Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 6675996, 13 pages
https://doi.org/10.1155/2020/6675996

Research Article

Dynamic Neighborhood-Based Particle Swarm Optimization for Multimodal Problems

Xu-Tao Zhang,[1] Biao Xu ,[2,3] Wei Zhang,[1] Jun Zhang,[4] and Xin-fang Ji[5]

[1]*Department of Electrical Engineering, Jiangsu College of Safety Technology, Xuzhou 221100, China*
[2]*Department of Electronic Engineering, Shantou University, Shantou 515041, China*
[3]*Key Laboratory of Digital Signal and Image Processing of Guangdong Province, Shantou 515041, China*
[4]*Xuzhou Kuangyi Automation Technology Co., Ltd., Xuzhou 221100, China*
[5]*School of Information and Control Engineering, China University of Mining and Technology, Xuzhou 221100, China*

Correspondence should be addressed to Biao Xu; xubiao@stu.edu.cn

Received 19 November 2020; Revised 4 December 2020; Accepted 8 December 2020; Published 21 December 2020

Academic Editor: Bo Yang

Various black-box optimization problems in real world can be classified as multimodal optimization problems. Neighborhood information plays an important role in improving the performance of an evolutionary algorithm when dealing with such problems. In view of this, we propose a particle swarm optimization algorithm based on dynamic neighborhood to solve the multimodal optimization problem. In this paper, a dynamic ε-neighborhood selection mechanism is first defined to balance the exploration and exploitation of the algorithm. Then, based on the information provided by the neighborhoods, four different particle position updating strategies are designed to further support the algorithm's exploration and exploitation of the search space. Finally, the proposed algorithm is compared with 7 state-of-the-art multimodal algorithms on 8 benchmark instances. The experimental results reveal that the proposed algorithm is superior to the compared ones and is an effective method to tackle multimodal optimization problems.

1. Introduction

Various black-box problems to be tackled difficultly in the real world have the characteristic of multimodal problem [1]. Strictly speaking, a multimodal optimization problem (MMOP) refers to an optimization problem with multiple global or local optima. Typical instances include drug molecular design [2], truss structure optimization [3], and protein structure prediction [4]. When solving MMOPs, we expect to locate several optimal solutions simultaneously, for the following reasons [5, 6]: (1) finding multiple optimal solutions in different regions of the search space at the same time is conducive to maintaining the diversity of the population and offset the influence of genetic drift. (2) For many real-world engineering optimization problems, designers hope to freely choose solutions to meet different needs from several excellent solutions with great differences. (3) For the black-box problem, in the absence of prior knowledge, the positions and the numbers of the global optima of the problem cannot be obtained. Locating multiple optimal solutions of the problem at the same time can improve the possibility of finding its global optimal solution and can also possibly provide multiple optimal solutions for decision makers. In view of this, the field of multimodal optimization has received more and more attention recently due to its scientific and technological significance.

Traditional population-based evolutionary algorithms (EAs), such as genetic algorithm (GA) [7, 8], differential evolutionary (DE) [9], and particle swarm optimization (PSO) [10], are natural candidates for solving MMOPs. However, it should be noted that, in the case of solving MMOPs without special treatment, these EAs can only converge to one optimal solution of the optimization problem at a single run. In order to find multiple optimal solutions of the problem, it is necessary to run them repeatedly and it is expected to find a different optimal solution each time.

In order to enable EAs to solve MMOPs effectively, researchers have proposed a series of techniques, most of which are aimed at enhancing the diversity of population and making it converge towards different search directions. These techniques are commonly referred to as niching [5]. Representative niche techniques include crowding [11], clustering [12–14], speciation [15], and stretching and shrinking method [16]. Recently, niching techniques are also embedded into PSO algorithm in a number of literatures. For example, MMOP is solved by changing the topology of PSO, such as niche PSO based on ring topology [13] and PSO based on star topology and ring topology [17]. In addition, the concept of speciation has also been introduced into the species-based PSO [18–20], where species can form adaptively in different optimal states. However, in order to define a species, a niching radius must be provided in advance. Accordingly, species can be merged or separated into new species in each iteration. Other methods, such as *nbest* PSO [21, 22] and multiswarms [23, 24], are also proposed.

To remedy the defect of requirement for providing the niching radius in advance, a parameter, which is easy to set or with little sensitivity to the performance of the algorithm, is employed to complete the clustering or grouping of individuals by replacing the niching radius. Schoeman and Engelbrecht proposed a vector-based PSO algorithm, which uses vector operations to demarcate the boundaries of niches and maintain subswarms without any prior knowledge of the problem domain [25]. A niche is determined by a radius value based on the distance between the optimal of group and the nearest particle. Qu et al. [26] proposed a distance-based locally informed PSO (LIPS), where the global best position is replaced by multiple local best positions to guide the update of particles for converging to different optimal subspaces. However, LIPS needs to specify the neighborhood size of the particles. Based on the locality sensitive hashing, Zhang et al. proposed a fast niching technique to find the neighborhood set of particles, which can keep a balance between the exploration and exploitation of the algorithm while reducing the computational complexity of EAs [27]. Nevertheless, the method depends on the constructed hash functions. Further, Zhang et al. proposed a concept of parameter-free Voronoi neighborhood; the information provided by the Voronoi neighbors is used to estimate the evolutionary state of an individual. And different types of individuals are assigned with different reproduction strategies to support the exploration and exploitation of the search space [28]. As the dimension of the problem increases, the computational complexity of Voronoi will increase dramatically.

In order to handle the above problems, a dynamic neighborhood-based multimodal PSO algorithm (DNPSO) is proposed in this paper. The aim is to design different evolutionary strategies based on neighborhood information to balance the exploration and exploitation of the algorithm without specifying an exact neighborhood size. The main contributions of DNPSO are provided as follows:

(1) A dynamic ε-neighborhood selection mechanism is proposed. Compared with the existing neighborhood-based method, the neighborhood size of a particle in this paper is dynamically changed, and it is different for different particles. In addition, the proposed mechanism may lead to a larger possibility of information interaction between distant particles, which is beneficial for generating an exploratory offspring and restoring the exploration ability of PSO.

(2) Inspired by the successful application of neighborhood information to the design of the multimodal EAs, four different position updating strategies are presented according to the particle's position and performance in its neighborhoods. Consequently, the search performance of the algorithm can be improved.

The rest of this paper is arranged as follows. Section 2 gives the related work for PSO. The proposed DNPSO is presented in Section 3, including the dynamic ε-neighborhood selection mechanism, four different position updating strategies, and the framework of the proposed algorithm. In order to verify the effectiveness of the proposed DNPSO, comparison experiments are presented in Section 4. Section 5 highlights main findings and future research opportunities as a result of the survey.

2. Particle Swarm Optimization Algorithm

Consider the following maximization problem:

$$\begin{aligned} &\max f(X), \\ &s.t. \quad X \in S \subseteq R^D. \end{aligned} \tag{1}$$

where X is a D-dimensional decision variable, S refers to the bound-constraint of the decision space, R^D means the D-dimensional space, and $f(\cdot)$ is the objective function.

PSO is a heuristic search algorithm proposed by Kennedy et al. [29]. The searching principle of PSO comes from the imitation of bird foraging behavior. Due to its simple structure and efficient searching performance, PSO has been successfully applied to solving various optimization problems [30–32]. In PSO, the position of each particle in the search space represents the "potential feasible solution" of the problem to be optimized. The particles are initially random in the feasible search space with a random velocity, which aims to converge to the global optimal solution of the optimization problem. During the optimization process, each particle tracks two optimal positions simultaneously. One is the best position that it has achieved so far, also known as the individual guider (*Pbest*), and the other is the global best position detected so far in the neighborhood of the current particle or in the entire swarm, also known as the global guider (*Gbest*) [29]. Among them, the tracking of individual guider can be regarded as the component of self-cognition, and the learning of global optimal position can be seen as the component of social cognition. In the next iteration, both the ego and the social cognition components randomly influence the velocity of each particle.

Since the PSO was proposed, researchers have proposed numerous PSO variants. Two variants of PSO are presented here, i.e., PSO with inertia weight [29] and LIPS proposed by Qu et al. [26]. At each iteration, the velocity and position of a particle in the PSO with inertia weight are updated as follows:

$$\begin{aligned} v_{i,d}(t+1) &= wv_{i,d}(t) + c_1 r_1\left(Pbest_{i,d}(t) - x_{i,d}(t)\right) \\ &\quad + c_2 r_2\left(Gbest_d(t) - x_{i,d}(t)\right), \\ x_{i,d}(t+1) &= x_{i,d}(t) + v_{i,d}(t+1), \end{aligned} \tag{2}$$

where t is the number of iterations; $X_i(t) = (x_{i,1}(t), x_{i,2}(t), \ldots, x_{i,D}(t))$ and $V_i(t) = (v_{i,1}(t), v_{i,2}(t), \ldots, v_{i,D}(t))$ mean the position and velocity of the i-th particle at the t-th iteration, respectively. $Pbest_i(t)$ represents the best position found by the i-th particle and refers to the best position of the swarm. w is the inertia weight; $c1$ and $c2$ are cognitive and social coefficients, respectively. $r1$ and $r2$ are two random values within [0, 1]. Figure 1 shows the flow chart of a conventional PSO.

The velocity of a particle in LIPS is determined by the information provided by its neighbors, and the velocity of each particle is updated as follows:

$$v_{i,d}(t+1) = w\left(v_{i,d}(t) + \varphi\left(P_{i,d}(t) - x_{i,d}(t)\right)\right), \tag{3}$$

where

$$P_i(t) = \frac{\sum_{j=1}^{nsize}\left(\varphi_j nbest_j(t)\right)/nsize}{\varphi}, \tag{4}$$

where φ_j refers to a random value uniformly distributed in $[0, 4.1/nsize]$ [26], $\varphi = \sum_{j=1}^{nsize}\varphi_j$, $nbest_j$ is the j-th nearest neighborhood to $Pbest_i$, and $nsize$ means the neighborhood size.

3. Dynamic Neighborhood-Based PSO for Multimodal Problems

In this section, two key techniques are presented in the proposed DNPSO. One is the definition of dynamic ε-neighborhood; the other is the four tailored particle update strategies based on the neighborhood information of the current particle, and the framework of the proposed DNPSO is also provided.

3.1. Dynamic ε-Neighborhood. At present, a number of researchers try to solve the MMOPs by using distance-based neighborhood to form different species in the search space, and most of them show that neighborhood information is crucial to enhance the diversity of the population [26, 33–35]. However, with neighborhood information, the offspring produced by these methods tend to be attracted to the inner regions enclosed by the initial population. Therefore, the number of optimal solutions is likely to depend on the initial distribution of individuals. To address this issue, we expect that particles can exchange information not only from their nearby particles, but also from the

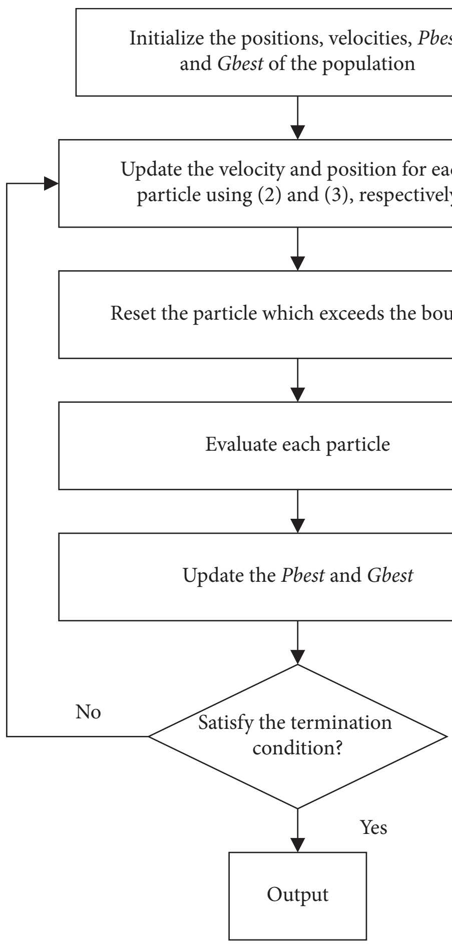

Initialize the positions, velocities, *Pbest*, and *Gbest* of the population
Update the velocity and position for each particle using (2) and (3), respectively
Reset the particle which exceeds the bounds
Evaluate each particle
Update the *Pbest* and *Gbest*
Satisfy the termination condition?
No
Yes
Output

FIGURE 1: Flow chart of a conventional PSO.

distant particles. Thereby, a balance can be achieved between the exploration and exploitation of the algorithm.

In view of this, this section presents a dynamic ε-neighborhood approach, where ε refers to the neighborhood radius. This approach is inspired by the idea of density clustering, and readers interested in this can refer to [36]. Some definitions of dynamic ε-neighborhood are presented as follows.

Definition 1. (Core object). X_i is a core object if the ε-neighborhood of X_i contains at least *MinPts* samples.

Definition 2. (Directly neighborhood-reachable). If X_j is located in the ε-neighborhood of X_i, and X_i is the core object, then X_j is directly reachable by X_i.

Definition 3. (Neighborhood-reachable). For X_i and X_j, if there exists a point p that makes X_i and p directly neighborhood-reachable and p and X_j are also directly neighborhood-reachable, then X_j is reachable by neighborhood of X_i.

Figure 2 further illustrates the definitions of dynamic ε-neighborhood. From Figure 2, we can easily obtain that if X is a core object, then A, B, C, D, and E are directly

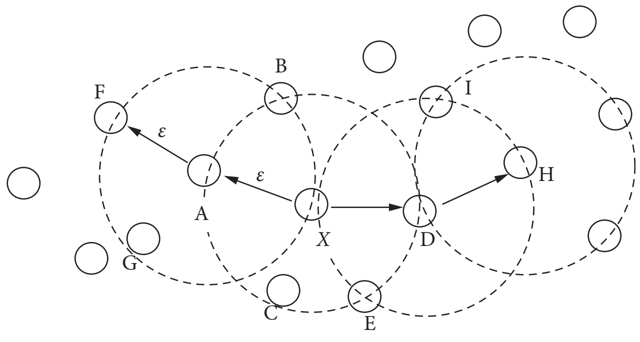


Figure 2: Diagram of dynamic ε-neighborhood.

neighborhood-reachable by X, and F, G, H, and I are neighborhood-reachable by X.

It should be noted that the purpose of dynamic ε-neighborhood selection mechanism is to obtain the neighbors of each particle in the population. Therefore, each particle in the population is regarded as the core object. According to Definition 1, the ε-neighborhood of a core object should include at least *MinPts* points. A fixed value for *MinPts* is first provided in this paper, as the value of ε is difficult to set without prior knowledge. Then, we set $\varepsilon_i = dis_{max}$, where dis_{max} refers to the maximum distance between X_i and the particles in the set of directly neighborhood-reachable. Based on this, the neighborhood-reachable particles of X_i are found, so as to form the dynamic neighborhood of particle X_i.

The proposed dynamic ε-neighborhood has three characteristics. (1) The neighborhood size of particle X_i varies dynamically in different evolutionary stage. (2) The neighborhood sizes of different particles in the same evolutionary stage may also be different. (3) Dynamic ε-neighborhood divides the neighborhood of the particle X_i into two levels. The particles in the group of directly neighborhood-reachable can be regarded as the close neighbors of X_i, while the particles in the group of neighborhood-reachable are regarded as the far neighbors of X_i. Algorithm 1 presents the pseudocode of dynamic ε-neighborhood selection mechanism.

3.2. PSO Based on Neighborhood Information. The neighborhood information of particles is employed to deal with the MMOPs in DNPSO. The detailed pseudocode of DNPSO is shown in Algorithm 2. At the beginning of DNPSO, Latin hypercube sampling (LHS) is utilized to generate the initial population with the size of N_{PSO} in the search space. Then, under *MaxFEs*, the particles are iteratively updated in the population. The detailed procedures are described as follows. Firstly, Algorithm 1 is employed to find the neighborhood of $Pbest_i$ in *Pbest*, including the close neighbors in the group of directly neighborhood-reachable (Ndr) and far neighbors in the group of neighborhood-reachable (Nr). Then, comparing $Pbest_i$ with the particles in Ndr and Nr, a position updating strategy is selected for X_i. After the new position of X_i is generated and evaluated, it will be compared with $Pbest_i$. If the performance of the updated X_i is better than that of $Pbest_i$, update $Pbest_i$; otherwise, $Pbest_i$ remains unchanged.

The main idea of DNPSO is to assign the most appropriate position updating strategy to the particles, so that multiple optimal solutions of the optimization problem can be located simultaneously more effectively. Specifically, this paper adopts four updating strategies to generate offspring.

Case 1. When $Pbest_i$ has the optimal performance in its directly neighborhood-reachable set, Ndr, and neighborhood-reachable set, Nr, it indicates that $Pbest_i$ is likely to be close to a peak in the search space, as shown in Figure 3(a). Therefore, $Pbest_i$ can be modified in a small scale by adding a Gaussian disturbance to it in the expectation that it will move in the direction of its nearest peak. The position update formula of X_i is given as follows:

$$x_{i,d}(t+1) = Pbest_{i,d}(t) + Gaussian(0,\sigma), \tag{5}$$

where Gaussian $(0,\sigma)$ is the Gaussian distribution with mean zero and standard deviation σ.

Case 2. When P_{besti} is optimal in its Ndr but is not the best in its Nr, it indicates that P_{besti} may be on a valley or on an unimportant local optimal peak, as shown in Figure 3(b). Therefore, it is necessary to exchange information with the particles in its distant neighbors, i.e., the neighbors in Nr, to make it jump out of the unimportant local optimum. The velocity update strategy of X_i adopts formula (3), where nbest stores all the particles which is better than $Pbest_i$ in Nr.

Case 3. When $Pbest_i$ is not optimal in its Ndr and not worst in its Nr, it indicates that $Pbest_i$ may be halfway up a hill in the search space, as shown in Figure 3(c). Therefore, the convergence speed can be improved and the exploitation of the algorithm can be improved by exchanging information with the superior particles in the nearest neighbors. Formula (3) is also used for the velocity update of X_i, where nbest is the nearest neighbor set, i.e., the particles which are better than P_{besti} in Ndr.

Case 4. When $Pbest_i$ has the worst performance in its Ndr and Nr, indicating that $Pbest_i$ may be close to a valley, as shown in Figure 3(d), at this point, it can learn multiple directions, so as to improve the exploration of the algorithm. Then, the velocity update formula of X_i adopts the following formula:

$$\begin{aligned} v_{i,d}(t+1) = {} & wv_{i,d}(t) + c_1 r_1\left(Nbest_{Ndr,d}(t) - x_{i,d}(t)\right) \\ & + c_2 r_2\left(Nbest_{Nr,d}(t) - x_{i,d}(t)\right), \end{aligned} \tag{6}$$

where $Nbest_{N\,dr}(t)$ is the optimal particle in Ndr and $Nbest_{Nr}(t)$ is the optimal particle in Nr of $Pbest_i$.

It should be noted that a neighborhood-reachable set, Nr, may be an empty set. In this case, two situations are discussed: (1) when $Pbest_i$ is optimal in its Ndr, it will be processed in accordance with Case 1; (2) when $Pbest_i$ is not the best in its Ndr, it will be processed in accordance with Case 3.

```
Input: Population size, $N_{\mathrm{PSO}}$, MinPts, the $i$-th particle in the population, $X_i$;
Output: Ndr: Directly neighborhood-reachable group of Xi; Nr: neighborhood-reachable group of $X_i$;
(1) For $j = 1$: $N_{\mathrm{PSO}}$;
(2)   While $X_i \sim = X_j$ do;
(3)     Calculate the Euclidean distance between $X_i$ and $X_j$;
(4)   End;
(5) End;
(6) Take the nearest $MinPts$ particles with $X_i$, $p1, \ldots, pMinPts$, and store them in Ndr;
(7) The maximum distance between $X_i$ and the particles in Ndr is denoted as dismax, and set $\varepsilon_i = $ dismax;
(8)   For $k = 1$: $MinPts$;
(9)     For $j = 1$: $N_{\mathrm{PSO}}$;
(10)      While $p1$-$MinPts \sim = X_j$ & $X_i \sim = X_j$ do;
(11)        Calculate the Euclidean distance between $pk$ and $X_j$. If the distance is less than $\varepsilon_i$, then put it into Nr.
(12)      End;
(13)    End;
(14) End.
```

Algorithm 1: Dynamic ε-neighborhood selection of particle X_i.

3.3. Complexity Analysis. The time complexity of multimodal EAs is governed by their niching components. Generally, it is estimated by the number of elementary operations performed at each generation. The computational complexity of the proposed DNPSO is composed of two parts: the construction of particle's neighborhood in Subsection 3.1 and the operation of PSO in Subsection 3.2. The complexity of constructing the neighborhood of a particle is $O(D \cdot N_{PSO} + MinPts \cdot N_{PSO})$. Then, the complexity of constructing all particle's neighborhood in the population is $O((D + MinPts) \cdot N_{PSO}^2)$, where D is the problem dimension and N_{PSO} is the population size. The time complexity of running PSO is $O(DN_{PSO})$. Hence, the total time complexity is $O((D + MinPts) \cdot N_{PSO}^2 + D \cdot N_{PSO})$. Since the value of N_{PSO} is large, the amortized cost of DNPSO for each generation is $O((D + MinPts) \cdot N_{PSO}^2)$.

4. Experiments and Analysis

In order to verify the effectiveness of the proposed DNPSO, the experiment is divided into the following two parts: (1) analyzing the sensitivity of the proposed algorithm to the value of *MinPts* and (2) comparing DNPSO with 7 state-of-the-art multimodal EAs to test its capability of tackling MMOPs. In this paper, eight widely used benchmark problems are selected to test the performance of the algorithm. These benchmark problems are derived from the IEEE CEC 2013 special section on multimodal optimization [37]. The characteristics of these instances are shown in Table 1, where the "Peak height" refers to the value of the global optimal solution. All the algorithms are implemented by MATLAB R2014b on a CPU with Intel Core i5 and 1.6 GHz, and the experimental results are the average of 30 independent runs.

4.1. Parameter Settings. In DNPSO, the population size, N_{PSO}, the maximum number of evaluations, *MaxFEs*, and the niching radius, r (which is used to distinguish the two neighboring global optimal solutions) are all set according to [34], as shown in Table 2. The amplitude accuracy is set to 1E-03. $MinPts = 3$; in formula (4), $w = 0.7298$; the standard deviation in formula (6) is 0.1; in formula (7), $w = 0.7298$, $c_1 = 2.05$, and $c_2 = 2.05$. According to [26], the neighborhood size (nsize) changes within {2, 3, 4, 5} as increase of generations.

4.2. Performance Metrics. In this paper, two commonly used performance indicators are used to evaluate the performance of an algorithm [28]:

(1) Peak ratio (PR): the peak ratio is the average percentage of the global optima found in multiple independent runs. The PR is calculated using the following formula:

$$\mathrm{PR} = \frac{\sum_{i=1}^{R} \mathrm{NPF}_i}{\mathrm{NPF} \times R}, \tag{7}$$

where NPF_i is the number of global optima found in the i-th run, NPF denotes the number of know global optima, and R is the number of runs.

(2) Success rate (SR): success rate is the ratio between the number of successful runs (NSR) and the total number of runs (R). A successful run of an algorithm is when all global peaks are found in a run. It is calculated as follows:

$$\mathrm{SR} = \frac{\mathrm{NSR}}{R}. \tag{8}$$

4.3. Analysis on Key Parameters. In this section, F2, F3, and F4 (when $D = 2$ and 3) and F6 and F7 (when $D = 5$) are selected as representatives to analyze the influence of the *MinPts* and σ, on the performance of DNPSO.

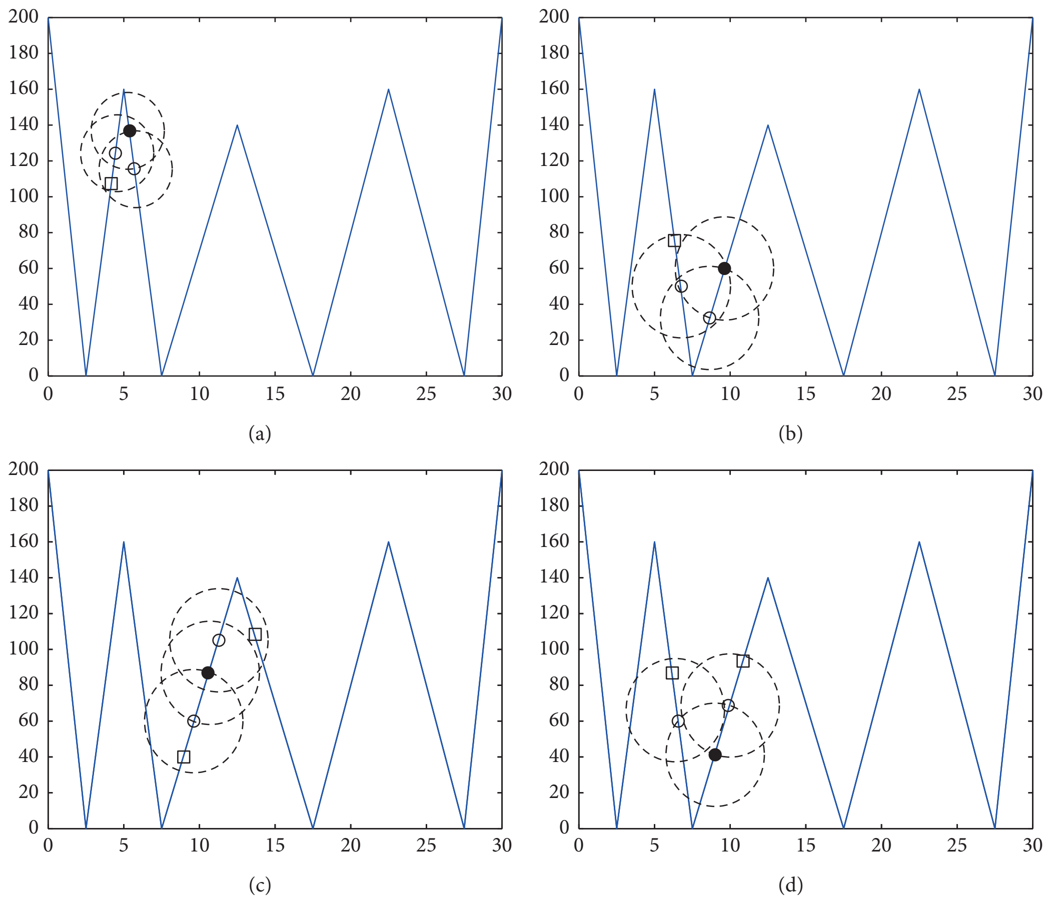


Figure 3: Illustration of the four cases of particles. The black solid circle is the core particle, the hollow circles are the direct neighborhood-reachable points of the core particle, and the squares are the neighborhood-reachable point of the core particle.

```
Input: The population size, $N_{\text{PSO}}$, the maximum number of evaluations, $MaxFEs$;
Output: $Pbest$;
(1) Generate an initial population with the size of NPSO with LHS, initialize $Pbest$;
(2) Evaluate the fitness of the particles in the initial population;
(3) $FEs = N_{\text{PSO}}$;
(4) While $FEs < MaxFEs$ do;
(5)    For $i = 1$: $N_{\text{PSO}}$;
(6)       Find the neighbors of $Pbest_i$ with Algorithm1, include Ndr and Nr;
(7)       If the fitness of $Pbest_i$ is better than that of each individual in Ndr and Nr, then;
(8)          Update the position of $X_i$ by strategy (6);
(9)       Elseif the fitness of $Pbest_i$ is the best in Ndr but is not the best in Nr, then;
(10)         Update the velocity and position of $X_i$ using strategies (4) and (2), respectively;
(11)      Elseif the fitness of $Pbest_i$ is not the best in Ndr and is not the worst in Nr, then;
(12)         Update the velocity and position of $X_i$ with strategies (4) and (2), respectively;
(13)      Elseif the fitness of $Pbest_i$ is worst in Ndr and Nr, then;
(14)         Update the velocity and position of $X_i$ with strategies (7) and (2), respectively;
(15)      End;
(16)      Evaluate the fitness of $X_i$;
(17)      Update $Pbest_i$;
(18)      $FEs = FEs + 1$;
(19)   End;
(20)   End.
```

Algorithm 2: DNPSO.

Table 1: Benchmark instances.

	Function	D	Decision space	No. of global/ local optima	Peak height
F1	Equal maxima	1	$x \in [0\ 1]$	5/0	1
F2	Uneven decreasing maxima	1	$x \in [0\ 1]$	¼	1
F3	Six-hump camel back	2	$x_1 \in [-1.9\ 1.9]$ $x_2 \in [-1.1\ 1.1]$	2/2	1.0316
F4	Shubert	2	$X \in [-10, 10]^2$	18/many	186.731
	Shubert	3	$X \in [-10, 10]^3$	81/many	2709.093
F5	Vincent	2	$X \in [0.25, 10]^2$	36	1
F6	Modified Rastrigin	2	$X \in [0, 1]^D$	12/0	-2
F7	Composition function 3	5/10	$X \in [-5, 5]^D$	6/many	0
F8	Composition function 4	5/10	$X \in [-5, 5]^D$	8/many	0

Table 2: Parameter settings.

	F1	F2	F3	F4(2D)	F4(3D)	F5	F6	F7	F8
MaxFEs	5 × 104	5 × 104	5 × 104	2 × 105	4 × 105	2 × 105	2 × 105	4 × 105	4 × 105
Npop	100	100	100	300	300	300	100	200	200
R	0.01	0.01	0.5	0.5	0.5	0.2	0.01	0.01	0.01

4.3.1. Analysis of MinPts. The value of MinPts determines the value of neighborhood radius, ε, in Subsection 3.1, which indirectly affects the number of particles in the neighborhood-reachable set, Nr. Here, we analyze the influence of MinPts on the performance of DNPSO, when its values set is 2, 3, 4, and 5, respectively. Table 3 shows the PR and SR values obtained by DNPSO with different values of *MinPts*.

It can be concluded from Table 3 that (1) for problems F2, F3, and F6, the proposed DNPSO obtains the same PR and SR when *MinPts* = 2, 3, 4, and 5. (2) For other problems, the PR value achieved from DNPSO when *MinPts* = 3 is the highest when *MinPts* = 2, 3, 4, and 5. (3) For F4 (2D), the SR value obtained from DNPSO when *MinPts* = 3 is higher than those when *MinPts* = 2, 4, and 5; for F4(3D) and F7(5D), when *MinPts* = 2, 3, 4, and 5, the SR values obtained from DNPSO are equal to 0. Considering the performance of DNPSO with different values of *MinPts*, the value of *MinPts* is set as 3 in subsequent experiments.

Table 3: Results obtained by DNPSO with different *MinPts* values.

	2		3		4		5	
	PR	SR	PR	SR	PR	SR	PR	SR
F2	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F3	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F4(2D)	0.92	0.2	**1.00**	**1.00**	0.97	0.6	0.93	0.2
F4(3D)	0.15	0.00	**0.69**	0.00	0.55	0.00	0.41	0.00
F6	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F7(5D)	0.13	0.00	**0.70**	0.00	0.68	0.00	0.68	0.00

4.3.2. Analysis of σ. The value of σ determines the size of Gaussian disturbance in formula (6), that is, the range of local search. This section analyzes the influence of σ on the performance of DNPSO, when σ set is 0.01, 0.05, 0.1, and 0.2. Table 4 shows the PR and SR values obtained by DNPSO under different σ values. It can be seen from Table 4 that (1) for F2, F3, and F6, the PR and SR values obtained by different values are similar. (2) For other problems, the PR value obtained when $\sigma = 0.1$ is higher than those obtained when $\sigma = 0.01$, 0.05, and 0.2. Based on these results, the value σ is set as 0.1 in subsequent experiments.

4.4. Comparison with Multimodal Evolutionary Algorithms. In order to verify the effectiveness of DNPSO, this section compares it with seven state-of-the-art multimodal EAs. These comparison algorithms include three multimodal algorithms based on PSO (R2PSO, R3PSO [15], and LIPS [26]), three multimodal algorithms based on DE (NCDE, NSDE [23], and VCNDE [28]), and one multiobjective EA (EMO-MMO [38–42]). In order to ensure the fairness of comparison, the population size, N_{PSO}, the maximum number of evaluations, *MaxFEs*, the niching radius, r, and amplitude accuracy of all comparison algorithms are consistent with the proposed DNPSO, and the remaining parameters are set according to their original literatures' suggestions.

Table 5 shows the mean value and standard deviation of *PR* obtained by DNPSO and 7 multimodal EAs when dealing with problems F1–F8, and the optimal results have been highlighted. In this paper, the Mann-Whitney test at a significance level of 5% is employed to evaluate significant difference between DNPSO and the compared algorithms, where "+" and "−" mean that DNPSO is significantly superior to and inferior to the compared one, respectively, and "=" indicates that there is no significant difference between them. In the table, "test" gives the results of the nonparametric test. In the penultimate row of Table 5, "win/tie/lose" is used to calculate the comparison result between DNSPO and each compared algorithm. Among them, "win" means the number of test problems that DNPSO dominates the compared one, "tie" indicates that the performance of DNPSO is similar to that of the compared one, and "lose" means the number of test problems that DNPSO is dominated by the compared one.

Table 4: Results obtained by DNPSO with different σ values.

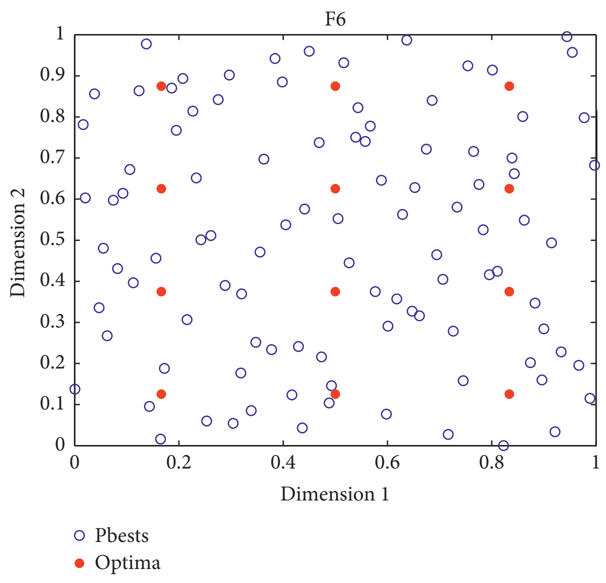
	0.01		0.05		0.1		0.2	
	PR	SR	PR	SR	PR	SR	PR	SR
F2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
F3	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
F4(2*D*)	0.98	0.80	1.00	1.00	0.98	0.80	1.00	1.00
F4(3*D*)	0.44	0.00	0.63	0.00	0.44	0.00	0.63	0.00
F6	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
F7(5*D*)	0.50	0.00	0.58	0.00	0.50	0.00	0.58	0.00

Table 5: Peak ratios (PRs) achieved from the compared algorithms.

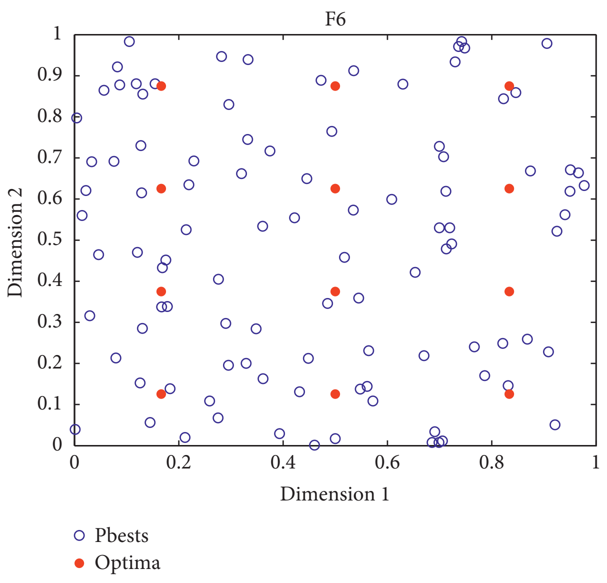
	PR	DNPSO	LIPS	EMO-MMO	R2PSO	R3PSO	NCDE	NSDE	VNCDE
F1	Mean	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.99	**1.00**
	Std	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Test	\	=	=	=	=	=	=	=
F2	Mean	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	Std	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Test	\	=	=	=	=	=	=	=
F3	Mean	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	Std	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Test	\	=	=	=	=	=	=	=
F4 (2*D*)	Mean	**1.00**	0.75	**1.00**	0.53	0.69	0.14	0.27	0.96
	Std	0.00	0.22	0.00	0.11	0.05	0.01	0.02	0.01
	Test	\	+	=	+	+	+	+	=
F4 (3*D*)	Mean	0.69	0.49	**0.80**	0.32	0.28	0.68	0.13	0.74
	Std	0.14	0.03	0.23	0.10	0.02	0.04	0.01	0.13
	Test	\	+	=	+	+	=	+	=
F5	Mean	**1.00**	0.20	**1.00**	0.01	0.16	0.50	0.06	0.55
	Std	0.00	0.05	0.00	0.00	0.01	0.01	0.04	0.05
	Test	\	+	=	+	+	+	+	+
F6	Mean	**1.00**	0.84	**1.00**	0.88	0.87	1.00	0.40	**1.00**
	Std	0.00	0.13	0.00	0.01	0.11	0.00	0.01	0.00
	Test	\	+	=	=	=	=	+	=
F7 (5*D*)	Mean	**0.70**	0.41	0.60	0.14	0.13	0.26	0.35	**0.70**
	Std	0.01	0.01	0.10	0.01	0.01	0.02	0.02	0.02
	Test	\	+	=	+	+	+	+	=
F7 (10*D*)	Mean	0.54	0.20	**0.66**	0.06	0.37	0.63	0.52	**0.66**
	Std	0.00	0.00	0.01	0.09	0.00	0.01	0.02	0.00
	Test	\	+	−	+	−	=	=	−
F8 (5*D*)	Mean	**0.38**	0.09	0.19	0.00	0.02	0.10	0.11	0.37
	Std	0.01	0.09	0.04	0.00	0.00	0.02	0.12	0.11
	Test	\	+	+	+	+	+	+	=
F8 (10*D*)	Mean	0.16	0.06	0.12	0.00	0.05	0.24	0.12	**0.29**
	Std	0.10	0.02	0.08	0.00	0.10	0.21	0.15	0.33
	Test	\	+	=	+	+	=	=	−
Win/tie/lose		\	8/3/0	1/9/1	7/4/0	6/4/1	4/7/0	6/5/0	1/8/2
Rank		2.81	5.18	2.86	6.27	5.90	4.32	5.95	**2.68**

Table 6: Post hoc analysis using DNPSO as control method.

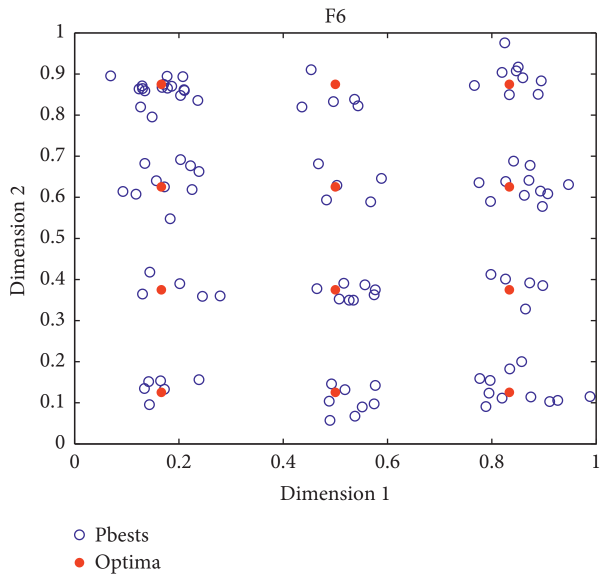
DNPSO vs LIPS	LIPS	EMO-MMO	R2PSO	R3PSO	NCDE	NSDE	VNCDE
Statistic	2.26	0.04	3.30	2.95	1.43	3.00	0.13
Adjusted p value	0.0409	0.9652	0.0065	0.0093	0.2047	0.0093	0.9287



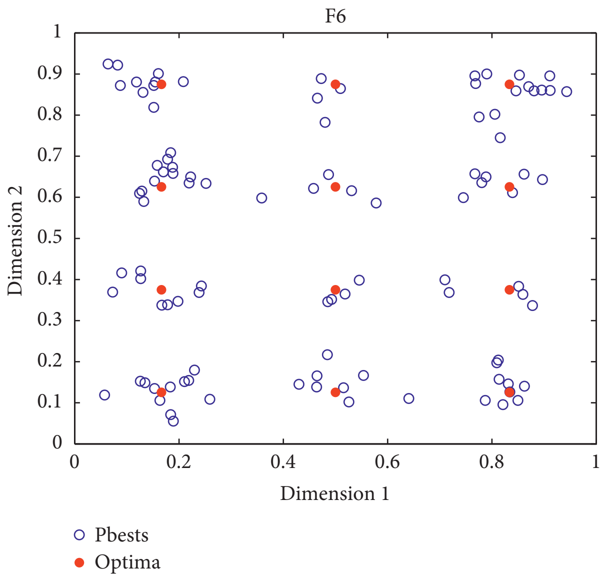
(a)



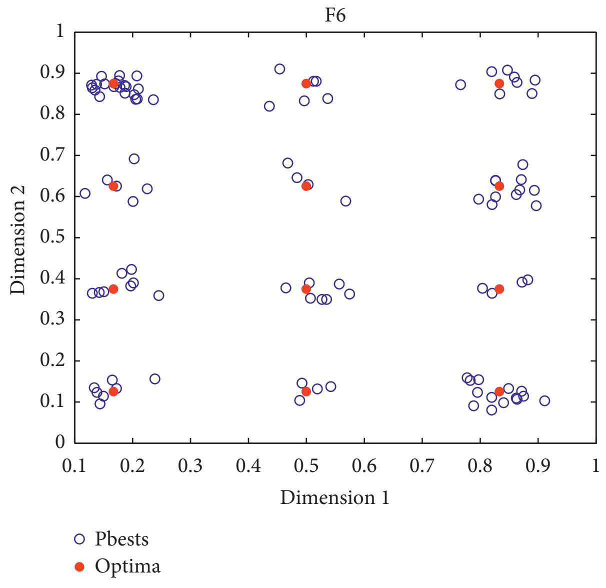
(b)



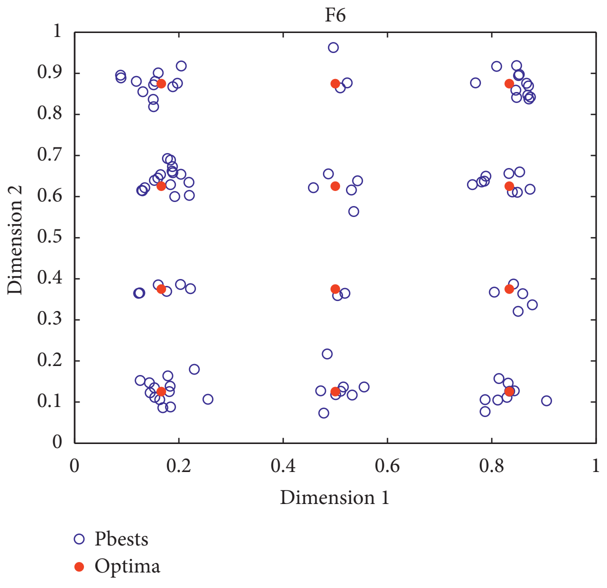
(c)



(d)

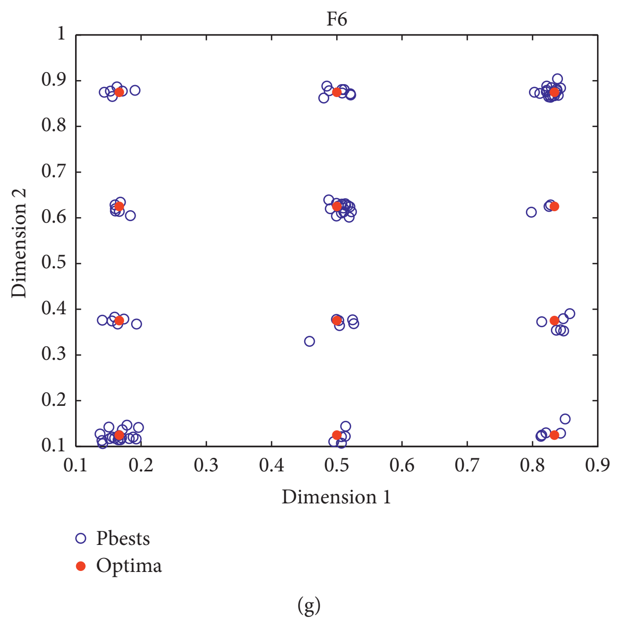


(e)



(f)

Figure 4: Continued.



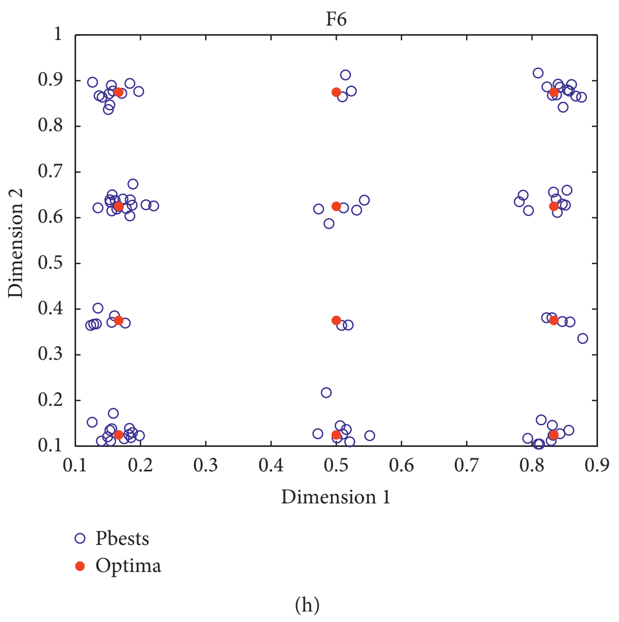


FIGURE 4: Distributions of Pbests of DNPSO and LIPS on different stages of F6. (a) Iteration 1 of DNPSO. (b) Iteration 1 of LIPS. (c) Iteration 5 of DNPSO. (d) Iteration 5 of LIPS. (e) Iteration 10 of DNPSO. (f) Iteration 10 of LIPS. (g) Iteration 15 of DNPSO. (h) Iteration 15 of LIPS.

It can be obtained from Table 5 that (1) for problems F1, F2, and F3 there is no significant difference between the PR obtained by DNPSO and all compared algorithms. (2) For F4 and F7(5*D*), except EMO-MMO and VNCDE, PRs obtained by the remaining five algorithms are significantly inferior to DNPSO. (3) For F5, there is no significant difference between DNPSO and EMO-MMO, but the PRs obtained by DNPSO are significantly better than those of the other six compared ones. (4) For F6, the PR obtained by DNPSO is significantly better than those of LIPS and NSDE, with no significant difference from the other 5 compared algorithms. (5) For F7 (10*D*), the PRs achieved by DNPSO are obviously better than those of LIPS and R2PSO, but worse than those of EMO-MMO, R3PSO, and VNCDE. (6) For F8 (5*D*), except VNCDE, PRs obtained by the remaining 6 algorithms are significantly inferior to DNPSO. (7) For F8 (10*D*), the PR achieved by DNPSO is obviously better than those of LIPS, R2PSO, and R3PSO, but worse than that of VNCDE. According to the overall statistical results, DNPSO dominated LIPS on at least 8 instances, superior than R2PSO on at least 7 instances, and outperformed R3PSO and NSDE on at least 6 instances, superior than NCDE on at least 4 instances.

In addition, STAC platform is used to test the difference between the proposed DNSPO and the compared algorithms, and Friedman test with significance level of 0.05 is used for analysis. The analysis results are shown in the last row of Table 5, and the comprehensive ranking values of all algorithms on *PR* are given. DNPSO is used as the control method, it can be seen that rank of DNPSO is slightly higher than that of VNCDE, but lower than those of the other 6 compared algorithms. Furthermore, we adjusted the *p* value of pairwise comparison by using Finner operation, and the results are listed in Table 6. As can be seen from Table 6, DNPSO is statistically superior to LIPS, R2PSO, R3PSO, and NSDE and has similar performance with other three compared algorithms.

Figure 4 shows the distribution of *Pbest* of the proposed DNPSO and LIPS on different population iterations for F6, where F6 has 12 global optimal solutions, as shown by red solid circles in the figure. It can be achieved from the figure that (1) both of them can converge to the vicinity of 12 global optimal solutions within 15 iterations; (2) the blue circles in Figure 4(g) are closer to the red solid circles than those in Figure 4(h), which indicates that the convergence precision of DNPSO is higher than that of LIPS.

Table 7 lists the SR obtained by DNPSO and 7 multimodal EAs when dealing with problems F1–F8. The optimal results have been highlighted. The last row shows the comprehensive ranking values of SR of all algorithms. It can be seen from Table 7 that (1) for problems F1, F2, and F3 DNPSO and all the compared algorithms can find all the optimal solutions in each run. (2) For F4 (2D), the SR of DNPSO and EMO-MMO is equal to 1; the SR of VNCDE is equal to 0.54, while the SR obtained by the remaining 5 algorithms is 0. (3) For F5, the SRs of DNPSO and EMO-MMO are 1, and those of other algorithms are 0. (4) For F6, the *SR* values of DNPSO, EMO-MMO, NCDE, and VNCDE are all higher than those of the other three compared algorithms. (5) For F4 (3*D*), F7, and F8, the SR values obtained by all algorithms is 0; that is, no single run can find all the global optimal solutions. (6) From the perspective of rank value, DNPSO and EMO-MMO have similar performance, and the rank value is superior to the other compared algorithms.

Table 8 lists the running time of DNPSO and 7 multimodal EAs on the problems F1-F8. It can be seen from Table 8 that (1) R2PSO has the shortest running time and VNCDE has the longest running time. (2) The proposed

Table 7: Success rates (SRs) achieved from the compared algorithms.

	DNPSO	LIPS	EMO-MMO	R2PSO	R3PSO	NCDE	NSDE	VNCDE
F1	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.98	**1.00**
F2	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F3	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F4 (2*D*)	**1.00**	0.00	**1.00**	0.00	0.00	0.00	0.00	0.54
F4 (3*D*)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
F5	**1.00**	0.00	**1.00**	0.00	0.00	0.00	0.00	0.00
F6	**1.00**	0.84	**1.00**	0.22	0.24	**1.00**	0.00	**1.00**
F7 (5*D*)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
F7 (10*D*)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
F8 (5*D*)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
F8 (10*D*)	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Rank	**3.72**	4.72	**3.72**	4.90	4.81	4.50	5.36	4.22

Table 8: Running time comparison of the multimodal algorithms.

	DNPSO	LIPS	EMO-MMO	R2PSO	R3PSO	NCDE	NSDE	VNCDE
F1	0.24	0.09	1.07	0.04	0.06	1.17	5.72	6.00
F2	0.24	0.09	1.08	0.03	0.08	1.12	6.48	5.90
F3	0.36	0.11	1.09	0.04	0.11	1.17	6.41	6.25
F4 (2*D*)	3.56	0.47	3.45	0.22	0.20	4.76	26.18	24.48
F4 (3*D*)	3.79	0.46	3.47	0.21	0.61	4.71	22.57	24.64
F5	6.91	1.25	8.90	0.60	0.93	10.01	51.42	46.88
F6	2.11	0.52	3.56	0.21	0.56	4.72	22.04	22.06
F7 (5*D*)	17.82	8.73	20.34	8.01	9.41	16.48	54.17	56.32
F7 (10*D*)	17.90	9.03	23.40	8.45	8.65	16.72	54.52	57.15
F8 (5*D*)	29.01	15.51	32.32	14.83	16.93	25.67	60.97	89.91
F8 (10*D*)	30.98	16.71	35.01	15.75	22.75	26.34	62.53	93.11

DNPSO has slightly longer running time than those of LIPS, R2PSO, and R3PSO, but less than those of the remaining four algorithms.

5. Conclusions

To tackle MMOPs, this paper presents a multimodal particle swarm optimization algorithm based on dynamic neighborhood, called DNPSO. The proposed DNPSO defines a dynamic neighborhood selection mechanism, which makes the neighborhood size of particles change dynamically in the process of evolution, and distinguishes the particles in the neighborhood between directly neighborhood-reachable and neighborhood-reachable at the same time, so as to balance the exploration and exploitation of the algorithm. Following that, based on the information provided by the neighbors, four different particle position updating strategies are designed to further support the algorithm's exploration and exploitation of the search space. The experimental results on eight test benchmark functions show that the proposed algorithm is competitive with several existing multimodal EAs and is an effective method to deal with MMOPs.

DNPSO, like other EAs, shows poor performance when dealing with high-dimensional MMOPs, which requires further study. It is also suggested to extend this dynamic neighborhood method to other EAs like DE in the future. In addition, it may be worthy to apply DNPSO to real-world problems such as feature selection.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This research was funded by the project of Jiangsu Provincial Department of Emergency Management (no. YJGL-YF-2020-15), the Scientific Research Staring Foundation of Shantou University (no. NTF20009), and the open project of Key Laboratory of Computational Intelligence and Signal Processing for Ministry of Education (no. 2019A002).

References

[1] C. Stoean, M. Preuss, R. Stoean, and D. Dumitrescu, "Multimodal optimization by means of a topological species conservation algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 6, pp. 842–864, 2011.

[2] J. W. Kruisselbrink, "Enhancing search space diversity in multi-objective evolutionary drug molecule design using niching," in *Proceedings of the 11th Annual Conference on Genetic Evolutionary Computation*, pp. 217–224, Montreal, QC, Canada, 2009.

[3] G.-C. Luh and C.-Y. Lin, "Optimal design of truss-structures using particle swarm optimization," *Computers & Structures*, vol. 89, no. 23-24, pp. 2221–2232, 2011.

[4] K. C. Wong, K. S. Leung, and M. H. Wong, "Protein structure prediction on a lattice model via multimodal optimization techniques," in *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2010*, pp. 155–162, Portland, OR, USA, July 2010.

[5] 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.

[6] X. Li, "Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization," in *Proceedings of the Genetic and Evolutionary Computation - GECCO 2004*, pp. 105–116, Seattle, WA, USA, 2004.

[7] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Boston, MA, USA, 1989.

[8] Y. Tian, T. Zhang, J. Xiao, X. Zhang, and Y. Jin, "A coevolutionary framework for constrained multi-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, p. 1, 2020.

[9] L. Tang, Y. Dong, and J. Liu, "Differential evolution with an individual dependent mechanism," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 4, pp. 560–574, 2015.

[10] H. Wu, C. Nie, F.-C. Kuo, H. Leung, and C. J. Colbourn, "A discrete particle swarm optimization for covering array generation," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 4, pp. 575–591, 2015.

[11] S. W. Mahfoud, "Crowding and preselection revisited," in *Parallel Problem Solving from Nature 2*, R. Männer and B. Manderick, Eds., pp. 27–36, North-Holland, Amsterdam, The Netherlands, 1992.

[12] X. Yin and N. Germay, "A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization," in *Artificial Neural Nets and Genetic Algorithms*, R. F. Albrecht, C. R. Reeves, and N. C. Steele, Eds., pp. 450–457, Springer, Vienna, Austria, 1993.

[13] X. Xiang, Y. Tian, J. Xiao, and X. Zhang, "A clustering-based surrogate-assisted multi-objective evolutionary algorithm for shelter location problem under uncertainty of road networks," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 12, pp. 7544–7555. In press, 2020.

[14] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 97–112, 2018.

[15] X. Li, "Niching without niching parameters: particle swarm optimization using a ring topology," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 4, p. 665, 2010.

[16] K. E. Parsopoulos, V. P. Plagianakos, G. D. Magoulas, and M. N. Vrahatis, "Objective function "stretching" to alleviate convergence to local minima," *Nonlinear Analysis: Theory, Methods & Applications*, vol. 47, no. 5, pp. 3419–3424, 2001.

[17] B. Qu, L. Xie, and C. Li, "Particle swarm optimization algorithms based on multi-mode optimization problem," *Journal of Zhongyuan University of Technology2018*, vol. 29, no. 4, pp. 70–76, 2018.

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

[19] Y. Zhang, D. W. Gong, and Z. H. Ding, "Handling multi-objective optimization problems with a multi-swarm cooperative particle swarm optimizer," *Expert Systems with Applications*, vol. 38, no. 11, pp. 3933–3941, 2011.

[20] X. Zhang, X. Zheng, R. Cheng, J. Qiu, and Y. Jin, "A competitive mechanism based multi-objective particle swarm optimizer with fast convergence," *Information Sciences*, vol. 427, pp. 63–76, 2018.

[21] R. Brits, A. P. Engelbrecht, and F. Van den Bergh, "Solving systems of unconstrained equations using particle swarm optimizers," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pp. 102–107, Hammamet, Tunisia, October 2002.

[22] J. Qiu, J. Wan, L. Zhang, F. Cheng, and Y. Luo, "Community-grouping based particle swarm optimisation algorithm for feature selection," in *Proceedings of the 2020 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, Glasgow, UK, July 2020.

[23] T. Blackwell and J. Branke, "Multiswarms, exclusion, and anti-convergence in dynamic environments," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 4, pp. 459–472, 2006.

[24] X. Zhang, T. Tan, B. Zhou, T. Yu, B. Yang, and X. Huang, "Adaptive distributed auction-based algorithm for optimal mileage based AGC dispatch with high participation of renewable energy," *International Journal of Electrical Power & Energy Systems*, vol. 124, 2020.

[25] I. L. Schoeman and A. P. Engelbrecht, "Using vector operations to identify niches for particle swarm optimization," in *IEEE Conference on Cybernetics and Intelligent Systems*, pp. 361–366, Singapore, December 2004.

[26] 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.

[27] Y. H. Zhang, Y. J. Gong, H. X. Zhang, T. L. Gu, and J. Zhang, "Toward fast niching evolutionary algorithms: a locality sensitive hashing-based approach," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 3, pp. 347–362, 2017.

[28] Y.-H. Zhang, Y.-J. Gong, Y. Gao, H. Wang, and J. Zhang, "Parameter-free Voronoi neighborhood for evolutionary multimodal optimization," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 2, pp. 335–349, 2020.

[29] J. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm Intelligence*, Morgan Kaufmann, San Francisco, CA, USA, 2001.

[30] I. Rodriguez-Fdez, A. Canosa, M. Mucientes, and A. Bugarin, "A web platform for the comparison of algorithms using statistical tests," in *IEEE International Conference on Fuzzy Systems*, Istanbul, Turkey, August 2015.

[31] Y. Zhang, D.-w. Gong, and J. Cheng, "Multi-objective particle swarm optimization approach for cost-based feature selection in classification," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 14, no. 1, pp. 64–75, 2017.

[32] Y. Hu, Y. Zhang, and D. Gong, "Multiobjective particle swarm optimization for feature selection with fuzzy cost," *IEEE Transactions on Cybernetics*, p. 1, 2020.

[33] 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.

[34] 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.

[35] F. Cheng, Q. Zhang, Y. Tian, and X. Zhang, "Maximizing receiver operating characteristics convex hull via dynamic

reference point-based multi-objective evolutionary algorithm," *Applied Soft Computing*, vol. 86, 2020.

[36] Z. Zhou, *Machine Learning*, Tsinghua University Press, Beijing, China, 2016.

[37] X. Li, A. Engelbrecht, and M. G. Epitropakis, "Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization," *Evolutionary Computation Machine Learning Group*, Technical Report, RMIT University, Melbourne, VIC, Australia, 2013.

[38] R. Cheng, M. Li, K. Li, and X. Yao, "Evolutionary multiobjective optimization-based multimodal optimization: fitness landscape approximation and peak detection," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 5, pp. 692–706, 2018.

[39] D. Gong, B. Xu, Y. Zhang, Y. Guo, and S. Yang, "A similarity-based cooperative co-evolutionary algorithm for dynamic interval multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 142–156, 2020.

[40] X. Zhang, Z. Xu, T. Yu, B. Yang, and H. Wang, "Optimal mileage based AGC dispatch of a GenCo," *IEEE Transactions on Power Systems*, vol. 35, no. 4, pp. 2516–2526, 2020.

[41] Y. Chen, X. Sun, D. Gong, Y. Zhang, J. Choi, and S. Klasky, "Personalized search inspired fast interactive estimation of distribution algorithm and its application," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 4, pp. 588–600, 2017.

[42] Y. Chen, X. Sun, D. Gong, and X. Yao, "DPM-IEDA: dual probabilistic model assisted interactive estimation of distribution algorithm for personalized search," *IEEE Access*, vol. 7, pp. 41006–41016, 2019.