Differential Evolution for Multimodal Optimization With Species by Nearest-Better Clustering

Xin Lin, Wenjian Luo, *Senior Member, IEEE*, and Peilan Xu

***Abstract*—Multimodal optimization problems (MMOPs) are common in real-world applications and involve identifying multiple optimal solutions for decision makers to choose from. The core requirement for dealing with such problems is to balance the ability of exploration in the global space and exploitation in the multiple optimal areas. In this paper, based on the differential evolution (DE), we propose a novel algorithm focusing on the formulation, balance, and keypoint of species for MMOPs, called FBK-DE. First, nearest-better clustering (NBC) is used to divide the population into multiple species with minimum size limitations. Second, to avoid placing too many individuals into one species, a species balance strategy is proposed to adjust the size of each species. Third, two keypoint-based mutation operators named DE/keypoint/1 and DE/keypoint/2 are proposed to evolve each species together with traditional mutation operators. The experimental results of FBK-DE on 20 benchmark functions are compared with 15 state-of-the-art multimodal optimization algorithms. The comparisons show that the proposed FBK-DE performs competitively with these algorithms.**

***Index Terms*—Differential evolution (DE), multimodal optimization problems (MMOPs), nearest-better clustering (NBC).**

I. Introduction

IN REAL-WORLD applications, some problems are very complex or have physical constraints. Thus, decision makers may need more than one optimal candidate solution to choose from; when a candidate solution is unsatisfactory, a decision maker can quickly select another candidate solution [1], [2].

Multimodal optimization problems (MMOPs) are problems that contain multiple global and local peaks. The goal of MMOPs is to find all optimal solutions to provide decision makers with multiple choices; thereby, decision makers can choose their preferred optimal solutions [2]. In reality, we usually care more about global optimal solutions instead of local optima. For the sake of convenience, the goal of MMOPs in this paper is to catch all the global optima.

Evolutionary algorithms (EAs) [3], [4] are a type of stochastic optimization methods that maintain a population for tracking the best solution for an optimization problem. In every iteration, the population evolves, converges to, and finally locates at the optimal solution. EAs have been used to solve various optimization problems, such as MMOPs [5] and dynamic optimization problems (DOPs) [6].

However, standard EAs are not suitable for MMOPs without modifications. Because standard EAs converge to one optimum, it is difficult to find all global optima of MMOPs. Owing to randomness, EAs may converge to different optima in different runs. Thus, to solve MMOPs, researchers have been working to drive the population toward multiple global optima while ensuring that the population converges. In other words, how to balance the ability of exploration in the global space and exploitation in multiple optimal areas for MMOPs by EAs are an important topic of study.

In fact, in order to solve various optimization problems including MMOPs, many strategies have designed for EAs to balance the exploration and the exploitation abilities. For example, Epitropakis *et al.* [7] adopted an additional space, which called archive strategy, to preserve some best historical solutions for dealing with MMOPs. Gong and Cai [8] proposed a modified mutation operator based on the ranking information to solve the global optimization problems with multimodal local optima. It means that some offspring are generated by the parents selected according to the ranking information. In [9], to improve the performance on MMOPs, Li divided the whole swarm into several subswarms. Each subswarms evolved independently so that the swarm would not converge into one optimal position. Besides, Bose *et al.* [10] proposed an improved version of artificial bee colony (ABC) algorithm for DOPs, where multiple subpopulations approach is used to effectively prevent fast convergence of the population.

For MMOPs, niching methods are common methods to find and preserve multiple optimal solutions [2], [5]. They implicitly or explicitly divide the population into several subpopulations where the algorithm searches for the best solutions. These methods include speciation [11], [12], crowding [13], [14], fitness sharing [15], clustering [16], derating [17], restricted tournament selection [18], and neighborhood mutation [19]–[21].

In nature, a species means a relatively independent ecological unit. Similarly, in the MMOPs, the individuals gather near the optimal positions, naturally forming multiple species.

Manuscript received November 13, 2018; revised February 2, 2019; accepted March 14, 2019. This work was supported in part by the National Natural Science Foundation of China under Grant 61573327. This paper was recommended by Associate Editor P. N. Suganthan. *(Corresponding author: Wenjian Luo.)*

The authors are with the School of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China, and also with the Anhui Province Key Laboratory of Software Engineering in Computing and Communication, University of Science and Technology of China, Hefei 230027, China (e-mail: iskcal@mail.ustc.edu.cn; wjluo@ustc.edu.cn; xpl@mail.ustc.edu.cn).

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

Digital Object Identifier 10.1109/TCYB.2019.2907657

Li [22] proposed the algorithm called species differential evolution (SDE), which combined the specification and differential evolution (DE). SDE looks for the seeds of all species and utilizes the species radius r_s to determine which species the other individuals belong to. In each species, a basic DE is performed to track multiple optima. However, the species radius r_s is hard to decide. If r_s is too small, there are few individuals in the species, which leads to low efficiency of DE, while too large, a species may contains too much individuals locating at different optima.

Nearest-better clustering (NBC) [23] method is another technology to divide the population into multiple species. The main idea is that the distance between each two individuals near different optima is much larger than the weighted average distance of the rest individuals to their nearest-better neighbors. NBC does not require the r_s to define the boundary of species. Instead, it would cut off some large edges to naturally form the species. However, it still has shortcomings. For example, if the distances among the optimal solutions vary greatly, it is hard to distinguish two individuals locating at two relatively close optima, which results in multiple optimal solutions in a species. The details would be discussed in Section II.

To deal with the disadvantages, we design some improvement mechanisms for the species. In this paper, we propose a novel algorithm named FBK-DE based on DE for MMOPs. In FBK-DE, we pay much attention on the formulation, balance, and keypoint of species. In other words, to find more global optima, several techniques are considered to enable FBK-DE to better balance the exploration and exploitation of the population as follows.

1) In this paper, NBC is adopted to divide the population into species. However, the standard NBC could generate some species with very small sizes, which is not beneficial to a fast convergence to the global optima. Therefore, we modified the standard NBC by limiting the minimum size of each species.
2) Because a few species could occupy most individuals of the population, we proposed a species balance strategy to balance the size of each species in FBK-DE. Thus, FBK-DE could prevent the population from converging to only a few optima with relatively large peak widths while neglecting others.
3) Since one species may contain several individuals locating at different global optima, we execute the original NBC over each species to recognize the internal seeds, which are defined as keypoints. To improve the performance of finding more than one optimum in a species, two keypoint-based mutation operators are proposed: DE/keypoint/1 and DE/keypoint/2. These mutation operators are very useful when two or more global optima are relatively close and cannot be separated by the NBC.

The rest of this paper is organized as follows. Section II reviews the basic concepts, including DE, NBC, and some other works about multimodal optimization. Section III illustrates the detailed process of FBK-DE. Benchmark problems, performance measurements, and experimental results of FBK-DE are presented in Section IV. The comparisons of different algorithms and differential components in FBK-DE and a discussion on FBK-DE are shown in Section V. Finally, a short conclusion is given in Section VI.

II. Related Work

In this section, we first provide a brief description of the standard DE. Following this, the NBC method is summarized. Finally, the work about multimodal optimization is reviewed.

A. Differential Evolution

As one of the EAs, DE [24], [25] has been used to handle a wide variety of optimization problems, such as the global optimization problem [24], MMOP [26], multiobjective optimization problem [27], [28], DOP [29], etc. In the process of evolution, DE spreads the distance and direction information among the current population to find the global optima. Similar to other EAs, DE also has four operators: 1) initialization; 2) mutation; 3) crossover; and 4) selection.

A DE begins by generating an initial population. In general, the initial population is randomly generated, which can be shown as

$$x_i^j(0) = \text{rand}(0, 1) \times \left(x_{\max}^j - x_{\min}^j\right) + x_{\min}^j \tag{1}$$

where $x_i^j(g)$ represents the jth dimension value of the ith individual $\boldsymbol{x}_i$ at gth generation. $x_{\max}^j$ and $x_{\min}^j$ represent the upper and lower boundaries of the jth dimension. Subscripts i and j are in the range [1, NP] and [1, D], where NP represents the size of the population and D represents the dimensions of the problem. rand(0, 1) represents a uniformly distributed random value generated from 0 to 1.

After the initialization, DE utilizes the mutation operator to generate the mutant individuals. For the individual $\boldsymbol{x}_i$, there are a variety of mutation operators to generate the corresponding mutant $\boldsymbol{v}_i$. Common mutation operators are as follows.

1) "DE/rand/1"

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

2) "DE/rand/2"

$$\begin{aligned}\boldsymbol{v}_i(g) = \boldsymbol{x}_{r1}(g) &+ F \cdot (\boldsymbol{x}_{r2}(g) - \boldsymbol{x}_{r3}(g)) \\ &+ F \cdot (\boldsymbol{x}_{r4}(g) - \boldsymbol{x}_{r5}(g)).\end{aligned} \tag{3}$$

3) "DE/best/1"

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

4) "DE/best/2"

$$\begin{aligned}\boldsymbol{v}_i(g) = \boldsymbol{x}_{\text{best}}(g) &+ F \cdot (\boldsymbol{x}_{r1}(g) - \boldsymbol{x}_{r2}(g)) \\ &+ F \cdot (\boldsymbol{x}_{r3}(g) - \boldsymbol{x}_{r4}(g))\end{aligned} \tag{5}$$

where the subscripts $r1$, $r2$, $r3$, $r4$, and $r5$ are randomly picked from $\{1, 2, 3, \ldots, \text{NP}\}\backslash\{i\}$ and mutually exclusive. The individual $\boldsymbol{x}_{\text{best}}$ represents the individual whose fitness value is the best in the current population. The scale factor F scales the difference vectors to control the mutation step. It is clear that more than one difference vector could be used in DE. The greater the number of difference vectors, the greater the

number of combinations, which leads to a high diversity. Joshi and Sanderson [30] discussed the relationship between the number of difference vectors and the size of the population.

After that the mutation operator generates the mutant individuals, the crossover operator implements a recombination between the parent individuals and the mutant ones to generate the offspring. The crossover operator used in this paper is given as follows:

$$u_i^j(g) = \begin{cases} v_i^j(g) & \text{if } \text{rnd}_i^j \le \text{CR or } j = j_{\text{rand}} \\ x_i^j(g) & \text{otherwise} \end{cases} \tag{6}$$

where CR represents the crossover rate, which is usually set to a fixed value lying in the range [0, 1]. rnd_i^j represents the random decimal obeying the uniform distribution between 0 and 1 (for $\boldsymbol{x}_i$ at jth dimension). j_{rand} represents the dimension index, which is randomly selected from 1 to D. If all the random values of the current individual are greater than CR, there must be one randomly selected dimension where the value comes from v_i^j.

Following the crossover operator, the selection operator is utilized to choose the best individual from $\boldsymbol{u}_i(g)$ and $\boldsymbol{x}_i(g)$ into the next generation shown as follows:

$$\boldsymbol{x}_i(g+1) = \begin{cases} \boldsymbol{u}_i(g) & f(\boldsymbol{u}_i(g)) \ge f(\boldsymbol{x}_i(g)) \\ \boldsymbol{x}_i(g) & \text{otherwise} \end{cases} \tag{7}$$

where $f(.)$ represents the fitness function that should be maximized. It should be noted that all the problems in this paper should be maximized.

Finally, DE continues to perform mutation, crossover, and selection operators until the termination condition is satisfied.

B. Nearest-Better Clustering

NBC is commonly adopted in MMOPs to divide the population into several species and simultaneously track multiple optimal solutions; ideally, each species tries to find an optimal solution. NBC was proposed by Preuss [23] and has some applications in practical problems, such as the question of deciding virtual camera composition (VCC) in a 3-D environment [31]. NBC also has some applications on the high-dimensional problems. For example, Luo *et al.* [32] utilized a differential grouping technique to divide high-dimensional problems into several low-dimensional subproblems, which are then solved by an NBC-based particle swarm optimization (PSO), where the parameter φ of NBC is initially set to 1.2 and then gradually increased to 2.0 during the algorithm run time.

The process of NBC is shown in Algorithm 1. First, the population is sorted by the fitness in descending order. Second, the Euclidean distances between each pair of individuals must be calculated. According to these distances, each individual (except for the best one) finds its own nearest-better neighbor and an edge is created to connect them (illustrated in lines 4–8). For convenience, the nearest-better neighbor is called the leader individual for two nodes connected by each edge, while the individual itself is referred to as the follower individual. Based on this connection, a spanning tree T is formed. Following this, the mean distance (i.e., length) μ_{dist} of these edges is calculated. Next, all the edges whose distance is greater than $\varphi \times \mu_{\text{dist}}$ are cut off, where φ is the scale factor of NBC. Finally, the entire spanning tree T is divided into several subtrees; each subtree represents a species and the root of the subtree is regarded as the seed of the corresponding species.

Algorithm 1 NBC [23]

1: Sort the population P by the fitness F from the best to the worst;
2: Calculate the distances between each pair of individuals;
3: Create an empty tree T;
4: **for** each $\boldsymbol{x}_i \in P$ **do**
5: Find the nearest better neighbor $\boldsymbol{x}_{i,nb}$;
6: Create an edge between $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i,nb}$
7: Add the edge to T;
8: **end for**
9: Calculate the mean distance μ_{dist} of all edges in T;
10: **for** each $e \in T$ **do**
11: **if** $dist(e) > \varphi \times \mu_{dist}$ **then**
12: Cut off the edge e and record the seed;
13: **end if**
14: **end for**

NBC performs very well when compared with existing niching methods. Moreover, NBC has several advantages, as shown in [33]. One particular advantage is that NBC does not require prior knowledge (e.g., the number of peaks). Another advantage is that NBC requires only one parameter φ to be set. The parameter φ controls the number of species. When φ is small, more species are obtained; while increasing φ results in fewer species. In general, φ is set to 2.0 [33].

However, each algorithm has its own unsuitable situations (including NBC). Such situation is the problem of large distance gaps among the peaks. As shown in Fig. 1, peak p_b is very close to peak p_a, but both are far from peak p_c. Assuming that all the peaks are located at by the current population (the individual $\boldsymbol{x}_a, \boldsymbol{x}_b$, and $\boldsymbol{x}_c$), the distance between individuals $\boldsymbol{x}_b$ and $\boldsymbol{x}_c$ is so large that the weighted mean distance is exceedingly large, which means that the edge between individuals $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ may not be cut off. Therefore, three individuals form two species (i.e., the species s_a and s_c). If a mutation operator with a strong convergence is adopted, all individuals in species s_a are attracted by peak p_b, resulting in the loss of peak p_a.

C. Multimodal Optimization

To deal with the problems with the multimodal environment, many strategies have been designed and embedded into the stochastic optimization algorithms.

Because niching methods are highly compatible with EAs, they have been embedded into various EAs to enhance the ability to search for multiple solutions. For example, Shir and Bäck [34] combined the dynamic niching method proposed in [35] with evolution strategies (ESs) to solve MMOPs. DE is one type of EAs used for solving various optimization problems [36]. Some work has been undertaken to combine DE with niching methods to tackle MMOPs.

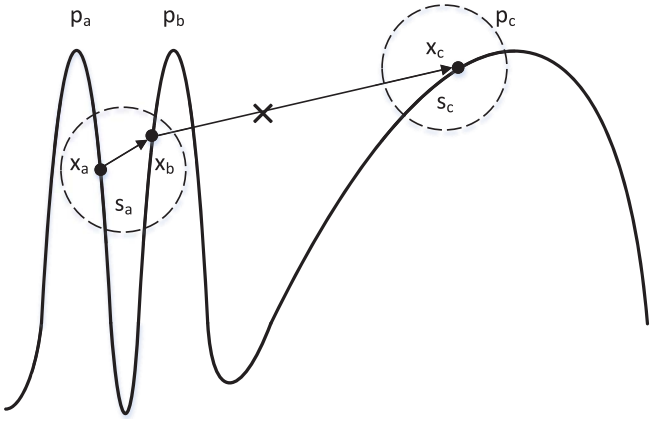


Fig. 1. Example of the problem where NBC is unsuitable.

Thomsen [26] embedded fitness sharing and crowding schemes into DE to form SharingDE and CDE, respectively. SharingDE utilizes the well-known sharing scheme from [15] in order to prevent individuals from converging to one peak. In addition, CDE adopts standard DE to generate offspring; however, the offspring is only competed with the most similar parents, and the best are put into the next generation. Qu *et al.* [19] proposed a neighborhood mutation strategy embedded into CDE and SDE. In their proposed algorithms (NCDE and NSDE), mutation is performed in each Euclidean neighborhood. Besides, Epitropakis *et al.* [37] modified the basic mutation operators which make the proposed algorithm better adapt to multimodal environments. For each individual, its offspring is generated by its nearest neighbor instead of itself. Gong *et al.* [38] utilized an operator called diversity preserving operator to deal with the neighborhoods which are converged or overlapped. Tuo *et al.* [39] adopted the local adjustment strategy to adjust the offspring which gives the proposed algorithm a stronger ability to explore the decision space on the high-dimensional MMOPs. Hui and Suganthan [40] proposed an ensemble scheme based on SDE. Because it is hard to decide the species radius r_s, the authors utilized different kinds of parameters and strategies to calculate the species radius. In these niching methods, some extra parameters are usually introduced. Recently, in [41], DE$_{cl}$ proposed by Bošković and Brest uses a self-adaptive parameter mechanism to control the algorithm parameters (including the population size).

Some variants of PSO have been applied to MMOPs. Parsopoulos and Vrahatis [42] introduced a stretching technique into PSO to properly separate the swarms and escape from the local optima. Sasaki *et al.* [43] organized several populations into a network. Each individual in each population not only acquires knowledge from its own *pbest* and *lbest* in the current population but also is evolved by the *gbest* among its neighbor populations. Li [44] considered that the *lbest* PSO with a ring topology naturally traced to the basins of attraction, thus, the swarms could finally catch optimal solutions. Brits *et al.* [45] proposed *nbest* PSO to deal with multimodal occasion. The *nbest* individual is the position center of several closest neighbor individuals. In addition, multiple swarms method is another strategy used in PSO to solve MMOPs. NichePSO [46], [47], proposed by Brits *et al.*, adopts a main swarm to search for the promising area. If some competitive individuals are found in the main swarm, these ones would be further evolved in a subswarm to find better solutions. In order to prevent multiple subswarms from locating the same optimal solution, NichePSO would merge these overlapping subwarms.

Besides, multiobjective evolutionary optimization algorithms have also been used to solve MMOPs [48], [49]. For example, recently, Wang *et al.* [50] designed a set of two contradictory objective functions, which guaranteed that the optimal solution must lie within the Pareto solution set.

III. Proposed Approach

In this section, we propose a novel algorithm based on DE named FBK-DE to solve MMOPs. First, the modified NBC with the limitation of minimum species size (called NBC-minsize for convenience) is proposed. Second, a species balance strategy is introduced. Next, two keypoint-based mutation operators are designed for MMOPs. Finally, the procedure and the time complexity analysis of FBK-DE are presented.

A. NBC-Minsize

As discussed in Section II-B, the parameter φ controls the number of species. When we set φ to a relatively small value, the population could be divided into more species. Thus, more promising areas could be located by different species. In this paper, the parameter φ is set to 1.0.

However, a small φ value results in excessive segments. Moreover, the number of species containing only a few individuals (e.g., one or two individuals) would greatly increase, making the species incapable of evolving with the mutation operators of DE.

Therefore, in this paper, the parameter *minsize* is used to limit the minimal number of individuals in the species. It is noted that the value of *minsize* must be carefully handled. If it is too small, the minimum size limitation mechanism is ineffective; if it is too large, it is very likely that two species locating at different peaks would be still linked together.

In order to obtain better results, *minsize* should take a small value in the early stage and relatively large values in the middle and later stages. In the early stage, since the population does not locate at the peaks, it should be divided into more species. In the middle and later stages, because the population approximately (or precisely) locates at multiple peaks, a larger *minsize* allows the species that locate the local optima to be incorporated into the near species that locate the global optima. Thus, the species converge to the global optima more quickly.

In this paper, the value of *minsize* is set as follows:

$$minsize(g) = 5 + g/2 \tag{8}$$

where g represents the current generation and $minsize(g)$ represents the minsize value at gth generation. Parameter *minsize* is initially set at 5, ensuring that the DE mutation operators (e.g., DE/rand/2) execute properly, then incremented by 1 for every two generations. However, in the later stages of the algorithm execution, since g is very large, it causes *minsize* to be very large. Therefore, we set a upper bound to *minsize*. In

Algorithm 2 NBC-Minsize

1: Set *minsize* by equation (8), (9);
2: Construct the spanning tree T;
3: Calculate the mean distance μ_{dist};
4: Calculate the *follow* vector;
5: Sort the edges in T from the longest to the shortest;
6: **for** each $e \in T$ **do**
7: **if** $dist(e) > \varphi \times \mu_{dist}$ **then**
8: Set e_f to the follower individual of e;
9: Set e_r to the root of the subtree containing e_f;
10: **if** $follow(e_f) \geq minsize$ **and** $follow(e_r) - follow(e_f) \geq minsize$ **then**
11: Cut off e;
12: Set e_l to the leader individual of e;
13: **for** each x on the path from e_l to e_r **do**
14: $follow(x) = follow(x) - follow(e_f)$;
15: **end for**
16: **end if**
17: **end if**
18: **end for**

Algorithm 3 Species Balance Strategy

1: $rest = 0$;
2: $\mu_{avg} = mean(nums)$;
3: $\mu_\lambda = round(\lambda \cdot \mu_{avg})$;
4: $s = \varnothing$
5: **for** $i = 1$ **to** $len(nums)$ **do**
6: **if** $nums(i) > \mu_\lambda$ **then**
7: $rest = rest + nums(i) - \mu_\lambda$;
8: $nums(i) = \mu_\lambda$;
9: **else if** $nums(i) < \mu_{avg}$ **then**
10: $s = s \cup \{i\}$;
11: **end if**
12: **end for**
13: $nums(s) = nums(s) + floor(rest/len(s))$;
14: $i = 0$;
15: **while** $rest > 0$ **do**
16: $nums(s(i)) = nums(s(i)) + 1$;
17: $i = i + 1$;
18: **end while**

other words, when *minsize* reaches the upper bound *bound*, *minsize* remains the same. For different problems, *bound* is different. The value of *bound* is defined as follows:

$$bound = \max(10, 3 * D). \tag{9}$$

That is to say, when D is less than 4, *minsize* increases from 5 to 10, when D is greater than or equal to 4, *minsize* increases from 5 to $3 * D$.

The NBC-minsize is shown in Algorithm 2. First, the value of *minsize* of species is set by (8) and (9). Second, a spanning tree is constructed identical to the standard NBC and the mean distance μ_{dist} is calculated. Following this, we calculate the *follow* vector. Each element of *follow* represents the number of nodes in the subtree rooted at the corresponding individual; specifically, each value of *follow* is initially set to 1. Next, the edges are sorted in descending order according to the fitness values of the follower individuals. Finally, for each edge, the *follow* value of each follower individual is added to that of a leader individual. Following this, the edges in T are sorted by their Euclidean distance in descending order. Subsequently, edges satisfying the conditions are cut off. In addition to the condition of standard NBC, that the distance exceeds the weighted mean distance, the other condition is that the sizes of the two species after the cutoff must be all greater than or equal to *minsize* (illustrated in lines 8–15). Finally, the partition scheme is returned and the species are obtained.

B. Species Balance Strategy

To prevent only a few species from occupying most individuals in the population, we propose a species balance strategy in this paper. Owing to different fitness landscapes surrounding different optima, the difficulty in locating different optima is also different. When the basin area of an optimum is large, more individuals will move to that optimum, and it is relatively easy to find. However, when the basin area of an optimum is small, fewer individuals will move to the corresponding optimum. Therefore, some excessive individuals should be deleted, and the species with small sizes should be expanded.

Algorithm 3 shows the species balance strategy, where *nums* is a vector that represents the sizes of all species and *round*(.) stands for the rounding function. First, we initialize a variable *rest* to record the number of individuals being removed from the large species. Next, the threshold value μ_λ and mean size μ_{avg} are calculated. Following this, an empty set s is used to record the index of species whose size is less than μ_{avg}. Furthermore, in lines 6–8, the values of *nums* that exceed the threshold are set to μ_λ and the reduced amounts are accumulated and recorded in *rest*. Finally, in the 13th–18th lines, all *nums* that are less than μ_{avg} are incremented by 1 in order until *rest* is exhausted, which ensured that the sum of *nums* is still equal to NP. It should be noted that the weight λ should be no less than 1.0; otherwise, a balance among the species could not be achieved.

According to Algorithm 3, the size of a species will either increase, decrease, or remain unchanged. If the size of a species decreases or remains unchanged, its offspring are still generated by the DE operators. However, if the size increases, the offspring with the original size of species are generated by the DE operators, but the increments are generated by the Gaussian distribution around $\boldsymbol{s}_{\text{seed}}$. Here, $\boldsymbol{s}_{\text{seed}}$ refers to the best individual in the species s. The details are as follows:

$$\begin{aligned} \boldsymbol{x}_{\text{rest}} &= \boldsymbol{s}_{\text{seed}} + N(0, 0.1) \\ x^j_{\text{rest}} &= \max\left(x^j_{\text{rest}}, s^j_{lb}\right) \\ x^j_{\text{rest}} &= \min\left(x^j_{\text{rest}}, s^j_{ub}\right) \end{aligned} \tag{10}$$

where $\boldsymbol{x}_{\text{rest}}$ represents an increased individual of the species and $N(0, 0.1)$ represents a Gaussian distribution with a mean of 0 and a standard deviation of 0.1. s^j_{lb} and s^j_{ub} refer to the minimum value and maximum value of the jth dimension of all

individuals in the corresponding species s. The latter two equations indicate that in each dimension, values above the upper boundary or below lower boundary are set to the corresponding boundary.

C. Keypoint-Based Mutation Operators

As stated in Section II-B, when the distances among multiple optima vary greatly, which is illustrated in Fig. 1, it is difficult for NBC to distinguish the nearby optima. After achieving the species s_a and s_b, it can be found that s_a contains $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ which locate at p_a and p_b, respectively. If we execute original NBC method in s_a, the edge between $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ could be cut off, where $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$, which should be the seed after separation, are regarded as two keypoints of the species s_a. In other words, the keypoints are the seeds in the species after the original NBC (i.e., the set of $\boldsymbol{x}_{kp}$). It is noting that the original NBC is only used to find the keypoints rather than to separate the individuals in the species again, because if the species is divided, the size may be too small for the species to evolve. Based on the situation, two keypoint-based mutation operators are proposed in this paper (DE/keypoint/1 and DE/keypoint/2) to improve the performance.

DE/keypoint/1 and DE/keypoint/2 are given as follows.

1) "DE/keypoint/1"

$$\boldsymbol{v}_i(g) = \boldsymbol{x}_{kp}(g) + F \cdot (\boldsymbol{x}_{r1}(g) - \boldsymbol{x}_{r2}(g)). \tag{11}$$

2) "DE/keypoint/2"

$$\begin{aligned}\boldsymbol{v}_i(g) = \boldsymbol{x}_{kp}(g) &+ F \cdot (\boldsymbol{x}_{r1}(g) - \boldsymbol{x}_{r2}(g)) \\ &+ F \cdot (\boldsymbol{x}_{r3}(g) - \boldsymbol{x}_{r4}(g))\end{aligned} \tag{12}$$

where $\boldsymbol{x}_{kp}$ represents an individual randomly selected from the keypoints in the species. When we execute NBC over a species, the parameter φ is denoted as φ_{kp}. It is worth noting that since the size of the species may be relatively small in the early stages, $r1, r2, r3, r4$, and $r5$ are randomly picked from $\{1, 2, 3, \ldots, sNP\}$ instead of $\{1, 2, 3, \ldots, sNP\}\backslash\{i\}$, where sNP is the size of the species.

Furthermore, the convergence of keypoint-based mutation operators is still so strong that other operators are needed to increase the diversity. In Section II, several common DE mutation operators are given. In FBK-DE, we adopt DE/rand/1, DE/rand/2, DE/keypoint/1, and DE/keypoint/2 to generate the mutant individuals.

The four mutation operators are separated into two groups. One group contains DE/rand/1 and DE/rand/2, whose exploration ability is strong and should be used with high probability in the early stages. The other group contains DE/keypoint/1 and DE/keypoint/2, with a high probability in the later stage. Within each group, two operators are randomly selected. The selection probabilities of the two groups are linked to the number of evaluations. Assuming that per is set to the first group's selection probability, per is set as follows:

$$per = 1 - (evals/\text{MaxFEs})^{\alpha} \tag{13}$$

where $evals$ is the current number of evaluations consumed and MaxFEs is the total number of fitness evolutions. Parameter α balances the ability of exploration and exploitation. When α exceeds 1, the exploration ability is strengthened, and the diversity of the population is improved. Conversely, when α is larger than 0 and less than 1, the exploitation ability is enhanced and the population converges more effectively. In this paper, the default value of α is set to (1/2).

Algorithm 4 Mutation Strategy in FBK-DE

```
Randomly generate two decimal numbers type1 and type2 between 0 and 1;
if type1 < per then
    if type2 < 0.5 then
        Generate the mutant v_i by DE/rand/1;
    else
        Generate the mutant v_i by DE/rand/2;
    end if
else
    if type2 < 0.5 then
        Generate the mutant v_i by DE/keypoint/1;
    else
        Generate the mutant v_i by DE/keypoint/2;
    end if
end if
```

Algorithm 5 FBK-DE

```
Set the population size NP;
Randomly generate the initial population P(0);
Evaluate the initial population P(0);
g = 0;
while the termination condition is not satisfied do
    Obtain the species by Algorithm 2;
    Obtain the size of the species by Algorithm 3;
    for each species s_i do
        for i=1 to min(sNP_i, nums(i)) do
            Generate v_i(g) by Algorithm 4;
            Generate u_i(g) by Equation (6);
            Put the better into P(g+1) by Equation (7);
        end for
        if nums(i) > sNP_i then
            Generate new individuals by Equation (10);
            Insert the new one into P(g+1);
        end if
    end for
    g = g + 1;
end while
```

Algorithm 4 shows the mutation strategy used in FBK-DE. First, two random numbers are generated. Next, $type1$ decides which group of mutation operators to use, and $type2$ decides which mutation operator is within the group. Finally, the mutant individual is produced by the corresponding mutation operator and returned. As for the other DE operators, they are the same as those in the standard DE.

D. FBK-DE

Based on these above components, the process of FBK-DE is shown in Algorithm 5.

First, the size of population is determined by the dimension of the problem (i.e., D) and the maximum number of evaluations (i.e., MaxFEs). The details are given in Section IV-C.

Second, the population is randomly generated by (1) and evaluated by the benchmark functions.

Third, the NBC-minsize (i.e., Algorithm 2) is utilized to divide the population into several species. Following this, the species balance scheme (i.e., Algorithm 3) is applied to decide the size of each species. After balancing the size of the species, $\min(sNP_i, nums(i))$ offspring are generated by the mutation operator (i.e., Algorithm 4) and crossover operator and the superior ones [from $\boldsymbol{u}_i(g)$ and $\boldsymbol{x}_i(g)$] are placed into the next generation population by the selection operator. When $nums(i)$ is larger than sNP_i, additional individuals are generated using (10) and directly inserted into the next generation population. Finally, this process is repeated until the termination condition is met.

As stated above, FBK-DE contains three mechanisms, including NBC-minsize, species balance strategy, and keypoint-based mutation operators. Here, we analyze the time complexity of each mechanism. In Algorithm 2, there are two time-consuming operations. One is to construct the spanning tree where the distance of each two points should be calculated, and the other is to check whether the edge should be cut off according to minsize. The time complexity of former operation is $O(n^2)$ and the latter is $O(n)$, where n is the population size. Therefore, the complexity of NBC-minsize is $O(n^2)$. From Algorithm 3, it can be seen that the time complexity of the species balance strategy is less than $O(n)$, because it is impossible that each individual forms a species. Finally, in the keypoint-based mutation operators, we would execute the original NBC to recognize the keypoints. However, there is no need to calculate the distances between each pair of individuals, since we obtain the distances in NBC-minsize. Based on that, the time complexity of identifying the keypoints is $O(n)$, and DE operators over each species requires at most $O(n)$. Therefore, the time complexity of FBK-DE is $O(\text{MaxGen} * n^2)$, where MaxGen is the maximum evolutionary generations allowed.

IV. Experiments

A. Benchmark Problems

In order to check the performance of FBK-DE, we use the 20 test functions from the competition on niching methods for multimodal function optimization [51]. Table I shows the information for the benchmark problems and the population size used in FBK-DE, where NKP is the number of the global peaks and MaxFEs is defined as the total number of fitness evaluations. r represents the niching radius and NP represents the population size, which are introduced in Section IV-C.

In these benchmark functions, the 1st–5th functions are simple functions, and the 6th–10th functions are functions which have a large number of global optima. Furthermore, the function $F7$ indicates a large gap between peaks. The 11th–20th functions are the composition functions with many local peaks, among which the 16th–20th functions are relatively high dimensional.

TABLE I
Information of the Benchmark Problems and the Population Size

Index	Function	NKP	Peak height	r	$MaxFEs$	NP
1	$F_1(1D)$	2	200.0	0.01	5.0E+4	250
2	$F_2(1D)$	5	1.0	0.01	5.0E+4	250
3	$F_3(1D)$	1	1.0	0.01	5.0E+4	250
4	$F_4(2D)$	4	200.0	0.01	5.0E+4	250
5	$F_5(2D)$	2	1.03163	0.5	5.0E+4	250
6	$F_6(2D)$	18	186.731	0.5	2.0E+5	1000
7	$F_7(2D)$	36	1.0	0.2	2.0E+5	1000
8	$F_6(3D)$	81	2709.0935	0.5	4.0E+5	2000
9	$F_7(3D)$	216	1.0	0.2	4.0E+5	2000
10	$F_8(2D)$	12	-2.0	0.01	2.0E+5	1000
11	$F_9(2D)$	6	0	0.01	2.0E+5	1000
12	$F_{10}(2D)$	8	0	0.01	2.0E+5	1000
13	$F_{11}(2D)$	6	0	0.01	2.0E+5	1000
14	$F_{11}(3D)$	6	0	0.01	4.0E+5	2000
15	$F_{12}(3D)$	8	0	0.01	4.0E+5	2000
16	$F_{11}(5D)$	6	0	0.01	4.0E+5	1334
17	$F_{12}(5D)$	8	0	0.01	4.0E+5	1334
18	$F_{11}(10D)$	6	0	0.01	4.0E+5	1334
19	$F_{12}(10D)$	8	0	0.01	4.0E+5	1334
20	$F_{12}(20D)$	8	0	0.01	4.0E+5	1334

TABLE II
Parameters in FBK-DE

Parameters	Values
φ	1.0
φ_{kp}	2.0
λ	2.0
α	0.5
CR	0.9
F	0.2∼0.8 (one differential vector) 0.5 (two differential vectors)
$MaxGen$	200 ($D < 5$) 300 ($D \geq 5$)

B. Performance Measurements

In addition to the benchmark problems, a quantified method is required to show the performance of FBK-DE on these functions. Thus, we use peak ratio (PR) and success rate (SR) mentioned in [51] to demonstrate our experimental results. PR represents the average percentage of the peaks found by the algorithm over all runs, while SR represents the average percentage of the runs where all global optima are found by the algorithm. The related formulas are as follows:

$$\text{PR} = \frac{\sum_{i=1}^{\text{NR}} \text{NPF}_i}{\text{NKP} * \text{NR}} \tag{14}$$

$$\text{SR} = \frac{\text{NSR}}{\text{NR}} \tag{15}$$

where NPF_i represents the number of global optima found by the algorithm at ith run, NKP is the total number of peaks in the problem (Table I), and NSR and NR are the number of runs where the algorithm finds all global optima and the total number of runs.

The calculation of NPF_i is also important; in [51], a detailed process was given for each individual. First, an empty set is initialized. Next, if two conditions are met, the individual is added to the set. Finally, the size of the set is returned as the number of global optima found by the algorithm. The following are the two conditions.

TABLE III
PRs AND SRs OF FBK-DE ON 20 PROBLEMS AT FIVE DIFFERENT ACCURACY LEVELS

ϵ	$F_1(1D)$		$F_2(1D)$		$F_3(1D)$		$F_4(2D)$		$F_5(2D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
$1e^{-1}$	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
$1e^{-2}$	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
$1e^{-3}$	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
$1e^{-4}$	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
$1e^{-5}$	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
ϵ	$F_6(2D)$		$F_7(2D)$		$F_6(3D)$		$F_7(3D)$		$F_8(2D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
$1e^{-1}$	0.990	0.820	0.813	0.000	0.826	0.000	0.426	0.000	1.000	1.000
$1e^{-2}$	0.990	0.820	0.813	0.000	0.826	0.000	0.426	0.000	1.000	1.000
$1e^{-3}$	0.990	0.820	0.813	0.000	0.825	0.000	0.426	0.000	1.000	1.000
$1e^{-4}$	0.990	0.820	0.813	0.000	0.824	0.000	0.425	0.000	1.000	1.000
$1e^{-5}$	0.000	0.000	0.813	0.000	0.823	0.000	0.425	0.000	1.000	1.000
ϵ	$F_9(2D)$		$F_{10}(2D)$		$F_{11}(2D)$		$F_{11}(3D)$		$F_{12}(3D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
$1e^{-1}$	1.000	1.000	0.935	0.480	1.000	1.000	0.930	0.600	0.733	0.000
$1e^{-2}$	1.000	1.000	0.935	0.480	1.000	1.000	0.923	0.560	0.730	0.000
$1e^{-3}$	1.000	1.000	0.935	0.480	1.000	1.000	0.920	0.540	0.730	0.000
$1e^{-4}$	1.000	1.000	0.935	0.480	1.000	1.000	0.907	0.460	0.730	0.000
$1e^{-5}$	1.000	1.000	0.935	0.480	1.000	1.000	0.890	0.380	0.728	0.000
ϵ	$F_{11}(5D)$		$F_{12}(5D)$		$F_{11}(10D)$		$F_{12}(10D)$		$F_{12}(20D)$	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
$1e^{-1}$	0.720	0.000	0.640	0.000	0.667	0.000	0.528	0.000	0.458	0.000
$1e^{-2}$	0.720	0.000	0.640	0.000	0.667	0.000	0.528	0.000	0.453	0.000
$1e^{-3}$	0.713	0.000	0.638	0.000	0.667	0.000	0.528	0.000	0.453	0.000
$1e^{-4}$	0.707	0.000	0.630	0.000	0.667	0.000	0.520	0.000	0.450	0.000
$1e^{-5}$	0.707	0.000	0.630	0.000	0.667	0.000	0.518	0.000	0.445	0.000

1) The absolute gap between its fitness and peak value is within ϵ, where ϵ represents the level of accuracy.
2) In the set, all the distances between itself and its superiors are greater than the niching radius r.

C. Results on FBK-DE

In this paper, FBK-DE is independently performed 50 times. The algorithm calculated the results in five levels of accuracy $\epsilon = \{1e^{-1}, 1e^{-2}, 1e^{-3}, 1e^{-4}, 1e^{-5}\}$.

Furthermore, we use the maximum evolutionary generations to obtain the population size NP. In other words, NP is related to the maximum evolutionary generations and the maximum fitness evaluations MaxFEs, which are calculated as follows:

$$\text{NP} = \left\lceil \frac{\text{MaxFEs}}{\text{MaxGen}} \right\rceil \tag{16}$$

where MaxGen indicates the maximum evolutionary generations decided by the dimension. When the dimension of problems is less than 5, MaxGen is set to 200, while the dimension is greater than or equal to 5, it is set to 300. For different problems, the value of NP is shown in Table I. It should be noted that although NP is set to 250 on the 1st–5th questions, this value is still much larger than in other literature [52], [53]. The parameters of FBK-DE are listed in Table II. Here, φ_{kp} is used to find the keypoints in one species, which is set to 2.0 as the original NBC and λ used in the species balance strategy is set as 2.0 to ensure the largest species have at most 2 times of the average size of all species. In general, F is fixed during the execution of the basic DE. However, in order to increase diversity, the value F of FBK-DE for one differential vector is randomly picked in the range [0.2, 0.8]. As for the two differential vectors, F is fixed to 0.5 because of large number of combinations. The source codes for this paper are available from the authors upon request.

The experimental results of FBK-DE for all benchmark problems are listed in Table III at all five accuracy levels. These results show that FBK-DE is very stable. Except for $F_6(2D)$ at the $1e^{-5}$ accuracy level, the difference between PR values at the highest and lowest accuracy levels does not exceed 4%. In addition, FBK-DE finds all peaks on the simple functions and most of the peaks in the low-dimensional composition problems. On the functions containing a large number of global peaks, except for $F_7(3D)$, FBK-DE finds more than 80% of all peaks. On the relatively high-dimensional problems, FBK-DE still finds more than a half of all peaks, along with 40% of all peaks on the 20-D problem.

V. DISCUSSION

A. Comparison With Other Algorithms

In this section, the results of various algorithms (including FBK-DE) are compared. As the results at $\epsilon = 1e^{-1}$ and $\epsilon = 1e^{-2}$ are not precise, the results at $\epsilon = 1e^{-3}$, $\epsilon = 1e^{-4}$, and $\epsilon = 1e^{-5}$ are chosen for this analysis. For simplicity, we compare the results at $\epsilon = 1e^{-4}$, which are commonly adopted in [53] and [54]. Comparison results at other accuracy levels with other algorithms as well as in other experiments are listed in the supplementary material.

In order to better evaluate the performance of FBK-DE, 15 popular comparison algorithms are selected, such as CDE [26], SDE [22], NCDE, NSDE [19], MOMMOP [50], LoICDE, LoISDE [55], PNPCDE [56], LIPS [57], and the

TABLE IV
EXPERIMENTAL RESULTS OF DIFFERENT ALGORITHMS ON THE BENCHMARK PROBLEMS AT THE ACCURACY LEVEL $\epsilon = 1e^{-4}$

Function	FBK-DE		CDE		SDE		NCDE		NSDE		MOMMOP		LoICDE		LoISDE	
Index	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1	**1.000**	1.000	**1.000**	1.000	0.657	0.373	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
2	**1.000**	1.000	**1.000**	1.000	0.737	0.529	**1.000**	1.000	0.776	0.667	**1.000**	1.000	**1.000**	1.000	0.235	0.039
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
4	**1.000**	1.000	**1.000**	1.000	0.284	0.000	**1.000**	1.000	0.240	0.000	**1.000**	1.000	0.975	0.902	0.250	0.000
5	**1.000**	1.000	**1.000**	1.000	0.922	0.843	**1.000**	1.000	0.745	0.490	**1.000**	1.000	**1.000**	1.000	0.667	0.333
6	0.990	0.820	**1.000**	1.000	0.056	0.000	0.305	0.000	0.056	0.000	**1.000**	1.000	**1.000**	1.000	0.056	0.000
7	0.813	0.000	0.861	0.000	0.054	0.000	0.873	0.000	0.053	0.000	**1.000**	1.000	0.705	0.020	0.029	0.000
8	0.824	0.000	0.000	0.000	0.015	0.000	0.001	0.000	0.013	0.000	**1.000**	1.000	0.000	0.000	0.012	0.000
9	0.425	0.000	0.474	0.000	0.011	0.000	0.461	0.000	0.006	0.000	**1.000**	1.000	0.187	0.000	0.005	0.000
10	**1.000**	1.000	**1.000**	1.000	0.147	0.000	0.989	0.863	0.098	0.000	**1.000**	1.000	**1.000**	1.000	0.083	0.000
11	**1.000**	1.000	0.330	0.000	0.314	0.000	0.729	0.059	0.248	0.000	0.716	0.020	0.660	0.000	0.167	0.000
12	0.935	0.480	0.002	0.000	0.208	0.000	0.252	0.000	0.135	0.000	0.939	0.549	0.495	0.000	0.125	0.000
13	**1.000**	1.000	0.141	0.000	0.297	0.000	0.667	0.000	0.225	0.000	0.667	0.000	0.510	0.000	0.167	0.000
14	**0.907**	0.460	0.026	0.000	0.216	0.000	0.667	0.000	0.190	0.000	0.667	0.000	0.657	0.000	0.167	0.000
15	0.730	0.000	0.005	0.000	0.108	0.000	0.319	0.000	0.125	0.000	0.618	0.000	0.299	0.000	0.125	0.000
16	**0.707**	0.000	0.000	0.000	0.108	0.000	0.667	0.000	0.170	0.000	0.650	0.000	0.559	0.000	0.167	0.000
17	0.630	0.000	0.000	0.000	0.076	0.000	0.250	0.000	0.108	0.000	0.505	0.000	0.223	0.000	0.076	0.000
18	**0.667**	0.000	0.167	0.000	0.026	0.000	0.500	0.000	0.163	0.000	0.497	0.000	0.219	0.000	0.157	0.000
19	**0.520**	0.000	0.000	0.000	0.105	0.000	0.348	0.000	0.098	0.000	0.223	0.000	0.037	0.000	0.027	0.000
20	**0.450**	0.000	0.000	0.000	0.000	0.000	0.250	0.000	0.123	0.000	0.125	0.000	0.123	0.000	0.088	0.000
bprs	**13**		7		1		5		2		10		6		2	

Function	PNPCDE		LIPS		DE_{cl}		DSDE/DSDE-C		LMCEDA		LMSEDA		LAMC-ACO		LAMS-ACO	
Index	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1	**1.000**	1.000	0.833	0.686	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
2	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
3	**1.000**	1.000	0.961	0.961	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
4	**1.000**	1.000	0.990	0.961	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
5	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
6	0.537	0.000	0.246	0.000	0.942	0.340	**1.000**	1.000	0.990	0.843	0.972	0.588	0.999	0.980	0.990	0.824
7	0.874	0.000	0.400	0.000	0.986	0.640	0.891	0.000	0.734	0.000	0.673	0.000	0.743	0.000	0.683	0.000
8	0.000	0.000	0.084	0.000	0.999	0.900	0.655	0.000	0.367	0.000	0.613	0.000	0.639	0.000	0.765	0.000
9	0.472	0.000	0.104	0.000	0.726	0.000	0.363	0.000	0.284	0.000	0.248	0.000	0.290	0.000	0.254	0.000
10	**1.000**	1.000	0.748	0.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	0.998	0.980	**1.000**	1.000	**1.000**	1.000
11	0.660	0.000	0.974	0.843	0.667	0.000	**1.000**	1.000	0.667	0.000	0.892	0.392	0.670	0.000	0.961	0.765
12	0.000	0.000	0.574	0.000	0.943	0.580	**1.000**	1.000	0.750	0.000	0.990	0.922	0.770	0.000	0.983	0.863
13	0.461	0.000	0.794	0.176	0.667	0.000	0.912	0.549	0.667	0.000	0.667	0.000	0.667	0.000	0.670	0.000
14	0.592	0.000	0.644	0.000	0.667	0.000	0.667	0.000	0.667	0.000	0.667	0.000	0.667	0.000	0.667	0.000
15	0.258	0.000	0.336	0.000	0.623	0.000	0.635	0.000	0.696	0.000	0.738	0.000	0.740	0.000	**0.748**	0.000
16	0.000	0.000	0.304	0.000	0.667	0.000	0.667	0.000	0.667	0.000	0.667	0.000	0.667	0.000	0.667	0.000
17	0.000	0.000	0.162	0.000	0.420	0.000	0.375	0.000	0.456	0.000	0.620	0.000	0.608	0.000	**0.708**	0.000
18	0.147	0.000	0.098	0.000	**0.667**	0.000	**0.667**	0.000	0.657	0.000	0.660	0.000	**0.667**	0.000	**0.667**	0.000
19	0.000	0.000	0.000	0.000	0.357	0.000	0.395	0.000	0.451	0.000	0.458	0.000	0.500	0.000	0.502	0.000
20	0.000	0.000	0.000	0.000	0.212	0.000	0.314	0.000	0.059	0.000	0.248	0.000	0.267	0.000	0.346	0.000
bprs	6		2		7		10		6		5		7		9	

recently proposed algorithms of DE_{cl} [41], DSDE, DSDE-C [53], LMCEDA, LMSEDA [52], LAMC-ACO, and LAMS-ACO [54]. Table IV shows the different PRs and SRs at the accuracy level $\epsilon = 1e^{-4}$, where the row "bprs" represents the number of the best PR results achieved by these algorithms. All the algorithms used their own default population sizes. Most of the recently compared results are from their corresponding papers, and the results of other algorithms are still from these papers.

From Table IV, it is clear that FBK-DE obtains most of the best PR results among the compared algorithms. The detailed analyses are given below.

1) For the first five problems, most algorithms including FBK-DE can find all global optimal solutions.
2) For the 6th–9th problems (with a large number of global peaks), FBK-DE does not perform as well; however, the results still exceed most of the compared algorithms. In particular for the 8th problem show that the PR result of FBK-DE is only exceeded by those of MOMMOP and DE_{cl}. In addition, for 10th problem, FBK-DE can find all the optimal solutions.
3) For the 11th–15th low-dimensional composition functions, FBK-DE can obtain satisfied results. Moreover, the best results for three functions are obtained by FBK-DE. Although on the 12th and 15th functions FBK-DE does not produce optimal results, the difference between FBK-DE and the best algorithm is small (below 7% and 2%). It is worth noting that FBK-DE finds all global peaks on the 13th problem while the other algorithms does not. In addition, FBK-DE finds 80% of the optimal peaks for the 14th problem; only two-thirds (at most) of all global peaks are found by the other algorithms.
4) For the 16th–20th composition functions (in the relatively high dimensions), FBK-DE achieves the best results for all except the 17th function. Especially on the 20-D problem, the best result obtained by FBK-DE exceeds 10% of the second-best result.

TABLE V
COMPARISON RESULTS OF THE DIFFERENT ALGORITHMS

Algorithm	PR_{mean}	PR_{std}	Rank
FBK-DE	**0.8299**	**0.1963**	**2.35**
CDE	0.4503	0.4510	8.7
SDE	0.2684	0.3023	12.9
NCDE	0.6139	0.3190	5.85
NSDE	0.2786	0.3124	12.1
MOMMOP	0.7804	0.2718	3.85
LoICDE	0.5825	0.3596	7.95
LoISDE	0.2302	0.2912	12.95
PNPCDE	0.4987	0.4087	8.75
LIPS	0.5126	0.3677	10.25
DEcl	0.7772	0.2404	3.65
DSDE/DSDE-C	0.7770	0.2493	2.7
LMCEDA	0.7056	0.2692	4.95
LMSEDA	0.7555	0.2409	4.65
LAMC-ACO	0.7447	0.2248	3.55
LAMS-ACO	0.7806	0.2255	2.95

Table V shows the comparison results of these multimodal optimization algorithms including FBK-DE. PR_{mean} and PR_{std} represent the average values and the standard deviations of PR on the 20 benchmark problems. For each problems, the results of the algorithms are sorted from the best to the worst, and each algorithm would obtain an index in the sorted data. The rank column in Table V represents the mean index on the 20 benchmark problems. From this table, it can be seen that our proposed algorithm achieves the best PR_{mean} and PR_{std}. In addition, from the rank perspective, FBK-DE performs the best among these algorithms.

B. Different Components of FBK-DE

1) Different Mutation Operators: First, we compare different mutation operators of FBK-DE. In order to show the balancing ability for exploration and exploitation, six algorithms denoted as FBK-DE, FBK-DE-r, FBK-DE-k, FBK-DE-b, FBK-DE-rb, and FBK-DE-n are compared. For mutation operators, FBK-DE-r uses only DE/rand/1 and DE/rand/2 to evolve the species. In addition, FBK-DE-k uses DE/keypoint/1 and DE/keypoint/2, and FBK-DE-b uses both DE/best/1 and DE/best/2. FBK-DE-n uses DE/nrand/1 and DE/nrand/2 for mutation, which are derived from [37]. The two mutation operators in the above algorithms are selected with an equal probability. The process of FBK-DE-rb is the same as FBK-DE; however, it uses DE/best/1 and DE/best/2 instead of DE/keypoint/1 and DE/keypoint/2. Except for the mutation operators, the other components are identical. The results are shown in Table VI at $\epsilon = 1e^{-4}$. The signs of "(++)" and "(+)" mean the result of FBK-DE are significantly and slightly better than of the corresponding algorithm, respectively.

From Table VI, it is clear that FBK-DE produces the best PR results for all benchmark problems except for the 7th and 9th functions. The performance of FBK-DE-k and FBK-DE-b are so poor that no peaks are found on the relatively high-dimensional functions. In addition, Table IV shows that most compared algorithms find all global optima on the 1st–5th functions, while FBK-DE-k and FBK-DE-b are unable to find the global peaks. FBK-DE-r finds all peaks in the first five functions. Except for the 10-D and 20-D problems, the results of FBK-DE-r are equal to or relatively poorer than that of FBK-DE. Compared with FBK-DE-n, FBK-DE performs better on the most problems, but does not perform well on the 7th and 9th problems.

Here, we specifically compare the results of FBK-DE and FBK-DE-rb to illustrate the effect of two keypoint-based mutation operators. Similar to the previous experiment, the comparisons on four type problems are discussed.

1) On the 1st–5th problems, considered as simple functions, both FBK-DE and FBK-DE-rb can achieve the desired results for all global optima.
2) On the many global peaks problems (the 6th–10th optimization functions), it is clear that the performances of FBK-DE are significantly better than that of FBK-DE-rb, which shows that the keypoint-based mutation operators are highly effective. As some distances between the peaks are very small in the 6th–9th functions, these results illustrate that a species would occasionally locate multiple basins of attraction. Thus, using NBC to find keypoints in the evolutionary process can effectively improve performance at the same time.
3) On the low-dimensional composition problems (the 11th–15th functions), the PR results for FBK-DE are superior to those of FBK-DE-kb. However, the differences on the six benchmark problems are generally smaller than on the 6th–9th problems with large amounts of global peaks. This shows that the probability of the species locating at multiple peaks is smaller than that of the previous problems.
4) On the relatively high-dimensional composition problems (the 16th–20th problems), FBK-DE performs better than or equal to FBK-DE-rb, indicating that the keypoint strategy can improve the overall performance of the algorithm. However, most gaps between the results of FBK-DE and FBK-DE-rb are not very large. Except for 19th function, other gaps are less than 4%.

2) Different Minimum Sizes of the Species: In this part, we discuss the minimum size of the species. In FBK-DE, *minsize* is set to an initial value of 5, which is incremented by 1 for every two generations until it reaches a maximum value. This maximum value is relative to the dimension of the problem. To design the comparison algorithms, we compare the results of the algorithms when *minsize* is set at fixed values of 10, 15, 30, and 60, which are denoted as FBK-DE-m10, FBK-DE-m15, FBK-DE-m30, and FBK-DE-m60. The results are shown in Table VII at $\epsilon = 1e^{-4}$.

From Table VII, it is clear that FBK-DE obtains the best results for 13 problems. In detail, all algorithms find all global peaks on the first five functions. On the 6th and 7th functions, FBK-DE does not perform well; however, the PR values are very close to those of the best PRs. On the 8th function, FBK-DE-m30 performs the best, with only a few peaks remaining hidden. This means that *minsize* should be larger on this function. On the 9th function, FBK-DE still performs better than all others. On the 1-D to 5-D composition functions (including 11th–15th functions), FBK-DE performs either better or slightly worse than the other algorithms. On the 18th–20th relatively high-dimensional functions, FBK-DE produces the best results. In particular, on the 19th problem,

TABLE VI
EXPERIMENTAL RESULTS OF DIFFERENT MUTATION OPERATORS ON THE BENCHMARK PROBLEMS AT THE ACCURACY LEVEL $\epsilon = 1e^{-4}$

Function	FBK-DE		FBK-DE-r		FBK-DE-k		FBK-DE-b		FBK-DE-rb		FBK-DE-n	
Index	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1	**1.000**	1.000	**1.000**	1.000	0.040(++)	0.000	0.050(++)	0.000	**1.000**	1.000	**1.000**	1.000
2	**1.000**	1.000	**1.000**	1.000	0.792(++)	0.260	0.876(++)	0.520	**1.000**	1.000	**1.000**	1.000
3	**1.000**	1.000	**1.000**	1.000	0.760(++)	0.760	0.820(++)	0.820	**1.000**	1.000	**1.000**	1.000
4	**1.000**	1.000	**1.000**	1.000	0.140(++)	0.000	0.225(++)	0.020	**1.000**	1.000	**1.000**	1.000
5	**1.000**	1.000	**1.000**	1.000	0.450(++)	0.220	0.580(++)	0.300	**1.000**	1.000	**1.000**	1.000
6	**0.990**	0.820	0.977(++)	0.640	0.003(++)	0.000	0.006(++)	0.000	0.954(++)	0.440	0.987(+)	0.800
7	0.813	0.000	0.742	0.000	0.196	0.000	0.222	0.000	0.714	0.000	**0.861**	0.000
8	**0.824**	0.000	0.750(++)	0.000	0.000(++)	0.000	0.000(++)	0.000	0.624(++)	0.000	0.418(++)	0.000
9	0.425	0.000	0.389	0.000	0.023	0.000	0.028	0.000	0.368	0.000	**0.520**	0.000
10	**1.000**	1.000	**1.000**	1.000	0.325(++)	0.000	0.423(++)	0.000	**1.000**	1.000	**1.000**	1.000
11	**1.000**	1.000	**1.000**	1.000	0.073(++)	0.000	0.117(++)	0.000	**1.000**	1.000	0.667(++)	0.000
12	**0.935**	0.480	0.932(+)	0.500	0.018(++)	0.000	0.028(++)	0.000	0.912(+)	0.340	0.735(++)	0.000
13	**1.000**	1.000	0.987(++)	0.920	0.080(++)	0.000	0.117(++)	0.000	0.990(+)	0.940	0.670(++)	0.000
14	**0.907**	0.460	0.823(++)	0.020	0.013(++)	0.000	0.013(++)	0.000	0.853(++)	0.200	0.667(++)	0.000
15	**0.730**	0.000	0.690(++)	0.000	0.002(++)	0.000	0.008(++)	0.000	0.692(++)	0.000	0.585(++)	0.000
16	**0.707**	0.000	0.667(++)	0.000	0.000(++)	0.000	0.000(++)	0.000	0.683(+)	0.000	0.667(++)	0.000
17	**0.630**	0.000	0.625(+)	0.000	0.000(++)	0.000	0.000(++)	0.000	0.625(+)	0.000	0.482(++)	0.000
18	**0.667**	0.000	**0.667**	0.000	0.000(++)	0.000	0.000(++)	0.000	**0.667**	0.000	0.593(++)	0.000
19	**0.520**	0.000	0.308(++)	0.000	0.000(++)	0.000	0.000(++)	0.000	0.432(++)	0.000	0.188(++)	0.000
20	**0.450**	0.000	0.000(++)	0.000	0.000(++)	0.000	0.000(++)	0.000	0.430(+)	0.000	0.000(++)	0.000
bprs	**18**		8		0		0		8		8	

TABLE VII
EXPERIMENTAL RESULTS OF DIFFERENT *minsize* ON THE BENCHMARK PROBLEMS AT THE ACCURACY LEVEL $\epsilon = 1e^{-4}$

Function	D	FBK-DE		FBK-DE-m10		FBK-DE-m15		FBK-DE-m30		FBK-DE-m60	
Index		PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
1	1	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
2	1	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
3	1	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
4	2	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
5	2	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
6	2	0.990	0.820	0.993	0.880	0.999	0.980	**1.000**	1.000	**1.000**	1.000
7	2	0.813	0.000	**0.819**	0.000	0.788	0.000	0.687	0.000	0.753	0.000
8	3	0.824	0.000	0.841	0.000	0.889	0.000	**0.979**	0.140	0.887	0.000
9	3	**0.425**	0.000	0.418(++)	0.000	0.373(++)	0.000	0.407(++)	0.000	0.401(++)	0.000
10	2	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
11	2	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
12	2	0.935	0.480	0.942	0.540	0.950	0.600	**0.958**	0.660	0.950	0.600
13	2	**1.000**	1.000	0.993(+)	0.960	**1.000**	1.000	0.947(++)	0.680	0.727(++)	0.000
14	3	**0.907**	0.460	0.877(+)	0.340	0.777(++)	0.040	0.667(++)	0.000	0.667(++)	0.000
15	3	0.730	0.000	0.730	0.000	0.720	0.000	**0.738**	0.000	0.712	0.000
16	5	0.707	0.000	**0.733**	0.000	0.697	0.000	0.670	0.000	0.667	0.000
17	5	0.630	0.000	0.640	0.000	**0.652**	0.000	0.620	0.000	0.598	0.000
18	10	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	0.650(++)	0.000	0.627(++)	0.000
19	10	**0.520**	0.000	0.450(++)	0.000	0.472(++)	0.000	0.465(++)	0.000	0.268(++)	0.000
20	20	**0.450**	0.000	0.000(++)	0.000	0.262(++)	0.000	0.300(++)	0.000	0.260(++)	0.000
bprs		**13**		10		10		11		8	

we compare the results of FBK-DE and FBK-DE-m30. It is clear that the performance of FBK-DE is superior to that of FBK-DE-m30. It is worth noting that the *minsize* of FBK-DE reaches 30 after about 50 generations. Following this, the *minsize* of FBK-DE is equal to that of FBK-DE-m30. This shows that in the early stages, *minsize* should not take a large value but should increase during evolutionary searches. This effect becomes more apparent on the 20-D problem. It is also clear that FBK-DE finds as many as 15% more peaks than FBK-DE-m30.

3) Different Values of α: Here, the different values of α are compared. In Section III, α is used to calculate the probability *per*, determining which group of mutation operators to use. Thus, we set α to four different values $(1/3)$, $(1/2)$, 1, and 2, which are denoted as FBK-DE-$(1/3)$, FBK-DE, FBK-DE-1, and FBK-DE-2. FBK-DE adopts the middle value, whose α is set to $(1/2)$. Table VIII shows the results at $\epsilon = 1e^{-4}$.

From Table VIII, it is clear that α is set to $(1/3)$, which achieves the best overall results. However, except for FBK-DE-2, most gaps among the results of the first three parameters are very small. On the simple functions, all algorithms find all global optima. On the 6th–10th functions with (large amounts of global peaks), smaller α produced superior PR results. This indicates that the algorithm requires more exploitation ability rather than exploration ability. For the low-dimensional composition functions, FBK-DE-$(1/3)$ and FBK-DE are equal in terms of performance. On the high-dimensional composition problems, the difference is still below 3%; the four algorithms all produces two or three best results.

TABLE VIII
EXPERIMENTAL RESULTS OF DIFFERENT α ON THE BENCHMARK PROBLEMS AT THE ACCURACY LEVEL $\epsilon = 1e^{-4}$

Function Index	FBK-DE-1/3		FBK-DE		FBK-DE-1		FBK-DE-2	
	PR	SR	PR	SR	PR	SR	PR	SR
1	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
2	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
3	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
4	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
5	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
6	**0.994**	0.900	0.990	0.820	0.984	0.800	0.983	0.740
7	**0.839**	0.000	0.813	0.000	0.773	0.000	0.742	0.000
8	**0.824**	0.000	**0.824**	0.000	0.804(++)	0.000	0.766(++)	0.000
9	**0.435**	0.000	0.425	0.000	0.407	0.000	0.390	0.000
10	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
11	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000	**1.000**	1.000
12	**0.942**	0.560	0.935	0.480	0.928	0.420	0.925	0.400
13	0.997(+)	0.980	**1.000**	1.000	0.990(+)	0.940	0.987(++)	0.920
14	0.893(+)	0.360	**0.907**	0.460	0.883(+)	0.300	0.883(+)	0.320
15	**0.732**	0.000	0.730	0.000	0.728	0.000	0.708	0.000
16	0.720	0.000	0.707	0.000	**0.723**	0.000	0.710	0.000
17	**0.632**	0.000	0.630	0.000	**0.632**	0.000	0.628	0.000
18	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000	**0.667**	0.000
19	0.495(+)	0.000	**0.520**	0.000	0.510(+)	0.000	**0.520**	0.000
20	0.445(+)	0.000	**0.450**	0.000	0.448(+)	0.000	0.432(+)	0.000
bprs	**15**		13		10		9	

TABLE IX
EXPERIMENTAL RESULTS OF DIFFERENT SPECIES BALANCE STRATEGIES ON THE BENCHMARK PROBLEMS AT THE ACCURACY LEVEL $\epsilon = 1e^{-4}$

Function Index	FBK-DE		no balance	
	PR	SR	PR	SR
1	**1.000**	1.000	**1.000**	1.000
2	**1.000**	1.000	**1.000**	1.000
3	**1.000**	1.000	**1.000**	1.000
4	**1.000**	1.000	**1.000**	1.000
5	**1.000**	1.000	**1.000**	1.000
6	0.990	0.820	**0.998**	0.960
7	**0.813**	0.000	0.675(++)	0.000
8	0.824	0.000	**0.886**	0.000
9	**0.425**	0.000	0.340(++)	0.000
10	**1.000**	1.000	**1.000**	1.000
11	**1.000**	1.000	**1.000**	1.000
12	0.935	0.480	**0.955**	0.640
13	**1.000**	1.000	0.997(+)	0.980
14	**0.907**	0.460	0.840(++)	0.140
15	0.730	0.000	**0.748**	0.000
16	**0.707**	0.000	0.673(++)	0.000
17	0.630	0.000	**0.718**	0.000
18	**0.667**	0.000	**0.667**	0.000
19	**0.520**	0.000	0.500(+)	0.000
20	**0.450**	0.000	0.298(++)	0.000
bprs	**15**		13	

4) Effect of Species Balance Strategy: In this part, we discuss the role of the species balance strategy in the algorithm. In the previous section, the species balance strategy is used to balance the sizes of the species if excessively large or small individuals existed in a species. Here, we test for two situations: 1) the current species balance strategy and 2) no balance strategy. The latter indicates that no species balance strategy is used during algorithm execution. Table IX shows the comparison.

It is clear that FBK-DE with a species balance strategy can achieve most of the best PR results. In detail, the gap between "no balance" and FBK-DE is very small when no balance strategy produces the best results (such as on the 6th and 15th functions). In addition, FBK-DE is far superior to FBK-DE without species balance strategy on the 7th, 16th, and 20th functions. On the 1st to 5th functions, all algorithms find all the global peaks. For the functions with many peaks, the data show the opposite. For the F_6 on 2-D and 3-D, it is clear that FBK-DE without a species balance strategy is superior to FBK-DE. For the F_7 on 2-D and 3-D, the results show that it is necessary to use a species balance strategy, as F_7 contains peaks of different widths. On the composition functions, FBK-DE with λ set to 2.0 can achieve a better balance of exploration and exploitation, whereas FBK-DE without a species balance strategy performs very poorly on several functions (such as on the 20th function).

5) Comparison With Original NBC: In this part, the comparison of two algorithms to show the performance of the NBC-minsize and the species balance strategy. Here, the comparison algorithm is denoted as FBK-DE-NBC, where the original NBC is used to generate the species and no balance strategy is adopted. The keypoint-based mutation operators are still used in FBK-DE-NBC. Table X shows the comparison results.

It can be seen that FBK-DE with NBC-minsize and the species balance strategy almost wins on all the benchmarks except for the 8th problem. Besides, except for the 8th problem and the problems where two algorithms can obtain all the global optimal solutions on all runs, FBK-DE performs significantly better than FBK-DE-NBC with the original NBC, especially on the composition functions. On the 20th problem, it is clear that FBK-DE has the ability to find more than 40% optimal solutions, while FBK-DE-NBC can hardly find the optimal solutions. Overall, FBK-DE performs much better than FBK-DE-NBC.

C. Limitations and Practicability

From Table IV, it can be seen that FBK-DE does not perform the best on the problems with many global peaks.

TABLE X
EXPERIMENTAL RESULTS OF FBK-DE AND FBK-DE-NBC ON THE BENCHMARK PROBLEMS AT THE ACCURACY LEVEL $\epsilon = 1e^{-4}$

Function Index	FBK-DE		FBK-DE-NBC	
	PR	SR	PR	SR
1	**1.000**	1.000	**1.000**	1.000
2	**1.000**	1.000	**1.000**	1.000
3	**1.000**	1.000	**1.000**	1.000
4	**1.000**	1.000	**1.000**	1.000
5	**1.000**	1.000	**1.000**	1.000
6	**0.990**	0.820	0.972(++)	0.620
7	**0.813**	0.000	0.614(++)	0.000
8	0.824	0.000	**0.922**	0.000
9	**0.425**	0.000	0.300(++)	0.000
10	**1.000**	1.000	**1.000**	1.000
11	**1.000**	1.000	**1.000**	1.000
12	**0.935**	0.480	0.780(++)	0.020
13	**1.000**	1.000	0.960(++)	0.760
14	**0.907**	0.460	0.773(++)	0.000
15	**0.730**	0.000	0.620(++)	0.000
16	**0.707**	0.000	0.667(++)	0.000
17	**0.630**	0.000	0.615(++)	0.000
18	**0.667**	0.000	0.620(++)	0.000
19	**0.520**	0.000	0.238(++)	0.000
20	**0.450**	0.000	0.068(++)	0.000
bprs	**19**		8	

Besides, as the dimension increases of MMOPs, the PR value decreases, which illustrates FBK-DE is not good enough over the high-dimensional problems.

For the problems with many global peaks, it is very difficult to find all the optimal solutions. However, too many choices are also a question for the decision makers, which is known as choice overload [58], [59]. The study in [60] demonstrates that the decision makers will feel very comfortable and then very uncomfortable when the optional solutions increase from one to several and then to dozens. For example, the 9th function has 216 global optima, it is a real problem of choice overload when all the global optima are presented to the decision makers.

For the high-dimensional MMOPs, it is a real challenge for existing EAs to find all the global optima in a run. Fortunately, from Table IV, it can be observed that FBK-DE performs the best when the dimension is no less than 10 (i.e., the 18th–20th problems). Therefore, FBK-DE is still an alternative in practice.

VI. CONCLUSION

In this paper, we proposed FBK-DE to handle MMOPs, where several species techniques were adopted to simultaneously find multiple optima. In FBK-DE, NBC was embedded to divide the population into several species to locate the optima. In order to overcome the disadvantages of NBC, an improved NBC strategy named NBC-minsize was presented to limit the minimum size of the species. In addition, a species balance strategy was adopted to balance the number of individuals generated by the species. Two DE mutation operators (DE/keypoint/1 and DE/keypoint/2) were proposed to make use of the multiple potential optima in a species. Finally, several state-of-the-art algorithms were compared with FBK-DE. The experimental results showed that FBK-DE performed better than other algorithms on most multimodal benchmark problems.

However, in our experiments, the highest dimension of MMOPs is 20. In the future, FBK-DE would be improved and tested on high-dimensional problems. Meanwhile, more complicated environments will be considered, such as the benchmark in [61] and the dynamic MMOPs [62].

REFERENCES

[1] J.-P. Li, X.-D. Li, and A. Wood, "Species based evolutionary algorithms for multimodal optimization: A brief review," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2010, pp. 1–8.

[2] X. Li, M. G. Epitropakis, K. Deb, and A. Engelbrecht, "Seeking multiple solutions: An updated survey on niching methods and their applications," *IEEE Trans. Evol. Comput.*, vol. 21, no. 4, pp. 518–538, Aug. 2017.

[3] T. Bäck, D. B. Fogel, and Z. Michalewicz, *Handbook of Evolutionary Computation*. Boca Raton, FL, USA: CRC Press, 1997.

[4] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, vol. 53. Heidelberg, Germany: Springer, 2003.

[5] S. Das, S. Maity, B. Qu, and P. N. Suganthan, "Real-parameter evolutionary multimodal optimization—A survey of the state-of-the-art," *Swarm Evol. Comput.*, vol. 1, no. 2, pp. 71–88, 2011.

[6] S. Biswas, S. Das, P. N. Suganthan, and C. A. Coello Coello, "Evolutionary multiobjective optimization in dynamic environments: A set of novel benchmark functions," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2014, pp. 3192–3199.

[7] 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. (CEC)*, 2013, pp. 79–86.

[8] W. Gong and Z. Cai, "Differential evolution with ranking-based mutation operators," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 2066–2081, Dec. 2013.

[9] X. Li, "Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization," in *Proc. Genet. Evol. Comput. Conf.*, 2004, pp. 105–116.

[10] D. Bose, S. Biswas, S. Kundu, and S. Das, "A strategy pool adaptive artificial bee colony algorithm for dynamic environment through multi-population approach," in *Proc. Int. Conf. Swarm Evol. Memetic Comput.*, 2012, pp. 611–619.

[11] A. Pétrowski, "A clearing procedure as a niching method for genetic algorithms," in *Proc. IEEE Int. Conf. Evol. Comput.*, 1996, pp. 798–803.

[12] 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, Sep. 2002.

[13] K. A. De Jong, "An analysis of the behavior of a class of genetic adaptive systems," Ph.D. dissertation, Dept. Comput. Commun. Sci., Univ. Michigan, Ann Arbor, MI, USA, 1975.

[14] S. W. Mahfoud, "Crowding and preselection revisited," in *Proc. 2nd Conf. Parallel Probl. Solving Nat.*, vol. 2, 1992, pp. 27–34.

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

[16] X. Yin and N. Germay, "A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization," in *Proc. Artif. Neural Nets Genet. Algorithms*, 1993, pp. 450–457.

[17] D. Beasley, D. R. Bull, and R. R. Martin, "A sequential niche technique for multimodal function optimization," *Evol. Comput.*, vol. 1, no. 2, pp. 101–125, Jun. 1993.

[18] G. R. Harik, "Finding multimodal solutions using restricted tournament selection," in *Proc. 6th Int. Conf. Genet. Algorithms*, 1995, pp. 24–31.

[19] B. 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.

[20] S. Das, A. Abraham, U. K. Chakraborty, and A. Konar, "Differential evolution using a neighborhood-based mutation operator," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 526–553, Jun. 2009.

[21] M. G. Epitropakis, D. K. Tasoulis, N. G. Pavlidis, V. P. Plagianakos, and M. N. Vrahatis, "Enhancing differential evolution utilizing proximity-based mutation operators," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 99–119, Feb. 2011.

[22] X. Li, "Efficient differential evolution using speciation for multimodal function optimization," in *Proc. 7th Annu. Conf. Genet. Evol. Comput.*, 2005, pp. 873–880.

[23] M. Preuss, "Niching the CMA-ES via nearest-better clustering," in *Proc. ACM 12th Annu. Conf. Companion Genet. Evol. Comput.*, 2010, pp. 1711–1718.

[24] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, 1997.

[25] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Berlin, Germany: Springer, 2005.

[26] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 2, 2004, pp. 1382–1389.

[27] H. A. Abbass, R. Sarker, and C. Newton, "PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 2, 2001, pp. 971–978.

[28] S. Biswas, D. Bose, S. Das, and S. Kundu, "Decomposition-based evolutionary multi-objective optimization approach to the design of concentric circular antenna arrays," *Progr. Electromagn. Res.*, vol. 52, pp. 185–205, 2013.

[29] R. Mendes and A. S. Mohais, "DynDE: A differential evolution for dynamic optimization problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 3, 2005, pp. 2808–2815.

[30] R. Joshi and A. C. Sanderson, "Minimal representation multisensor fusion using differential evolution," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 29, no. 1, pp. 63–76, Jan. 1999.

[31] M. Preuss, P. Burelli, and G. N. Yannakakis, "Diversified virtual camera composition," in *Proc. Eur. Conf. Appl. Evol. Comput.*, 2012, pp. 265–274.

[32] W. Luo, B. Yang, C. Bu, and X. Lin, "A hybrid particle swarm optimization for high-dimensional dynamic optimization," in *Proc. Asia–Pac. Conf. Simulat. Evol. Learn.*, 2017, pp. 981–993.

[33] M. Preuss, *Multimodal Optimization by Means of Evolutionary Algorithms*. Cham, Switzerland: Springer, 2015.

[34] O. M. Shir and T. Bäck, "Dynamic niching in evolution strategies with covariance matrix adaptation," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, vol. 3, 2005, pp. 2584–2591.

[35] O. M. Shir and T. Bäck, "Niching in evolution strategies," in *Proc. 7th Annu. Conf. Genet. Evol. Comput.*, 2005, pp. 915–916.

[36] S. Das, S. S. Mullick, and P. N. Suganthan, "Recent advances in differential evolution—An updated survey," *Swarm Evol. Comput.*, vol. 27, pp. 1–30, Apr. 2016.

[37] M. G. Epitropakis, V. P. Plagianakos, and M. N. Vrahatis, "Finding multiple global optima exploiting differential evolution's niching capability," in *Proc. IEEE Symp. Differ. Evol. (SDE)*, 2011, pp. 1–8.

[38] Y.-J. Gong, J. Zhang, and Y. Zhou, "Learning multimodal parameters: A bare-bones niching differential evolution approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 7, pp. 2944–2959, Jul. 2018.

[39] S. Tuo, J. Zhang, X. Yuan, and L. Yong, "A new differential evolution algorithm for solving multimodal optimization problems with high dimensionality," *Soft Comput.*, vol. 22, no. 13, pp. 4361–4388, 2018.

[40] 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.

[41] B. Bošković and J. Brest, "Clustering and differential evolution for multimodal optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2017, pp. 698–705.

[42] K. E. Parsopoulos and M. N. Vrahatis, "Modification of the particle swarm optimizer for locating all the global minima," in *Proc. Artif. Neural Nets Genet. Algorithms*, 2001, pp. 324–327.

[43] T. Sasaki, H. Nakano, A. Miyauchi, and A. Taguchi, "Particle swarm optimizer networks with stochastic connection for improvement of diversity search ability to solve multimodal optimization problems," *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, vol. 100, no. 4, pp. 996–1007, 2017.

[44] 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.

[45] R. Brits, A. Engelbrecht, and F. Van den Bergh, "Solving systems of unconstrained equations using particle swarm optimization," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, vol. 3, 2002, p. 6.

[46] R. Brits, A. P. Engelbrecht, and F. Van den Bergh, "Locating multiple optima using particle swarm optimization," *Appl. Math. Comput.*, vol. 189, no. 2, pp. 1859–1883, 2007.

[47] R. Brits, A. P. Engelbrecht, and F. Van den Bergh, "A niching particle swarm optimizer," in *Proc. Asia–Pac. Conf. Simulat. Evol. Learn.*, vol. 2, 2002, pp. 692–696.

[48] A. Basak, S. Das, and K. C. Tan, "Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 666–685, Oct. 2013.

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

[50] Y. Wang, H. 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.

[51] 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 University, Rep., Melbourne, VIC, Australia, Rep., 2013.

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

[53] Z.-J. Wang *et al.*, "Dual-strategy differential evolution with affinity propagation clustering for multimodal optimization problems," *IEEE Trans. Evol. Comput.*, vol. 22, no. 6, pp. 894–908, Dec. 2018.

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

[55] 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.

[56] 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.

[57] 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.

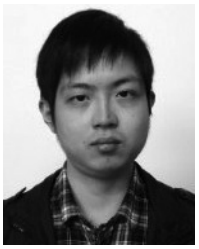
[58] B. Schwartz, *The Paradox of Choice: Why More Is Less*. New York, NY, USA: Ecco, 2004.

[59] E. Reutskaja and R. M. Hogarth, "Satisfaction in choice as a function of the number of alternatives: When 'goods satiate'," *Psychol. Market.*, vol. 26, no. 3, pp. 197–203, 2009.

[60] D. Bollen, B. P. Knijnenburg, M. C. Willemsen, and M. Graus, "Understanding choice overload in recommender systems," in *Proc. ACM Conf. Recommender Syst.*, 2010, pp. 63–70.

[61] B. Y. Qu, J.-J. Liang, Z. Y. Wang, Q. Chen, and P. N. Suganthan, "Novel benchmark functions for continuous multimodal optimization with comparative results," *Swarm Evol. Comput.*, vol. 26, pp. 23–34, Feb. 2016.

[62] W. Luo, X. Lin, T. Zhu, and P. Xu, "A clonal selection algorithm for dynamic multimodal function optimization," *Swarm Evol. Comput.*, Oct. 2018. [Online]. Available: https://doi.org/10.1016/j.swevo.2018.10.010

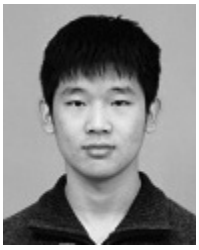


Xin Lin received the B.E. degree from Wuhan Textile University, Wuhan, China, in 2013. He is currently pursuing the master's degree with the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China.

His current research interest includes evolutionary optimization and applications.

Wenjian Luo (SM'15) received the B.S. and Ph.D. degrees from the Department of Computer Science and Technology, University of Science and Technology of China, Hefei, China, in 1998 and 2003, respectively.

He is currently an Associate Professor with the School of Computer Science and Technology, University of Science and Technology of China. His current research interests include computational intelligence and applications, machine learning and data mining, and information security and data privacy.



Peilan Xu received the B.S. degree from Dalian Polytechnic University, Dalian, China, in 2017. He is currently pursuing the master's degree with the School of Computer Science and Technology, University of Science and Technology of China, Hefei, China.

His current research interest includes evolutionary optimization and applications.