Hindawi
International Journal of Intelligent Systems
Volume 2023, Article ID 3961336, 25 pages
https://doi.org/10.1155/2023/3961336

WILEY | Hindawi

Research Article

Differential Evolution with a Level-Based Learning Strategy for Multimodal Optimization

Yuhui Zhang,[1] **Wenhong Wei**,[1] **Tiezhu Zhao**,[1] **and Zijia Wang**[2]

[1]*School of Computer Science and Technology, Dongguan University of Technology, Dongguan 523808, China*
[2]*School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou 510006, China*

Correspondence should be addressed to Yuhui Zhang; yhzhang@dgut.edu.cn

Received 27 May 2023; Revised 10 September 2023; Accepted 9 October 2023; Published 30 October 2023

Academic Editor: Fabio Caraffini

Multimodal optimization aims at efficiently finding multiple optimal solutions of a problem. Owing to the population-based search mechanism, evolutionary algorithms (EAs) are becoming increasingly popular in solving multimodal optimization problems (MOPs). Most existing work focuses on designing and incorporating niching techniques into EAs so that multiple subpopulations can be formed and assigned to locate different optima. To further enhance the exploration and exploitation abilities of existing EAs, this paper developed a multimodal level-based learning strategy. The basic idea is that individuals should be treated differently according to their positions in the subpopulation. In the evolutionary process, a subpopulation is formed for each candidate solution by grouping its neighboring solutions. Then, individuals in the subpopulation are sorted according to their fitness. Subsequently, the multimodal level-based learning strategy applies different mutation operators to different individuals according to their rankings. Experiments are conducted on a set of benchmark problems to verify the efficacy of the multimodal level-based learning strategy. The results show that the proposed learning strategy can significantly enhance the performance of the existing algorithm. In addition, the algorithm integrated with the proposed strategy is applied to the task of finding multiple roots of nonlinear equation systems (NESs). The results indicate that with the support of the proposed learning strategy, the integrated algorithm compares favorably with state-of-the-art root finding algorithms.

1. Introduction

Multimodal optimization problems (MOPs) are commonly encountered in scientific research and engineering design. Some examples of MOPs include RNA secondary structure prediction [1], motion planning of robot arms [2], trajectory planning of railway vehicles [3], image segmentation [4], and root finding of nonlinear equation systems [5]. Therefore, efficient algorithms for solving MOPs are of great importance for progressing the development of new technologies.

Evolutionary algorithms (EAs) are a sort of nature-inspired meta-heuristic search algorithms that mimic the evolution of life or the collective behavior of social animals. EAs maintain a group of candidate solutions (individuals) in the optimization process and iteratively update the solutions through genetic operators (e.g., crossover and mutation). When the termination criterion is met, the algorithms return the best solution found in the search process. Different from traditional optimization techniques that use the first-order or second-order derivatives, EAs do not require any problem-specific information and have strong global search abilities. They have been shown to be powerful optimization techniques for solving various types of black box optimization problems [6–10].

It is worth noting that EAs are mostly designed for global optimization, in which the goal is to locate a single global optimum. However, since a population of candidate solutions is maintained in the search process, it is straightforward to extend EAs to simultaneously locate multiple optimal solutions. The techniques used for extending EAs are commonly referred to as "niching." The basic idea of "niching" is to divide the population into small groups, and each group is responsible for locating one optimum. In this way, multiple optimal solutions can be located in a single

run. Over the past decades, various niching techniques have been developed in the literature [11, 12]. Integrated with these niching techniques, EAs have raised as a mainstream approach for solving MOPs.

However, most of the existing niching techniques focus on the improvement of genetic operators to induce niching behavior of the population. This paper explores another avenue to enhance the performance of multimodal evolutionary algorithms. The contributions of this work are twofold.

(1) We develop a multimodal level-based learning strategy that takes into account the working principle of niching algorithms. For each individual, a subpopulation is formed by grouping its neighboring solutions. Then, the subpopulation is divided into multiple levels according to the fitness of individuals.

(2) For each level of individuals, a tailored mutation operator is devised to enhance the search efficiency of EAs. Individuals with higher fitness values are assigned for exploitation while individuals with lower fitness values are assigned for exploration. This way, a better balance between exploitation and exploration can be achieved when solving multimodal optimization problems.

Comprehensive experiments are conducted on a set of benchmark problems and several real-world problems to examine the efficacy of the strategy. The experimental results show that the multimodal level-based learning strategy can significantly increase the performance of a popular differential evolution- (DE-) based multimodal algorithm.

The remainder of this paper is organized as follows. Section 2 gives a brief review on evolutionary multimodal optimization and introduces the principle of the level-based learning strategy. In Section 3, we propose a multimodal level-based learning strategy. The integration of the strategy with DE is described in detail. Comprehensive experiments are conducted in Section 4 to investigate the efficacy of the proposed strategy. The experimental results are presented and analyzed in this section as well. Finally, Section 5 concludes this paper and gives some future research directions.

2. Background

In this section, we first review some classical multimodal evolutionary algorithms. Then, some recently proposed niching techniques are introduced to uncover the research trend of evolutionary multimodal optimization. Subsequently, we elaborate on the level-based learning strategy, which is the basis of our proposed strategy. The principle and procedures of the level-based learning strategy are described in detail.

2.1. Evolutionary Multimodal Optimization. Recent years have witnessed the rapid development of EAs in various optimization domains. Jiang et al. [14] developed a self-adaptive niching DE (SaNDE) with ring topology. A new mutation operator called "current-to-pnbest" is proposed to increase the search efficiency. Moreover, an adaptive restart mechanism is used to handle the problem of stagnation. Sun et al. [15] proposed an adaptive regeneration framework based on search space adjustment to address the problem of premature convergence encountered by DE algorithms. Parouha and Verma [16] provided a comprehensive overview of the recent developments of DE and particle swarm optimization (PSO). Then, a hybrid algorithm of advanced DE and PSO is developed to solve unconstrained optimization problems. To improve the performance of DE, Deng et al. [17] proposed a dynamic combination-based mutation operator in which the base vector is constructed using the current optimal individual and an elite individual. A two-level parameter regularization strategy is used to adjust the scale factor and crossover rate by combing the effect of the population-level parameter and the individual-level parameter. Zhao et al. [18] implemented a Gaussian perturbation operation to expand the search neighborhood of candidate individuals produced by the opposition-based learning strategy. Different sized neighborhood is adopted in different evolutionary stages to balance exploration and exploitation.

In real-world scenario, it is not uncommon to see that a problem has the multimodality property, which means that the problem has multiple satisfactory solutions. Simultaneously finding multiple solutions has several benefits to decision makers. First, if an optimal solution cannot be deployed due to physical restrictions, decision makers can switch to other optimal solutions. Second, the obtained solutions can be combined to form a more robust solution. To find multiple optimal solutions, traditional optimization techniques need to run multiple times with different starting points, which is very inefficient and time-consuming. In comparison, EAs maintain a population of candidate solutions in the search process. We can disperse the population to search for multiple optima in a single run. Since EAs have this natural advantage over traditional techniques, they have become one of the mainstream methods for multimodal optimization and have been applied to a variety of real-world MOPs.

The primitive versions of EAs are designed for global optimization. Niching techniques have to be incorporated into the algorithms to induce multiple convergence behavior of the population. During the past decades, many classical niching methods have been developed. The most prominent ones include crowding [19], fitness sharing [20], speciation [21], and clearing [22]. The idea of crowding is that similar individuals need to compete for limited resources, where the similarity between individuals is measured by their distance in the search space. Fitness sharing first divides the population into several niches. Its underlying principle is that individuals in a niche should share information with others in the same niche. Fitness sharing is implemented by dividing the fitness of an individual by the number of individuals in its niche. Different from crowding and fitness sharing, speciation explicitly divides the population into a number of species. A species seed with the best fitness value is first extracted from the population. Then, individuals

falling within a circle of the species seed are classified into one species. The above process is repeated until all the individuals have been assigned to a species. Clearing is another popular niching technique whose procedure is similar to that of speciation. However, except for the best individual, it eliminates all other individuals in the same niche. The classical niching techniques often serve as building blocks for advanced methods and have been widely used in the design of new algorithms.

For DE, Qu et al. [23] proposed a neighborhood mutation operator that poses restrictions on the generation of mutation vectors. When generating a mutation vector $\mathbf{V}_i$ for the individual $\mathbf{X}_i$, vectors $\mathbf{X}_{r1}$, $\mathbf{X}_{r2}$, and $\mathbf{X}_{r3}$ are selected from the neighborhood of $\mathbf{X}_i$, instead of from the entire population. Qu et al. [23] integrated the neighborhood mutation operator with fitness sharing, speciation, and crowding and proposed three algorithms called neighborhood-based sharing DE (NShDE), neighborhood-based speciation DE (NSDE), and neighborhood-based crowding DE (NCDE). Epitropakis et al. [24] introduced two mutation operators called DE/nrand/1 and DE/nrand/2. These two operators are similar to DE/rand/1 and DE/rand/2 except that the nearest neighbor of the individual $\mathbf{X}_i$ is used as $\mathbf{X}_{r1}$. In a later work [25], they further enhanced the algorithm by adding an external archive and a reinitialization mechanism. In this way, the influence of the population size is reduced. Biswas et al. [26] developed an algorithm called parent-centric normalized mutation with proximity-based crowding DE (PNPCDE). In PNPCDE, a new parent-centric mutation operator is combined with a synchronous crowding replacement strategy to maintain population diversity. In [13], they also developed a new information sharing mechanism to increase the search efficiency of crowding DE. The idea is depicted in Figure 1. Instead of randomly selecting parents for mutation, distance and fitness information is used to control the selection probability of parents. Gao et al. [27] proposed a cluster-based DE and a self-adaptive strategy to tackle multimodal problems. The clustering partition and the self-adaptive strategy are combined with crowding DE (CDE) and species-based DE (SDE). The resulting algorithms are termed self-adaptive strategy-based clustering crowding DE (Self-CCDE) and self-adaptive strategy-based clustering speciation DE (Self-CSDE), respectively. Similarly, a double-layer clustering speciation method and a self-adaptive strategy are developed in [28]. Lin et al. [29] proposed two key-point-based mutation operators and used nearest-better clustering to balance the exploration ability in the global space and the exploitation ability in multiple optimal areas. Sheng et al. [30] designed a niching competition strategy that encourages high potential niches for exploitation and low potential niches for exploration. Moreover, an archive supporting strategy is implemented at the niche level to maintain potential optima. Wang et al. [31] proposed a parameter-free automatic niching technique based on the affinity propagation clustering. They developed an automatic niching DE (ANDE) and enhanced it with a contour prediction approach and a two-level local search strategy.

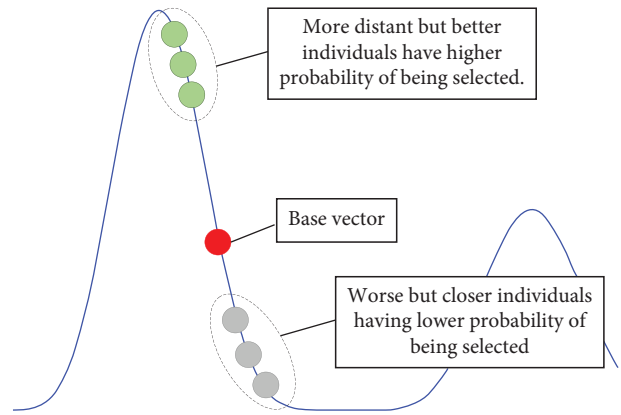

Figure 1: Basic idea of the local information sharing mechanism [13]. When selecting parents for mutation, better individuals have higher probabilities of being selected, even though their distances to the base vector are larger than those of inferior individuals.

For PSO, Li [32] showed that PSO with the ring topology is able to simultaneously locate multiple solutions. Qu et al. [33] proposed a distance-based locally informed particle swarm optimizer (LIPS). To enhance the fine search ability of PSO, instead of using **gBest**, several local best particles are combined to form learning exemplars. Liu et al. [34] proposed a niching PSO based on the hierarchical clustering. A small world topology is used in the final stage of the algorithm to improve the exploitation ability. Zheng et al. [35] integrated multiobjective technique and mean-shift clustering method with PSO. The developed algorithm is called multiobjective clustering PSO (MO-C-PSO). A switching evolutionary process is incorporated to refine solutions in each subpopulation. Zou et al. [36] developed a close neighbor mobility strategy that uses Euclidean-based ring topology. The elite mechanism and DE mutation operators are also adopted in the algorithm to improve solution accuracy.

More recently, the crowding technique is embedded into an improved artificial bee colony algorithm (IABC) to make the algorithm capable of tracking and maintaining multiple optima [37]. Two new search mechanisms are designed to enhance the population diversity so that the algorithm can efficiently explore different regions of the search space. Dai et al. [38] developed an optima-identified framework (OIF) for multimodal optimization. The framework is combined with brainstorm optimization (BSO) to handle the task of locating and maintaining multiple optimal solutions. A max-fitness clustering method (MCM) is first proposed to divide the population into a number of clusters. Subsequently, a modified disruption strategy (MDS) is used to identify potential optima within the cluster centers. Two kinds of redistribution strategies (RS) are developed to make the most of the individuals according to which type of clusters they belong to. The experimental results carried out in [38] showed that OIF-BSO reached state-of-the-art performance.

Besides the abovementioned algorithms, considerable research effort has been devoted to the development of new techniques [39–47]. For a comprehensive survey of evolutionary multimodal optimization techniques, interested readers can refer to [11, 12]. From the above review, it can be

noticed that most of the existing work focuses on the development of new genetic operators that are able induce stable niching behavior. One critical problem is how to balance the exploration and exploitation abilities of the multimodal algorithm. This paper explores another path to address the problem by introducing a novel level-based learning strategy.

2.2. Level-Based Learning Strategy. The level-based learning (LL) strategy was developed by Yang et al. [48] to tackle large-scale optimization problems. When handling large-scale optimization problems, the search space grows dramatically as the number of dimensions increases. Moreover, the landscape becomes very complicated. This poses great challenges to the efficiency and efficacy of the existing EAs. On the one hand, to avoid premature convergence, it is required that the algorithms have strong diversity preservation capabilities. On the other hand, to obtain solutions with high accuracy, it is required that the algorithms have fast convergence speed. Therefore, a critical problem encountered by population-based search algorithms is how to balance the exploration and exploitation abilities. The level-based learning strategy is developed in this context.

Taking inspiration from the education activities in schools, Yang et al. [48] developed a level-based learning swarm optimizer. In pedagogy, it is a common practice to separate mixed-level students into different levels [49, 50]. Noting that different students have different cognitive and learning abilities, teachers teach these students in accordance with their aptitude. The mixed-level learning methodology has been widely deployed in various education scenarios. Following this teaching and learning paradigm, Yang et al. [48] divided the swarm into different levels and treated the particles in different levels differently. The update of each particle is guided by two predominant particles in higher levels of the swarm. A new exemplar selection method is developed accordingly.

In the search process, particles are distributed in different regions of the search space. They have different potential for exploring and exploiting the search space. To tell the particles apart, the LL strategy separates the particles into different levels according to their fitness values. Suppose we are solving a minimization problem and there are NP particles in the swarm; the LL strategy first sorts the NP particles in ascending order based on their fitness values. Then, the population is divided into NL levels. The i-th level is denoted by L_i. Better particles belong to higher levels. It is worth noting that the level index starts by 1 and L_1 is the highest level. The number of particles in a level is called "level size." For simplicity, all the levels have the same size, namely, NP/NL.

There are two ideas behind the design of the LL strategy. On the one hand, more promising candidate solutions can be found around better individuals [51]. Therefore, individuals in higher levels possess useful information to guide the evolution of individuals in lower levels. Fast convergence can be achieved by incorporating the information into the position update operator of PSO. On the other hand, individuals in different levels have different potential for exploration and exploitation. It is worth noting that exploitation and exploration are two opposite activities. For individuals in higher levels, they are more likely to be close to the global optimum and have higher potential for exploitation. Conversely, individuals in lower levels have more potential for exploration. Therefore, individuals in lower levels should learn from individuals in multiple higher levels to reach a compromise between exploration and exploitation.

The level-based learning strategy combines the two ideas. Figure 2 illustrates the strategy. From the figure, it can be observed that individuals in lower levels learn from those in higher levels. Moreover, the number of candidate exemplars for learning differs from level to level. As the level goes higher, the number of candidate exemplars becomes smaller. This matches the expectation that better individuals should devote more efforts to exploitation than to exploration. On the other hand, the LL strategy encourages individuals in low levels to pay more attention to exploration.

An exemplar selection method is designed accordingly by applying the LL strategy to the PSO algorithm. For each particle in level L_s, two particles randomly selected from two different higher levels are used in the velocity and position update. Specifically, suppose that there is a particle $\mathbf{X}_i$ belonging to level L_s; its velocity and position are updated as follows:

$$v_{i,j} = r_1 v_{i,j} + r_2\left(x_{p,j} - x_{i,j}\right) + \phi r_3\left(x_{q,j} - x_{i,j}\right), \quad (1)$$

$$x_{i,j} = x_{i,j} + v_{i,j}, \quad (2)$$

where $\mathbf{X}_i = [x_{i,1}, x_{i,2}, \ldots, x_{i,D}]$ and $\mathbf{V}_i = [v_{i,1}, v_{i,2}, \ldots, v_{i,D}]$ are the position and velocity of the particle, respectively. p and q are two mutually exclusive integers randomly selected from the range $[1, s-1]$. We can always make the inequality $p < q < s$ hold through the exchange of p and q. $\mathbf{X}_p = [x_{p,1}, x_{p,2}, \ldots, x_{p,D}]$ and $\mathbf{X}_q = [x_{q,1}, x_{q,2}, \ldots, x_{q,D}]$ are two particles randomly selected from level L_p and level L_q, respectively. In this way, we can guarantee that the learning exemplars of $\mathbf{X}_i$ come from higher levels. r_1, r_2, and r_3 are three real numbers randomly drawn from the interval $[0, 1]$. ϕ is the control parameter of the algorithm. From (1), it can be noted that the velocity update contains three terms. The first term is the inertia of the particle. The second and third terms in the right hand of (1) can be viewed as cognitive terms. Using the update formula, superior particles have higher potential in exploring the search space, while inferior particles have higher potential in exploiting local search regions. This is because the second term of (1) encourages exploitation by learning from a superior exemplar and the third term encourages exploration by learning from a relatively inferior exemplar.

According to the theoretical analysis conducted in [48], the strategy is able to increase the exploration and exploitation capabilities of PSO. It is worth noting that the level-based learning strategy is designed for PSO to solve large-scale optimization problems. The integration of the strategy

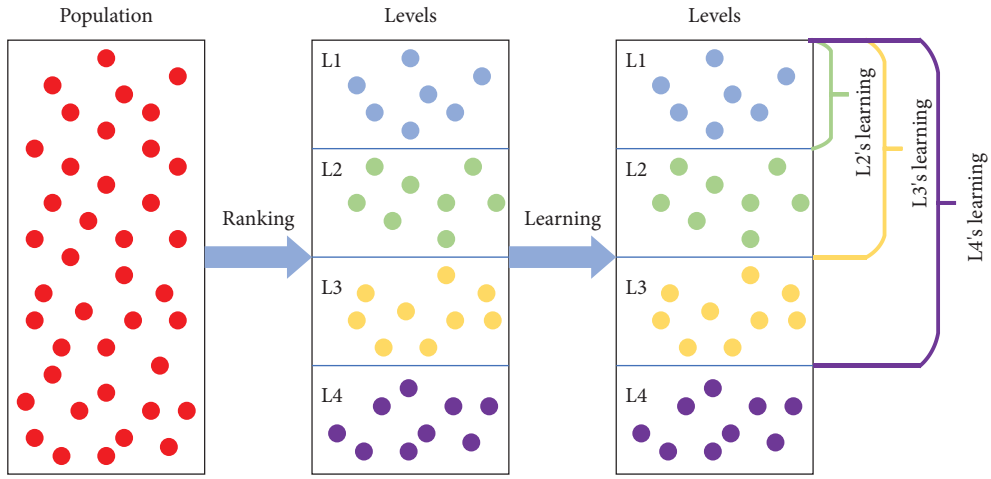

FIGURE 2: Illustration of the level-based learning strategy. The population is sorted and divided into four levels. Individuals in L_4 learn from those in L_1, L_2, and L_3. Analogously, individuals in L_3 learn from those in L_1 and L_2, and so on and so forth.

with other paradigms of EAs is left as a future work. Following this research avenue, in this paper, we developed a multimodal level-based learning strategy and incorporated it into a popular niching DE algorithm.

3. Level-Based Learning Differential Evolution for Multimodal Optimization

In this section, we first present the multimodal level-based learning strategy. Then, we integrate it with a popular multimodal differential evolution algorithm named NCDE [23].

3.1. Multimodal Level-Based Learning Strategy. Multimodal optimization problems have multiple optimal solutions that satisfy the requirements of decision makers. These solutions are generally distributed over the entire search space [52]. On the one hand, to locate all the optimal solutions, optimizers need to have adequate exploration capability to explore the search space. On the other hand, to guarantee the accuracy of the obtained solutions, algorithms need to have adequate exploitation capability to exploit a detected optimal area. Therefore, it is a critical challenge for multimodal evolutionary algorithms to achieve a balance between exploration and exploitation. This is the same challenge encountered by large-scale optimization. Therefore, the level-based learning strategy can be transplanted here to improve the performance of multimodal evolutionary algorithms. Motivated by the observation, we develop a multimodal level-based learning strategy.

In multimodal optimization, multiple optimal solutions need to be located simultaneously. The multiple convergence capability of population-based algorithms is induced by niching methods which divide the whole population into multiple subpopulations. Each subpopulation is directed to its nearby optimum by employing local crossover and mutation operators. In the search landscape of multimodal optimization problems, the optimal regions may have different sizes and shapes. In addition, the evolutionary states of subpopulations may vary from each other due to the stochastic nature of genetic operators. Therefore, a single update operator cannot handle subpopulation diversity. Furthermore, within each subpopulation, the individuals have various distances to the optimum. They should be treated differently to maximize the use of limited computational resources.

The proposed multimodal level-based learning strategy tackles the challenges by dividing the individuals in each subpopulation into multiple levels. For each level, a tailored mutation operator is devised to generate offspring solutions. The utilization of multiple mutation operators handles the diversity of subpopulations. On the other hand, individuals are assigned distinct roles based on which level they belong to. Individuals in higher levels are closer to optimal solutions and are assigned for exploitation. Individuals in lower levels are relatively far from optimal solutions and are assigned for exploration. In this way, the algorithm can strike a balance between exploration and exploitation, which is of great importance when tracking and locating multiple optimal solutions.

Figure 3 illustrates the proposed multimodal level-based learning strategy. The detailed procedures are as follows. For each individual $\mathbf{X}_i$, a subpopulation containing the individual is first constructed. Then, we sort the individuals in the subpopulation in descending order with respect to their fitness values (suppose that we are solving maximization problems). Subsequently, the subpopulation is evenly divided into three levels. We find out which level the individual $\mathbf{X}_i$ belongs to. According to the level index, we conduct different genetic operators to create offspring for the individual $\mathbf{X}_i$. Specifically, if the individual belongs to level L_1, then a self-learning strategy is adopted. If the individual belongs to level L_2, then a directional learning strategy is adopted. Otherwise (i.e., the individual $\mathbf{X}_i$ belongs to level L_3), an exploratory learning strategy is adopted.

The multimodal level-based learning strategy has several characteristics. First, instead of directly taking the entire population into consideration, the learning strategy is

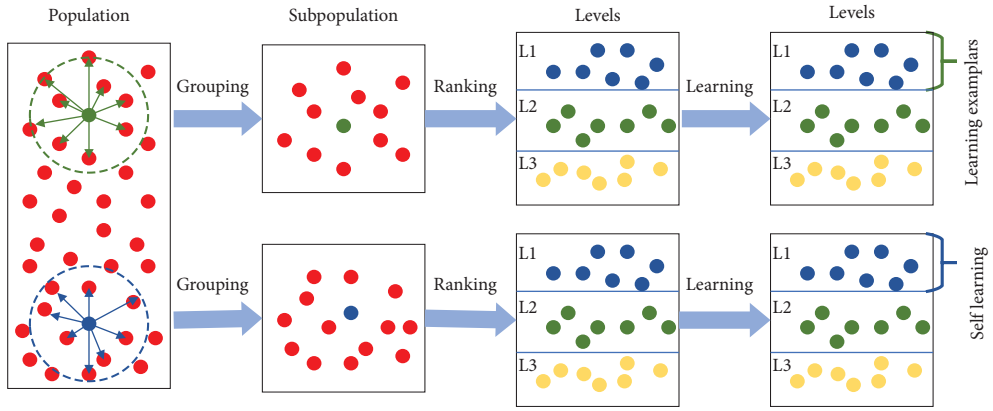

FIGURE 3: Illustration of the proposed multimodal level-based learning strategy. For each individual $\mathbf{X}_i$, we first construct a subpopulation by grouping the neighboring individuals of $\mathbf{X}_i$. Then, the subpopulation is sorted and divided into three different levels.

deployed in the subpopulation level to facilitate the location of multiple optimal solutions. Second, the proposed strategy alleviates the need of setting any control parameters. In the original level-based learning strategy, we need to determine the number layers NL. In the proposed strategy, the control parameter NL is fixed at three since only a small portion of the population is involved in the ranking procedure at each time.

3.2. Level-Based Learning Differential Evolution. We integrate the proposed multimodal level-based learning strategy with a popular niching DE called NCDE [23] to show how the strategy can help improve the performance of existing niching algorithms. NCDE is selected as the basic algorithm mainly for two reasons.

(1) NCDE is a popular niching algorithm that has been widely examined in the literature. It has received recognition from researchers as an effective baseline algorithm for solving multimodal optimization problems.

(2) NCDE has a simple structure and is easy to implement. It only makes small modifications to the canonical DE and is a good testbed for examining new ideas. We can minimize the influence of other operators and investigate the pure effect of the proposed multimodal level-based learning strategy.

The pseudocode of NCDE is presented in Algorithm 1. In each iteration of NCDE, the neighborhood mutation operator is used for reproduction and the crowding technique is used for replacement. Specifically, for an individual $\mathbf{X}_i$, a subpopulation subpop_i is constructed by grouping the m nearest neighbors of $\mathbf{X}_i$, where m is a control parameter of the neighborhood mutation operator. Then, three distinct vectors $\mathbf{X}_{r1}$, $\mathbf{X}_{r2}$, and $\mathbf{X}_{r3}$ are randomly selected from subpop_i. A mutation vector $\mathbf{V}_i = [v_{i,1}, v_{i,2}, \ldots, v_{i,D}]$ is generated using the following formula:

$$\mathbf{V}_i = \mathbf{X}_{r1} + F \cdot (\mathbf{X}_{r2} - \mathbf{X}_{r3}), \tag{3}$$

where F is a scale factor of the mutation operator. Crossover is applied on the mutation vector $\mathbf{V}_i$ to obtain a trial vector $\mathbf{U}_i = [u_{i,1}, u_{i,2}, \ldots, u_{i,D}]$.

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if } \text{rand}_j \le CR \text{ or } j = k, \\ x_{i,j}, & \text{otherwise}, \end{cases} \tag{4}$$

where rand_j is a random real value within the interval [0, 1] and k is a random integer within the interval [1, D]. CR is the crossover rate. We find the individual $\mathbf{X}_j$ that has the smallest distance to $\mathbf{U}_i$. If the fitness of $\mathbf{U}_i$ is better than $\mathbf{X}_j$, then replace $\mathbf{X}_j$ with $\mathbf{U}_i$. From the pseudocode, it can be observed that NCDE uses one single mutation operator for all the individuals to produce offspring solutions regardless of the different roles of the individuals. To achieve better balance between exploration and exploitation in the subpopulation level, we integrate the proposed multimodal level-based learning strategy into NCDE. The integrated algorithm is named LLNCDE and its pseudocode is presented in Algorithm 2. Figure 4 depicts the flowchart of LLNCDE.

LLNCDE starts with a set of NP randomly initialized candidate solutions. After evaluating the fitness of the candidate solutions, the algorithm iteratively updates the solutions until the predefined termination criterion is satisfied. In each iteration, for each candidate solution $\mathbf{X}_i$, the multimodal level-based learning strategy is used to produce an offspring solution $\mathbf{U}_i$. Specifically, the new strategy creates a subpopulation subpop_i by grouping m nearest neighbors of $\mathbf{X}_i$. Then, the subpopulation is sorted and divided into three levels. According to which level the individual $\mathbf{X}_i$ is classified to, a different update strategy is adopted for reproduction. After the offspring solution $\mathbf{U}_i$ has been generated, the crowding selection mechanism is used to update the population. We find the nearest neighbor $\mathbf{X}_j$ of $\mathbf{U}_i$ in the population and replace $\mathbf{X}_j$ with $\mathbf{U}_i$ if $\mathbf{U}$ has better fitness.

Before generating the offspring population, the guide selection method is used to select parental solutions. The guide selection method is also referred to as mating selection in EAs. EAs are popular meta-heuristic search (MHS) algorithms that mimic the biological evolution of living things. The iterative search process of EAs is illustrated in Figure 5. EAs maintain a population of candidate solutions. A group of parental solutions is selected from the whole population through mating selection. Then, genetic operators (crossover and mutation) are performed to produce offspring solutions.

(1) Generate an initial population of individuals POP = $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N\}$ by uniformly and randomly sample NP individuals from the search space;
(2) Evaluate the fitness values of the NP individuals;
(3) FEs = NP;
(4) **while** FEs < MaxFEs **do**:
(5) **for** $i = 1$ to N:
(6) Construct a subpopulation subpop_i by grouping m nearest neighbors of individual $\mathbf{X}_i$;
(7) Generate an offspring solution $\mathbf{U}_i$ using canonical genetic operators of DE (shown in (3) and (4)) where the vectors $\mathbf{X}_{r1}$, $\mathbf{X}_{r2}$, and $\mathbf{X}_{r3}$ are randomly selected from subpop_i;
(8) Evaluate the fitness of $\mathbf{U}_i$;
(9) FEs = FEs + 1;
(10) Find the individual $\mathbf{X}_j$ in the population that has the smallest distance to $\mathbf{U}_i$;
(11) **if** the fitness of $\mathbf{U}_i$ is better than $\mathbf{X}_j$ **then**:
(12) Replace $\mathbf{X}_j$ with $\mathbf{U}_i$;
(13) **end if**
(14) **end for**
(15) **end while**

Algorithm 1: NCDE.

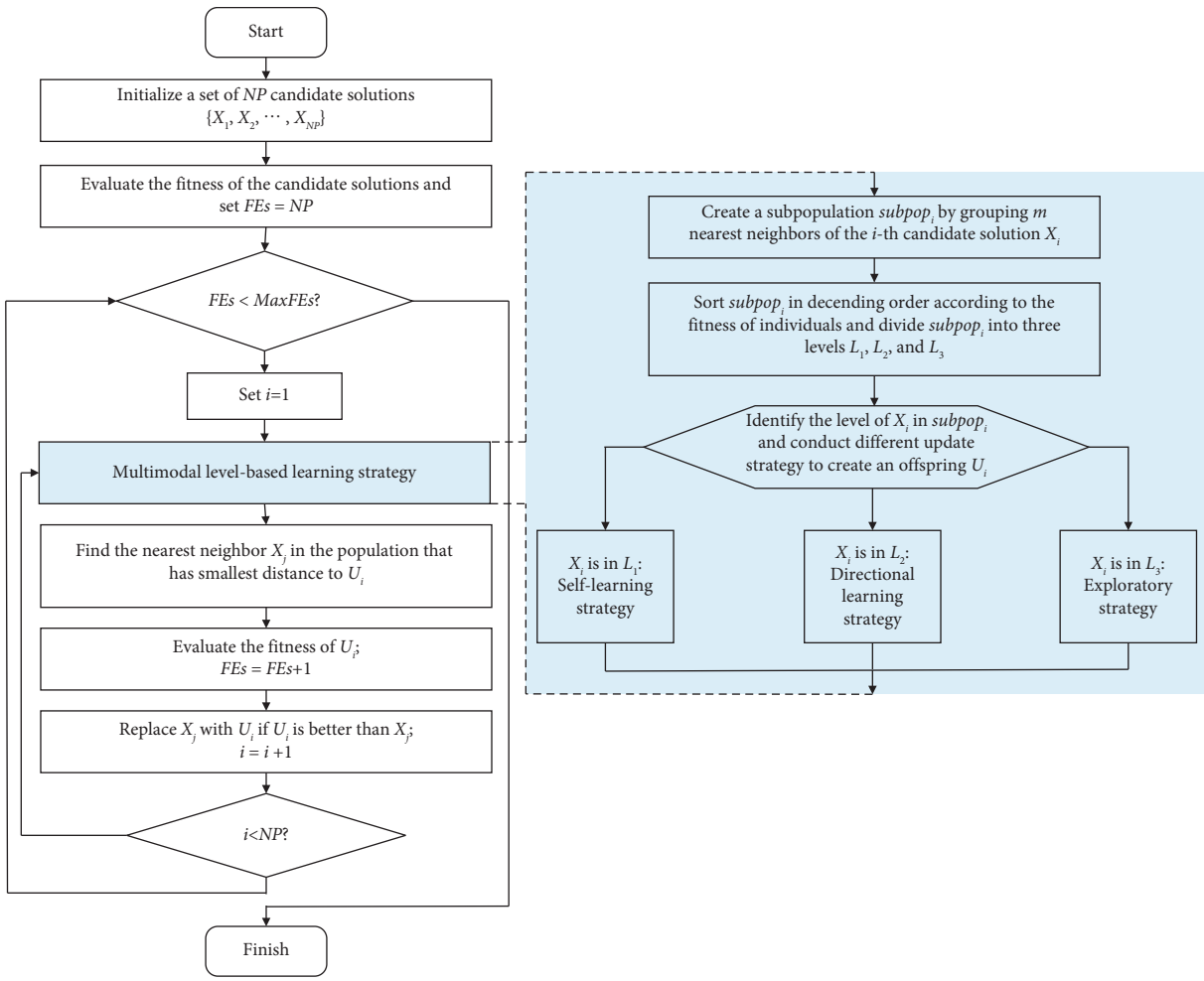

Figure 4: Flowchart of LLNCDE.

(1) Generate an initial population of individuals POP = $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_{NP}\}$ by uniformly and randomly sample NP individuals in the search space;
(2) Evaluate the fitness values of the NP individuals;
(3) FEs = NP;
(4) **while** FEs < MaxFEs **do**:
(5) **for** $i = 1$ to NP:
(6) Create a subpopulation subpop_i by grouping m nearest neighbors of individual $\mathbf{X}_i$;
(7) Sort individuals in subpop_i in descending order with respect to their fitness;
(8) Evenly divide subpop_i into three levels L_1, L_2, and L_3, check which level $\mathbf{X}_i$ belongs to;
(9) **if** $\mathbf{X}_i$ is in level L_1 **then**:
(10) Generate an offspring solution $\mathbf{U}_i$ with the self-learning strategy (5) (6);
(11) **end if**
(12) **if** $\mathbf{X}_i$ is in level L_2 **then**:
(13) Generate an offspring solution $\mathbf{U}_i$ with the directional learning strategy (7);
(14) **end if**
(15) **if** $\mathbf{X}_i$ is in level L_3 **then**:
(16) Generate an offspring solution $\mathbf{U}_i$ with the exploratory strategy (3) (4);
(17) **end if**
(18) Evaluate the fitness of $\mathbf{U}_i$;
(19) FEs = FEs + 1;
(20) Find the individual $\mathbf{X}_j$ in the population that has the smallest distance to $\mathbf{U}_i$;
(21) **if** the fitness of $\mathbf{U}_i$ is better than $\mathbf{X}_j$ **then**:
(22) Replace $\mathbf{X}_j$ with $\mathbf{U}_i$;
(23) **end if**
(24) **end for**
(25) **end while**

Algorithm 2: LLNCDE.

Subsequently, the offspring solutions and the current population are combined, and environmental selection is performed on the combined population to choose a new set of candidate solutions that enters the next iteration.

The roulette wheel selection and the tournament selection are two commonly used selection methods in the EA literature [53]. Note that candidate solutions selected for reproduction directly affect the direction and success of the search; a novel selection method based on fitness-distance balance (FDB) is developed in [54]. The new method assigns different scores to different individuals according to their fitness values, as well as their distances to the best solution. Extensive experiments have been conducted in [54], and the results showed that the FDB-based selection method is able to identify candidate solutions with the highest potential and improve the search efficiency. In the FDB selection method, for a candidate solution, the weighted combination of the normalized fitness and distance values is computed as the score of the solution. The weighting coefficient w influences the degree of exploitation and exploration. However, different search periods require different intensities of exploitation and exploration. To address this problem, an improved version called dynamic FDB selection method is proposed in [55]. The improved selection method is applied to the coordination of directional overcurrent relay (DOCR) problem. The numerical results demonstrated that the improved method is very effective in minimizing the relay operating time. More recently, an adaptive FDB selection method is invented in [56] and is successfully applied to the economical operation of modern power grids with uncertainties.

In [55], the existing selection methods are divided into four categories, namely, random, ordinal-based, greedy, and probabilistic methods. The method used in the proposed algorithm belongs to the second category, which selects solutions from the population in an ordinal manner. Note that the goal of this paper is to improve the performance of existing multimodal optimization algorithms through the introduction of better learning strategy. Following this research line, the proposed approach does not change the basic structure of the baseline algorithm and LLNCDE inherits the ordinal-based selection method from DE.

The core of the multimodal level-based learning strategy lies in the population creation method used in the design of MHS algorithms. In the literature, a number of MHS algorithms have been developed to solve various complex optimization problems [57, 58]. The population creation methods used in different MHS algorithms vary significantly. Evolution-based GA and DE use crossover and mutation operators to produce offspring solutions. Many variants of the crossover and mutation operators have been developed to enhance the search efficiency of GA and DE [59–61]. Different from GA and DE, swarm-based PSO, ACO, and ABC use foraging operators to explore the area of the solution space. PSO [62] conducts velocity and position update operators to refresh particles in the swarm. ACO [63] uses the pheromone-guided route construction operator and the pheromone update operator to construct new solutions

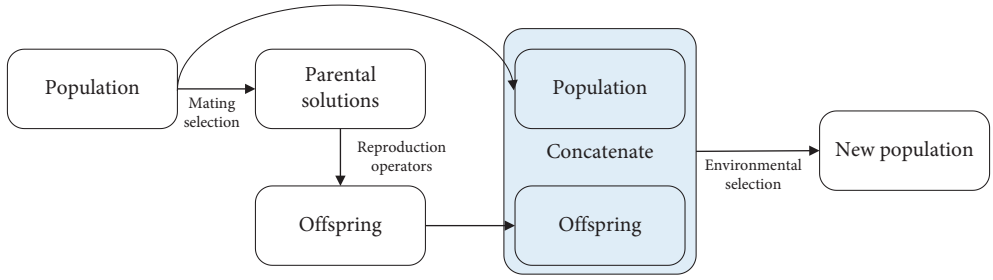

Figure 5: Iterative search process of EAs.

to combinatorial optimization problems. ABC [64] divides the search phase into three parts, i.e., employed bee phase, onlooker bee phase, and scout bee phase. In each phase, a different update operator is used to find food sources with higher nectar amount.

Since the baseline algorithm is built upon DE, the proposed learning strategy adopts the basic genetic operators of DE. To increase the efficiency of the algorithm in finding multiple optimal solutions, the proposed learning strategy adopts different update operators for different types of individuals. The basic principle is shown in Figure 6. For a candidate solution $\mathbf{X}_i$, we first create a subpopulation subpop_i by grouping m nearest neighbors of $\mathbf{X}_i$. Then, the subpopulation is divided into three levels L_1, L_2, and L_3. If $\mathbf{X}_i$ is in level L_1, then it is in the part of subpop_i that is most close to an optimum. Therefore, we try to improve $\mathbf{X}_i$ by adding a small perturbation generated from a Gaussian distribution. If $\mathbf{X}_i$ is in L_2, then we try to improve $\mathbf{X}_i$ by adding a directional vector pointing toward a better individual in L_1. If $\mathbf{X}_i$ is in L_3, then we allow the individual to explore the search space freely by employing the traditional mutation operator of DE. The intention is to check whether there are other optima in the search space. In this way, the subpopulation can efficiently exploit the nearby peak region while at the same time spare efforts to explore other optima. The detailed offspring generation steps for the three levels of individuals are described as follows.

(a) $\mathbf{X}_i$ belongs to level L_1: This means that $\mathbf{X}_i$ is a dominant individual in the subpopulation and is possibly near an optimal solution. Therefore, a self-learning strategy (i.e., local search) is adopted by the individual to produce new solutions. This can enhance the exploitation capability of LLNCDE and increase the solution accuracy. The self-learning strategy is formulated as follows:

$$u_{i,j} = x_{i,j} + Gr_j, \tag{5}$$

where Gr_j is a random number drawn from the Gaussian distribution $N(\mu, \sigma)$. In the Gaussian distribution, the mean μ is set to 0 and the variance σ is set to 10^p, where p is a real value that dynamically decreases from -1 to -6 as the number of fitness evaluations increases. More specifically, p is set as follows:

$$p = -1 + (-5) \times \frac{\text{FEs}}{\text{MaxFEs}}. \tag{6}$$

The principle of the self-learning strategy is to improve the solutions by making small random perturbations. The Gaussian distribution is very suitable for the task of generating random perturbations owing to its well-studied properties. With the Gaussian distribution, the probabilities of generating random values above and below the mean value are equal and the variance of the random values is controllable. Since the optimum may appear in any direction of the candidate solution, the mean μ of the Gaussian distribution is fixed at 0 to avoid search biases. As for the standard deviation σ, the parameter p is dynamically adjusted in a way that σ will gradually decrease as the number of fitness evaluations increases. The principle behind the setting is that as the search progresses, the individuals will gradually approach the optimum. As the distance between the optimum and the individual becomes small, we need to shrink the size of perturbations to avoid overshooting the optimum. In addition, gradually reducing the size of perturbations is also helpful for the algorithm to find solutions that reach the predefined accuracy requirement.

(b) $\mathbf{X}_i$ belongs to level L_2: This means that $\mathbf{X}_i$ is in a medium position of its subpopulation and is likely to be improved by learning from a dominant individual in level L_1. Therefore, a directional learning strategy is adopted by $\mathbf{X}_i$ to produce an offspring solution. This strategy turns uninformed search into informed search and is helpful for increasing the convergence speed of LLNCDE. The directional learning strategy is formulated as follows:

$$\mathbf{U}_i = \mathbf{X}_i + F \cdot (\mathbf{X}_{r1} - \mathbf{X}_i) + F \cdot (\mathbf{X}_{r2} - \mathbf{X}_{r3}), \tag{7}$$

where $\mathbf{X}_{r1}$, $\mathbf{X}_{r2}$, and $\mathbf{X}_{r3}$ are individuals randomly selected from level L_1 of subpop_i. It is worth noting that $\mathbf{X}_{r1} - \mathbf{X}_i$ is a vector pointing from $\mathbf{X}_i$ to $\mathbf{X}_{r1}$. This vector guides the individual $\mathbf{X}_i$ to evolve toward a dominant individual in level L_1.

(c) $\mathbf{X}_i$ belongs to level L_3: This means that $\mathbf{X}_i$ is an inferior individual in its subpopulation. We can assign exploration tasks to the individual without

significantly hindering the evolution of the subpopulation. Therefore, an exploratory learning strategy is adopted by $\mathbf{X}_i$ to produce an offspring solution. This strategy can promote the exploration capability of LLNCDE so that more optimal solutions can be detected. The exploratory strategy is the same as the canonical DE operator formulated in (3), except that the individuals $\mathbf{X}_{r1}$, $\mathbf{X}_{r2}$, and $\mathbf{X}_{r3}$ are randomly selected from the combination of L_1 and L_2. After obtaining the mutation vector $\mathbf{V}_i$, $\mathbf{V}_i$ and $\mathbf{X}_i$ undergo crossover to generate an offspring solution $\mathbf{U}_i$.

It is worth noting that the offspring solution may cross the search boundary. Many different bound handling techniques have been proposed in the literature [65, 66]. Note that the performance, disruptiveness, and population diversity of DE are greatly affected by the choice of constraint handling techniques [67, 68]. To guarantee reproducibility of results, we specify the way of dealing with solutions generated outside the search space. Two bound handling techniques (i.e., the nearest method and the random method) are combined in a probabilistic manner. Figure 7 illustrates the two methods. In the nearest method (illustrated in Figure 7(a)), for the j-th dimension, the newborn individual is set back to the boundary. It is formulated as follows:

$$u_{i,j} = \begin{cases} x_j^{\min}, & \text{if } u_{i,j} < x_j^{\min}, \\ x_j^{\max}, & \text{if } u_{i,j} > x_j^{\max}, \end{cases} \tag{8}$$

where $x_j^{\min}$ and $x_j^{\max}$ are the lower bound and upper bound of the j-th dimension, respectively. In the random method (illustrated in Figure 7(b)), the j-th dimension of the newborn individual is set to a random value within the feasible range $[x_j^{\min}, x_j^{\max}]$. The two ways of fixing infeasible values are assigned equal probabilities by means of coin tosses. The reason of using these boundary handling techniques is that in multimodal landscape, some peaks may locate in the boundary or places near the boundary. By employing two different boundary handling techniques, the algorithm can tackle a wide variety of problems with different optima distributions.

The existing boundary constraint handling techniques can be put into two categories, namely, deterministic techniques and stochastic techniques [68]. The deterministic techniques always return the same feasible value when the same infeasible value is presented. In comparison, the stochastic techniques may return different values when the same infeasible value is inputted. From this point of view, the boundary handling technique used in the paper belongs to the second category.

By identifying the role of $\mathbf{X}_i$ in its subpopulation, we can choose the most appropriate genetic operator for $\mathbf{X}_i$ to improve the search efficiency. The differences between the method developed in [48] and the proposed multimodal level-based learning strategy lie in two aspects.

(1) The method developed in [48] is used for large-scale optimization, in which the goal is to find a single global optimum. Therefore, the entire population is divided into multiple levels. In multimodal optimization, individuals are assigned to search for different optima. Considering this, in our strategy, for each individual $\mathbf{X}_i$, a subpopulation is formed by grouping the neighbors of $\mathbf{X}_i$. Then, the subpopulation is divided into multiple levels. The proposed strategy is more suitable for the goal of finding multiple optimal solutions.

(2) In [48], all the individuals use the same update operator to produce new candidate solutions. For an individual in a specific level, the update operator selects learning exemplars from the collection of higher level individuals. In our multimodal level-based learning strategy, a tailored mutation operator is designed for each level of individuals. Individuals with higher fitness values are assigned for exploitation while those with lower fitness values are assigned for exploration. In this way, the algorithm can achieve a better balance between exploration and exploitation.

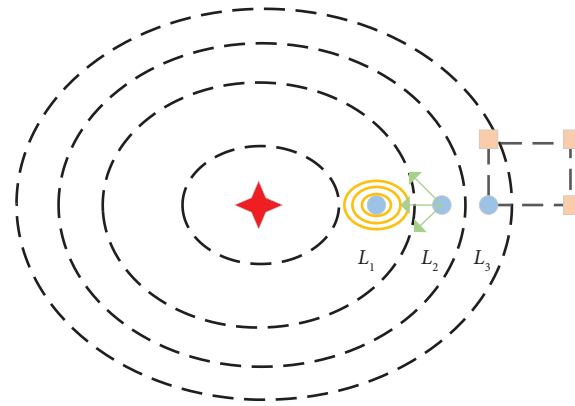

Figure 6: Another perspective on the multimodal level-based learning strategy.

After generating an offspring solution $\mathbf{U}_i$, the crowding selection mechanism is used to update the population. The update mechanism plays a very important role in the population management of MHS algorithms. To our knowledge, the update mechanism is also referred to as environmental selection in the community of evolutionary computation, as illustrated in Figure 5. Existing update mechanisms are mostly designed based on the assumption of "survival of the fittest." However, the fitness value used in the design of update mechanisms cannot accurately reflect the environmental adaptability in nature. With the greedy selection method, individuals with better fitness values are admitted to the next iteration regardless of their similarities. This may cause diversity loss and lead to premature convergence. To address the issue, a multicriteria survival mechanism is developed in [69] to effectively mimic the functioning of nature. Specifically, a new natural survivor method (NSM) is devised to calculate the score of an individual by combining three factors, namely, the fitness value of the individual, the individual's contribution to the population, and the individual's contribution to the mating

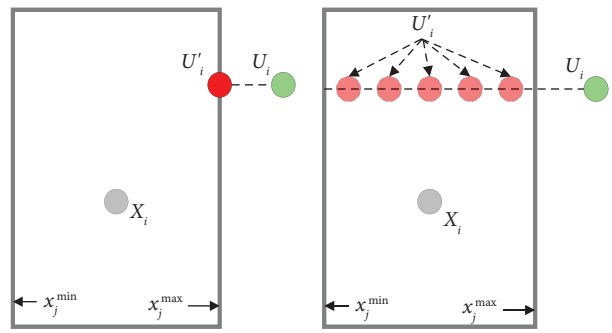

FIGURE 7: Bound handling techniques used in the level-based learning strategy. $\mathbf{X}_i$ represents the parental individual. $\mathbf{U}_i$ represents the newborn individual that crosses the search boundary. $\mathbf{U}'_i$ represents the rectified individual. (a) The nearest method. (b) The random method.

pool. The three factors are combined using three weighting coefficients, and a dynamic weighting algorithm is devised to meet the search requirements in different search periods. The experimental analysis conducted in [69] showed that the novel NSM method provides a superior performance improvement to the existing MHS algorithms.

In the proposed algorithm, the fitness is used as an indicator to assess the quality of candidate solutions. In [69], the survival selection methods are divided into three categories, i.e., the fitness value-based method, the direct method, and the NSM score-based method. From this point of view, the method used in the proposed algorithm belongs to the first category. In the proposed algorithm, the crowding selection mechanism is adopted to avoid genetic drift phenomenon and maintain population diversity. The working principle of the crowding selection mechanism is depicted in Figure 8. Suppose that there are two peaks in the search landscape (denoted by stars) and each peak is surrounded by several candidate solutions (denoted by circles). After an offspring solution (denoted by squares) has been generated, it is compared with the nearest neighbor in the population to compete for admission to the next iteration. Figure 8 shows two cases of the crowding competition. In case A, the offspring is located in the same niche as its parent. The parental solution is replaced by its offspring which is closer to the peak. In case B, the offspring appears in another niche and competes with population members in that niche. The genetic drift is avoided by restricting the competition between individuals that are far apart. From the examples shown in Figure 8, the crowding selection mechanism takes into account both the fitness values and the relationship between individuals. It is beneficial to the maintenance of multiple optimal solutions. From this perspective, the crowding selection mechanism can be viewed as a primitive version of the NSM method developed in [69].

It is worth noting that LLNCDE is not a simple combination of the learning strategy of PSO and the clearing method. The differences between the proposed approach and the ensemble of PSO and the clearing method mainly lie in two aspects.

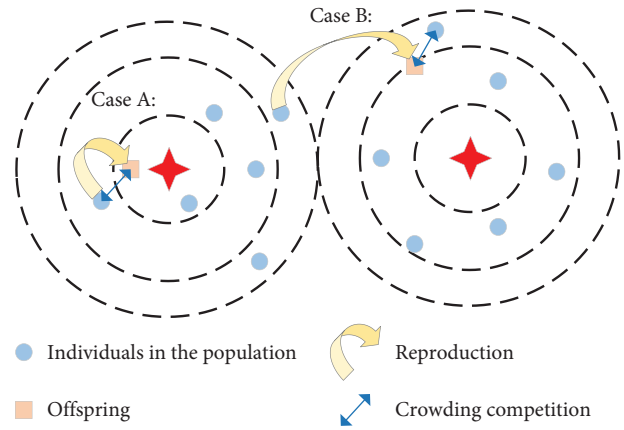

FIGURE 8: Illustration of the crowding selection mechanism.

(1) In the clearing procedure, each subpopulation contains a dominant individual that has the best fitness value. An individual belongs to a given subpopulation when its distance with the dominant is less than a threshold value σ. In the proposed approach, each individual $\mathbf{X}_i$ will serve as a subpopulation center and $\mathbf{X}_i$ is not necessarily the dominant of the subpopulation. The position of $\mathbf{X}_i$ in the subpopulation varies according to its fitness.

(2) As for the update strategy, the major difference lies in the way of selecting learning exemplars. In the canonical PSO, for a candidate solution $\mathbf{X}_i$, the global and local best individuals are used to update $\mathbf{X}_i$. In comparison, the multimodal level-based learning strategy selects the learning exemplars from the group of individuals in higher levels of the same subpopulation. The purpose of this modification is to ensure that the individuals can be guided toward their nearby optimum in a more efficient manner.

The proposed learning strategy is integrated with DE to enhance its performance in solving multimodal optimization problems. One benefit of the proposed strategy is that it does not change the structure of the integrated algorithm. Its function is to identify the role of individuals in their

corresponding subpopulations. Based on the identified role, we apply a tailored genetic operator for individuals in each level to produce new candidate solutions. From this point of view, the level-based learning strategy can be applied to various types of population-based algorithms. If the base algorithm uses the same mutation operator for all individuals, then it can be improved by the proposed learning strategy. However, since the update operator of different population-based algorithms may vary significantly, we need to come up with new update operators for the algorithms. Therefore, extra work needs to be done before applying the proposed learning strategy to algorithms other than DE.

In LLNCDE, two algorithm configurations may affect the exploration and exploitation capabilities of the search procedure.

(1) Neighborhood size m: The neighborhood size m is a parameter inherited from NCDE. When producing an offspring solution for candidate solution $\mathbf{X}_i$, a subpopulation subpop_i is formed by grouping m nearest neighbors of $\mathbf{X}_i$. Individuals in the subpopulation serve as a gene pool to provide basic material for reproduction. From this point of view, a large neighborhood size is beneficial to the exploration of the search space since a set of diverse individuals can produce diverse donor vectors. On the other hand, a small neighborhood size is beneficial to the exploitation of local regions since the donor vector generated by a small number of neighboring individuals might still be in the same local region.

(2) Reproduction operators for each level of individuals: The subpopulation subpop_i is divided into three levels according to the fitness of individuals. Three mutation operators are specified for the three levels of individuals to achieve a balance between exploration and exploitation. The operator used for each level of individuals can be adjusted to bias the search toward exploration or exploitation.

There are several major components of LLNCDE, namely, the initialization procedure, the construction, sorting, and division of subpopulations, the reproduction procedure, and the environmental selection operator. The initialization procedure is only executed once, and the time spent on the procedure is $O(D{\cdot}\text{NP})$. Other components are in the main loop of LLNCDE and are executed repeatedly until the termination criterion is satisfied. For each individual, we construct a subpopulation by grouping its m nearest neighbors. To achieve this, we need to first calculate the individual's distances to all other members in the population. This takes $O(D{\cdot}\text{NP})$ time. Then, we find the m smallest numbers among the distances. $O(m{\cdot}\text{NP})$ time is required in this process. Overall, the complexity of subpopulation construction is $O(\max(D{\cdot}\text{NP}, m{\cdot}\text{NP}))$. The sorting of a subpopulation takes $O(m\log m)$ time and the division takes constant time. Subsequently, the reproduction and environmental selection procedures take $O(D)$ and $O(D{\cdot}\text{NP})$ time, respectively. Since we need to process all NP individuals in each iteration, the overall time complexity of LLNCDE is $O(\max(m{\cdot}\text{NP}^2, D{\cdot}\text{NP}^2))$. It is worth noting that the time complexity of NCDE is $O(D{\cdot}\text{NP}^2)$. Therefore, the proposed multimodal level-based learning strategy does not impose serious burden on the time complexity.

4. Experimental Study

In this section, we carry out experiments to study the performance of the multimodal algorithm developed using the proposed multimodal level-based learning strategy. We test LLNCDE on a set of benchmark multimodal optimization problems and compare it with several popular niching algorithms. The convergence speeds of the algorithms are examined in this section as well. Furthermore, we test LLNCDE on multiple root finding problems extracted from real-world tasks and compare it with several state-of-the-art algorithms.

4.1. Experimental Setup

4.1.1. Algorithms in Comparison. We compare LLNCDE with several popular multimodal evolutionary algorithms, namely, r3pso-lhc [32], LIPS [33], NCDE [23], dADE/nrand/1 [25], PNPCDE [26], LoICDE [13], IABC [37], and OIF-BSO [38]. The first two algorithms are PSO-based algorithms. LIPS uses a locally informed strategy to construct learning exemplar for each particle, while r3pso-lhc is the ring topology PSO in which the neighbors of each particle are defined by the two neighboring nodes in the ring. The next four algorithms are DE-based algorithms. NCDE is the crowding DE with a neighborhood mutation strategy. Similarly, both LoICDE and PNPCDE adopt crowding DE as the basic framework. LoICDE incorporates a novel information sharing mechanism. When selecting parents for mutation, distance and fitness information is taken into consideration. Individuals that have better fitness and smaller distances to the base vector $\mathbf{X}_{r1}$ have higher probabilities of being selected. To facilitate tracking and maintaining optima, PNPCDE incorporates a parent-centric mutation operator that makes use of normalized neighborhood. The last two algorithms are based on other evolutionary computation paradigms. IABC is an improved ABC algorithm embedded with the crowding technique. OIF-BSO is a novel BSO algorithm with new clustering, disruption, and redistribution strategies. The algorithms are implemented in C++ and are compiled using the Microsoft compiler. They are executed on a computer running Windows 10, with Intel Xeon Gold 5218R 2.10-GHz CPU and 64 GB RAM.

4.1.2. Test Functions. A set of benchmark functions are adopted in the experiment to examine the niching performance of the algorithms. The benchmark function set contains 20 functions with different characteristics [52]. All the functions are to be maximized, and therefore the optimal solutions are also referred to as "peaks." The first ten functions ($F1$–$F10$) are simple, well-known functions

collected from the literature. The remaining functions ($F11$–$F20$) are complex functions constructed using composition methods. These functions are scalable with respect to dimensions. Moreover, the number of global optima can be adjusted by users, providing additional flexibility. Another important feature of the benchmark function set is that the shapes and distributions of peaks vary significantly from function to function. Some test functions have evenly distributed peaks (e.g., $F6$, $F8$, and $F10$), while others have an uneven peak distribution (e.g., $F7$, $F9$, and $F10$–$F20$). A large number of optima may reside in a small portion of the search space. Furthermore, a test function may have different types of peaks in its search space. For example, both symmetrical Gaussian peaks and asymmetrical sharp peaks can be observed in the search space of test functions $F11$–$F20$. Table 1 lists the information of the test functions. More detailed descriptions of the test functions (including mathematical formula and properties) can be found in [52].

4.1.3. Performance Measure. For each benchmark function, the number of global optima and their fitness (i.e., peak height) are available for evaluating the performance of algorithms. To determine whether an optimum has been successfully located, we need to specify an accuracy level ε. If the difference between the fitness of an obtained solution and the peak height is smaller than ε, the solution is identified as an optimum. A niche radius sufficient to separate two close peaks is also provided by the benchmark set to avoid duplicate counting of found peaks. We use two performance metrics to assess the niching performance of evolutionary multimodal algorithms, namely, peak ratio (PR) and success rate (SR).

(a) Peak ratio (PR): PR measures the average percentage of peaks located by the algorithm over all the runs. Specifically, PR is formulated as follows:

$$\mathrm{PR} = \frac{\sum_{i=1}^{\mathrm{NR}} \mathrm{NFP}_i}{\mathrm{NKP} \cdot \mathrm{NR}}, \tag{9}$$

where NR is the total number of runs, NKP is the number of known peaks of the test function, and NFP_i denotes the number of found peaks in the i-th run.

(b) Success rate (SR): SR measures the ratio of the number of successful runs to the total number of runs. A successful run is a run in which all the optima are detected. Specifically, SR is formulated as follows:

$$\mathrm{SR} = \frac{\mathrm{NSR}}{\mathrm{NR}}, \tag{10}$$

where NSR represents the number of successful runs.

4.1.4. Parameter Settings. The proposed multimodal level-based learning strategy does not introduce new parameters. The algorithms involved in the comparison are specially designed for solving multimodal optimization problems.

Table 1: Test functions used in the experiment.

Test function	Name	NPK
$F1$	Five-uneven-peak trap (1D)	2
$F2$	Equal maxima (1D)	5
$F3$	Uneven decreasing maxima (1D)	1
$F4$	Himmelblau (2D)	4
$F5$	Six-hump camel back (2D)	2
$F6$	Shubert (2D)	18
$F7$	Vincent (2D)	36
$F8$	Shubert (3D)	81
$F9$	Vincent (3D)	216
$F10$	Modified Rastrigin (2D)	12
$F11$	Composition function 1 (2D)	6
$F12$	Composition function 2 (2D)	8
$F13$	Composition function 3 (2D)	6
$F14$	Composition function 3 (3D)	6
$F15$	Composition function 4 (3D)	8
$F16$	Composition function 3 (5D)	6
$F17$	Composition function 4 (5D)	8
$F18$	Composition function 3 (10D)	6
$F19$	Composition function 4 (10D)	8
$F20$	Composition function 4 (20D)	8

The developers have conducted extensive experiments on a large set of problems to find the most suitable parameters for their algorithms. Therefore, we follow the recommendation of the algorithm developers. The population size NP of the algorithms is fixed at 100. In LIPS, the parameter $n$size gradually increases from 2 to 5 as the number of fitness evaluations increases. For DE-based algorithms, the scale parameter F and the crossover rate CR are set to 0.5 and 0.9, respectively. In NCDE, the neighborhood size parameter m is set to 10% of the population size NP. In LoICDE, the parameter k that controls the neighborhood size is dynamically decreased from 12.5% NP to 5% NP. In PNPCDE, the parameter k is set to 10% NP. In OIF-BSO, the cluster size M is set to 3. The termination criterion is defined by the maximum number of fitness evaluations (MaxFEs). We use the recommended setting provided in [52]. The setting of MaxFEs for all the test functions is given in Table 2. The accuracy level used to determine whether an optimum is detected is set to a challenging value, i.e., $1E-04$, to examine the niching performance of the algorithms. The algorithms are not only required to locate the coarse-grained regions of peaks but also required to fine-tune the solutions to ensure that their fitness is within the error margin. This brings significant challenges to the exploration and exploitation capabilities of multimodal algorithms. Generally, the higher the accuracy level, the more challenging the optimization task. Each algorithm is executed 50 times for each test function to obtain statistically reliable results.

4.2. Overall Performance. The experimental results of the compared algorithms are listed in Table 3. Both PR and SR values are reported in the table. In addition, Wilcoxon rank-sum tests with the Bonferroni correction [70] are conducted to check whether there are significant differences between the results of LLNCDE and those of the compared algorithms. The notation "+" represents that the proposed

Table 2: Setting of MaxFEs for the test functions.

Test function	MaxFEs
$F1$–$F5$	$5.00E+04$
$F6$–$F7$	$2.00E+05$
$F8$–$F9$	$4.00E+05$
$F10$–$F13$	$2.00E+05$
$F14$–$F20$	$4.00E+05$

LLNCDE is significantly better than the compared algorithm, while the notation "–" indicates the opposite. The last row of the table provides the B/E/W counts of the comparison results, which means that the number of test functions that LLNCDE performs is better than, equal to, or worse than the corresponding algorithm.

From the table, it can be concluded that LLNCDE outperforms the other algorithms on at least eight test functions. On the other hand, LLNCDE is beaten by other algorithms on at most four test functions. For the simple test functions $F1$–$F5$, all the algorithms are able to achieve approximately 100% PR and SR. $F6$–$F9$ are test functions with an enormous number of peaks. Since the population size NP is fixed at 100, it is very difficult for the algorithms to locate all the optima without additional archive techniques. Therefore, the PR and SR values obtained by the algorithms drop rapidly on these functions. $F10$ is a test function with evenly distributed Gaussian peaks and is relatively easy to solve. The PR and SR values of the algorithms are close to 100%. The remaining functions are complex composite functions whose landscapes are rugged and have many local optima. The algorithms cannot find all the optima for these functions, and therefore the SR values drop to 0. Nevertheless, the PR values achieved by LLNCDE are higher than those of the compared algorithms. With the enhancement of the multimodal level-based learning strategy, LLNCDE is able to achieve a better balance between global exploration and local exploitation and locate more peaks. It can be inferred from the PR values that LLNCDE can find more than one optimum for all the complex composite functions.

To intuitively show the multiple convergence ability of LLNCDE, the distribution of individuals in the final iteration is depicted in Figure 9. Figure 9 shows the individuals (represented by red dots) on two-dimensional test functions ($F4$–$F7$, and $F10$–$F13$). From the figure, it can be observed that the algorithm can successfully locate most of the global peaks, as well as some local peaks.

To show the distribution of the numerical results obtained through 50 independent runs, the box plots of the algorithms on each test function are presented in Figure 10. From the figure, it can be observed that on a relatively large number of test functions, the medians of the box plots associated with LLNCDE are higher than those of the compared algorithms. In addition, the interquartile ranges (box lengths) of LLNCDE are generally smaller than those of its competitors. These results indicate that the proposed algorithm with the multimodal level-based learning strategy possesses more stable performance than the compared algorithms.

To analyze the overall performance of the algorithms, the Friedman test is conducted to detect differences in algorithms across multiple test functions. The statistical test results obtained using the KEEL software [71] are tabulated in Table 4. According to the Friedman test, LLNCDE has the highest ranking, following by dADE/nrand/1, IABC, and OIF-BSO.

The proposed multimodal level-based learning strategy intends to improve the algorithm performance by refining the learning strategies for individuals in different levels of the subpopulation. The niching approach adopted in LLNCDE is inherited from NCDE. To investigate the individual contributions of the niching approach and the new learning strategy, we compare the experimental results of CDE, NCDE, and LLNCDE. It is worth noting that the difference between CDE and NCDE is that a niching approach called neighborhood-based mutation is incorporated in NCDE. The difference between NCDE and LLNCDE is that the multimodal level-based learning strategy is incorporated in LLNCDE. Table 5 lists the PR and SR values of the three algorithms. The rankings of the algorithms obtained by Friedman's test are presented in Table 6.

From the tables, it can be noticed that NCDE performs better than CDE and is outperformed by LLNCDE. Owing to the neighborhood-based mutation, NCDE shows an edge over CDE when handling complex composite multimodal optimization problems. LLNCDE further enhances the performance of NCDE by refining learning strategies for each level of individuals. The pairwise comparisons illustrate the performance improvement brought by the use of the niching approach and the multimodal level-based learning strategy, respectively.

4.3. Convergence Speed. In this section, we move on to examine the convergence speed of the algorithms. After specifying an accuracy level ε, the convergence speed is defined by the average number of fitness evaluations required to find all the optimal solutions over multiple runs. It is calculated as follows:

$$\text{AvgFEs} = \frac{\sum_{i=1}^{\text{NR}} \text{FEs}_i}{\text{NR}}, \tag{11}$$

where FEs_i is the number of fitness evaluations spent by the algorithm in the i-th run to locate all the global optima. If the algorithm fails to locate all the global optima after exhausting the given fitness evaluations (i.e., MaxFEs), then MaxFEs is counted as FEs_i.

Table 7 records the convergence speeds of the algorithms. Since locating all the optima of complex functions is a very difficult task and it is very hard for the algorithms to achieve nonzero SR values, Table 7 only shows the results on test functions $F1$–$F6$. The best results are highlighted in bold. It can be observed that LLNCDE has the fastest convergence speed on three out of the six test functions. To clearly demonstrate the search efficiency of the algorithms, convergence graphs on the most challenging test functions ($F15$–$F20$) are shown in Figure 11. The x-axis represents the number of consumed FEs, and the y-axis represents the

Table 3: Peak ratios and success rates achieved by the algorithms.

Function	r3pso-lhc		NCDE		LIPS		dADE/nrand/1		LoICDE		PNPCDE		IABC		OIF-BSO		LLNCDE	
	PR	SR	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	1.000	1.000
*F*2	0.996	0.980	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*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	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	1.000	1.000	0.995	0.980	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	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
*F*6	0.713+	0.000	0.146+	0.000	0.757+	0.000	1.000	1.000	0.482+	0.000	0.343+	0.000	0.947+	0.480	0.671+	0.000	0.993	0.800
*F*7	0.366+	0.000	0.688−	0.000	0.489+	0.000	0.496+	0.000	0.728−	0.000	0.604	0.000	0.724−	0.000	0.558+	0.000	0.601	0.000
*F*8	0.112+	0.000	0.504−	0.000	0.200+	0.000	0.449	0.020	0.546−	0.000	0.074+	0.000	0.619−	0.000	0.357	0.000	0.394	0.000
*F*9	0.096+	0.000	0.267−	0.000	0.124+	0.000	0.153+	0.000	0.282−	0.000	0.256−	0.000	0.369−	0.000	0.155+	0.000	0.243	0.000
*F*10	0.933+	0.380	1.000	1.000	0.992	0.900	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	0.995	0.920
*F*11	0.700	0.000	0.797−	0.180	0.863−	0.240	0.667+	0.000	0.653+	0.000	0.827−	0.120	0.663+	0.000	0.667+	0.000	0.717	0.000
*F*12	0.648+	0.000	0.330+	0.000	0.823	0.140	0.900−	0.420	0.070+	0.000	0.525+	0.000	0.135+	0.000	0.733+	0.000	0.810	0.080
*F*13	0.643	0.000	0.593+	0.000	0.690	0.000	0.670	0.000	0.513+	0.000	0.667	0.000	0.613+	0.000	0.667	0.000	0.660	0.000
*F*14	0.563+	0.000	0.637+	0.000	0.627	0.000	0.667	0.000	0.657	0.000	0.663	0.000	0.667	0.000	0.730	0.000	0.667	0.000
*F*15	0.213+	0.000	0.265+	0.000	0.418+	0.000	0.493+	0.000	0.275+	0.000	0.270+	0.000	0.363+	0.000	0.740+	0.000	0.650	0.000
*F*16	0.030+	0.000	0.637	0.000	0.203+	0.000	0.623+	0.000	0.650	0.000	0.600+	0.000	0.663	0.000	0.667	0.000	0.663	0.000
*F*17	0.028+	0.000	0.248+	0.000	0.208+	0.000	0.300+	0.000	0.250+	0.000	0.225+	0.000	0.153+	0.000	0.535+	0.000	0.413	0.000
*F*18	0.000+	0.000	0.290+	0.000	0.060+	0.000	0.453	0.000	0.300+	0.000	0.190+	0.000	0.400+	0.000	0.623+	0.000	0.497	0.000
*F*19	0.000+	0.000	0.095+	0.000	0.008+	0.000	0.048+	0.000	0.090+	0.000	0.125+	0.000	0.100+	0.000	0.415	0.000	0.280	0.000
*F*20	0.000+	0.000	0.245	0.000	0.000+	0.000	0.088+	0.000	0.205+	0.000	0.125+	0.000	0.060+	0.000	0.225+	0.000	0.250	0.000
B/E/W	13/7/0		8/8/4		10/9/1		8/11/1		9/8/3		9/9/2		9/8/3		9/11/0		NA	

The notation "+" represents that the PR value achieved by LLNCDE is significantly better than that of the corresponding algorithm, while the notation "–" denotes the opposite. The differences are detected using the Wilcoxon rank-sum test with the Bonferroni correction at significance level $\alpha = 0.05$. The last row of the table summarizes the B/E/W counts of LLNCDE against its competitor.

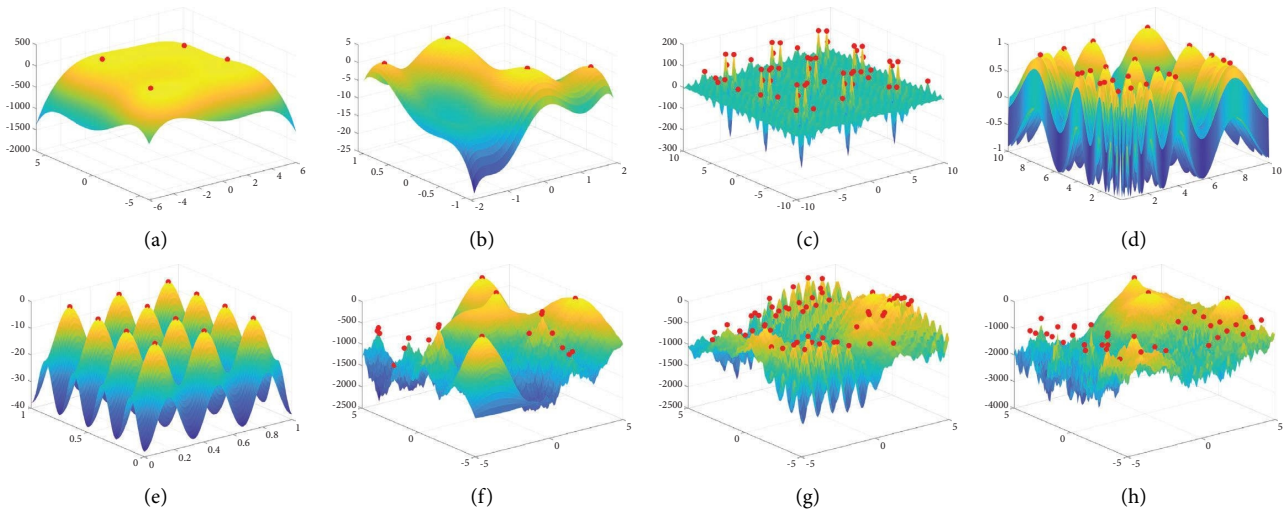

Figure 9: Population distribution in the final iteration of LLNCDE on two-dimensional test functions. (a) *F*4. (b) *F*5. (c) *F*6. (d) *F*7. (e) *F*10. (f) *F*11. (g) *F*12. (h) *F*13.

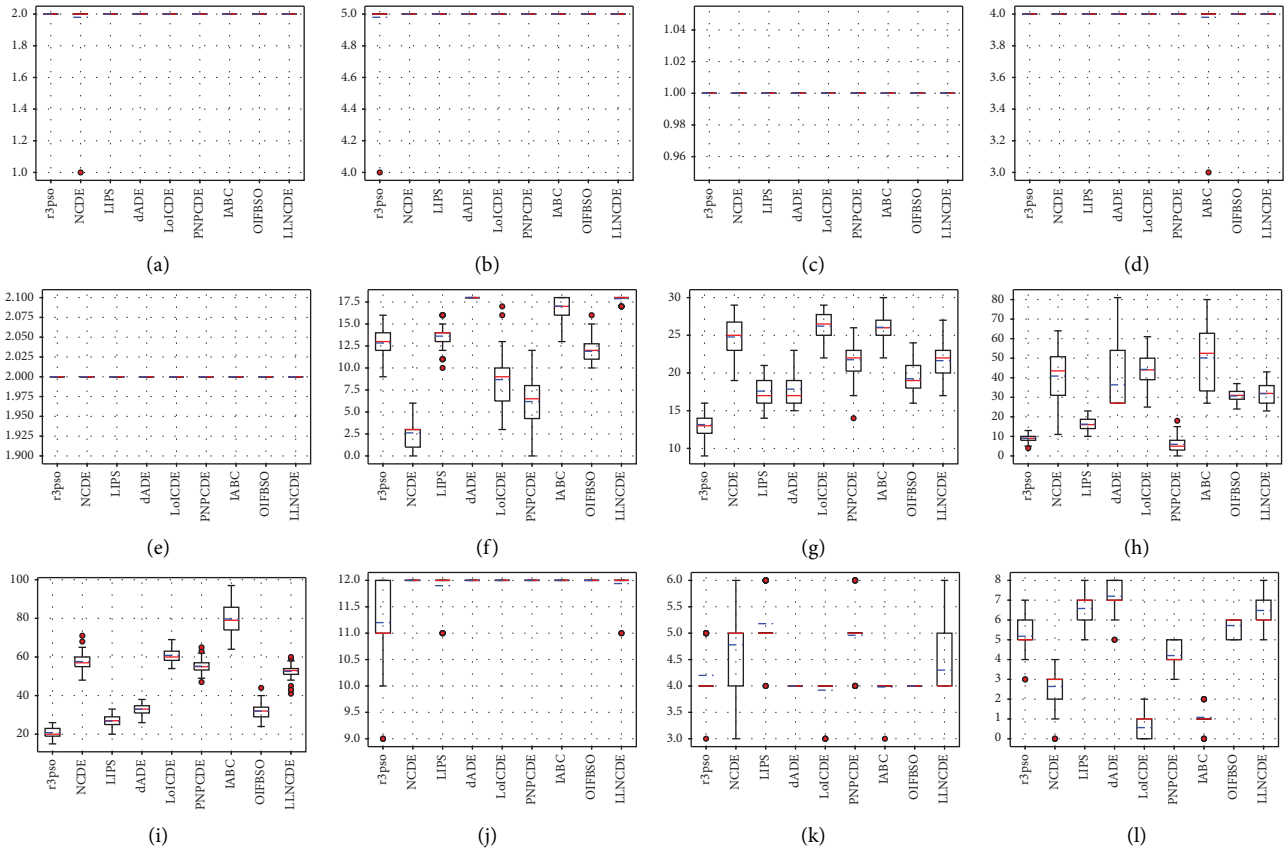

Figure 10: Continued.

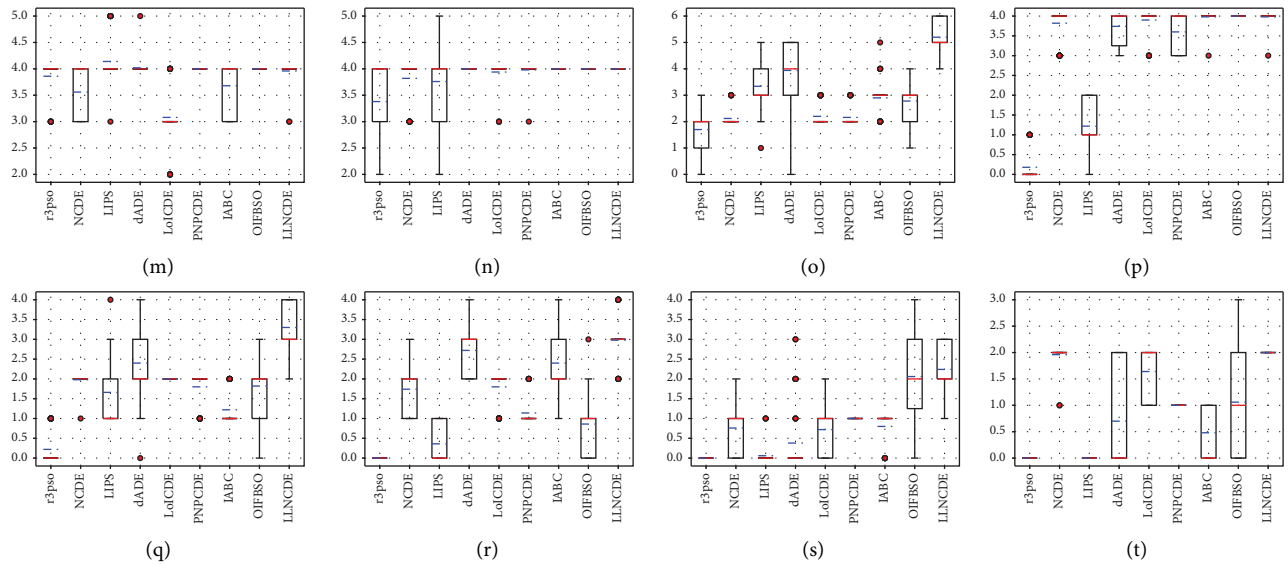

Figure 10: Box plots of the number of peaks located by the algorithms over 50 independent runs: (a) $F1$, (b) $F2$, (c) $F3$, (d) $F4$, (e) $F5$, (f) $F6$, (g) $F7$, (h) $F8$, (i) $F9$, (j) $F10$, (k) $F11$, (l) $F12$, (m) $F13$, (n) $F14$, (o) $F15$, (p) $F16$, (q) $F17$, (r) $F18$, (s) $F19$, and (t) $F20$.

Table 4: Rankings of the algorithms by Friedman's test.

Algorithm	Ranking
LLNCDE	3.5
dADE/nrand/1	4.125
IABC	4.65
OIF-BSO	4.65
LoICDE	4.925
NCDE	4.975
PNPCDE	5.15
LIPS	5.675
r3pso-lhc	7.35

number of found peaks. The data used to plot the curves are averaged over 50 independent runs of the algorithms. It can be noticed from the figures that LLNCDE has relatively fast convergence speed. The curves representing LLNCDE become flat much later than the other algorithms. This is because LLNCDE makes full use of computing resources (i.e., fitness evaluations) by continuously assigning exploration and exploitation tasks to different individuals. Hence, with the assistance of the multimodal level-based learning strategy, LLNCDE is capable of finding more peaks with relatively small number of fitness evaluations.

4.4. Effect of the Multimodal Level-Based Learning Strategy. In this section, we investigate the effect of the multimodal level-based learning strategy. In the literature, it is a common practice to enhance the performance of DE by employing multiple distinct mutation operators [72, 73]. To examine whether the improvement comes from the use of multiple mutation operators or from the multimodal level-based learning strategy, we compare LLNCDE with its variant that employs the mutation operators randomly. When generating new candidate solutions, we randomly select one of the three mutation operators specified in formulas (3)–(7). The variant is denoted as LLNCDE-R. Moreover, three simplified variants of LLNCDE that only employ one of the three mutation operators are also

Table 5: Peak ratios and success rates achieved by CDE, NCDE, and LLNCDE.

Function	CDE		NCDE		LLNCDE	
	PR	SR	PR	SR	PR	SR
$F1$	1.000	1.000	1.000	1.000	1.000	1.000
$F2$	1.000	1.000	1.000	1.000	1.000	1.000
$F3$	1.000	1.000	1.000	1.000	1.000	1.000
$F4$	0.985	0.980	1.000	1.000	1.000	1.000
$F5$	1.000	1.000	1.000	1.000	1.000	1.000
$F6$	0.068+	0.480	0.146+	0.000	0.993	0.800
$F7$	0.523+	0.000	0.688−	0.000	0.601	0.000
$F8$	0.029+	0.000	0.504−	0.000	0.394	0.000
$F9$	0.220+	0.000	0.267−	0.000	0.243	0.000
$F10$	1.000	1.000	1.000	1.000	0.995	0.920
$F11$	0.667+	0.000	0.797−	0.000	0.717	0.000
$F12$	0.000+	0.000	0.330+	0.000	0.810	0.080
$F13$	0.667	0.000	0.593+	0.000	0.660	0.000
$F14$	0.667	0.000	0.637+	0.000	0.667	0.000
$F15$	0.503+	0.000	0.265+	0.000	0.650	0.000
$F16$	0.667	0.000	0.637+	0.000	0.663	0.000
$F17$	0.000+	0.000	0.248+	0.000	0.413	0.000
$F18$	0.180+	0.000	0.290+	0.000	0.497	0.000
$F19$	0.000+	0.000	0.095+	0.000	0.280	0.000
$F20$	0.000+	0.000	0.245	0.000	0.250	0.000
B/E/W	11/9/0		8/8/4		NA	

The notation "+" represents that the PR value achieved by LLNCDE is significantly better than that of the competitor, while the notation "−" denotes the opposite. The differences are detected using the Wilcoxon rank-sum test with the Bonferroni correction at significance level $\alpha = 0.05$. The last row of the table summarizes the B/E/W counts of LLNCDE against its competitor.

Table 6: Rankings of CDE, NCDE, and LLNCDE by Friedman's test.

Algorithm	Ranking
LLNCDE	1.65
NCDE	1.95
CDE	2.4

Table 7: Convergence speed of the algorithms on test functions $F1$–$F6$.

Algorithm	Func	$F1$	$F2$	$F3$	$F4$	$F5$	$F6$
dADE/nrand/1	Avg	214.52	8742.22	1760.94	28555.98	6632.52	166917.48
	Std	4.14	3151.03	951.60	10310.97	2125.69	50648.81
LIPS	Avg	**200.00**	**1618.00**	1212.00	9316.00	3664.00	200000.00
	Std	**0.00**	**520.27**	702.75	1250.34	749.34	0.00
LoICDE	Avg	240.00	2670.00	1082.00	20608.00	5068.00	200000.00
	Std	74.83	1365.03	807.39	3855.07	1268.77	0.00
NCDE	Avg	350.00	3028.00	2268.00	11232.00	4088.00	200000.00
	Std	222.93	1423.94	2561.21	3012.34	771.40	0.00
PNPCDE	Avg	**200.00**	2134.00	**1066.00**	23938.00	4916.00	200000.00
	Std	**0.00**	603.19	**773.72**	5648.85	1304.20	0.00
r3pso-lhc	Avg	**200.00**	1896.00	1020.00	6806.00	2650.00	196454.00
	Std	**0.00**	564.26	495.18	6214.45	344.24	24822.00
IABC	Avg	209.50	2677.54	1789.92	31556.04	5019.36	179195.64
	Std	37.61	1262.27	1550.16	5699.34	1654.65	28085.09
OIF-BSO	Avg	468.68	4009.06	3466.80	34238.40	13490.58	200000.00
	Std	194.04	2203.60	3901.36	4980.45	4101.65	0.00
LLNCDE	Avg	554.00	2428.00	1710.00	**5918.00**	**3394.00**	**120022.00**
	Std	282.99	584.14	1145.99	**912.95**	**598.80**	**52532.13**

included in the comparison. The simplified variants are denoted as LLNCDE-O1, LLNCDE-O2, and LLNCDE-O3, respectively. The experimental results are listed in Table 8. The rankings of the algorithms obtained by Friedman's test are tabulated in Table 9.

From the tables, it can be observed that both LLNCDE and LLNCDE-R perform better than the three simplified variants (LLNCDE-O1, LLNCDE-O2, and LLNCDE-O3) with only one mutation operator. This indicates that the use of multiple mutation operators helps handle the diverse search scenarios contained in multimodal landscape. On the other hand, the overall performance of LLNCDE is better than that of LLNCDE-R. LLNCDE surpasses LLNCDE-R on eight out of 20 test problems. This observation reveals that within each subpopulation, identifying which level the individuals belong to provides useful information on the selection of suitable update operators. By associating individuals in different levels with different mutation operators, the search efficiency of the algorithm can be significantly improved.

4.5. Effect of the Parameter Setting. In this section, we conduct experiments to investigate the effect of the parameter settings. Specifically, we test LLNCDE with a fixed value of p ranging from -1 to -6. For the sake of clarity, the variant with setting $p = -k$ is denoted as LLNCDE-pnk. The PR and SR values achieved by LLNCDE and its variants are summarized in Table 10. Table 11 presents the rankings of the variants obtained by Friedman's test.

According to the rankings of the algorithms, LLNCDE with dynamically decreasing σ is able to achieve the overall best performance in terms of PR values. In addition, it can be observed that LLNCDE is able to outperform its variants in most of the test problems. These results indicate that gradually decreasing the size of perturbations better fits the task of guiding individuals toward their nearby optimum.

4.6. Finding Multiple Roots of Nonlinear Equation Systems. In this section, we apply the multimodal evolutionary algorithms to solve nonlinear equation systems (NESs). An NES can be formulated as follows:

$$\begin{cases} e_1(\mathbf{X}) = 0, \\ e_2(\mathbf{X}) = 0, \\ \vdots, \\ e_M(\mathbf{X}) = 0, \end{cases} \tag{12}$$

where $\mathbf{X} = [x_1, x_2, \ldots, x_D]$ is the decision vector. M represents the number of equations. At least one of the equations is nonlinear. A decision vector $\mathbf{X}$ is called a root if it satisfies the condition that for any $i \in \{1, 2, \ldots, M\}$, $e_i(\mathbf{X}) = 0$. An NES generally has multiple roots, especially in cases that $D > M$. Our goal is to find all the roots of the system. Four nonlinear equation systems extracted from real-world problems, i.e., the multiple steady states problem [74], the molecular conformation problem [75], the robot kinematic problem [76], and the interval arithmetic problem [77], are used to

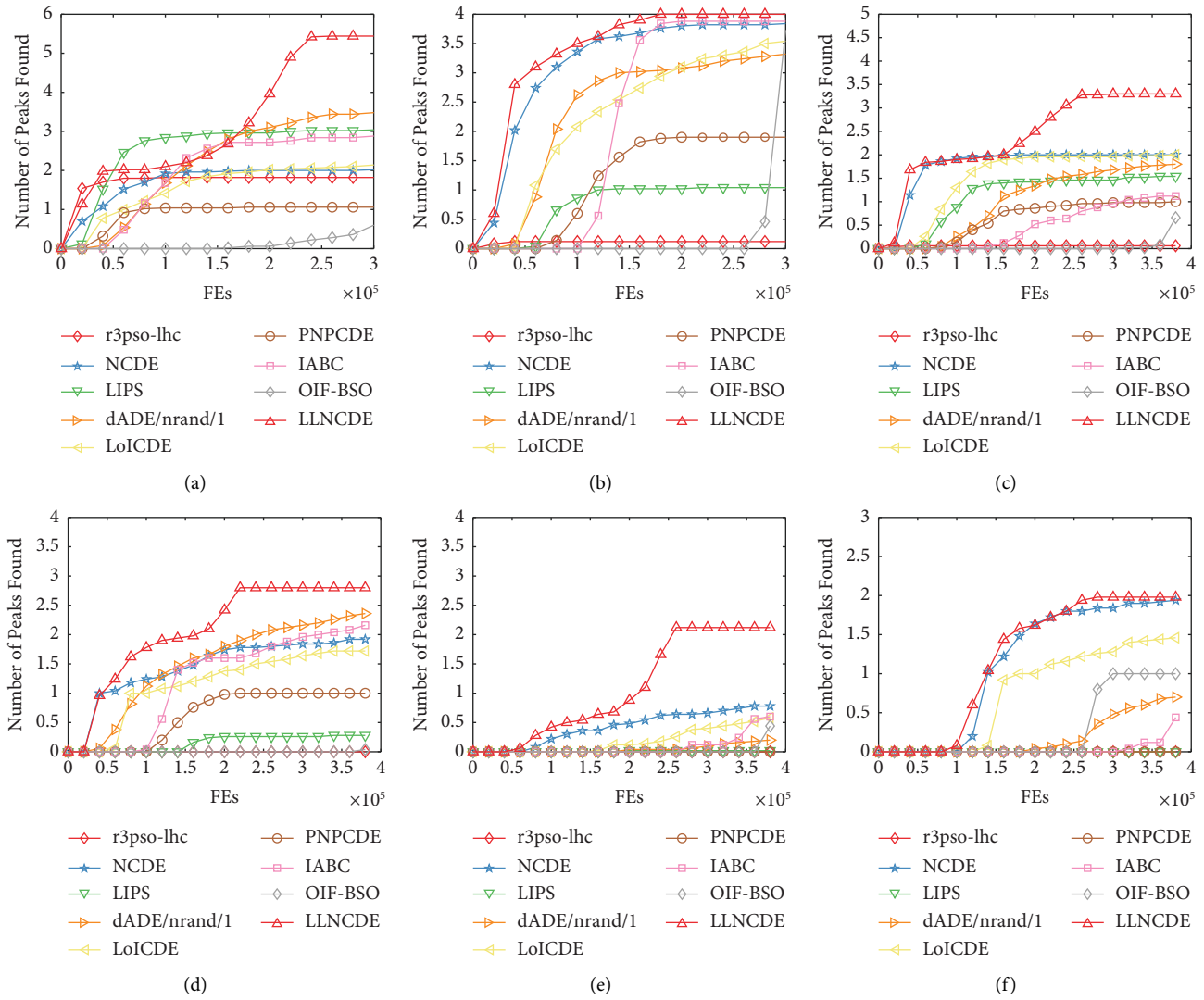

Figure 11: Convergence graphs of the algorithms on complex test functions $F15$–$F20$. The x-axis represents the number of consumed FEs, and the y-axis represents the number of found optima. (a) $F15$. (b) $F16$. (c) $F17$. (d) $F18$. (e) $F19$. (f) $F20$.

examine the effectiveness of the proposed level-based learning strategy. We compare LLNCDE with state-of-the-art algorithms specifically designed for solving NES, namely, repulsion-based adaptive DE (RADE) [78], dynamic repulsion-based JADE (DR-JADE) [79], multiobjective NES (MONES) [80], and adaptive multiobjective DE with weighted biobjective transformation technique (A-WeB) [81]. RADE combines the strengths of the repulsion technique, diversity preservation technique, and adaptive parameter control technique to tackle the challenge posed by NES problems. DR-JADE incorporates a dynamic repulsion technique that adjusts the repulsion radius in the evolutionary process. Moreover, the population diversity is preserved through population reinitialization. MONES transforms the problem of root finding into a biobjective optimization problem. All roots of the original NES are Pareto optimal solutions of the transformed problem. In this way, we can solve NES with any multiobjective evolutionary algorithm. A-WeB transforms an NES to a weighted biobjective optimization problem. Not only the roots of the original NES but also their images are mapped to the Pareto front of the transformed problem. An adaptive multi-objective DE is then adopted to solve the transformed problems.

Table 12 provides the basic information of the real-world nonlinear equation systems, including the number of linear and nonlinear equations (denoted by LE and NE), as well as the number of decision variables (D). NOR represents the number of roots. The maximum number of fitness evaluations for each problem is set according to the problem complexity. For ease of description, the four problems are denoted as $P1$, $P2$, $P3$, and $P4$ respectively. We continue to use PR and SR to evaluate the performance of the algorithms. A root is deemed to be found if there exists an individual whose Euclidean distance to the root is smaller than $1E-04$. Table 13 lists the experimental results of the algorithms. The best PR and SR values are marked in bold. The numerical results of the compared algorithms are taken from the

Table 8: Peak ratios and success rates achieved by LLNCDE and its variants.

Function	LLNCDE-R		LLNCDE-O1		LLNCDE-O2		LLNCDE-O3		LLNCDE	
	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	0.980	0.980	1.000	1.000	1.000	1.000
*F*4	1.000	1.000	1.000	1.000	0.855+	0.500	1.000	1.000	1.000	1.000
*F*5	1.000	1.000	1.000	1.000	0.950	0.900	1.000	1.000	1.000	1.000
*F*6	0.789+	0.000	0.208+	0.000	0.000+	0.000	0.104+	0.000	0.993	0.800
*F*7	0.603	0.000	0.615	0.000	0.092+	0.000	0.566	0.000	0.601	0.000
*F*8	0.380	0.000	0.006+	0.000	0.000+	0.000	0.293+	0.000	0.394	0.000
*F*9	0.218+	0.000	0.227+	0.000	0.001+	0.000	0.221+	0.000	0.243	0.000
*F*10	1.000	1.000	1.000	1.000	0.527+	0.000	0.978	0.780	0.995	0.920
*F*11	0.710	0.020	0.667+	0.000	0.530+	0.000	0.823−	0.260	0.717	0.000
*F*12	0.648+	0.000	0.365+	0.000	0.020+	0.000	0.313+	0.000	0.810	0.080
*F*13	0.677	0.000	0.647	0.000	0.443+	0.000	0.697−	0.000	0.660	0.000
*F*14	0.667	0.000	0.450+	0.000	0.000+	0.000	0.663	0.000	0.667	0.000
*F*15	0.475+	0.000	0.325+	0.000	0.003+	0.000	0.303+	0.000	0.650	0.000
*F*16	0.637	0.000	0.303+	0.000	0.000+	0.000	0.630+	0.000	0.663	0.000
*F*17	0.348+	0.000	0.100+	0.000	0.000+	0.000	0.238+	0.000	0.413	0.000
*F*18	0.380+	0.000	0.010+	0.000	0.000+	0.000	0.340+	0.000	0.497	0.000
*F*19	0.140+	0.000	0.000+	0.000	0.000+	0.000	0.185+	0.000	0.280	0.000
*F*20	0.003+	0.000	0.000+	0.000	0.000+	0.000	0.223+	0.000	0.250	0.000
B/E/W	8/12/0		12/8/0		16/4/0		10/8/2		NA	

The notation "+" represents that the PR value achieved by LLNCDE is significantly better than that of the variant, while the notation "−" denotes the opposite. The differences are detected using the Wilcoxon rank-sum test with the Bonferroni correction at significance level $\alpha = 0.05$. The last row of the table summarizes the B/E/W counts of LLNCDE against its variants.

Table 9: Rankings of LLNCDE and its variants by Friedman's test.

Algorithm	Ranking
LLNCDE	1.8
LLNCDE-R	2.375
LLNCDE-O3	2.875
LLNCDE-O1	3.2
LLNCDE-O2	4.75

Table 10: Peak ratios and success rates achieved by LLNCDE and its variants with fixed values of p.

Function	LLNCDE-*pn*1		LLNCDE-*pn*2		LLNCDE-*pn*3		LLNCDE-*pn*4		LLNCDE-*pn*5		LLNCDE-*pn*6		LLNCDE	
	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	0.990	0.980	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	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
*F*4	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
*F*5	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
*F*6	0.021+	0.000	0.450+	0.000	0.984	0.720	0.964+	0.540	0.338+	0.000	0.024+	0.000	0.993	0.800
*F*7	0.639−	0.000	0.612	0.000	0.602	0.000	0.612	0.000	0.578	0.000	0.583	0.000	0.601	0.000
*F*8	0.002+	0.000	0.015+	0.000	0.184+	0.000	0.402	0.000	0.093+	0.000	0.003+	0.000	0.394	0.000
*F*9	0.246	0.000	0.244	0.000	0.249	0.000	0.247	0.000	0.236+	0.000	0.237	0.000	0.243	0.000
*F*10	0.998	0.980	0.995	0.940	0.998	0.980	0.995	0.940	0.998	0.980	1.000	1.000	0.995	0.920
*F*11	0.713	0.000	0.707	0.020	0.723	0.000	0.707	0.040	0.713	0.020	0.730	0.060	0.717	0.000
*F*12	0.473+	0.000	0.553+	0.000	0.813	0.060	0.795	0.080	0.638+	0.000	0.458+	0.000	0.810	0.080
*F*13	0.643	0.000	0.647	0.000	0.670	0.000	0.667	0.000	0.620+	0.000	0.623+	0.000	0.660	0.000
*F*14	0.653	0.000	0.667	0.000	0.667	0.000	0.667	0.000	0.667	0.000	0.663	0.000	0.667	0.000
*F*15	0.295+	0.000	0.308+	0.000	0.505+	0.000	0.625	0.000	0.418+	0.000	0.318+	0.000	0.650	0.000
*F*16	0.607+	0.000	0.643	0.000	0.660	0.000	0.657	0.000	0.643	0.000	0.620+	0.000	0.663	0.000
*F*17	0.245+	0.000	0.238+	0.000	0.280+	0.000	0.305+	0.000	0.250+	0.000	0.223+	0.000	0.413	0.000
*F*18	0.367+	0.000	0.353+	0.000	0.407+	0.000	0.407+	0.000	0.380+	0.000	0.340+	0.000	0.497	0.000
*F*19	0.100+	0.000	0.090+	0.000	0.128+	0.000	0.240+	0.000	0.188+	0.000	0.128+	0.000	0.280	0.000
*F*20	0.245	0.000	0.230+	0.000	0.248	0.000	0.245	0.000	0.248	0.000	0.233+	0.000	0.250	0.000
B/E/W	8/11/1		8/12/0		5/15/0		4/16/0		9/11/0		10/10/0		NA	

The notation "+" represents that the PR value achieved by LLNCDE is significantly better than that of the variant, while the notation "−" denotes the opposite. The differences are detected using the Wilcoxon rank-sum test with the Bonferroni correction at significance level $\alpha = 0.05$. The last row of the table summarizes the B/E/W counts of LLNCDE against its variants.

Table 11: Rankings of LLNCDE and its variants with fixed values of p by Friedman's test.

Algorithm	Ranking
LLNCDE	2.775
LLNCDE-*pn*3	2.85
LLNCDE-*pn*4	3.225
LLNCDE-*pn*5	4.3
LLNCDE-*pn*2	4.8
LLNCDE-*pn*1	4.875
LLNCDE-*pn*6	5.175

Table 12: Nonlinear equation systems extracted from real-world scenarios.

Problem	D	LE	NE	NOR	MaxFEs
Multiple steady states problem	2	0	2	7	50,000
Molecular conformation	3	0	3	16	500,000
Robot kinematic problem	8	1	7	16	100,000
Interval arithmetic problem	10	0	10	1	50,000

Table 13: Peak ratios and success rates achieved by LLNCDE and state-of-the-art root finding algorithms when solving real-world NESs.

Problems	RADE		MONES		DR-JADE		A-WeB		LLNCDE	
	PR	SR	PR	SR	PR	SR	PR	SR	PR	SR
*P*1	0.997	0.980	0.439	0.000	**1.000**	**1.000**	0.837	0.120	0.997	0.980
*P*2	0.562	0.000	0.144	0.000	NA	NA	NA	NA	**0.740**	**0.000**
*P*3	0.944	0.430	0.166	0.000	0.840	0.033	0.669	0.000	**0.963**	**0.620**
*P*4	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**

The best PR and SR values are marked in bold.

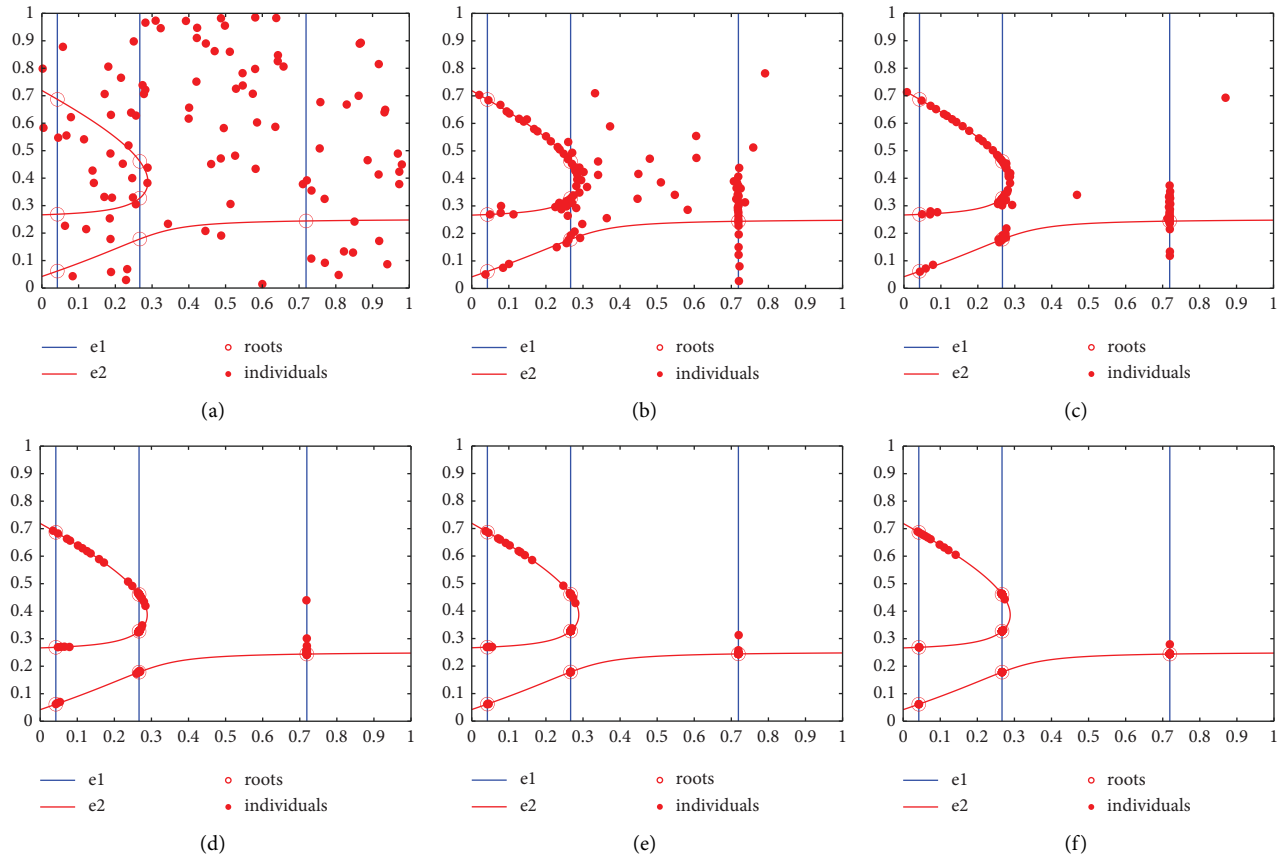

Figure 12: Population distribution in different iterations of LLNCDE when solving the multiple steady states problem. (a) 1st iteration. (b) 50th iteration. (c) 100th iteration. (d) 200th iteration. (e) 300th iteration. (f) 500th iteration.

literature [78, 79]. From the table, it can be observed that LLNCDE achieves the best results on three out of the four problems (i.e., $P2$, $P3$, and $P4$). According to the PR values, LLNCDE can find all the roots for $P1$, $P3$, and $P4$ in most of its runs. Compared with $P1$, $P3$, and $P4$, $P2$ is relatively hard to solve. LLNCDE is able to find more than 70% of the roots on average and compares favorably with state-of-the-art algorithms. This indicates that LLNCDE is suitable for the task of finding multiple roots of NES.

To show the search process of LLNCDE, Figure 12 depicts the population distribution in different iterations when solving $P1$ (i.e., the multiple steady states problem). The red circles and red dots represent the roots and individuals, respectively. From the figure, it can be seen that the individuals are divided into different subpopulations. Individuals in each subpopulation gradually gather around a root in the search space. In the final stage, all the roots are successfully located by LLNCDE.

5. Conclusions and Discussion

In this paper, we developed a multimodal level-based learning strategy to enhance the performance of niching algorithm. Taking the nature of multimodal optimization into consideration, the learning strategy is applied in the subpopulation level. After dividing the whole population into a bunch of subpopulations, individuals in each subpopulation are sorted and grouped into three levels. Then, genetic operators with different exploration and exploitation tendencies are assigned to individuals according to their levels. In this way, a better balance between exploration and exploitation is achieved. The proposed strategy is embedded into a popular multimodal differential evolution algorithm. Comprehensive experiments are carried out to investigate the effect of the proposed learning strategy. The experimental results demonstrate that the strategy is able to improve the performance of existing algorithms with respect to peak ratio (PR) and success rate (SR). It can increase the convergence speed, and therefore more peaks can be located with a relatively small number of fitness evaluations. Furthermore, we test the integrated algorithm by applying it to several root finding problems extracted from real-world scenarios. According to the experimental results, compared with several state-of-the-art root finding algorithms, more roots can be located by the proposed algorithm.

The advantage of the multimodal level-based learning strategy is that it is able to identify the role of individuals in the subpopulation. Different mutation operators are tailored for individuals in different levels of the subpopulation. By treating individuals differently based on their levels, the search efficiency of the subpopulations can be increased. Although the proposed approach is able to enhance the performance of niching algorithms, it has its limitations. The subpopulation is only divided into three levels, which is a coarse division of the subpopulation. Moreover, the learning strategy for each level of individuals is designed manually based on empirical studies and is fixed during the search process. It cannot fully exploit the potential of candidate solutions according to their evolutionary states. A future research direction toward extending the study is to divide the subpopulations in a fine-grained manner and design adaptive learning strategy using historical search information to further enhance the search efficiency. Besides, it is necessary to apply the proposed algorithm to different types of real-world problems to evaluate the practical value of the multimodal level-based learning strategy.

Another issue associated with the proposed algorithm is that the guiding mechanism, convergence equations, and update mechanism are inherited from the canonical DE algorithm. They are not completely applicable to multimodal optimization problems. LLNCDE selects candidate solutions from the population in an ordinal manner for reproduction. Moreover, the fitness value-based update mechanism is adopted to refresh the population without considering the individuals' contribution to the population and the mating pool. Therefore, it is a promising avenue to further improve the algorithm performance by incorporating dynamic or adaptive selection methods based on fitness-distance balance [54–56], as well as update mechanisms based on novel natural survivor methods that assign appropriate scores to different individuals [69].

Data Availability

The data presented in this study are available on request from the corresponding author.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was funded in part by the National Natural Science Foundation of China under grant nos. 62106046 and 62106055, in part by the Guangdong Basic and Applied Basic Research Foundation under grant nos. 2019A1515110474 and 2022A1515011825, in part by the Ministry of Science and Technology of China under grant no. 2018AAA0101301, in part by the Special Project in Key Fields of Guangdong Colleges and Universities under grant no. 2021ZDZX3007, in part by the Key Research Platforms and Projects of High School in Guangdong Province under grant nos. 2023ZDZX1028 and 2023ZDZX1050, in part by the Dongguan Social Development Science and Technology Project under grant no. 20211800904722, in part by the Dongguan Science and Technology Special Commissioner Project under grant no. 2021180050007, and in part by the Guangzhou Science and Technology Planning Project under grant nos. 2023A04J0388 and 2023A03J0662.

References

[1] Y. Hu and K. Zhang, "Multimodal optimization evolutionary algorithm for RNA secondary structure prediction," in *The Fifth International Conference on Biological Information and*

Biomedical Engineering. BIBE2021, Association for Computing Machinery, Hangzhou, China, 2021.

[2] T. Osa, "Multimodal trajectory optimization for motion planning," *The International Journal of Robotics Research*, vol. 39, no. 8, pp. 983–1001, 2020.

[3] L. Pröhl and H. Aschemann, "Multimodal optimization for the trajectory planning of railway vehicles," in *Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC)*, Glasgow, UK, July 2020.

[4] T. R. Farshi, J. H. Drake, and E. Özcan, "A multimodal particle swarm optimization-based approach for image segmentation," *Expert Systems with Applications*, vol. 149, Article ID 113233, 2020.

[5] J. Liang, B. Qu, B. Li, K. Yu, and C. Yue, "Locating multiple roots of nonlinear equation systems via multi-strategy optimization algorithm with sequence quadratic program," *Science China Information Sciences*, vol. 65, no. 7, Article ID 179102, 2022.

[6] J. Dong, W. Gong, F. Ming, and L. Wang, "A two-stage evolutionary algorithm based on three indicators for constrained multi-objective optimization," *Expert Systems with Applications*, vol. 195, Article ID 116499, 2022.

[7] H. Jiang, M. Lu, Y. Tian, J. Qiu, and X. Zhang, "An evolutionary algorithm for solving capacitated vehicle routing problems by using local information," *Applied Soft Computing*, vol. 117, Article ID 108431, 2022.

[8] J. Jabari Lotf, M. Abdollahi Azgomi, and M. R. Ebrahimi Dishabi, "An improved influence maximization method for social networks based on genetic algorithm," *Physica A: Statistical Mechanics and its Applications*, vol. 586, Article ID 126480, 2022.

[9] A. Maskooki, K. Deb, and M. Kallio, "A customized genetic algorithm for Bi-objective routing in a dynamic network," *European Journal of Operational Research*, vol. 297, no. 2, pp. 615–629, 2022.

[10] J. Pereira, J. Mendes, J. S. S. Júnior, C. Viegas, and J. R. Paulo, "A review of genetic algorithm approaches for wildfire spread prediction calibration," *Mathematics*, vol. 10, no. 3, p. 300, 2022.

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

[12] X. Li, M. G. Epitropakis, K. Deb, and A. Engelbrecht, "Seeking multiple solutions: an updated survey on niching methods and their applications," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 4, pp. 518–538, 2017.

[13] S. Biswas, S. Kundu, and S. Das, "Inducing niching behavior in differential evolution through local information sharing," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 2, pp. 246–263, 2015.

[14] R. Jiang, J. Zhang, Y. Tang, J. Feng, and C. Wang, "Self-adaptive DE algorithm without niching parameters for multimodal optimization problems," *Applied Intelligence*, vol. 52, no. 11, pp. 12888–12923, 2022.

[15] G. Sun, C. Li, and L. Deng, "An adaptive regeneration framework based on search space adjustment for differential evolution," *Neural Computing & Applications*, vol. 33, no. 15, pp. 9503–9519, 2021.

[16] R. P. Parouha and P. Verma, "A systematic overview of developments in differential evolution and particle swarm optimization with their advanced suggestion," *Applied Intelligence*, vol. 52, no. 9, pp. 10448–10492, 2022.

[17] L. Deng, C. Li, Y. Lan, G. Sun, and C. Shang, "Differential evolution with dynamic combination based mutation operator and two-level parameter adaptation strategy," *Expert Systems with Applications*, vol. 192, Article ID 116298, 2022.

[18] X. Zhao, S. Feng, J. Hao, X. Zuo, and Y. Zhang, "Neighborhood opposition-based differential evolution with Gaussian perturbation," *Soft Computing*, vol. 25, no. 1, pp. 27–46, 2021.

[19] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," *Proceedings of the 2004 Congress on Evolutionary Computation IEEE Cat. No.04TH8753*, vol. 2, p. 1382, 2004.

[20] D. E. Goldberg and J. Richardson, "Genetic algorithms with sharing for multimodal function optimization," in *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, Lawrence Erlbaum, Hillsdale, NJ, USA, 1987.

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

[22] A. Pétrowski, "A clearing procedure as a niching method for genetic algorithms," in *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp. 798–803, IEEE, Nagoya, Japan, May 1996.

[23] B.-Y. Qu, P. N. Suganthan, and J.-J. Liang, "Differential evolution with neighborhood mutation for multimodal optimization," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 5, pp. 601–614, 2012.

[24] M. G. Epitropakis, V. P. Plagianakos, and M. N. Vrahatis, "Finding multiple global optima exploiting differential evolution's niching capability," in *Proceedings of the 2011 IEEE Symposium on Differential Evolution (SDE)*, IEEE, Paris, France, April 2011.

[25] M. G. Epitropakis, X. Li, and E. K. Burke, "A dynamic archive niching differential evolution algorithm for multimodal optimization," in *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, IEEE, Cancun, Mexico, June 2013.

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

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

[28] Q. Liu, S. Du, B. J. van Wyk, and Y. Sun, "Double-layer-clustering differential evolution multimodal optimization by speciation and self-adaptive strategies," *Information Sciences*, vol. 545, pp. 465–486, 2021.

[29] X. Lin, W. Luo, and P. Xu, "Differential evolution for multimodal optimization with species by nearest-better clustering," *IEEE Transactions on Cybernetics*, vol. 51, no. 2, pp. 970–983, 2021.

[30] W. Sheng, X. Wang, Z. Wang, Q. Li, and Y. Chen, "Adaptive memetic differential evolution with niching competition and supporting archive strategies for multimodal optimization," *Information Sciences*, vol. 573, pp. 316–331, 2021.

[31] Z.-J. Wang, Z.-H. Zhan, Y. Lin et al., "Automatic niching differential evolution with contour prediction approach for

multimodal optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 114–128, 2020.

[32] X. Li, "Niching without niching parameters: particle swarm optimization using a ring topology," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 150–169, 2009.

[33] B.-Y. Qu, P. N. Suganthan, and S. Das, "A distance-based locally informed particle swarm model for multimodal optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 3, pp. 387–402, 2013.

[34] Q. Liu, S. Du, B. J. Van Wyk, and Y. Sun, "Niching particle swarm optimization based on euclidean distance and hierarchical clustering for multimodal optimization," *Nonlinear Dynamics*, vol. 99, no. 3, pp. 2459–2477, 2020.

[35] T. Zheng, J. Liu, Y. Liu, and S. Tan, "Hybridizing multi-objective, clustering and particle swarm optimization for multimodal optimization," *Neural Computing & Applications*, vol. 34, no. 3, pp. 2247–2274, 2022.

[36] J. Zou, Q. Deng, J. Zheng, and S. Yang, "A close neighbor mobility method using particle swarm optimizer for solving multimodal optimization problems," *Information Sciences*, vol. 519, pp. 332–347, 2020.

[37] S. Ma, Y. Wang, and S. Zhang, "Improved artificial bee colony algorithm for multimodal optimization based on crowding method," *Journal of Organizational and End User Computing*, vol. 34, no. 3, pp. 1–18, 2022.

[38] Z. Dai, W. Fang, K. Tang, and Q. Li, "An optima-identified framework with brain storm optimization for multimodal optimization problems," *Swarm and Evolutionary Computation*, vol. 62, Article ID 100827, 2021.

[39] T. Huang, Y. J. Gong, W. N. Chen, H. Wang, and J. Zhang, "A probabilistic niching evolutionary computation framework based on binary space partitioning," *IEEE Transactions on Cybernetics*, vol. 52, no. 1, pp. 51–64, 2022.

[40] H. Zhao, Z. H. Zhan, Y. Lin et al., "Local binary pattern-based adaptive differential evolution for multimodal optimization problems," *IEEE Transactions on Cybernetics*, vol. 50, no. 7, pp. 3343–3357, 2020.

[41] Z. G. Chen, Z. H. Zhan, H. Wang, and J. Zhang, "Distributed individuals for multiple peaks: a novel differential evolution for multimodal optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 4, pp. 708–719, 2020.

[42] H. Lu, S. Sun, S. Cheng, and Y. Shi, "An adaptive niching method based on multi-strategy fusion for multimodal optimization," *Memetic Computing*, vol. 13, no. 3, pp. 341–357, 2021.

[43] A. Ahrari and K. Deb, "Multimodal optimization by evolution strategies with repelling subpopulations," in *Metaheuristics for Finding Multiple Solutions*, M. Preuss, M. G. Epitropakis, X. Li, and J. E. Fieldsend, Eds., Springer International Publishing, Berlin, Germany, 2021.

[44] R. Ahmed, A. Nazir, S. Mahadzir, M. Shorfuzzaman, and J. Islam, "Niching grey wolf optimizer for multimodal optimization problems," *Applied Sciences*, vol. 11, p. 4795, 2021.

[45] H. Xia, C. Li, S. Zeng, Q. Tan, J. Wang, and S. Yang, "A reinforcement-learning-based evolutionary algorithm using solution space clustering for multimodal optimization problems," in *Proceedings of the 2021 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1938–1945, Kraków, Poland, July 2021.

[46] A. Basak, S. Das, and K. C. Tan, "Multimodal optimization using a biobjective differential evolution algorithm enhanced with mean distance-based selection," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 5, pp. 666–685, 2013.

[47] Y. Wang, H. X. Li, G. G. Yen, and W. Song, "MOMMOP: multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems," *IEEE Transactions on Cybernetics*, vol. 45, no. 4, pp. 830–843, 2015.

[48] Q. Yang, W. N. Chen, J. D. Deng, Y. Li, T. Gu, and J. Zhang, "A level-based learning swarm optimizer for large-scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 4, pp. 578–594, 2018.

[49] J. Harty, "Differentiation in teaching and learning: principles and practice," *Reaching Out: Journal of Inclusive Education in Ireland*, vol. 16, p. 1, 2002.

[50] J. C. Richards and W. A. Renandya, *Methodology in Language Teaching: An Anthology of Current Practice*, Cambridge University Press, Cambridge, UK, 2002.

[51] R. Cheng and Y. Jin, "A competitive swarm optimizer for large scale optimization," *IEEE Transactions on Cybernetics*, vol. 45, no. 2, pp. 191–204, 2015.

[52] X. Li, A. Engelbrecht, and M. G. Epitropakis, *Benchmark Functions for CEC'2013 Special Session and Competition on Niching Methods for Multimodal Function Optimization*, RMIT University, Evolutionary Computation and Machine Learning Group, Melbourne, Australia, 2013.

[53] K. Deb, "Introduction to selection," *Evolutionary Computation*, vol. 1, pp. 166–171, 2000.

[54] H. T. Kahraman, S. Aras, and E. Gedikli, "Fitness-distance balance (FDB): a new selection method for meta-heuristic search algorithms," *Knowledge-Based Systems*, vol. 190, Article ID 105169, 2020.

[55] H. T. Kahraman, H. Bakir, S. Duman, M. Katı, S. Aras, and U. Guvenc, "Dynamic FDB selection method and its application: modeling and optimizing of directional overcurrent relays coordination," *Applied Intelligence*, vol. 52, no. 5, pp. 4873–4908, 2022.

[56] S. Duman, H. T. Kahraman, and M. Kati, "Economical operation of modern power grids incorporating uncertainties of renewable energy sources and load demand using the adaptive fitness-distance balance-based stochastic fractal search algorithm," *Engineering Applications of Artificial Intelligence*, vol. 117, Article ID 105501, 2023.

[57] Z. Beheshti and S. M. H. Shamsuddin, "A review of population-based meta-heuristic algorithms," *International Journal of Advances in Soft Computing and Its Applications*, vol. 5, no. 1, pp. 1–35, 2013.

[58] T. Dokeroglu, E. Sevinc, T. Kucukyilmaz, and A. Cosar, "A survey on new generation metaheuristic algorithms," *Computers & Industrial Engineering*, vol. 137, Article ID 106040, 2019.

[59] G. Pavai and T. Geetha, "A survey on crossover operators," *ACM Computing Surveys*, vol. 49, no. 4, pp. 1–43, 2016.

[60] S. M. Lim, A. B. M. Sultan, M. N. Sulaiman, A. Mustapha, and K. Y. Leong, "Crossover and mutation operators of genetic algorithms," *International Journal of Machine Learning and Computing*, vol. 7, no. 1, pp. 9–12, 2017.

[61] M. Pant, M. Pant, H. Zaheer, L. Garcia-Hernandez, and A. Abraham, "Differential evolution: a review of more than

two decades of research," *Engineering Applications of Artificial Intelligence*, vol. 90, Article ID 103479, 2020.

[62] J. Nayak, H. Swapnarekha, B. Naik, G. Dhiman, and S. Vimal, "25 Years of particle swarm optimization: flourishing voyage of two decades," *Archives of Computational Methods in Engineering*, vol. 30, no. 3, pp. 1663–1725, 2023.

[63] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[64] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga, "A comprehensive survey: artificial bee colony (ABC) algorithm and applications," *Artificial Intelligence Review*, vol. 42, no. 1, pp. 21–57, 2014.

[65] R. Biedrzycki, J. Arabas, and D. Jagodziński, "Bound constraints handling in differential evolution: an experimental study," *Swarm and Evolutionary Computation*, vol. 50, Article ID 100453, 2019.

[66] S. Helwig, J. Branke, and S. Mostaghim, "Experimental analysis of bound handling techniques in particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 2, pp. 259–271, 2013.

[67] F. Caraffini, A. V. Kononova, and D. Corne, "Infeasibility and structural bias in differential evolution," *Information Sciences*, vol. 496, pp. 161–179, 2019.

[68] A. V. Kononova, D. Vermetten, F. Caraffini, M.-A. Mitran, and D. Zaharie, "The importance of being constrained: dealing with infeasible solutions in differential evolution and beyond," 2023, https://arxiv.org/abs/2203.03512.

[69] H. T. Kahraman, M. Katı, S. Aras, and D. A. Taşci, "Development of the natural survivor method (NSM) for designing an updating mechanism in metaheuristic search algorithms," *Engineering Applications of Artificial Intelligence*, vol. 122, Article ID 106121, 2023.

[70] 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 and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.

[71] J. Alcalá-Fdez, L. Sanchez, S. Garcia et al., "KEEL: a software tool to assess evolutionary algorithms for data mining problems," *Soft Computing*, vol. 13, no. 3, pp. 307–318, 2009.

[72] Y. F. Ge, M. Orlowska, J. Cao, H. Wang, and Y. Zhang, "Knowledge transfer-based distributed differential evolution for dynamic database fragmentation," *Knowledge-Based Systems*, vol. 229, Article ID 107325, 2021.

[73] J. Y. Li, K. J. Du, Z. H. Zhan, H. Wang, and J. Zhang, "Distributed differential evolution with adaptive resource allocation," *IEEE Transactions on Cybernetics*, vol. 53, no. 5, pp. 2791–2804, 2023.

[74] C. A. Floudas, "Recent advances in global optimization for process synthesis, design and control: enclosure of all solutions," *Computers & Chemical Engineering*, vol. 23, pp. S963–S973, 1999.

[75] I. Z. Emiris and B. Mourrain, "Computer algebra methods for studying and computing molecular conformations," *Algorithmica*, vol. 25, no. 2-3, pp. 372–402, 1999.

[76] C. Wang, R. Luo, K. Wu, and B. Han, "A new filled function method for an unconstrained nonlinear equation," *Journal of Computational and Applied Mathematics*, vol. 235, no. 6, pp. 1689–1699, 2011.

[77] C. Grosan and A. Abraham, "A new approach for solving nonlinear equations systems," vol. 38, no. 3, pp. 698–714, 2008.

[78] W. Gong, Y. Wang, Z. Cai, and L. Wang, "Finding multiple roots of nonlinear equation systems via a repulsion-based adaptive differential evolution," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 4, pp. 1499–1513, 2020.

[79] Z. Liao, W. Gong, X. Yan, L. Wang, and C. Hu, "Solving nonlinear equations system with dynamic repulsion-based evolutionary algorithms," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 4, pp. 1590–1601, 2020.

[80] W. Song, Y. Wang, H. X. Li, and Z. Cai, "Locating multiple optimal solutions of nonlinear equation systems based on multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 3, pp. 414–431, 2014.

[81] W. Gong, Y. Wang, Z. Cai, and S. Yang, "A weighted biobjective transformation technique for locating multiple optimal solutions of nonlinear equation systems," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 5, pp. 697–713, 2017.