Dual-Strategy Differential Evolution With Affinity Propagation Clustering for Multimodal Optimization Problems

Zi-Jia Wang, *Student Member, IEEE*, Zhi-Hui Zhan, *Member, IEEE*, Ying Lin, *Member, IEEE*, Wei-Jie Yu, *Member, IEEE*, Hua-Qiang Yuan, Tian-Long Gu, Sam Kwong, *Fellow, IEEE*, and Jun Zhang, *Fellow, IEEE*

***Abstract*—Multimodal optimization problem (MMOP), which targets at searching for multiple optimal solutions simultaneously, is one of the most challenging problems for optimization. There are two general goals for solving MMOPs. One is to maintain population diversity so as to locate global optima as many as possible, while the other is to increase the accuracy of the solutions found. To achieve these two goals, a novel dual-strategy differential evolution (DSDE) with affinity propagation clustering (APC) is proposed in this paper. The novelties and advantages of DSDE include the following three aspects. First, a dual-strategy mutation scheme is designed to balance exploration and exploitation in generating offspring. Second, an adaptive selection mechanism based on APC is proposed to choose diverse individuals from different optimal regions for locating as many peaks as possible. Third, an archive technique is applied to detect and protect stagnated and converged individuals. These individuals are stored in the archive to preserve the found promising solutions and are reinitialized for exploring more new areas. The experimental results show that the proposed DSDE algorithm is better than or at least comparable to the state-of-the-art multimodal algorithms when evaluated on the benchmark problems from CEC2013, in terms of locating more global optima, obtaining higher accuracy solution, and converging with faster speed.**

Manuscript received January 11, 2017; revised June 2, 2017 and August 31, 2017; accepted October 19, 2017. Date of publication November 17, 2017; date of current version November 28, 2018. This work was supported in part by the National Natural Science Foundations of China under Grant 61772207, Grant 61402545, and Grant 61332002, in part by the Natural Science Foundations of Guangdong Province for Distinguished Young Scholars under Grant 2014A030306038, in part by the Project for Pearl River New Star in Science and Technology under Grant 201506010047, in part by the GDUPS (2016), in part by the Fundamental Research Funds for the Central Universities, and in part by the Hong Kong RGC General Research Fund under Grant 9042038 and Grant CityU 11205314. *(Corresponding authors: Zhi-Hui Zhan; Jun Zhang).*

Z.-J. Wang, Z.-H. Zhan, Y. Lin, W.-J. Yu, and J. Zhang are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China and also with the Guangdong Provincial Key Laboratory of Computational Intelligence and Cyberspace Information, China (e-mail: zhanapollo@163.com; junzhang@ieee.org).

H.-Q. Yuan is with the School of Computer Science and Network Security, Dongguan University of Technology, Dongguan 523808, China.

T.-L. Gu is with the School of Computer Science and Engineering, Guilin University of Electronic Technology, Guilin 541004, China.

S. Kwong is with the Department of Computer Science, City University of Hong Kong, Hong Kong.

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2017.2769108

***Index Terms*—Affinity propagation clustering (APC), archive technique, differential evolution (DE), dual-strategy differential evolution (DSDE), multimodal optimization problems (MMOPs).**

I. Introduction

MANY real-world optimization problems require finding as many optimal solutions as possible, such as electromagnetic optimization [1], [2], data mining [3]–[5], power system [6], pattern recognition [7], [8], and protein structure prediction [9]. These problems are commonly known as multimodal optimization problems (MMOPs), which are more challenging than the traditional unimodal optimization problems. Over the past decades, MMOP has drawn considerable attention [10]–[13]. Owing to the success of evolutionary algorithms (EAs) [14]–[26], many efforts have been paid to extend EAs to MMOPs [27]–[29]. However, since traditional EAs are designed to search for a single optimal solution, modifications and multimodality-specific mechanisms are necessary for EAs to locate multiple optima simultaneously.

Many multimodal algorithms follow the framework in Fig. 1. The population is initialized and evaluated similar to traditional EAs. However, during the evolutionary process, to tackle MMOPs, the "niching" techniques have been proposed to maintain the population diversity, so as to locate multiple peaks [10]–[13], [27]–[29]. The working principle of niching is to partition the whole population into several subpopulations. There are many kinds of niching methods, such as speciation [10]–[13], crowding [28], [29], fitness sharing [30], clustering [31], [32], hill-valley [33], recursive middling [34], topological species conservation [35], and dynamic cluster size niching [36]. The algorithm in this paper adopts the dynamic cluster size niching [36], since the strategy is able to find a good balance between exploration and exploitation.

After the partition, the evolutionary operators are executed within subpopulations to generate offspring. Then, the fitness values of offspring are obtained and a new population is formed by a selection operator. As shown in Fig. 1, two kinds of selection operators are widely used in multimodal algorithms. One is the combination selection that first combines the N parents and the N offspring (N is the population size), then

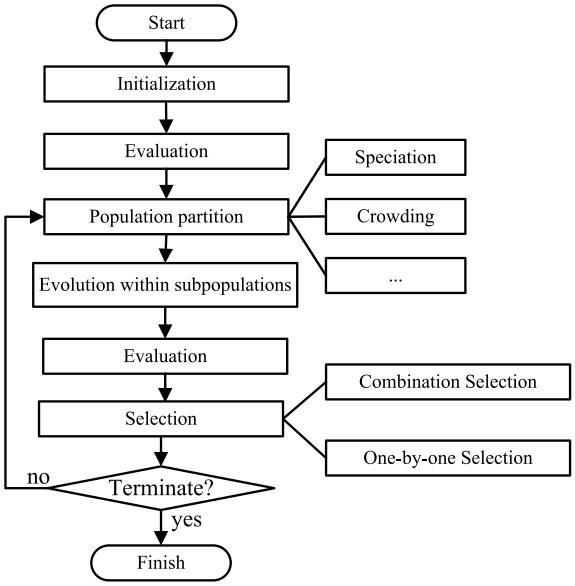


Fig. 1. Flowchart of multimodal algorithm framework.

selects the N best ones from the $2N$ individuals. The other is the one-by-one selection that compares each offspring's fitness value with its nearest parent (measured by Euclidean distance), and then replaces the parent if the offspring is better.

Under the framework of Fig. 1, many EAs have been successfully extended to solve MMOP, like genetic algorithm [11], [30], [37], ant colony optimization [38], estimation of distribution algorithm (EDA) [36], particle swarm optimization (PSO) [39], [40], and differential evolution (DE) [10], [29], [31], [32], [41]–[47]. Among these EAs, the DE variants have shown promising performance [10], [29], [31], [32], [41]–[47]. Therefore, we focus on the DE-based multimodal algorithm in this paper.

Despite the efforts put into utilizing EAs/DEs to solve MMOP, there are still some drawbacks of the current multimodal algorithms as follows.

1) It is a dilemma to choose or design an appropriate reproduction scheme that favors both exploration (finding more globally optimal regions) and exploitation (refining solution in each globally optimal region). For example in a DE-based algorithm, a mutation scheme with high randomness focuses on exploration, while a greedy mutation scheme concentrates on exploitation. However, this leads to deficiency when tackling MMOP, since the algorithms not only require high diversity for exploring multiple global optima, but also need fast convergence to refine the solutions in each globally optimal region.
2) Regarding the selection operator, it is strange that almost all the existing multimodal algorithms do not consider selecting individuals according to different peaks. If we partition the population before the evolutionary operators, why not partition the evolved population for better selection? The existing methods like combination selection and one-by-one selection both have their disadvantages. For example, suppose that the landscape of the problem and the distribution of population are as shown in Fig. 2. In Fig. 2(a), there are five global optima. If we use the combination selection, some peaks might be missed due to the strong selection pressure. That is, among the combined population denoted by the five circles and five stars, the five circles located in only two peaks may be selected. Alternatively, if we use the one-by-one selection in Fig. 2(b) that contains three global optima and two local optima, some local optima may also be selected. That is, the two circles in the two local optimal regions may also be selected, which may mislead the evolution.
3) Moreover, most multimodal algorithms are not aware of the convergence situation of different search regions. If some individuals have stagnated and converged in a specific region, it is not necessary to allocate fitness evaluations (FEs) to these converged individuals.

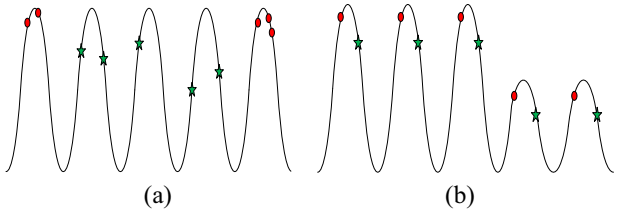


Fig. 2. Two examples of problem landscape and distribution of population. (a) Case of combination selection. (b) Case of one-by-one selection.

To solve these drawbacks, a dual-strategy DE (named DSDE) with affinity propagation clustering (APC) based selection and archive technique is proposed. Specifically, the three major novel characteristics and advantages that help DSDE balance diversity and convergence for locating more global peaks and increasing solution accuracy are described as follows.

1) In the individual evolution, a dual-strategy mutation scheme is adopted, which enables each individual to choose its own suitable mutation strategy specifically to meet the exploration and exploitation search requirements of its own. This way, the entire population is able to achieve a good balance between locating as many peaks as possible by exploring and refining the solution accuracy by exploiting.
2) In the selection, an adaptive probabilistic selection mechanism based on APC is proposed. By partitioning the population and choosing suitable individuals from different clusters to match different peaks in the landscape, the selection is good at diversity preservation. Meanwhile, we can select high-quality individuals that are close to the global optima by using a novel probabilistic model.
3) Moreover, an archive technique is proposed to detect and protect the converged individuals. These converged individuals will be reinitialized to increase the population diversity for searching more areas. Meanwhile, to preserve and avoid losing the found optima by reinitialization, these converged individuals are stored in the archive at the same time.

Experiments are performed on the 20 benchmark multimodal functions from the CEC2013 test suite. The results

are better than, or at least comparable to those obtained by the state-of-the-art multimodal algorithms, showing the effectiveness of our algorithms.

The rest of this paper is organized as follows. Section II reviews the basic DE and its variants for multimodal optimization. Section III gives a detailed description of the proposed DSDE algorithm. Experimental setup and results are presented and compared in Section IV. Finally, conclusions and future works are given in Section V.

II. DE AND MMOP

A. DE

DE realizes the evolution of individuals according to the difference between individuals. The initial population is randomly generated according to a uniform distribution in the search space as

$$x_{i,j} = L_j + \text{rand} \times \left(U_j - L_j\right) \tag{1}$$

where L_j and U_j are the lower and upper bounds of the jth dimension, rand is a random number in the range of [0, 1]. After initialization, DE updates the population of individuals via mutation, crossover, and selection operators. The three operators constitute the main loop of DE until the algorithm terminates.

1) Mutation: In each generation, a mutation vector is obtained for each individual based on the difference of other individuals. Four frequently used mutation strategies are listed below.

1) DE/rand/1

$$\boldsymbol{v}_i = \boldsymbol{x}_{r1} + F \times (\boldsymbol{x}_{r2} - \boldsymbol{x}_{r3}). \tag{2}$$

2) DE/best/1

$$\boldsymbol{v}_i = \boldsymbol{x}_{\text{best}} + F \times (\boldsymbol{x}_{r1} - \boldsymbol{x}_{r2}). \tag{3}$$

3) DE/current-to-best/1

$$\boldsymbol{v}_i = \boldsymbol{x}_i + F \times (\boldsymbol{x}_{\text{best}} - \boldsymbol{x}_i) + F \times (\boldsymbol{x}_{r1} - \boldsymbol{x}_{r2}). \tag{4}$$

4) DE/current-to-rand/1

$$\boldsymbol{u}_i = \boldsymbol{x}_i + \text{rand} \times (\boldsymbol{x}_{r1} - \boldsymbol{x}_i) + F \times (\boldsymbol{x}_{r2} - \boldsymbol{x}_{r3}) \tag{5}$$

where $r1$, $r2$, and $r3$ are the indexes of three different individuals randomly selected from the population. The parameter F is a positive real control parameter called the amplification factor, which controls the amplification of the difference vectors. $\boldsymbol{x}_{\text{best}}$ is the individual with the best fitness. Note that (5) for the "DE/current-to-rand/1" mutation actually includes a rotationally invariant arithmetic line recombination operator and therefore generates $\boldsymbol{u}_i$ directly.

2) Crossover: After mutation, DE generally performs a binomial crossover operation that exchanges some components from $\boldsymbol{x}_i$ and $\boldsymbol{v}_i$ to form a trial vector $\boldsymbol{u}_i$, with each dimension determined as

$$u_{i,j} = \begin{cases} v_{i,j} & \text{if rand} \le \text{CR or } j = j_{\text{rand}} \\ x_{i,j} & \text{otherwise} \end{cases} \tag{6}$$

where j_{rand} is an integer uniformly selected from $\{1, 2, \ldots, D\}$ to make sure that at least one dimension of $\boldsymbol{u}_i$ comes from $\boldsymbol{v}_i$.

Algorithm 1 Speciation Clustering

Begin
1. Sort the population from better to worse according to fitness value;
2. **For** i=1 to N/M //N is the population size, M is the cluster size
3. The best individual $\boldsymbol{X}$ is set as the species seed;
4. The $\boldsymbol{X}$ and its nearest $M-1$ individuals form a new species;
5. Remove the M individuals from the population;
6. **End of For**

End

Algorithm 2 Crowding Clustering

Begin
1. Randomly generate a reference point $\boldsymbol{R}$ in the search space;
2. **For** $i = 1$ to N/M //N is the population size, M is the cluster size
3. Find the nearest individual $\boldsymbol{X}$ to $\boldsymbol{R}$ in the population;
4. The $\boldsymbol{X}$ and its nearest $M-1$ individuals form a new crowd;
5. Remove the M individuals from the population;
6. **End of For**

End

The crossover rate CR is another parameter, which decides the proportion of the trial vector inherited from the mutation vector $\boldsymbol{v}_i$.

3) Selection: This operator selects the better individual from the trial vector $\boldsymbol{u}_i$ and the original vector $\boldsymbol{x}_i$ to enter the next generation. For example, for a maximization problem, the vector with a higher fitness value is selected into the next generation, which can be expressed by

$$\boldsymbol{x}_i = \begin{cases} \boldsymbol{u}_i, & \text{if } f(\boldsymbol{u}_i) \ge f(\boldsymbol{x}_i) \\ \boldsymbol{x}_i, & \text{otherwise} \end{cases} \tag{7}$$

where $f(\boldsymbol{x})$ is the FE function for a solution $\boldsymbol{x}$.

B. MMOP

MMOP aims at simultaneously locating the global optima as many as possible. This requires the algorithm to maintain higher diversity so that it can locate different global optima. Moreover, due to the limited FEs budget, the algorithm should also have fast convergence speed to increase the accuracy of solutions in each globally optimal region. Nowadays, niching techniques have been widely used and incorporated into EAs for solving MMOPs. Speciation [10]–[13] and crowding [28], [29] are two of the most famous niching methods.

In speciation, the population is divided into some species (subpopulations). Each subpopulation is formed by a species seed and the neighbors within the niching radius r. When combined with DE, the algorithm is named species-based DE (SDE) [10].

In crowding, each offspring is compared with its nearest parent from a crowd formed by randomly selecting C individuals. Then, the offspring will replace the compared parent if it is better. Otherwise, the offspring will be ignored. When combined with DE, the algorithm is named crowding DE (CDE) [29].

Although these two niching strategies have shown their effectiveness in SDE [10] and CDE [29], respectively, they both have two drawbacks. First, their performance is very sensitive to the niching parameters (i.e., the species radius r in speciation and the crowding size C in crowding). Second, when the dimensionality or the complexity of problem increases, leading to a huge number of local optima, the

feasibility of these strategies is severely impaired. These two limitations have restricted their applications.

Recently, research on improving these two niching techniques has drawn considerable attention. For instance, in order to enhance partition, Qu *et al.* [31] and Gao *et al.* [32] have integrated clustering methods with niching strategies. The basic clustering framework for speciation and crowding niching are shown in Algorithms 1 and 2, respectively. These algorithms are called neighborhood mutation-based DE (NSDE and NCDE) [31] and self-adaptive clustering-based DE (Self-CSDE and Self-CCDE) [32]. However, these algorithms also introduce a new parameter M, which is the neighborhood size (cluster size). To reduce the influence of this parameter, Biswas *et al.* [41] developed an improved local information sharing mechanism in niching DE (LoINDE, including LoISDE and LoICDE), where the neighborhood size is gradually reduced in a nonlinear manner. They also presented a new mutation strategy called parent-centric mutation operator in niching DE (PNPCDE) [47]. In [36], the clustering niching methods are incorporated with EDA, termed as LMSEDA and LMCEDA, where the cluster size is dynamically changed from a certain interval, which can find a potential balance between diversity and convergence. Parameter-free niching strategies are also developed, such as hill-valley niching technique [33], which detects hill valleys by sampling and evaluating some intermediate points between two individuals. However, to get more accurate detection, enough points should be sampled and evaluated, which will cause consumption of extra FEs.

Furthermore, various traditional EAs are also presented with other techniques to tackle MMOPs. For instance, Li [39] proposed RPSO (including R2PSO and R3PSO) by using ring topology according to particles' indexes in PSO for niching. Qu *et al.* [40] brought up a distance-based locally informed PSO (LIPS), which uses the information provided by particles' neighborhoods in terms of Euclidean distance. Moreover, some researchers have utilized multiobjective techniques to tackle MMOPs. In [37] and [48], the MMOP was converted into a bi-objective optimization problem, where the first objective is the fitness function itself, and the second objective is generated according to gradient. Wang *et al.* [49] designed two conflict objectives in each dimension to solve MMOPs, termed as MOMMOP.

However, when using DE or other EAs for multimodal optimization, it still confronts various difficulties. The algorithms are required to locate all the global optima, to avoid getting trapped in the local optima, and to refine the solution accuracy of each optimal region. These difficulties are especially challenging when dealing with problems with high dimension or complexity. Thus, how to find a balance between exploration (finding more globally optimal regions) and exploitation (refining solution in each globally optimal region) in DE or other EAs is still greatly desirable.

III. DSDE for MMOP

In order to deal with MMOPs efficiently, two major goals need to be considered and achieved: 1) diversity, which helps discover more regions containing the global optima and 2) convergence, which helps increase the accuracy of the solutions in each globally optimal region. To accomplish these two goals, our proposed DSDE uses an innovative approach to find a balance between exploration and exploitation. DSDE follows the multimodal algorithm framework discussed in Fig. 1, together with the following novel characteristics and advantages.

First, we design a dual-strategy mutation scheme for different individuals to generate offspring according to their different search requirements. Second, we propose a new probabilistic selection mechanism based on APC. On the one hand, the selection operator based on clustering can preserve population diversity by choosing suitable individuals from different clusters for locating different peaks. On the other hand, the probabilistic model can enhance the solution accuracy and choose more suitable individuals close to the global optimum. Third, the archive technique is used to detect converged and stagnant individuals so as to make good use of limited FEs.

Before we introduce DSDE, we first describe the dynamic cluster sizing in [36], which is also utilized in our DSDE, as the speciation cluster niching method shown in Algorithm 1. Specifically, in every generation, an integer is chosen randomly from a fixed interval as the cluster size M, while the number of niches is N/M. Note that if $N\%M \neq 0$, the last niche will have $M + N\%M$ individuals. The interval for M is set as [4, 20], due to the fact that DE must have at least four individuals. This scheme is not only intuitive but also effective and readily applicable to a wide range of uses. For example, if the current niche gets trapped in the local optima, the cluster size has a chance to become larger to improve the niching diversity. In contrast, if multiple peaks are found, the cluster size can be lowered to enhance the capability for exploitation. As a result, we can achieve a potential balance between population diversity and fast convergence.

A. Dual-Strategy Individual Evolution

After partitioning the whole population into several subpopulations, we subdivide each subpopulation into two equal parts according to individuals' fitness. The fitness division criterion can be readily implemented and can efficiently divide individuals into two distinct parts. The first part, namely "superior individuals", consists of individuals that are suitable for exploitation, while the second part, "inferior individuals," contains individuals that are suitable for exploration. The idea of the dual mutation strategy based on the two parts of individuals is illustrated in Fig. 3 for a maximization problem, where the set A contains the superior individuals with better fitness values, while the set B contains the inferior individuals with poorer fitness values. If the current individual belongs to the superior individuals, e.g., the individual A' in set A, it should further exploit its neighborhood to speed up the convergence and to refine the solutions accuracy. On the contrary, if the current individual belongs to inferior individuals, e.g., the individual B' in set B, it should explore further areas in the search space and maintain population diversity.

1) Scheme 1-Using DE/lbest/1 if the Current Individual Is Superior: DE/lbest/1 is derived from DE/best/1 in (3).

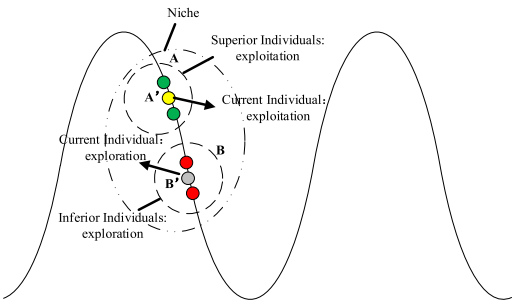


Fig. 3. Illustration of dual-strategy.

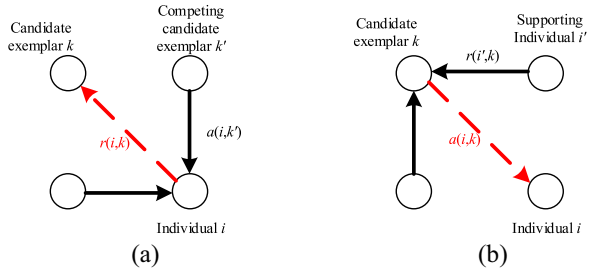


Fig. 4. Message-passing in APC. (a) Sending responsibilities. (b) Sending availabilities.

Since the population is divided into several subpopulations, DE/lbest/1 uses the local best vector from the subpopulation instead of the entire population. The superior individuals have good fitness, therefore, DE/lbest/1 can keep their good performance on exploitation, which can exploit the neighborhoods of the superior individuals to accelerate the convergence speed and enhance local search ability.

2) Scheme 2-Using DE/current-to-rand/1 if the Current Individual Is Inferior: As for the inferior individuals, there is no need to exploit their neighborhoods. But they can be used to explore further areas. DE/current-to-rand/1 shows good diversity [50] and is suitable for inferior individuals. Thus, using DE/current-to-rand/1 can ensure the diversity of the population and explore more peaks.

B. APC-Based Selection

As we wish to find as many distinct global peaks as possible in multimodal optimization, it is prudent to select individuals according to different peaks. Thus, we can use the clustering method to partition the entire population into a number of nonoverlapping groups. Each group will focus on converging to one or a small number of optima.

Motivated by this observation, the selection mechanism based on clustering is proposed for the first time. After generating N individuals, we first execute the clustering operator on the combined population. However, some traditional clustering methods, such as k-means clustering [51], [52] are quite sensitive to the predefined parameters like the number of clusters. Herein, we take a quite different clustering approach, named APC, which does not need to determine the number of clusters and simultaneously considers all individuals as potential exemplars [53].

APC is a relatively new clustering approach proposed by Frey and Dueck [53] in *Science*. It has also been applied in EAs [54]. Instead of requiring the number of clusters to be predefined, the clusters in APC emerge automatically by a message-passing procedure. There are two kinds of message communicated between individuals, responsibility and availability. The "responsibility" $r(i, k)$ is sent from each individual i to a candidate exemplar individual k. It shows the suitable degree of individual k to be the exemplar for individual i. The "availability" $a(i, k)$ is sent from a candidate exemplar individual k to individual i. It reflects how appropriate it is for individual i to choose individual k as its exemplar. Particularly, $r(i, k)$ is associated with the availabilities sent from other potential exemplars to individual i, as illustrated in Fig. 4(a). Similarly, $a(i, k)$ is related to the responsibilities sent from other supporting individuals to the candidate exemplar individual k, as shown in Fig. 4(b). In the beginning, the $a(i, k)$ is set to 0. Then in every iteration, the current responsibilities and availabilities are computed using the rules

$$r(i,k) = s(i,k) - \max_{k \neq k'} \{a(i,k') + s(i,k')\} \tag{8}$$

$$a(i,k) = \min\left\{0, r(k,k) + \sum_{i' \neq i,k} \max\{0, r(i',k)\}\right\} \tag{9}$$

where $s(i, k)$ is the similarity between individuals i and k, which is set as the negative squared error (Euclidean distance): $s(i,k) = -||\boldsymbol{x}_i - \boldsymbol{x}_k||^2$.

The current values will combine with the values in the last iteration to preserve and utilize the message from last iteration so as to avoid numerical oscillations. That is, during the information transmission process, each message is set as λ times of its value in the last iteration plus $(1-\lambda)$ times of its currently updated value

$$r(i,k) = (1-\lambda) \times r(i,k) + \lambda \times r(i,k)_{\text{last}} \tag{10}$$

$$a(i,k) = (1-\lambda) \times a(i,k) + \lambda \times a(i,k)_{\text{last}}. \tag{11}$$

For each individual i, find the k that maximizes $a(i, k) + r(i, k)$. If $k = i$, individual i is identified as an exemplar. Otherwise, the individual k is identified as the exemplar for individual i. The message-passing procedure may be terminated after a fixed maximum iterations *max_its*, or when the estimated exemplars stay constant for a predefined number of iterations *con_its*. In all of our experiments, we use the fixed parameters $\lambda = 0.9$, *max_its* $= 100$, and *con_its* $= 30$. Herein, the λ with relatively large value will keep more message from last iteration, making the clustering results more stable and effectively avoid numerical oscillations. Moreover, relatively smaller *max_its* and *con_its* values make the computational burden of APC relatively light. For more details of APC, please refer to [53].

After executing the clustering operator on the combined population, we will have several clusters, so called subpopulations. In each subpopulation, we first sort the individuals in decreasing order according to the individuals' fitness values. Then, each subpopulation will have a species seed with the best fitness. We always select and move the best individual from each subpopulation to the population of the next generation.

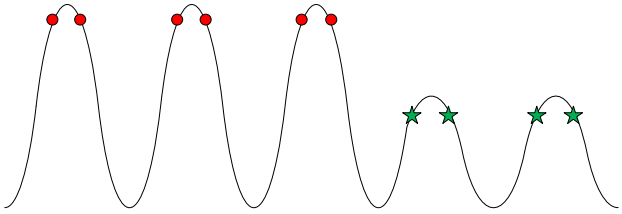


Fig. 5. Example of the population distribution after clustering, the circle individuals are more preferred than the star individuals.

It is obvious that fitter species seeds are more likely to be close to the global optima and thus should have a larger opportunity to be chosen. For example, suppose that the landscape of the problem is as Fig. 5, which contains three global optima and two local optima. The circles represent the individuals which have higher fitness values and are close to the global optima, while the stars indicate the individuals which are close to the local optima. Apparently, we prefer to choose the circles rather than the stars. In order to do this, we proposed a probabilistic model for the selection operator, where the probability of carrying out the selection operator on a subpopulation is associated with the fitness of its species seed. Mathematically, the probability P_i for the ith subpopulation can be formulated as

$$P_i = \frac{f_i - f_{\min} + \phi}{f_{\max} - f_{\min} + \phi} \tag{12}$$

where f_i is the fitness value of the species seed in the ith subpopulation, $f_{\max}$ and $f_{\min}$ are the maximum and minimum fitness values of all the species seeds. To meet the case where $f_{\min} = f_{\max}$, φ is utilized here and is set as 1E−4.

Next, the selection is performed in a queueing manner, as illustrated in Fig. 6. Different from the deterministic queueing selection proposed in [55], our operator is probabilistically based on the above-defined P_i. In Fig. 6, each rectangle represents a subpopulation, and different shapes denote different individuals, which are sorted in decreasing order of their fitness values. The arrows provide the choosing sequences of the subpopulation in each round. If the subpopulation i is not empty, we generate a random value in [0, 1]. If the value is smaller than the probability P_i, the first individual in the subpopulation i (i.e., the one with the best fitness) will be selected and moved to the next generation. Otherwise, no individual will be selected from this subpopulation in this round. For example, in Fig. 6(a), the label "Y" means the individual is selected and moved to the next generation, while the label "N" means the individual is temporarily ignored. After the first round of the selection, the remaining individuals are shown in Fig. 6(b). Then, we continue to select the remaining best individuals in each subpopulation based on the probability P_i. The selection process repeats until N individuals are selected. There are two advantages of our selection operator to solve MMOPs.

1) Picking individuals from different subpopulations can help maintain the population diversity, which can choose the individuals from different peaks to match the multiple peaks search requirements. Besides, we always choose the top-ranking individuals in each subpopulation, the quality and accuracy of selected individuals is guaranteed. In other words, we can find a balance between diversity and convergence.
2) The selection operator on a subpopulation i is carried out based on P_i, meaning that the better the species seed is, the higher probability the subpopulation is chosen and vice versa. In this way, we can choose more suitable individuals close to global optima and avoid getting trapped in local optima.

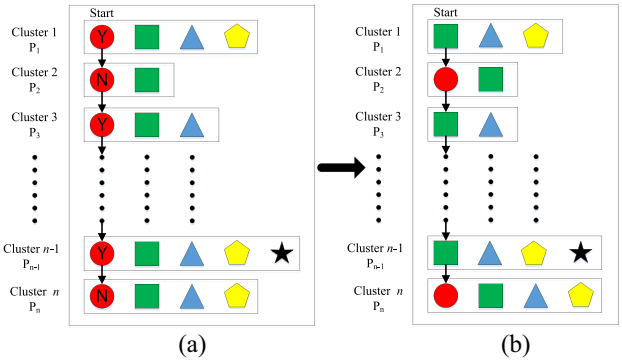


Fig. 6. Illustration of the selection operator in DSDE. (a) Before selection. (b) After the first round selection.

Algorithm 3 Adaptive Selection Mechanism

Begin
1. Execute APC on the combined population;
2. Calculate the choosing probability P_i for each subpopulation;
3. $N_0 = 0$; $i = 1$;
4. **While** $N_0 < N$
5. **If** Subpopulation i is not empty
6. **If** *rand* $< P_i$
7. Remove the best individual $\boldsymbol{X}$ in subpopulation i and add $\boldsymbol{X}$ to the next generation;
8. $N_0 = N_0 + 1$;
9. **End of If**
10. **End of If**
11. $i = i + 1$;
12. **If** $i > n$ (n is the number of subpopulations identified by APC)
13. $i = 1$;
14. **End of If**
15. **End of While**

End

The framework of the adaptive selection mechanism is shown in Algorithm 3.

C. Archive Technique

The archive technique has been widely used in multiobjective optimization to store nondominated solutions [56]–[60]. Besides, cultural algorithm can also archive previous knowledge to control the search [61], [62]. However, it is rarely used in multimodal algorithms. In [63], an archive technique was proposed to detect converged subpopulation and reinitialize them for searching other regions. However, the converged subpopulation is obtained by the hill-valley method, which would cause a waste of FEs. Differently, in this paper, we proposed to detect the stagnant individuals and to obtain the converged subpopulation by simple distance information. The process is described as follows.

Algorithm 4 DSDE

Begin
1. Randomly initialize the population;
2. Randomly generate an integer as the cluster size M;
3. Partition the population into several species using **Algorithm 1**;
4. **For** each species
5. **For** each individual $\boldsymbol{x}_i$ in the current species
6. **If** $\boldsymbol{x}_i$ is a superior individual
7. Using DE/lbest/1 strategy for $\boldsymbol{x}_i$;
8. **Else**
9. Using DE/current-to-rand/1 strategy for $\boldsymbol{x}_i$;
10. **End of If**
11. **End of For**
12. **End of For**
13. Perform adaptive selection mechanism using **Algorithm 3**;
14. **For** each individual $\boldsymbol{x}_i$
15. **If** $sc_i \geq T$
16. Move $\boldsymbol{x}_i$ and its neighbors whose fitness values are worse than $\boldsymbol{x}_i$ to archive and reinitialize these individuals;
17. **End of If**
18. **End of For**
19. Repeat Steps 2 to 18 until the termination criterion is satisfied.

End

Algorithm 5 DSDE-C

Begin
1. Randomly initialize the population;
2. Randomly generate an integer as the cluster size M;
3. Partition the population into several species using **Algorithm 1**;
4. **For** each species
5. **For** each individual $\boldsymbol{x}_i$ in the current species
6. **If** $\boldsymbol{x}_i$ is a superior individual
7. Using DE/lbest/1 strategy for $\boldsymbol{x}_i$;
8. **Else**
9. Using DE/current-to-rand/1 strategy for $\boldsymbol{x}_i$;
10. **End of If**
11. **End of For**
12. **End of For**
13. **For** each offspring $\boldsymbol{u}_i$
14. Compare $\boldsymbol{u}_i$ with the nearest parental individual and replace it if $\boldsymbol{u}_i$ has a better fitness value;
15. **End of For**
16. **For** each individual $\boldsymbol{x}_i$
17. **If** $sc_i \geq T$
18. Move $\boldsymbol{x}_i$ and its neighbors whose fitness values are worse than $\boldsymbol{x}_i$ to archive and reinitialize these individuals;
19. **End of If**
20. **End of For**
21. Repeat Steps 2 to 20 until the termination criterion is satisfied.

End

Each individual $\boldsymbol{x}_i$ in the population is associated with a stagnation counter sc_i, which is initialized as 0 and is used to record the number of generations that $\boldsymbol{x}_i$ has not been improved. Once $\boldsymbol{x}_i$ is improved, sc_i is reset to 0. Once sc_i exceeds a certain threshold T, $\boldsymbol{x}_i$ is treated to be stagnant. For each stagnant individual $\boldsymbol{x}_i$, we find its M nearest neighbors (M is the current cluster size in this generation) in the population to form a converged subpopulation with $M + 1$ individuals. In such a condition, in order to avoid wasting the FEs budget on these stagnated individuals, some of the individuals in the converged subpopulations will be reinitialized to increase the population diversity for searching other areas. However, these stagnant individuals are probably the optima of the search space. To preserve and avoid losing the found optima by reinitialization, these individuals are stored in the archive at the same time. Herein, the individuals to be reinitialized and stored are only $\boldsymbol{x}_i$ and those worse than $\boldsymbol{x}_i$. The individuals better than $\boldsymbol{x}_i$ are still kept in the population so as to maintain the exploitation ability. These stagnated individuals are maintained in the archive until the end of the search. Therefore, by doing so, we are also able to preserve the converged solutions or the probably optimal solutions found during the search.

D. Complete DSDE Algorithm

Based on all the components described above, the pseudo code of the complete procedure of DSDE is outlined in Algorithm 4. The superiority of DSDE is shown as follows.

1) Each individual chooses a mutation strategy suitable for itself by using the dual-strategy mutation technique, leading to a balance between diversity and convergence.
2) The adaptive selection operator based on APC can choose more suitable individuals distributed in different optimal regions. It fulfills the requirement of MMOP for locating all the optimal regions. Besides, the proposed probabilistic model can ensure the accuracy of solutions and avoid getting trapped in local optima.
3) The archive technique can detect and protect the converged individuals. These converged individuals will be reinitialized to increase the population diversity for searching more areas. Meanwhile, storing these converged individuals in the archive can preserve and avoid losing the found optima by reinitialization.

E. DSDE for Higher Dimensional Problems

DSDE performs particularly well on many MMOPs, which will be further discussed in Section IV. However, the selection mechanism based on APC in DSDE does not work well enough in high-dimensional functions, mainly due to the "curse of dimensionality" that limits the efficiency of APC. For the high-dimensional space, the individual distribution is more dispersed. Therefore, the distance or similarity between any two individuals is not significantly different. The failure of the similarity estimation in APC causes the difficulties in forming stable clusters to locate different optimal regions.

Aiming at this problem, when solving high-dimensional problems, we incorporate the selection operator like in CDE in our method, termed as DSDE-C. That is, after generating a new individual $\boldsymbol{u}_i$, we will compare the fitness of $\boldsymbol{u}_i$ with the most similar parental individual (denoted by Euclidean distance) and replace the parent if $\boldsymbol{u}_i$ is better. The complete pseudo code of the procedure of DSDE-C is outlined in Algorithm 5.

F. Complexity Analysis

Herein, we denote the population size and the dimension of problem as N and D, respectively. First, in our niching method for both DSDE and DSDE-C, the time complexity is $O(N \times \log(N)) + O(N^2 \times D)$, which are obtained by line 1 and lines 2–6 in Algorithm 1, respectively. This time-complexity is actually reduced to the $O(N^2 \times D)$ and is similar to many other

TABLE I
PARAMETER SETTINGS

Function	Max_FEs	N
F_1-F_5	5.00E+04	80
F_6	2.00E+05	100
F_7	2.00E+05	300
F_8-F_9	4.00E+05	300
F_{10}	2.00E+05	100
F_{11}-F_{13}	2.00E+05	200
F_{14}-F_{20}	4.00E+05	200

state-of-the-art multimodal algorithms because of the calculations of pairwise distance. As for individual evolution, both DSDE and DSDE-C have the same complexity as $O(N \times D)$, as obtained by lines 4–12 in both Algorithms 4 and 5. In the selection operator, DSDE uses the selection mechanism based on APC, so we first analyze the time complexity of APC. To implement APC, we first need to calculate the similarity matrix $s(i, k)$, the time complexity of which is $O(N^2 \times D)$. Then, we should calculate message $r(i, k)$ and $a(i, k)$ in each iteration, the time complexity of which is $O(N^3)$. Therefore, the time complexity of APC is $O(N^2 \times D + N^3 \times NT)$, where NT is the number of iterations in APC. As a result, the time complexity of the selection in DSDE is $O(N^2 \times D + N^3 \times NT) + O(N \times D)$, which are obtained by line 1 and lines 2–15 in Algorithm 3, respectively. Therefore the time-complexity is actually reduced to the $O(N^2 \times D + N^3 \times NT)$, while DSDE-C takes $O(N^2 \times D)$, as obtained by lines 13 and 14 in Algorithm 5. Furthermore, the time complexity of the archive technique in both DSDE and DSDE-C is $O(N^2 \times D)$, as obtained by lines 14–18 in Algorithm 4 or lines 16–20 in Algorithm 5. Therefore, the overall time complexities of DSDE and DSDE-C are $O(N^2 \times D + N^3 \times NT)$ and $O(N^2 \times D)$, respectively.

It should be noted that the time complexity of most state-of-the-art multimodal algorithms is $O(N^2 \times D)$ due to the calculations of pairwise distance, which is the same to the time complexity of the proposed DSDE-C. Detailed comparisons of time complexity are listed in Table S.I in the supplementary file.

IV. EXPERIMENT STUDIES

A. Benchmark Functions and Performance Metrics

In this section, 20 widely used benchmark functions from CEC2013 test suite [64] are used to test the performance of DSDEs (also including DSDE-C). These functions can be classified into three groups. The first group includes the first ten functions $F_1 - F_{10}$ which are low-dimensional base functions. The second group consists of the next five functions $F_{11} - F_{15}$ which are low-dimensional composition function with a huge number of local optima. The third group consists of the last five functions $F_{16} - F_{20}$ that are high-dimensional composition function. Due to the space limitation, the properties of these functions are given in Table S.II in the supplementary material. For more details about these test functions, please refer to [64].

Three popular evaluation criteria in [64] called peak ratio (PR), success rate (SR), and convergence speed (AveFEs) are used to evaluate the performance of DSDEs and other state-of-the-art multimodal algorithms. For a given maximum number of FEs (denoted as Max_FEs) and an accuracy level ε, PR refers to the average percentage of all global optima found over multiple runs. SR denotes the percentage of successful runs out of all runs. Here, a successful run means a run, where all the global optima are found. AveFEs is the average number of function evaluations required in the successful runs. If an algorithm cannot locate all the global optima under the Max_FEs, then the Max_FEs is used to calculate AveFEs.

There are five frequently adopted accuracy levels, $\varepsilon = 1.0\text{E}-01$, $\varepsilon = 1.0\text{E}-02$, $\varepsilon = 1.0\text{E}-03$, $\varepsilon = 1.0\text{E}-04$, and $\varepsilon = 1.0\text{E}-05$. However, the accuracy levels with $\varepsilon = 1.0\text{E}-01$ and $\varepsilon = 1.0\text{E}-02$ are easy to accomplish and are imprecise. So in our experiments, only the accuracy levels with $\varepsilon = 1.0\text{E}-03$, $\varepsilon = 1.0\text{E}-04$, and $\varepsilon = 1.0\text{E}-05$ are chosen. Besides, in this paper, we mainly discuss the results at $\varepsilon = 1.0\text{E}-04$, which is commonly used in [31], [32], and [39]–[49]. Furthermore, the Max_FEs is set the same as proposed in CEC2013 competition [64], while the population size N is set the same as previous works [36], [49], as shown in Table I, to make the algorithm easily applicable and fairly compared with other algorithms. Moreover, the certain threshold T used in the archive technique is set as 40 and 80 in DSDE and DSDE-C, respectively. The amplification factor F and crossover rate CR in our algorithm are set as 0.5 and 0.9, respectively. In addition, due to the stochastic characteristic of the algorithms, each algorithm is run independently for 51 times. The Wilcoxon's rank sum test statistics and the mean results are calculated to evaluate the performance of the algorithms.

B. Comparisons With State-of-the-Art Multimodal Algorithms

The results of DSDEs on $F_1 - F_{20}$ with respect to PR and SR at all three accuracy levels (i.e., $\varepsilon = 1.0\text{E}-03$, $\varepsilon = 1.0\text{E}-04$, and $\varepsilon = 1.0\text{E}-05$) are given in Table II. The PR that equals to 1 are emphasized in boldface. As DSDE-C is designed for higher dimensional problems, we presented the results of DSDE-C for the functions $F_{16} - F_{20}$.

To further evaluate the performance of DSDEs, we compare the results obtained by DSDEs with those obtained by several state-of-the-art multimodal algorithms, i.e., SDE [10], CDE [29], NSDE, NCDE [31], Self-CSDE, Self-CCDE [32], LoICDE, LoISDE [41], PNPCDE [47], R2PSO, R3PSO [39], LIPS [40], MOMMOP [49], and LMCEDA, LMSEDA [36]. The results of these state-of-the-art multimodal algorithms are from [36], which are obtained under the same Max_FEs.

Tables III–V present the comparison results in terms of PR, SR, and AveFEs at the accuracy level $\varepsilon = 1.0\text{E}-04$, while the comparison results at the other two accuracy levels are listed in Tables S.III–S.VI in the supplementary material. For clarity, the best PRs are highlighted in boldface. In addition, Wilcoxon's rank sum test [65] at $\alpha = 0.05$ with respect to PR between DSDEs and other state-of-the-art algorithms is performed to evaluate the statistical significance of the results. The symbols "+," "≈," and "−" indicate DSDEs perform significantly better (+), similarly (≈), or significantly worse (−) than the corresponding algorithm in comparison.

TABLE II
EXPERIMENTAL RESULTS IN PR AND SR OF DSDES ON 20 PROBLEMS $f_1 - f_{20}$ AT ALL THE THREE ACCURACY LEVELS

DSDE										
ε	F_1		F_2		F_3		F_4		F_5	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1E-03	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
1E-04	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
1E-05	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
ε	F_6		F_7		F_8		F_9		F_{10}	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1E-03	**1.000**	1.000	0.899	0.000	0.682	0.000	0.391	0.000	**1.000**	1.000
1E-04	**1.000**	1.000	0.891	0.000	0.655	0.000	0.363	0.000	**1.000**	1.000
1E-05	**1.000**	1.000	0.878	0.000	0.629	0.000	0.337	0.000	**1.000**	1.000
ε	F_{11}		F_{12}		F_{13}		F_{14}		F_{15}	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1E-03	**1.000**	1.000	**1.000**	1.000	0.922	0.549	0.667	0.000	0.679	0.000
1E-04	**1.000**	1.000	**1.000**	1.000	0.912	0.549	0.667	0.000	0.635	0.000
1E-05	**1.000**	1.000	0.985	0.882	0.912	0.549	0.667	0.000	0.615	0.000
DSDE-C										
ε	F_{16}		F_{17}		F_{18}		F_{19}		F_{20}	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1E-03	0.667	0.000	0.375	0.000	0.667	0.000	0.395	0.000	0.314	0.000
1E-04	0.667	0.000	0.375	0.000	0.667	0.000	0.395	0.000	0.314	0.000
1E-05	0.667	0.000	0.375	0.000	0.667	0.000	0.395	0.000	0.314	0.000

TABLE III
EXPERIMENTAL RESULTS IN PR AND SR ON PROBLEMS $f_1 - f_{15}$ AT ACCURACY LEVEL $\varepsilon = 1.0\text{E} - 04$

Func	DSDE		CDE		SDE		LIPS		R2PSO		R3PSO		NCDE		NSDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F_1	**1.000**	1.000	**1.000(≈)**	1.000	0.657(+)	0.373	0.833(+)	0.686	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_2	**1.000**	1.000	**1.000(≈)**	1.000	0.737(+)	0.529	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.776(+)	0.667
F_3	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.961(≈)	0.961	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_4	**1.000**	1.000	**1.000(≈)**	1.000	0.284(+)	0.000	0.990(≈)	0.961	0.946(+)	0.784	0.966(+)	0.863	**1.000(≈)**	1.000	0.240(+)	0.000
F_5	**1.000**	1.000	**1.000(≈)**	1.000	0.922(+)	0.843	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.745(+)	0.490
F_6	**1.000**	1.000	**1.000(≈)**	1.000	0.056(+)	0.000	0.246(+)	0.000	0.537(+)	0.000	0.688(+)	0.000	0.305(+)	0.000	0.056(+)	0.000
F_7	0.891	0.000	0.861(+)	0.000	0.054(+)	0.000	0.400(+)	0.000	0.484(+)	0.000	0.436(+)	0.000	0.873(+)	0.000	0.053(+)	0.000
F_8	0.655	0.000	0.000(+)	0.000	0.015(+)	0.000	0.084(+)	0.000	0.023(+)	0.000	0.421(+)	0.000	0.001(+)	0.000	0.013(+)	0.000
F_9	0.363	0.000	0.474(**-**)	0.000	0.011(+)	0.000	0.104(+)	0.000	0.122(+)	0.000	0.125(+)	0.000	0.461(**-**)	0.000	0.006(+)	0.000
F_{10}	**1.000**	1.000	**1.000(≈)**	1.000	0.147(+)	0.000	0.748(+)	0.000	0.905(+)	0.353	0.850(+)	0.157	0.989(+)	0.863	0.098(+)	0.000
F_{11}	**1.000**	1.000	0.330(+)	0.000	0.314(+)	0.000	0.974(+)	0.843	0.641(+)	0.000	0.650(+)	0.000	0.729(+)	0.059	0.248(+)	0.000
F_{12}	**1.000**	1.000	0.002(+)	0.000	0.208(+)	0.000	0.574(+)	0.000	0.392(+)	0.000	0.537(+)	0.000	0.252(+)	0.000	0.135(+)	0.000
F_{13}	**0.912**	0.549	0.141(+)	0.000	0.297(+)	0.000	0.794(+)	0.176	0.627(+)	0.000	0.647(+)	0.000	0.667(+)	0.000	0.225(+)	0.000
F_{14}	**0.667**	0.000	0.026(+)	0.000	0.216(+)	0.000	0.644(+)	0.000	0.408(+)	0.000	0.637(+)	0.000	**0.667(≈)**	0.000	0.190(+)	0.000
F_{15}	0.635	0.000	0.005(+)	0.000	0.108(+)	0.000	0.336(+)	0.000	0.167(+)	0.000	0.213(+)	0.000	0.319(+)	0.000	0.125(+)	0.000
+(DSDE is significantly better)			7		14		11		11		11		8		13	
-(DSDE is significantly worse)			1		0		0		0		0		1		0	
≈			7		1		4		4		4		6		2	
Func	PNPCDE		Self-CCDE		Self-CSDE		LoICDE		LoISDE		LMCEDA		LMSEDA		MOMMOP	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F_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	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_2	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.235(+)	0.039	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_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	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_4	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.686(+)	0.294	0.975(+)	0.902	0.250(+)	0.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_5	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.961(+)	0.922	**1.000(≈)**	1.000	0.667(+)	0.333	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_6	0.537(+)	0.000	0.942(+)	0.490	0.699(+)	0.020	**1.000(≈)**	1.000	0.056(+)	0.000	0.990(+)	0.843	0.972(+)	0.588	**1.000(≈)**	1.000
F_7	0.874(+)	0.000	0.884(≈)	0.020	0.695(+)	0.000	0.705(+)	0.020	0.029(+)	0.000	0.734(+)	0.000	0.673(+)	0.000	**1.000(-)**	1.000
F_8	0.000(+)	0.000	0.994(**-**)	0.882	0.695(**-**)	0.000	0.000(+)	0.000	0.012(+)	0.000	0.367(+)	0.000	0.613(+)	0.000	**1.000(-)**	1.000
F_9	0.472(**-**)	0.000	0.459(**-**)	0.000	0.265(+)	0.000	0.187(+)	0.000	0.005(+)	0.000	0.284(+)	0.000	0.248(+)	0.000	**1.000(-)**	1.000
F_{10}	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.992(+)	0.922	**1.000(≈)**	1.000	0.083(+)	0.000	**1.000(≈)**	1.000	0.998(≈)	0.980	**1.000(≈)**	1.000
F_{11}	0.660(+)	0.000	0.778(+)	0.137	0.399(+)	0.000	0.660(+)	0.000	0.167(+)	0.000	0.667(+)	0.000	0.892(+)	0.392	0.716(+)	0.020
F_{12}	0.000(+)	0.000	0.422(+)	0.000	0.321(+)	0.000	0.495(+)	0.000	0.125(+)	0.000	0.750(+)	0.000	0.990(≈)	0.922	0.939(+)	0.549
F_{13}	0.461(+)	0.000	0.660(+)	0.000	0.317(+)	0.000	0.510(+)	0.000	0.167(+)	0.000	0.667(+)	0.000	0.667(+)	0.000	0.667(+)	0.000
F_{14}	0.592(+)	0.000	0.657(+)	0.000	0.304(+)	0.000	0.657(+)	0.000	0.167(+)	0.000	**0.667(≈)**	0.000	**0.667(≈)**	0.000	**0.667(≈)**	0.000
F_{15}	0.258 (+)	0.000	0.343(+)	0.000	0.186(+)	0.000	0.299(+)	0.000	0.125(+)	0.000	0.696(**-**)	0.000	**0.738(-)**	0.000	0.618(+)	0.000
+	8		6		11		9		13		7		6		4	
-	1		2		1		0		0		1		1		3	
≈	6		7		3		6		2		7		8		8	

From Tables III–V and Tables S.III–S.VI in the supplementary material, we can see that with the accuracy level increasing from $\varepsilon = 1.0\text{E} - 03$ to $\varepsilon = 1.0\text{E} - 05$, the performances of many algorithms are largely weakened, except DSDEs.

1) For the first 15 low-dimensional functions $F_1 - F_{15}$, DSDE performs significantly better on most functions than the other algorithms, no matter at which accuracy level. Moreover, the advantage of DSDE over the other algorithms becomes increasingly obvious as the accuracy level increases. Take the accuracy level $\varepsilon = 1.0\text{E} - 04$ as an example, we find DSDE is significantly superior to SDE, LIPS, R2PSO, R3PSO, NSDE,

TABLE IV
EXPERIMENTAL RESULTS IN PR AND SR ON PROBLEMS f_{16}–f_{20} AT ACCURACY LEVEL $\varepsilon = 1.0\text{E}-04$

Func	DSDE-C		CDE		SDE		LIPS		R2PSO		R3PSO		NCDE		NSDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F_{16}	**0.667**	0.000	0.000(+)	0.000	0.108(+)	0.000	0.304(+)	0.000	0.095(+)	0.000	0.431(+)	0.000	**0.667(≈)**	0.000	0.170(+)	0.000
F_{17}	0.375	0.000	0.000(+)	0.000	0.076(+)	0.000	0.162(+)	0.000	0.015(+)	0.000	0.096(+)	0.000	0.250(+)	0.000	0.108(+)	0.000
F_{18}	**0.667**	0.000	0.167(+)	0.000	0.026(+)	0.000	0.098(+)	0.000	0.036(+)	0.000	0.101(+)	0.000	0.500(+)	0.000	0.163(+)	0.000
F_{19}	0.395	0.000	0.000(+)	0.000	0.105(+)	0.000	0.000(+)	0.000	0.000(+)	0.000	0.032(+)	0.000	0.348(+)	0.000	0.098(+)	0.000
F_{20}	**0.314**	0.000	0.000(+)	0.000	0.000(+)	0.000	0.000(+)	0.000	0.002(+)	0.000	0.078(+)	0.000	0.250(+)	0.000	0.123(+)	0.000
+(DSDE-C is significantly better)			5		5		5		5		5		4		5	
-(DSDE-C is significantly worse)			0		0		0		0		0		0		0	
≈			0		0		0		0		0		1		0	
Func	PNPCDE		Self-CCDE		Self-CSDE		LoICDE		LoISDE		LMCEDA		LMSEDA		MOMMOP	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F_{16}	0.000(+)	0.000	0.657(+)	0.000	0.072(+)	0.000	0.559(+)	0.000	0.167(+)	0.000	**0.667(≈)**	0.000	**0.667(≈)**	0.000	0.650(+)	0.000
F_{17}	0.000(+)	0.000	0.248(+)	0.000	0.056(+)	0.000	0.223(+)	0.000	0.076(+)	0.000	0.456(-)	0.000	**0.620(-)**	0.000	0.505(-)	0.000
F_{18}	0.147(+)	0.000	0.337(+)	0.000	0.003(+)	0.000	0.219(+)	0.000	0.157(+)	0.000	0.657(≈)	0.000	0.660(≈)	0.000	0.497(+)	0.000
F_{19}	0.000(+)	0.000	0.113(+)	0.000	0.000(+)	0.000	0.037(+)	0.000	0.027(+)	0.000	0.451(-)	0.000	**0.458(-)**	0.000	0.223(+)	0.000
F_{20}	0.000(+)	0.000	0.027(+)	0.000	0.000(+)	0.000	0.123(+)	0.000	0.088(+)	0.000	0.059(+)	0.000	0.248(+)	0.000	0.125(+)	0.000
+	5		5		5		5		5		1		1		4	
-	0		0		0		0		0		2		2		1	
≈	0		0		0		0		0		2		2		0	

TABLE V
EXPERIMENTAL RESULTS IN AVEFES ON PROBLEMS $f_1 - f_5$ AT ALL THE THREE ACCURACY LEVELS

ε=1.0E-03									
Func	DSDE	CDE	NCDE	Self-CCDE	LoICDE	PNPCDE	MOMMOP	LMCDEA	LMSDEA
F_1	1.62E+02	**1.60E+02(≈)**	6.86E+02(+)	3.55E+02(+)	1.73E+02(+)	1.62E+02(≈)	1.66E+02(+)	3.23E+02(+)	3.41E+02(+)
F_2	**4.22E+02**	1.39E+03(+)	9.27E+02(+)	7.61E+02(+)	1.24E+03(+)	1.08E+03(+)	1.09E+03(+)	1.07E+03(+)	8.97E+02(+)
F_3	**2.40E+02**	1.11E+03(+)	4.94E+02(+)	5.55E+02(+)	5.84E+02(+)	8.33E+02(+)	9.58E+02(+)	5.06E+02(+)	4.84E+02(+)
F_4	**3.48E+03**	2.71E+04(+)	3.73E+03(≈)	7.07E+03(+)	1.48E+04(+)	2.28E+04(+)	3.50E+04(+)	1.18E+04(+)	6.37E+03(+)
F_5	**8.45E+02**	5.42E+03(+)	1.57E+03(+)	1.97E+03(+)	1.88E+03(+)	3.69E+03(+)	1.48E+04(+)	3.78E+03(+)	2.51E+03(+)
+(DSDE is significantly better)		4	4	5	5	4	5	5	5
-(DSDE is significantly worse)		0	0	0	0	0	0	0	0
≈		1	1	0	0	1	0	0	0
ε=1.0E-04									
Func	DSDE	CDE	NCDE	Self-CCDE	LoICDE	PNPCDE	MOMMOP	LMCDEA	LMSDEA
F_1	1.62E+02	**1.60E+02(≈)**	6.86E+02(+)	3.55E+02(+)	1.73E+02(+)	1.62E+02(≈)	1.66E+02(+)	3.23E+02(+)	3.41E+02(+)
F_2	**6.40E+02**	3.23E+03(+)	1.84E+03(+)	1.46E+03(+)	3.00E+03(+)	2.68E+03(+)	3.09E+03(+)	1.37E+03(+)	1.17E+03(+)
F_3	**3.58E+02**	3.35E+03(+)	9.84E+02(+)	9.55E+02(+)	1.52E+03(+)	2.42E+03(+)	2.47E+03(+)	6.51E+02(+)	6.28E+02(+)
F_4	8.62E+03	3.93E+04(+)	**4.95E+03(-)**	1.15E+04(+)	2.71E+04(+)	3.36E+04(+)	3.69E+04(+)	1.40E+04(+)	7.88E+03(-)
F_5	**2.24E+03**	1.13E+04(+)	2.45E+03(+)	3.66E+03(+)	3.50E+03(+)	8.97E+03(+)	1.78E+04(+)	6.00E+03(+)	3.71E+03(+)
+(DSDE is significantly better)		4	4	5	5	4	5	5	4
-(DSDE is significantly worse)		0	1	0	0	0	0	0	1
≈		1	0	0	0	1	0	0	0
ε=1.0E-05									
Func	DSDE	CDE	NCDE	Self-CCDE	LoICDE	PNPCDE	MOMMOP	LMCDEA	LMSDEA
F_1	1.62E+02	**1.60E+02(≈)**	6.86E+02(+)	3.55E+02(+)	1.73E+02(+)	1.62E+02(≈)	1.66E+02(+)	3.23E+02(+)	3.41E+02(+)
F_2	**1.03E+03**	6.68E+03(+)	2.52E+03(+)	2.35E+03(+)	6.50E+03(+)	6.60E+03(+)	5.78E+03(+)	1.53E+03(+)	1.29E+03(+)
F_3	**4.61E+02**	7.93E+03(+)	1.80E+03(+)	1.94E+03(+)	3.74E+03(+)	5.37E+03(+)	5.06E+03(+)	7.62E+02(+)	7.21E+02(+)
F_4	1.77E+04	4.87E+04(+)	**6.04E+03(-)**	1.76E+04(≈)	3.77E+04(+)	4.41E+04(+)	3.92E+04(+)	1.52E+04(-)	8.66E+03(-)
F_5	**2.60E+03**	1.93E+04(+)	3.42E+03(+)	6.97E+03(+)	5.65E+03(+)	1.60E+04(+)	1.87E+04(+)	7.72E+03(+)	4.97E+03(+)
+(DSDE is significantly better)		4	4	4	5	4	5	4	4
-(DSDE is significantly worse)		0	1	0	0	0	0	1	1
≈		1	0	1	0	1	0	0	0

Self-CSDE, and LoISDE on more than ten functions, while none of these algorithms can surpass DSDE on more than one function. When compared with CDE, NCDE, PNPCDE, Self-CCDE, LoICDE, LMCEDA, and LMSEDA, our DSDE significantly outperforms them on at least six functions, while beaten by them on no more than two functions. It seems that the performance between DSDE and MOMMOP are almost equivalent, DSDE still performs better than MOMMOP on four functions while worse than MOMMOP on three functions.

2) For the last five high-dimensional functions $F_{16}-F_{20}$, which are more complicated with a huge number of local optima, our DSDE variant DSDE-C almost outperforms all the algorithms on all the five test functions. Similarly, such phenomenon also becomes more apparent when the accuracy level increases, which shows the feasibility and superiority of DSDE-C in dealing with high-dimensional and more complicated problems. Take the accuracy level $\varepsilon = 1.0\text{E}-04$ as an example again. DSDE-C performs better than CDE, SDE, LIPS, R2PSO, R3PSO, NSDE, PNPCDE, Self-CCDE, Self-CSDE, LoICDE, and LoISDE on all the five functions. Meanwhile, it surpasses NCDE and MOMMOP on four functions while only worse than MOMMOP on one function. Although DSDE-C performs slightly

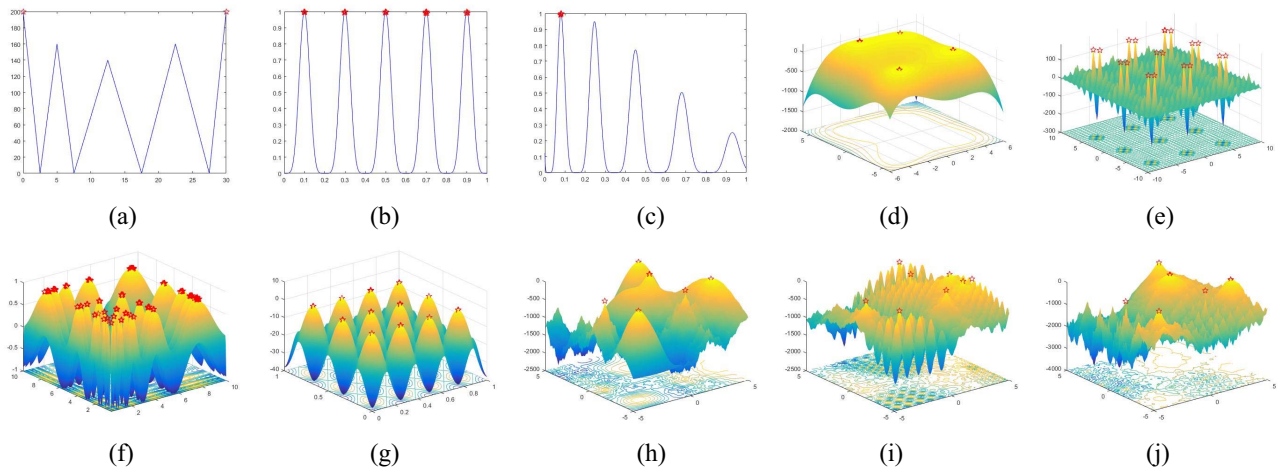


Fig. 7. Results in final fitness landscapes on ten functions that can be visualized. (a) F_1. (b) F_2. (c) F_3. (d) F_4. (e) F_6. (f) F_7. (g) F_{10}. (h) F_{11}. (i) F_{12}. (j) F_{13}.

worse than LMCEDA and LMSEDA on these five functions, DSDE performs much better than LMCEDA and LMSEDA on the first 15 low-dimensional functions.

3) For the comparison in AveFEs, we only choose the simple functions F_1-F_5 to make comparison, because there is nearly no sense to make comparisons using this criterion when there is no successful run for algorithms on complicated functions. Besides, if the PR results of different algorithms are not comparable, it is also meaningless. Herein, we compare DSDE with CDE, NCDE, Self-CCDE, LoICDE, PNPCDE, MOMMOP, LMCEDA, and LMSEDA on F_1-F_5, because they can achieve the equivalent performance on PR and SR. As we can see, we find that DSDE is superior to all the eight compared algorithms in AveFEs. DSDE achieves fewer AveFEs on almost all functions than the other algorithms, regardless of accuracy level. Take accuracy level $\varepsilon = 1.0\text{E}-04$ for example, from Table V, DSDE performs better than all the eight algorithms on at least four functions. As a result, our algorithm has a fast convergence speed to locate all the global optima on these MMOPs.

Overall, we can conclude DSDEs generally outperform the state-of-the-art multimodal algorithms in comparison, in terms of PR, SR, and AveFEs. Moreover, the superiority of DSDEs becomes more prominent as the accuracy level increases. Besides, the experimental results show that DSDEs can find a balance between diversity and convergence, as a result of the novel techniques proposed in this paper, which will be investigated in the next part.

In addition, to have a more intuitive view of DSDE, in Fig. 7, we show the fitness landscape and the distribution of the final individuals when the algorithm terminated on ten selected functions. In Fig. 7, we only show the individuals whose fitness value is similar to the best individual. That is, the difference in their fitness values is no larger than 0.01.

As can be seen from Fig. 7, DSDE can still find all the global optima, even though the number of global optima is very large, such as Fig. 7(e). Further, when there are many or even a huge number of local optima, such as Fig. 7(h)–(j), DSDE is capable of maintaining high population diversity and locating all the global optima.

From all the tables and figures listed above, DSDEs can make good balance between exploration and exploitation. This helps the algorithms to keep population diversity for locating more global optima and to accelerate the convergence speed for improving the accuracy of solutions. Therefore, DSDEs outperform other state-of-the-art multimodal algorithms.

TABLE VI
EXPERIMENTAL RESULTS IN PR AND SR BETWEEN DSDES AND THEIR VARIANTS WITH SINGLE MUTATION STRATEGY ON PROBLEMS f_1-f_{20} AT ACCURACY LEVEL $\varepsilon = 1.0\text{e}-04$

Func	DSDE		DSDE-lbest		DSDE-rand	
	PR	SR	PR	SR	PR	SR
F_1	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_2	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_3	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_4	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_5	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_6	**1.000**	1.000	0.700(+)	0.000	0.978(+)	0.627
F_7	**0.891**	0.000	0.839(+)	0.000	0.754(+)	0.000
F_8	**0.655**	0.000	0.345(+)	0.000	0.204(+)	0.000
F_9	**0.363**	0.000	0.362(≈)	0.000	0.221(+)	0.000
F_{10}	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_{11}	**1.000**	1.000	0.869(+)	0.314	0.931(+)	0.627
F_{12}	**1.000**	1.000	0.836(+)	0.157	0.980(+)	0.824
F_{13}	**0.912**	0.549	0.748(+)	0.000	0.846(+)	0.196
F_{14}	**0.667**	0.000	**0.667(≈)**	0.000	0.621(+)	0.000
F_{15}	**0.635**	0.000	0.520(+)	0.000	0.515(+)	0.000
+(DSDE is significantly better)			7		9	
-(DSDE is significantly worse)			0		0	
≈			8		6	
Func	DSDE-C		DSDE-C-lbest		DSDE-C-rand	
	PR	SR	PR	SR	PR	SR
F_{16}	**0.667**	0.000	0.379(+)	0.000	**0.667(≈)**	0.000
F_{17}	**0.375**	0.000	0.125(+)	0.000	0.294(+)	0.000
F_{18}	**0.667**	0.000	0.147(+)	0.000	0.533(+)	0.000
F_{19}	**0.395**	0.000	0.000(+)	0.000	0.350(+)	0.000
F_{20}	**0.314**	0.000	0.000(+)	0.000	0.025(+)	0.000
+(DSDE-C is significantly better)			5		4	
-(DSDE-C is significantly worse)			0		0	
≈			0		1	

C. Effects of DSDE Components

The main components of DSDEs are: 1) dual-strategy mutation; 2) adaptive probabilistic selection mechanism based on APC; and 3) archive technique. In this section, we will discuss the property and effect of each component.

1) Dual-Strategy Mutation: In order to investigate the effect of dual-strategy, we consider two DSDEs variants with only one mutation strategy (i.e., DE/current-to-rand/1 or DE/lbest/1), which are denoted as DSDEs-rand and DSDEs-lbest, respectively. They are compared with DSDEs on $F_1 - F_{20}$, with the detailed comparison results with respect to PR and SR at accuracy level $\varepsilon = 1.0\text{E}-04$ being listed in Table VI. We can see that DSDEs totally outperform than DSDEs-rand and DSDEs-lbest on 13 and 12 functions, respectively. These results demonstrate that our dual strategy scheme is helpful for improving the performance of the algorithm.

It is established knowledge that DE/lbest/1 is a greedy mutation strategy in niching. However, it performs poorly on the complicated problems, especially on $F_{17} - F_{20}$, which poses a strong requirement of population diversity for the optimizers. The effectiveness of DE/current-to-rand/1 has been verified when applied to solve rotated problems and multiobjective optimization problems [50], because it can maintain good diversity of population. However, it also slows the speed of convergence on some functions which can be seen from our results.

As we can see, DSDEs achieve higher accuracy than other DSDEs variants with only one strategy. The better performance of DSDEs is due to making full use of evolution information from individuals, which can select a proper mutation strategy for each individual and can achieve a better balance between exploration and exploitation. As a result, the DSDE in our algorithm is useful.

2) APC-Based Selection: To date, there are mainly two selection operators. The one-by-one selection operator, which compares the fitness of the new generated individual with its most similar parent and replaces it if the new individual's fitness is better, used in CDE, NCDE, PNPCDE, Self-CCDE, Self-CSDE, LoICDE, LMCEDA, and LMSEDA. Another one is combination selection, which is choosing N fitter individuals from the combined population (offspring and parents), used in SDE, NSDE, and LoISDE. Although the results of these two methods are promising, there is still much room for improvement. Besides, from the above results in Tables III–V and Tables S.III–S.VI in the supplementary material, it seems that the results of the one-by-one selection operator are generally better than the combination selection. This may be due to the fact that the one-by-one selection operator can guarantee the comparison to be carried out in the same niche, which can protect the individuals in other niches from being covered.

Here, to evaluate the effectiveness of the new proposed adaptive selection mechanism, the DSDE variant with the one-by-one selection operator is compared with DSDE on $F_1 - F_{20}$. Interestingly, such a DSDE variant is actually the DSDE-C. The averaged results of PR and SR at accuracy level $\varepsilon = 1.0\text{E}-04$ over 51 runs in each problem between DSDE and DSDE-C are presented in Table VII in detail.

TABLE VII
EXPERIMENTAL RESULTS IN PR AND SR BETWEEN DSDE AND DSDE-C ON PROBLEMS $f_1 - f_{20}$ AT ACCURACY LEVEL $\varepsilon = 1.0\text{E}-04$

Func		F_1	F_2	F_3	F_4	F_5
DSDE	PR	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	SR	1.000	1.000	1.000	1.000	1.000
DSDE-C	PR	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
	SR	1.000	1.000	1.000	1.000	1.000
Func		F_6	F_7	F_8	F_9	F_{10}
DSDE	PR	**1.000**	**0.891**	**0.655**	**0.363**	**1.000**
	SR	1.000	0.000	0.000	0.000	1.000
DSDE-C	PR	0.000	0.471	0.000	0.119	**1.000**
	SR	0.000	0.000	0.000	0.000	1.000
Func		F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
DSDE	PR	**1.000**	**1.000**	**0.912**	**0.667**	**0.635**
	SR	1.000	1.000	0.549	0.000	0.000
DSDE-C	PR	0.667	0.260	0.667	**0.667**	0.375
	SR	0.000	0.000	0.000	0.000	0.000
Func		F_{16}	F_{17}	F_{18}	F_{19}	F_{20}
DSDE	PR	0.627	0.255	0.248	0.020	0.000
	SR	0.000	0.000	0.000	0.000	0.000
DSDE-C	PR	**0.667**	**0.375**	**0.667**	**0.395**	**0.314**
	SR	0.000	0.000	0.000	0.000	0.000

As we can see, there is nearly no difference between DSDE and DSDE-C on the first five functions. However, on functions $F_6 - F_{15}$ that involve many global optima and a huge number of local optima, DSDE performs much better than DSDE-C. This is because the adaptive selection operator based on APC can choose more suitable individuals distributed in different optimal regions by partitioning the population, which matches the requirement of MMOP for locating all the optimal regions. Moreover, the probabilistic selection can make sure the chosen individuals are close to the global optima. These results illustrate the effectiveness of our adaptive selection strategy.

However, for the last five functions $F_{16} - F_{20}$, DSDE performs worse than DSDE-C, especially on $F_{19} - F_{20}$. We first look at F_{16} and F_{17} with dimensions of 5, the performance difference between DSDE and DSDE-C is not obvious, which indicates that the performance of APC is not seriously affected when the dimension increases to 5. But the performance of DSDE degrades drastically on $F_{18} - F_{20}$ with the increasing dimension. It cannot find any global optima on F_{20}. While DSDE-C always keeps a stable performance and shows the consistent superiority than DSDE on the higher dimensional functions. Such an observation is caused by the property of APC, which is somehow sensitive to the dimension of problem and is not appropriate for solving high-dimensional problems, as we have mentioned in Section III-E.

3) Archive Technique: DSDEs detect the stagnated and converged individual x_i by a stagnation counter sc_i. Once the sc_i exceeds a certain threshold T, x_i is judged to be stagnant. Thus, the effect of the archive technique may be sensitive to the threshold T. If T is small, the recognized stagnant individual may not completely converged, while if T is large, we may waste many FEs.

In order to investigate the sensitivity of the threshold T to DSDEs, we compare the performance of DSDEs with different parameters T from 20 to 100. The DSDEs with parameter variant $T = a$ is denoted as DSDEs(a). For example, DSDE with $T = 20$ is denoted as DSDE(20). Note that DSDE in this paper is with $T = 40$ and DSDE-C in this paper is with

TABLE VIII
EXPERIMENTAL RESULTS IN PR AND SR BETWEEN DSDES AND THEIR VARIANTS WITH DIFFERENT PARAMETER T ON PROBLEMS $f_1 - f_{20}$ AT ACCURACY LEVEL $\varepsilon = 1.0\text{E}-04$

Func	DSDE		DSDE(20)		DSDE(30)		DSDE(60)		DSDE(80)	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
F_1	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_2	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_3	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_4	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_5	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_6	**1.000**	1.000	**1.000(≈)**	1.000	0.998(≈)	0.961	0.984(+)	0.745	0.958(+)	0.431
F_7	**0.891**	0.000	0.883(≈)	0.000	0.872(+)	0.000	0.837(+)	0.000	0.822(+)	0.000
F_8	0.655	0.000	0.649(≈)	0.000	**0.669(≈)**	0.000	0.614(+)	0.000	0.605(+)	0.000
F_9	0.363	0.000	**0.394(-)**	0.000	0.371(-)	0.000	0.344(+)	0.000	0.331(+)	0.000
F_{10}	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000
F_{11}	**1.000**	1.000	**1.000(≈)**	1.000	**1.000(≈)**	1.000	0.980(+)	0.882	0.993(≈)	0.961
F_{12}	**1.000**	1.000	0.985(+)	0.882	0.980(+)	0.843	0.995(≈)	0.961	0.961(+)	0.667
F_{13}	**0.912**	0.549	**0.912(≈)**	0.510	0.892(+)	0.392	0.866(+)	0.275	0.908(+)	0.510
F_{14}	**0.667**	0.000	**0.667(≈)**	0.000	**0.667(≈)**	0.000	**0.667(≈)**	0.000	**0.667(≈)**	0.000
F_{15}	0.635	0.000	0.610(+)	0.000	0.605(+)	0.000	**0.640(≈)**	0.000	0.591(+)	0.000
+(DSDE is significantly better)			2		4		6		7	
-(DSDE is significantly worse)			1		1		0		0	
≈			12		10		9		8	
Func	DSDE-C		DSDE-C(50)		DSDE-C(60)		DSDE-C(100)			
	PR	SR	PR	SR	PR	SR	PR	SR		
F_{16}	**0.667**	0.000	**0.667(≈)**	0.000	**0.667(≈)**	0.000	**0.667(≈)**	0.000		
F_{17}	0.375	0.000	0.350(+)	0.000	0.346(+)	0.000	**0.385(≈)**	0.000		
F_{18}	**0.667**	0.000	0.614(+)	0.000	0.647(≈)	0.000	0.647(≈)	0.000		
F_{19}	0.395	0.000	0.346(+)	0.000	**0.404(≈)**	0.000	**0.404(≈)**	0.000		
F_{20}	0.314	0.000	0.179(+)	0.000	**0.326(≈)**	0.000	0.260(+)	0.000		
+(DSDE-C is significantly better)			4		1		1			
-(DSDE-C is significantly worse)			0		0		0			
≈			1		4		4			

$T = 80$. The detailed experimental results between DSDEs and their variants with respect to PR and SR at accuracy level $\varepsilon = 1.0\text{E}-04$ averaged over 51 runs are listed in Table VIII.

As we can see, on the first 15 functions, different T values make nearly no difference in DSDE on $F_1 - F_5$, F_{10}, and F_{14}. However, on the other functions, DSDE with $T = 40$ performs better than DSDE(20), DSDE(30), DSDE(60), and DSDE(80) on 2, 4, 6, and 7 functions, respectively. On $F_{16} - F_{20}$, DSDE-C with $T = 80$ performs similarly to DSDE-C(60) and DSDE-C(100) on four functions, but DSDE-C performs better than DSDE-C(60) and DSDE-C(100) on F_{17} and F_{20}, respectively.

Overall, DSDE with $T = 40$ and DSDE-C with $T = 80$ can find a balance between high-accuracy and low-FEs, which are the most suitable parameters to achieve a good performance in our testing.

V. CONCLUSION

This paper develops a DSDE with APC for tackling MMOPs, which can achieve a better balance between exploration and exploitation. Three novel techniques are developed to improve the performance of the algorithm: 1) dual-strategy mutation; 2) adaptive probabilistic selection mechanism based on APC; and 3) archive technique. Note that the selection mechanism using APC is not efficient for solving high-dimensional problems, so we also proposed a DSDE variant named DSDE-C that uses the one-by-one selection operator for solving high-dimensional problems.

The dual-strategy mutation can achieve a better balance between exploration and exploitation. Furthermore, the adaptive selection mechanism based on APC can choose more suitable individuals that are from different peaks and are also close to the global optima. Finally, the archive technique can detect and protect the converged individuals, so as to reinitialize them for improving the exploration ability and preserve the found optimal solutions by storing them in the archive.

Based on these three novel techniques, DSDEs can achieve a promising performance when dealing with MMOPs regardless of the accuracy level. The experimental results also show the efficiency and feasibility of DSDEs for quickly locating more accurate global optima.

For future work, we will extend the DSDEs to solve more complicated optimization problems and real-world applications. Meanwhile, we can also use the parallel/distributed computing resources to reduce the executing time of DSDEs and further improve the performance of DSDEs.

REFERENCES

[1] D.-K. Woo, J.-H. Choi, M. Ali, and H.-K. Jung, "A novel multimodal optimization algorithm applied to electromagnetic optimization," *IEEE Trans. Magn.*, vol. 47, no. 6, pp. 1667–1673, Jun. 2011.
[2] C.-H. Yoo, D.-K. Lim, and H.-K. Jung, "A novel multimodal optimization algorithm for the design of electromagnetic machines," *IEEE Trans. Magn.*, vol. 52, no. 3, pp. 1–4, Mar. 2016.
[3] W. Sheng, S. Swift, L. Zhang, and X. Liu, "A weighted sum validity function for clustering with a hybrid niching genetic algorithm," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 6, pp. 1156–1167, Dec. 2005.
[4] Y. Li, "Using niche genetic algorithm to find fuzzy rules," in *Proc. Int. Symp. Web Inf. Syst. Appl.*, Xuzhou, China, 2009, pp. 64–67.
[5] M. Boughanem and L. Tamine, "A study on using genetic niching for query optimisation in document retrieval," in *Advances in Information Retrieval*. Heidelberg, Germany: Springer, 2002, pp. 135–149.

[6] A. Y. Goharrizi *et al.*, "A parallel multimodal optimization algorithm for simulation-based design of power systems," *IEEE Trans. Power Del.*, vol. 30, no. 5, pp. 2128–2137, Oct. 2015.

[7] J. Yao, N. Kharma, and P. Grogono, "A multi-population genetic algorithm for robust and fast ellipse detection," *Pattern Anal. Appl*., vol. 8, nos. 1–2, pp. 149–162, 2005.

[8] E. Cuevas, M. González, D. Zaldívar, and M. Pérez-Cisneros, "Multi-ellipses detection on images inspired by collective animal behavior," *Neural Comput. Appl.*, vol. 24, no. 5, pp. 1019–1033, 2014.

[9] K.-C. Wong, K.-S. Leung, and M.-H. Wong, "Protein structure prediction on a lattice model via multimodal optimization techniques," in *Proc. Conf. Genet. Evol. Comput.*, Portland, OR, USA, 2010, pp. 155–162.

[10] X. Li, "Efficient differential evolution using speciation for multimodal function optimization," in *Proc. Conf. Genet. Evol. Comput.*, Washington, DC, USA, 2005, pp. 873–880.

[11] J.-P. Li, M. E. Balazs, G. T. Parks, and P. J. Clarkson, "A species conserving genetic algorithm for multimodal function optimization," *Evol. Comput.*, vol. 10, no. 3, pp. 207–234, 2002.

[12] C. L. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu, "Disburdening the species conservation evolutionary algorithm of arguing with radii," in *Proc. Conf. Genet. Evol. Comput.*, London, U.K., 2007, pp. 1420–1427.

[13] J. Gan and K. Warwick, "Dynamic niche clustering: A fuzzy variable radius niching technique for multimodal optimisation in GAs," in *Proc. IEEE Congr. Evol. Comput.*, Seoul, South Korea, 2001, pp. 215–222.

[14] W.-N. Chen *et al.*, "A novel set-based particle swarm optimization method for discrete optimization problems," *IEEE Trans. Evol. Comput.*, vol. 14, no. 2, pp. 278–300, Apr. 2010.

[15] Z.-H. Zhan *et al.*, "Cloudde: A heterogeneous differential evolution algorithm and its distributed cloud version," *IEEE Trans. Parallel Distrib. Syst.*, vol. 28, no. 3, pp. 704–716, Mar. 2017.

[16] X.-F. Liu *et al.*, "An energy efficient ant colony system for virtual machine placement in cloud computing," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 113–128, Feb. 2018.

[17] Y.-L. Li *et al.*, "Fast micro-differential evolution for topological active net optimization," *IEEE Trans. Cybern.*, vol. 46, no. 6, pp. 1411–1423, Jun. 2016.

[18] Y.-L. Li *et al.*, "Differential evolution with an evolution path: A deep evolutionary algorithm," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1798–1810, Sep. 2015.

[19] Y. H. Li, Z.-H. Zhan, S. J. Lin, J. Zhang, and X. N. Luo, "Competitive and cooperative particle swarm optimization with information sharing mechanism for global optimization problems," *Inf. Sci.*, vol. 293, pp. 370–382, Feb. 2015.

[20] G. Karafotias, M. Hoogendoorn, and A. E. Eiben, "Parameter control in evolutionary algorithms: Trends and challenges," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 167–187, Apr. 2015.

[21] R. A. Sarker, S. M. Elsayed, and T. Ray, "Differential evolution with dynamic parameters selection for optimization problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 5, pp. 689–707, Oct. 2014.

[22] N. M. Hamza, D. L. Essam, and R. A. Sarker, "Constraint consensus mutation-based differential evolution for constrained optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 447–459, Jun. 2016.

[23] T. Liao, K. Socha, M. A. Montes de Oca, T. Stützle, and M. Dorigo, "Ant colony optimization for mixed-variable optimization problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 503–518, Aug. 2014.

[24] M. A. Muñoz, M. Kirley, and S. K. Halgamuge, "Exploratory landscape analysis of continuous space optimization problems using information content," *IEEE Trans. Evol. Comput.*, vol. 19, no. 1, pp. 74–87, Feb. 2015.

[25] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano, "A distance-based ranking model estimation of distribution algorithm for the flow-shop scheduling problem," *IEEE Trans. Evol. Comput.*, vol. 18, no. 2, pp. 286–300, Apr. 2014.

[26] W.-N. Chen *et al.*, "Particle swarm optimization with an aging leader and challengers," *IEEE Trans. Evol. Comput.*, vol. 17, no. 2, pp. 241–258, Apr. 2013.

[27] G. R. Harik, "Finding multimodal solutions using restricted tournament selection," in *Proc. Int. Conf. Genet. Algorithms*, Pittsburgh, PA, USA, 1995, pp. 24–31.

[28] O. J. Mengshoel and D. E. Goldberg, "The crowding approach to niching in genetic algorithms," *Evol. Comput.*, vol. 16, no. 3, pp. 315–354, Sep. 2008.

[29] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," in *Proc. IEEE Congr. Evol. Comput.*, Portland, OR, USA, 2004, pp. 1382–1389.

[30] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Proc. 2nd Int. Conf. Genet. Algorithms*, Cambridge, MA, USA, 1987, pp. 41–49.

[31] B. Y. Qu, P. N. Suganthan, and J. J. Liang, "Differential evolution with neighborhood mutation for multimodal optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 5, pp. 601–614, Oct. 2012.

[32] W. Gao, G. G. Yen, and S. Liu, "A cluster-based differential evolution with self-adaptive strategy for multimodal optimization," *IEEE Trans. Cybern.*, vol. 44, no. 8, pp. 1314–1327, Aug. 2014.

[33] R. K. Ursem, "Multinational evolutionary algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Washington, DC, USA, 1999, pp. 1633–1640.

[34] J. Yao, N. Kharma, and Y. Q. Zhu, "On clustering in evolutionary computation," in *Proc. IEEE Congr. Evol. Comput.*, Vancouver, BC, Canada, 2006, pp. 1752–1759.

[35] C. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu, "Multimodal optimization by means of a topological species conservation algorithm," *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 842–864, Dec. 2010.

[36] Q. Yang *et al.*, "Multimodal estimation of distribution algorithms," *IEEE Trans. Cybern.*, vol. 47, no. 3, pp. 636–650, Mar. 2017.

[37] J. Yao, N. Kharma, and P. Grogono, "Bi-objective multipopulation genetic algorithm for multimodal function optimization," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 80–102, Feb. 2010.

[38] Q. Yang *et al.*, "Adaptive multimodal continuous ant colony optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 2, pp. 191–205, Apr. 2017.

[39] X. Li, "Niching without niching parameters: Particle swarm optimization using a ring topology," *IEEE Trans. Evol. Comput.*, vol. 14, no. 1, pp. 150–169, Feb. 2010.

[40] B. Y. Qu, P. N. Suganthan, and S. Das, "A distance-based locally informed particle swarm model for multimodal optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 3, pp. 387–402, Jun. 2013.

[41] S. Biswas, S. Kundu, and S. Das, "Inducing niching behavior in differential evolution through local information sharing," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 246–263, Apr. 2015.

[42] D. Zaharie, "A multipopulation differential evolution algorithm for multimodal optimization," in *Proc. Mendel*, 2004, pp. 17–22.

[43] S. Hui and P. N. Suganthan, "Ensemble and arithmetic recombination-based speciation differential evolution for multimodal optimization," *IEEE Trans. Cybern.*, vol. 46, no. 1, pp. 64–74, Jan. 2016.

[44] D. Zaharie, "Extensions of differential evolution algorithms for multimodal optimization," in *Proc. SYNASC*, Timişoara, Romania, 2004, pp. 523–534.

[45] B.-Y. Qu and P. N. Suganthan, "Novel multimodal problems and differential evolution with ensemble of restricted tournament selection," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, 2010, pp. 1–7.

[46] M. G. Epitropakis, X. Li, and E. K. Burke, "A dynamic archive niching differential evolution algorithm for multimodal optimization," in *Proc. IEEE Congr. Evol. Comput.*, Cancún, Mexico, 2013, pp. 79–86.

[47] S. Biswas, S. Kundu, and S. Das, "An improved parent-centric mutation with normalized neighborhoods for inducing niching behavior in differential evolution," *IEEE Trans. Cybern.*, vol. 44, no. 10, pp. 1726–1737, Oct. 2014.

[48] K. Deb and A. Saha, "Multimodal optimization using a bi-objective evolutionary algorithm," *Evol. Comput.*, vol. 20, no. 1, pp. 27–62, Mar. 2012.

[49] Y. Wang, H.-X. Li, G. G. Yen, and W. Song, "MOMMOP: Multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems," *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 830–843, Apr. 2015.

[50] K. V. Price, "An introduction to differential evolution," in *New Ideas in Optimization*. London, U.K.: McGraw-Hill, 1999, pp. 79–108.

[51] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl, "Constrained k-means clustering with background knowledge," in *Proc. Int. Conf. Mach. Learn.*, Williamstown, MA, USA, 2001, pp. 577–584.

[52] P. S. Bradley and U. M. Fayyad, "Refining initial points for k-means clustering," in *Proc. Int. Conf. Mach. Learn.*, Madison, WI, USA, 1998, pp. 91–99.

[53] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[54] B. H. Chen and J. L. Hu, "An adaptive niching EDA based on clustering analysis," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, 2010, pp. 1–7.

[55] Y.-H. Zhang, Y.-J. Gong, W.-N. Chen, and J. Zhang, "Composite differential evolution with queueing selection for multimodal optimization," in *Proc. IEEE Congr. Evol. Comput.*, Sendai, Japan, 2015, pp. 425–432.

[56] Z. H. Zhan *et al.*, "Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems," *IEEE Trans. Cybern.*, vol. 43, no. 2, pp. 445–463, Apr. 2013.

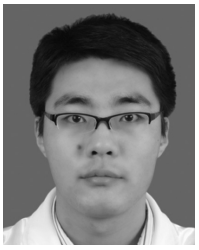
[57] Y.-L. Li, Y.-R. Zhou, Z.-H. Zhan, and J. Zhang, "A primary theoretical study on decomposition-based multiobjective evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 20, no. 4, pp. 563–576, Aug. 2016.

[58] M. Asafuddoula, T. Ray, and R. Sarker, "A decomposition-based evolutionary algorithm for many objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 3, pp. 445–460, Jun. 2015.

[59] S. Yang, M. Li, X. Liu, and J. Zheng, "A grid-based evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 721–736, Oct. 2013.

[60] J. Cheng, G. G. Yen, and G. Zhang, "A many-objective evolutionary algorithm with enhanced mating and environmental selections," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 592–605, Aug. 2015.

[61] M. Z. Ali, N. H. Awad, P. N. Suganthan, and R. G. Reynolds, "A modified cultural algorithm with a balanced performance for the differential evolution frameworks," *Knowl. Based Syst.*, vol. 111, pp. 73–86, Nov. 2016.
[62] N. H. Awad, M. Z. Ali, P. N. Suganthan, and R. G. Reynolds, "CADE: A hybridization of cultural algorithm and differential evolution for numerical optimization," *Inf. Sci.*, vol. 378, pp. 215–241, Feb. 2017.
[63] Y.-H. Zhang, M.-T. Li, Y.-J. Gong, and J. Zhang, "Differential evolution with random walk mutation and an external archive for multimodal optimization," in *Proc. IEEE Symp. Series Comput. Intell.*, Cape Town, South Africa, 2015, pp. 1868–1875.
[64] X. Li, A. Engelbrecht, and M. G. Epitropakis, "Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization," Evol. Comput. Mach. Learn. Group, RMIT Univ., Melbourne, VIC, Australia, Tech. Rep., 2013. [Online]. Available: http://goanna.cs.rmit.edu.au/~xiaodong/cec13-lsgo/competition/
[65] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.



Zi-Jia Wang (S'15) received the B.S. degree in automation from Sun Yat-sen University, Guangzhou, China, in 2015, where he is currently pursuing the Ph.D. degree.

He is currently a Research Fellow with the South China University of Technology, Guangzhou. His current research interests include evolutionary computation algorithms like differential evolution and particle swarm optimization and their applications in design and optimization, such as cloud computing resources scheduling.

Zhi-Hui Zhan (S'09–M'13) received the bachelor's and Ph.D. degrees from the Department of Computer Science, Sun Yat-sen University, Guangzhou, China, in 2007 and 2013, respectively.

He is currently a Professor with the School of Computer Science and Engineering, South China University of Technology, Guangzhou. He is also appointed as the Pearl River Scholar Young Professor in 2016. His current research interests include evolutionary computation algorithms, swarm intelligence algorithms and their applications in real-world problems, and in environments of cloud computing and big data.

Dr. Zhan was a recipient of the China Computer Federation Outstanding Dissertation and the IEEE CIS Outstanding Dissertation for his doctoral dissertation, the Natural Science Foundation for Distinguished Young Scientists of Guangdong Province, China, in 2014, the Pearl River New Star in Science and Technology in 2015, the Youth Talent in Science and Technology Innovation of Guangdong Province in 2016, and the Wu Wen Jun Artificial Intelligence Excellent Youth from the Chinese Association for Artificial Intelligence in 2017. He is listed as one of the Most Cited Chinese Researchers in Computer Science.

Ying Lin (M'13) received the B.Sc. degree in computer science and the Ph.D. degree in computer applied technology from Sun Yat-sen University, Guangzhou, China, in 2007 and 2012, respectively.

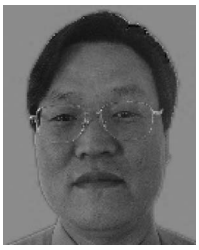
She is currently an Assistant Professor with the Department of Psychology, Sun Yat-sen University and also a Research Fellow with the South China University of Technology, Guangzhou. Her current research interests include computational intelligence and its applications in psychology.

Wei-Jie Yu (S'10–M'14) received the bachelor's and Ph.D. degrees from Sun Yat-sen University, Guangzhou, China, in 2009 and 2014, respectively.

He is currently a Lecturer with the School of Information Management, Sun Yat-sen University and also a Research Fellow with the South China University of Technology, Guangzhou. His current research interests include computational intelligence and its applications on intelligent information processing, big data, and cloud computing.

Hua-Qiang Yuan received the Ph.D. degree from Shanghai Jiao Tong University, Shanghai, China, in 1996.

He is currently a Professor with the School of Computer Science and Network Security, Dongguan University of Technology, Dongguan, China. His current research interests include computational intelligence and cyberspace security.



Tian-Long Gu received the M.Eng. degree from Xidian University, Xi'an, China, in 1987 and the Ph.D. degree from Zhejiang University, Hangzhou, China, in 1996.

From 1998 to 2002, he was a Research Fellow with the School of Electrical and Computer Engineering, Curtin University of Technology, Perth, WA, Australia, and a Post-Doctoral Fellow with the School of Engineering, Murdoch University, Murdoch, WA, Australia. He is currently a Professor with the School of Computer Science and Engineering, Guilin University of Electronic Technology, Guilin, China. His current research interests include formal methods, data and knowledge engineering, software engineering, and information security protocol.

Sam Kwong (F'13) received the B.Sc. degree in electrical engineering from the State University of New York at Buffalo, Buffalo, NY, USA, in 1983, the M.A.Sc. degree in electrical engineering from the University of Waterloo, Waterloo, ON, Canada, in 1985, and the Ph.D. degree from the University of Hagen, Hagen, Germany, in 1996.

From 1985 to 1987, he was a Diagnostic Engineer with Control Data Canada, where he designed the diagnostic software to detect the manufacture faults of the VLSI chips in the Cyber 430 machine. He later joined as a Member of Scientific Staff with Bell Northern Research, Ottawa, ON, Canada. He joined as a Lecturer with the Department of Electronic Engineering, City University of Hong Kong, Hong Kong, in 1990. He is currently a Professor with the Department of Computer Science. His current research interests include pattern recognition, evolutionary computations, and video analytics.

Prof. Kwong is appointed as a IEEE Distinguished Lecturer for IEEE SMC Society in 2017. He has been the Vice President for IEEE Systems, Man and Cybernetics for conferences and meetings since 2014.

Jun Zhang (F'17) received the Ph.D. degree from the City University of Hong Kong, Hong Kong, in 2002.

He is currently a Changjiang Chair Professor with the School of Computer Science and Engineering, South China University of Technology, Guangzhou. His current research interests include computational intelligence, cloud computing, high performance computing, data mining, wireless sensor networks, operations research, and power electronic circuits. He has published over 100 technical papers in the above areas.

Dr. Zhang was a recipient of the China National Funds for Distinguished Young Scientists from the National Natural Science Foundation of China in 2011 and the First-Grade Award in Natural Science Research from the Ministry of Education, China, in 2009. He is currently an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, the IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, and the IEEE TRANSACTIONS ON CYBERNETICS.