.000	0.125	0.000	0.380	0.000	0.520	0.000	**0.750**	0.000
F16	0.017	0.000	0.663	0.000	0.153	0.000	**0.667**	0.000	0.165	0.000	**0.667**	0.000	0.582	0.000	**0.667**	0.000
F17	0.000	0.000	0.245	0.000	0.085	0.000	0.274	0.000	0.078	0.000	0.253	0.000	0.308	0.000	**0.675**	0.000
F18	0.185	0.000	0.335	0.000	0.085	0.000	0.260	0.000	0.148	0.000	0.405	0.000	0.285	0.000	**0.667**	0.000
F19	0.000	0.000	0.124	0.000	0.008	0.000	0.139	0.000	0.015	0.000	0.246	0.000	0.180	0.000	**0.500**	0.000
F20	0.000	0.000	0.231	0.000	0.000	0.000	0.129	0.000	0.074	0.000	0.235	0.000	0.158	0.000	**0.244**	0.000
bprs	6		7		2		9		2		9		7		17	

optima are found. Five different accuracy levels, i.e., ε = {1.0E−1, 1.0E−2, 1.0E−3, 1.0E−4, 1.0E−5} has been used to report the results.

To make a fair comparison, the maximum number of function evaluations (*Max_FEs*) and the population size (*Np*) are set to be the same for all algorithms as shown in Table 1. The population size of our proposed method and its variants is set to 3**Np*, due to the employment of multilevel sampling strategy. The number of level *NL* in the proposed MS strategy is set to be 15. The size of niches *M* is set to be 5. In the XLS mechanism, the scale parameter of Cauchy distribution and crossover probability are set to be 1.0E−4 and 0.5, respectively.

4.2. Exploring the MMDE

First, the significance of the MS and XLS mechanisms in proposed algorithm is examined. We therefore compare our proposed algorithm with its variants: MMDE without XLS (denoted as MMDE_1) and MMDE without XLS and MS (denoted as MMDE_2). The proposed algorithm and its variants are compared using the same parameter configurations, except the MMDE_2 in which a population size of *Np* is used due to no MS being employed. Table 2 shows the results in terms of PR of the three algorithms. By employing the method proposed by Cheng et al. [50], some representative results have also been visually shown in Fig. 1, compared to optimal points in the relevant test functions.

As can be seen from the results, the MS is able to greatly enhance the performance of the proposed algorithm. By incorporating the MS mechanism, MMDE_1 can generally locate more optima than MMDE_2. By comparing the MMDE and MMDE_1, it can also be found that the XLS can help to recover more optima in the search space. Specifically, the MMDE gives better and comparable PR values than MMDE_1 on all of the functions, except function F20. From the above results, it can be concluded that the

Table 6
Comparing results in terms of PR and SR between MMDE and state-of-the-art multimodal methods on 20 functions at accuracy level ε = 1.0E−4. The best PR is highlighted in bold.

F	CDE		NCDE		NSDE		LoICDE		LoISDE		Self_CCDE		Self_CSDE		MMDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F2	**1.000**	1.000	**1.000**	1.000	0.396	0.000	**1.000**	1.000	0.252	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F3	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F4	**1.000**	1.000	**1.000**	1.000	0.250	0.000	**1.000**	1.000	0.250	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F5	**1.000**	1.000	**1.000**	1.000	0.500	0.000	**1.000**	1.000	0.500	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F6	0.411	0.000	0.510	0.000	0.056	0.000	0.474	0.000	0.056	0.000	0.998	0.970	0.444	0.000	**1.000**	1.000
F7	0.606	0.000	0.860	0.000	0.036	0.000	0.880	0.000	0.030	0.000	0.872	0.020	0.452	0.000	**0.917**	0.000
F8	0.000	0.000	**0.999**	0.950	0.012	0.000	0.282	0.000	0.012	0.000	**0.999**	0.900	0.187	0.000	0.978	0.150
F9	0.404	0.000	0.447	0.000	0.005	0.000	**0.472**	0.000	0.005	0.000	0.451	0.000	0.121	0.000	0.463	0.000
F10	**1.000**	1.000	0.996	0.950	0.083	0.000	**1.000**	1.000	0.087	0.000	**1.000**	1.000	0.994	0.930	**1.000**	1.000
F11	0.330	0.000	0.670	0.000	0.168	0.000	0.990	0.940	0.173	0.000	0.852	0.340	0.990	0.940	**1.000**	1.000
F12	0.000	0.000	0.133	0.000	0.125	0.000	0.371	0.000	0.125	0.000	0.546	0.000	0.803	0.140	**1.000**	1.000
F13	0.067	0.000	0.648	0.000	0.183	0.000	0.667	0.000	0.168	0.000	0.667	0.000	**0.847**	0.250	0.667	0.000
F14	0.183	0.000	**0.667**	0.000	0.173	0.000	**0.667**	0.000	0.167	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000
F15	0.011	0.000	0.283	0.000	0.125	0.000	0.393	0.000	0.125	0.000	0.360	0.000	0.518	0.000	**0.750**	0.000
F16	0.000	0.000	0.660	0.000	0.153	0.000	**0.667**	0.000	0.165	0.000	**0.667**	0.000	0.573	0.000	**0.667**	0.000
F17	0.000	0.000	0.244	0.000	0.084	0.000	0.265	0.000	0.078	0.000	0.250	0.000	0.299	0.000	**0.638**	0.000
F18	0.168	0.000	0.328	0.000	0.085	0.000	0.250	0.000	0.148	0.000	0.382	0.000	0.283	0.000	**0.658**	0.000
F19	0.000	0.000	0.094	0.000	0.006	0.000	0.136	0.000	0.015	0.000	0.180	0.000	0.175	0.000	**0.500**	0.000
F20	0.000	0.000	**0.231**	0.000	0.000	0.000	0.129	0.000	0.074	0.000	0.190	0.000	0.153	0.000	0.013	0.000
bprs	6		8		2		9		2		9		7		16	

Table 7
Comparing results in terms of PR and SR between MMDE and state-of-the-art multimodal methods on 20 functions at accuracy level ε = 1.0E−5. The best PR is highlighted in bold.

F	CDE		NCDE		NSDE		LoICDE		LoISDE		Self_CCDE		Self_CSDE		MMDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F1	**1.000**	**1.000**	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F2	**1.000**	**1.000**	**1.000**	1.000	0.396	0.000	**1.000**	1.000	0.252	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F3	**1.000**	**1.000**	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F4	0.815	0.260	**1.000**	1.000	0.250	0.000	**1.000**	1.000	0.250	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F5	**1.000**	1.000	**1.000**	1.000	0.500	0.000	**1.000**	1.000	0.500	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
F6	0.099	0.000	0.341	0.000	0.056	0.000	0.313	0.000	0.056	0.000	0.994	0.920	0.444	0.000	**1.000**	1.000
F7	0.266	0.000	0.851	0.000	0.036	0.000	0.826	0.000	0.030	0.000	0.872	0.020	0.452	0.000	**0.914**	0.000
F8	0.000	0.000	**0.999**	0.950	0.012	0.000	0.176	0.000	0.012	0.000	**0.999**	0.900	0.187	0.000	0.943	0.000
F9	0.135	0.000	0.445	0.000	0.005	0.000	0.424	0.000	0.005	0.000	0.451	0.000	0.121	0.000	**0.463**	0.000
F10	**1.000**	1.000	0.993	0.920	0.083	0.000	0.999	0.990	0.087	0.000	**1.000**	1.000	0.994	0.930	**1.000**	1.000
F11	0.058	0.000	0.668	0.000	0.168	0.000	0.973	0.840	0.173	0.000	0.835	0.270	0.990	0.940	**1.000**	1.000
F12	0.000	0.000	0.098	0.000	0.125	0.000	0.214	0.000	0.125	0.000	0.466	0.000	0.803	0.140	**1.000**	1.000
F13	0.008	0.000	0.628	0.000	0.183	0.000	0.667	0.000	0.168	0.000	0.665	0.000	**0.847**	0.250	0.667	0.000
F14	0.018	0.000	**0.667**	0.000	0.173	0.000	**0.667**	0.000	0.167	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000
F15	0.000	0.000	0.275	0.000	0.125	0.000	0.379	0.000	0.125	0.000	0.358	0.000	0.518	0.000	**0.750**	0.000
F16	0.000	0.000	0.660	0.000	0.153	0.000	0.662	0.000	0.165	0.000	**0.667**	0.000	0.568	0.000	**0.667**	0.000
F17	0.000	0.000	0.244	0.000	0.084	0.000	0.256	0.000	0.078	0.000	0.250	0.000	0.295	0.000	**0.594**	0.000
F18	0.163	0.000	0.327	0.000	0.080	0.000	0.245	0.000	0.148	0.000	0.363	0.000	0.275	0.000	**0.650**	0.000
F19	0.000	0.000	0.080	0.000	0.005	0.000	0.135	0.000	0.015	0.000	0.131	0.000	0.170	0.000	**0.500**	0.000
F20	0.000	0.000	**0.231**	0.000	0.000	0.000	0.129	0.000	0.074	0.000	0.139	0.000	0.153	0.000	0.006	0.000
bprs	5		8		2		6		2		10		7		17	

MS mechanism help to balance the exploration and exploitation of evolutionary search, thus effectively identifying the optima in solution space. While XLS helps to accurately recover the optima in search space. By incorporating these two mechanisms, the resulting algorithm is therefore able to effectively and accurately address the MMOPs.

4.3. Comparisons with state-of-the-art algorithms

Then, we compare our proposed algorithm with several state-of-the-art multimodal optimization algorithms. The algorithms to be compared include crowding-based DE (CDE) [51], neighborhood based CDE (NCDE) and neighborhood based speciation DE (NSDE) proposed in [8], locally informative crowding-based DE (LoICDE) and locally informative speciation-based DE (LoISDE) given by Biswas et al. [52] as well as cluster-based CDE with self-adaptive strategy (Self_CCDE) and cluster-based SDE with self-adaptive strategy (Self_CSDE) provided in [9]. Before discussing the comparison results, we firstly give a brief description of the algorithms to be compared. The CDE [51] is a crowding based DE for MMOPs. The NCDE [8] also employs a crowding DE, but incorporated with a newly designed neighborhood mutation. While the NSDE [8] relies on a species niching method based DE to search multimodal space. In the LoICDE and LoISDE [52], a local information sharing based neighborhood mutation strategy is incorporated into a crowding and species based niching method, respectively, for MMOPs. On the other hand, the Self_CCDE and Self_CSDE [9] incorporate a variant of crowding and species niching method, respectively, into DE along with a self-adaptive parameter setting strategy for MMOPs. To make the comparisons meaningful, the same *Max_FEs* and *Np* values are used for all algorithms. The rest parameters of the algorithms are specified according to the original source papers.

Tables 3–7 show the comparison results of the methods in terms of PR and SR at different levels of accuracy. The term "*bprs*" in the tables is used to account the number of functions that the

corresponding algorithm obtains the best PR results across all the functions and algorithms. As can be seen from Tables 3-7 that, in terms of *bprs*, the proposed algorithm is generally able to more effectively identify the optima in the solution space comparing with the seven methods. Particularly, it can also be observed that the MMDE can be more effective on higher accuracy level of experiments. For example, for the CDE, NCDE, NSDE, LoICDE, LoISDE, Self_CCDE and Self_CSDE at level of accuracy of 1.0E−5, the value of *bprs* turns out to be 5, 8, 2, 6, 2, 10 and 7, respectively. By comparison, the MMDE achieves a much better value of 17. By closely examining the results, we can find that the MMDE locate all known optima on functions F1–F6 and F10–F12. Further, on functions F7, F15, F17–F19, the MMDE can always give the best performance at all levels of accuracy expect the 1.0E−1. From the above results, it is clear that our proposed algorithm performs the best among the eight methods to be compared.

5. Conclusions

In this paper, we have reported the implementation of a DE based evolution for MMOPs. In the proposed method, the DE is incorporated with a multilevel sampling strategy and a crossover-based local search scheme for MMOPs. The multilevel sampling strategy is devised with the purpose of appropriately balancing the exploration and exploitation of evolutionary search, thus effectively identify the optima in the search space. While the crossover-based local search scheme is designed and employed to improve accuracy of seeds in niches. The performance of proposed method has been evaluated via a series of experiments. The results show that proposed mechanisms are able to greatly improve the niching based DE for addressing the MMOPs and the resulting algorithm outperform state-of-the-art algorithms to be compared.

The work in this paper can be certainly extended further. For example, it would be desirable to employ the proposed multilevel sampling strategy to deal with large-scale optimization problems [53] and multi-objective optimization problems [54]. It is also interesting to employ an alternative sampling strategy to achieve a balanced evolution with large population as well as limited computational resources.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant nos.61573316, 61873082 and U1509207 and Natural Science Foundation of Zhejiang Province under Grant nos. LY18F030023 and LY17F020017.

References

[1] N. Zeng, H. Zhang, Y. Chen, B. Chen, Y. Liu, Path planning for intelligent robot based on switching local evolutionary PSO algorithm, Assem. Autom. 36 (2) (2016) 120–126.
[2] N. Zeng, H. Zhang, W. Liu, J. Liang, F. Alsaadi, A switching delayed PSO optimized extreme learning machine for short-term load forecasting, Neurocomputing 240 (2017) 175–182.
[3] N. Zeng, H. Qiu, Z. Wang, W. Liu, H. Zhang, Y. Li, A new switchingdelayed-PSO-based optimized svm algorithm for diagnosis of alzheimer's disease, Neurocomputing 320 (2018) 195–202.
[4] X. Li, Niching without niching parameters: particle swarm optimization using a ring topology, IEEE Trans. Evolut. Comput. 14 (1) (2010) 150–169.
[5] Y. Gong, J. Zhang, Y. Zhou, Learning multimodal parameters: a bare-bones niching differential evolution approach, IEEE Trans. Neural Netw. Learn. Syst. 29 (7) (2018) 2944–2959.
[6] Z. Wang, Z. Zhan, Y. Lin, W. Yu, H. Yuan, T. Gu, S. Kwong, J. Zhang, Dual-strategy differential evolution with affinity propagation clustering for multimodal optimization problems, IEEE Trans. Evolut. Comput. 22 (6) (2018) 894–908.
[7] J. Li, M. Balazs, G. Parks, P. Clarkson, A species conserving genetic algorithm for multimodal function optimization, Evolut. Comput. 10 (3) (2002) 207–234.
[8] B. Qu, P. Suganthan, J. Liang, Differential evolution with neighborhood mutation for multimodal optimization, IEEE Trans. Evolut. Comput. 16 (5) (2012) 601–614.
[9] W. Gao, G. Yen, S. Liu, A cluster-based differential evolution with selfadaptive strategy for multimodal optimization, IEEE Trans. Cybern. 14 (8) (2014) 1314–1327.
[10] R. Cheng, M. Li, K. Li, X. Yao, Evolutionary multiobjective optimizationbased multimodal optimization: Fitness landscape approximation and peak detection, IEEE Trans. Evolut. Comput. 22 (5) (2018) 692–706.
[11] Y. Wang, H. Li, G. Yen, W. Song, Mommop: multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems, IEEE Trans. Cybern. 45 (4) (2015) 830–843.
[12] A. Basak, S. Das, K.C. Tan, Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection, IEEE Trans. Cybern. 17 (5) (2013) 666–685.
[13] G. Harik, Finding multimodal solutions using restricted tournament selection, in: Proceedings of the International Conference on Genetic Algorithms, 1995, pp. 24–31.
[14] D. Goldberg, J. Richardson, Genetic algorithms with sharing for multimodal function optimization, in: Proceedings of the International Conference on Genetic Algorithms, 1987, pp. 41–49.
[15] A. Cioppa, C. Stefano, A. Marcell, Where are the niches? dynamic fitness sharing, IEEE Trans. Evolut. Comput. 11 (4) (2007) 453–465.
[16] M. Epitropakis, X. Li, E. Burke, A dynamic archive niching differential evolution algorithm for multimodal optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2013, pp. 79–86.
[17] B. Lacroix, D. Molina, F. Herrera, Region-based memetic algorithm with archive for multimodal optimisation, Inf. Sci. 367 (C) (2016) 719–746.
[18] S. Kalra, S. Rahnamayan, K. Deb, Enhancing clearing-based niching method using delaunay triangulation, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2017, pp. 2328–2337.
[19] P. Moscato, Memetic algorithms: a short introduction, New Ideas in Optimization, McGraw-Hill, 1999, pp. 219–234.
[20] J. Vitela, O. Castanos, A real-coded niching memetic algorithm for continuous multimodal function optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2008, pp. 2170–2177.
[21] B. Qu, J. Liang, P. Suganthan, Niching particle swarm optimization with local search for multi-modal optimization, Inf. Sci. 197 (2012) 131–143.
[22] Q. Yang, W. Chen, Y. Li, C. Chen, X. Xu, J. Zhang, Multimodal estimation of distribution algorithms, IEEE Trans. Cybern. 47 (3) (2017) 636–650.
[23] S. Kazarlis, S. Papadakis, I. Theocharis, V. Petridis, Microgenetic algorithms as generalized hill-climbing operators for ga optimization, IEEE Trans. Evolut. Comput. 5 (3) (2001) 204–217.
[24] M. Lozano, F. Herrera, N. Krasnogor, D. Molina, Real-coded memetic algorithms with crossover hill-climbing, Evolut. Comput. 12 (3) (2004) 273–302.
[25] K. Deb, A. Anand, D. Joshi, A computationally efficient evolutionary algorithm for real-parameter evolution, Evolut. Comput. 10 (4) (2002) 371–395 495.
[26] M. Dietzfelbinger, B. Naudts, C.V. Hoyweghen, I. Wegener, The analysis of a recombinative hill-climber on H-IFF, IEEE Trans. Evolut. Comput. 7 (5) (2003) 417–423.
[27] R. Storn, K. Price, Differential evolution: a simple and efficient heuristic for global optimization over continues spaces, J. Glob. Optim. 11 (4) (1997) 341–359.
[28] S. Das, P. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evolut. Comput. 15 (1) (2011) 4–31.
[29] K. Jong, An analysis of the behavior of a class of genetic adaptive systems, Ph.D. dissertation, University of Michigan.
[30] S. Mahfoud, Crowding and Preselection Revisited, in: Proceedings of Parallel Problem Solving from Nature PPSN-II, 1992, pp. 27–36.
[31] A. Petrowski, A clearing procedure as a niching method for genetic algorithms, in: Proceedings of the IEEE Congress on Evolutionary Computation, 1996, pp. 798–803.
[32] G. Singh, K. Deb, Comparisons of multi-modal optimization algorithms based on evolutionary algorithms, in: Proceedings of the Conference on Genetic and Evolutionary Computation, 2006, pp. 1305–1312.
[33] D. Beasley, D. Bull, R. Martin, A sequential niche technique for multimodal function optimization, Evolut. Comput. 1 (2) (1993) 101–125.
[34] M. Bessaou, A. Petrowski, P. Siarry, Island model cooperating with speciation for multimodal optimization, in: Proceedings of the International Conference on Parallel Problem Solving from Nature, Springer-Verlag, 2000, pp. 437–446.
[35] X. Li, M. Epitropakis, K. Deb, A. Engelbrecht, Seeking multiple solutions: an updated survey on niching methods and their applications, IEEE Trans. Evolut. Comput. 21 (4) (2017) 518–538.
[36] M. Epitropakis, V. Plagianakos, M.N. Vrahatis, Multimodal optimization using niching differential evolution with index-based neighborhoods, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2012, pp. 1–8.
[37] R. Lung, D. Dumitrescu, A new evolutionary model for detecting multiple optima, in: Proceedings of the Genetic and Evolutionary Computation Conference, 2007, pp. 1296–1303.
[38] Y. Zhang, Y. Gong, W. Chen, Z. Zhan, J. Zhang, A generic archive technique for enhancing the niching performance of evolutionary computation, in: Proceedings of the IEEE Symposium on Swarm Intelligence, 2014, pp. 1–8.
[39] J. Ni, L. Li, F. Qiao, Q. Wu, A novel memetic algorithm based on the comprehensive learning PSO, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2012, pp. 1–8.
[40] H. Wang, N. Wang, D. Wang, A memetic particle swarm optimization algorithm for multimodal optimization problems, in: Proceedings of the Chinese Control and Decision Conference, 2011, pp. 3839–3845.
[41] J. Kennedy, The particle swarm: social adaptation of knowledge, in: Proceed-

ings of the IEEE International Conference on Evolution Computation, 1997, pp. 303–308.

[42] Y. Petalas, K. Parsopoulos, M. Vrahatis, Memetic particle swarm optimization, Ann. Oper. Res. 156 (1) (2007) 99–127.

[43] Z. Ren, A. Zhang, C. Wen, Z. Feng, A scatter learning particle swarm optimization algorithm for multimodal problems, IEEE Trans. Cybern. 44 (7) (2014) 1127–1140.

[44] F. Solis, R. Wets, Minimization by random search techniques, Math. Oper. Res. 6 (1) (1981) 19–30.

[45] A. Sharifi, J. Kordestani, M. Mahdaviani, M. Meybodi, A novel hybrid adaptive collaborative approach based on particle swarm optimization and local search for dynamic optimization problems, Appl. Soft Comput. 32 (2015) 432–448.

[46] A. Sharifi, V. Noroozi, M. Bashiri, A.B. Hashemi, M.R. Meybodi, Two phased cellular PSO: a new collaborative cellular algorithm for optimization in dynamic environments, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2012, pp. 1–8.

[47] N. Hansen, S.D. Muller, P. Koumoutsakos, Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES), Evolut. Comput. 1 (11) (2003) 1–8.

[48] R. Tanabe, A. Fukunaga, Success-history based parameter adaptation for differential evolution, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2013, pp. 71–78.

[49] X. Li, A. Engelbrecht, M.G. Epitropakis, Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization, Evolutionary Computation Machine Learning Group, RMIT University, Melbourne, VIC, Australia, Technical Report. http://goanna.cs.rmit.edu.au/~xiaodong/cec13-lsgo/competition/.

[50] R. Cheng, M. Li, X. Yao, Parallel peaks: a visualization method for benchmark studies of multimodal optimization, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2017, pp. 263–270.

[51] R. Thomsen, Multimodal optimization using crowding-based differential evolution, in: Proceedings of the IEEE Congress on Evolutionary Computation, 2, 2004, pp. 1382–1389.

[52] S. Biswas, S. Kundu, S. Das, Inducing niching behavior in differential evolution through local information sharing, IEEE Trans. Evolut. Comput. 19 (2) (2015) 246–263.

[53] R. Cheng, Y. Jin, A competitive swarm optimizer for large scale optimization, IEEE Trans. Cybern. 45 (2) (2015) 191–204.

[54] L. Li, W. Wang, X. Xu, Multi-objective particle swarm optimization based on global margin ranking, Inf. Sci. 375 (2017) 30–47.

Xi Wang received the B.Sc. Degree in computer science in 2016 from Zhejiang University of Technology, Hangzhou, China, where he is currently pursuing his M.Sc. degree in Computer Science. His current research interests mainly focus on evolutionary computation and data mining.

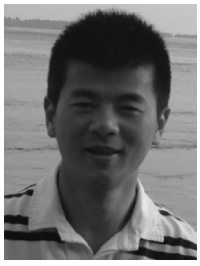
Mengmeng Sheng received the M.Sc. Degree in computer science from the University of Hong Kong, Hong Kong, in 2008. Then, she worked at China Telecom as a Data Engineer. In August 2013, she moved to Zhejiang Police College, China, where she is a lecturer in Computer Science. Currently, she is also pursuing her Ph.D. degree in Computer Science at Zhejiang University of Technology, Hangzhou, China. Her research interests focus on data mining, machine learning and computer vision.

Kangfei Ye received the B.Sc. Degree in computer science in 2016 from Zhejiang University of Technology, Hangzhou, China, where he is currently pursuing his M.Sc. degree in Computer Science. His research interests focus on evolutionary computation and data clustering.

Jian Lin received the B.Sc. Degree in computer science in 2016 from Zhejiang University of Technology, Hangzhou, China, where he is currently a postgraduate student in Computer Science. His research interests focus on machine learning and data mining.

Jiafa Mao received the Ph.D. degree in pattern recognition from East China University of Science and Technology, China, in 2009. Since then, he has worked as a Post-doc Researcher at Beijing University of Posts and Telecommunications, China. In July 2011, he joined Zhejiang University of Technology, China, where he is currently a Professor in the School of Computer Science & Technology. His research interests include pattern recognition, digital image processing and information hiding. He has published over 40 papers in the scientific literature.

Shengyong Chen (M'01, SM'10) received the Ph.D. degree in computer vision from City University of Hong Kong, Hong Kong, in 2003. He joined Zhejiang University of Technology, China, in Feb. 2004, where he is currently a Professor in the Department of Computer Science. He is also working as a Professor at Tianjin University of Technology, China. He received a fellowship from the Alexander von Humboldt Foundation of Germany and worked at University of Hamburg in 2006–2007. His research interests include computer vision, robotics, and image analysis. He is a Fellow of IET, a senior member of IEEE, and a committee member of IET Shanghai Branch. He has published over 100 scientific papers in international journals and conferences. He received the National Outstanding Youth Foundation Award of China in 2013.



Weiguo Sheng received the M.Sc. degree in information technology from the University of Nottingham, U.K., in 2002 and the Ph.D. degree in computer science from Brunel University, U.K., in 2005. Then, he worked as a Researcher at the University of Kent, U.K. and Royal Holloway, University of London, U.K. He is currently a Professor at Hangzhou Normal University. His research interests include evolutionary computation, data mining/clustering, pattern recognition and machine learning.