Hindawi
Complexity
Volume 2020, Article ID 7824785, 19 pages
https://doi.org/10.1155/2020/7824785

Research Article

Improved Laplacian Biogeography-Based Optimization Algorithm and Its Application to QAP

Xinming Zhang,[1] Doudou Wang,[1] Haiyan Chen,[2] Wentao Mao,[1] Shangwang Liu,[1] Guoqi Liu,[1] and Zhi Dou[1]

[1] *College of Computer and Information Engineering, Henan Normal University, Xinxiang, Henan 453007, China*
[2] *Department of Gynecological Tumor, Hubei Cancer Hospital, Wuhan, Hubei 430079, China*

Correspondence should be addressed to Haiyan Chen; cleverbear88@126.com

Received 27 April 2019; Revised 9 January 2020; Accepted 17 January 2020; Published 6 April 2020

Academic Editor: Quanmin Zhu

Laplacian Biogeography-Based Optimization (LxBBO) is a BBO variant which improves BBO's performance largely. When it solves some complex problems, however, it has some drawbacks such as poor performance, weak operability, and high complexity, so an improved LxBBO (ILxBBO) is proposed. First, a two-global-best guiding operator is created for guiding the worst habitat mainly to enhance the exploitation of LxBBO. Second, a dynamic two-differential perturbing operator is proposed for the first two best habitats' updating to improve the global search ability in the early search phase and the local one in the late search one, respectively. Third, an improved Laplace migration operator is formulated for other habitats' updating to improve the search ability and the operability. Finally, some measures such as example learning, mutation operation removing, and greedy selection are adopted mostly to reduce the computation complexity of LxBBO. A lot of experimental results on the complex functions from the CEC-2013 test set show ILxBBO obtains better performance than LxBBO and quite a few state-of-the-art algorithms do. Also, the results on Quadratic Assignment Problems (QAPs) show that ILxBBO is more competitive compared with LxBBO, Improved Particle Swarm Optimization (IPSO), and Improved Firefly Algorithm (IFA).

1. Introduction

Optimization has always been a hot topic for various researchers, engineers, and others. In solving some complex problems, traditional optimization methods mainly rely on empirical analysis or the use of accurate mathematical models. However, they cannot find the optimal solution on other problems or within a reasonable time. In recent decades, inspired by nature, many Intelligent Optimization Algorithms (IOAs) such as Particle Swarm Optimization (PSO) [1], Shuffled Frog Leaping Algorithm (SFLA) [2], Differential Evolution (DE) [3], Krill Herd (KH) [4], Harmony Search (HS) [5], Cuckoo Search (CS) [6], Grey Wolf Optimizer (GWO) [7], and Biogeography-Based Optimization (BBO) [8], have sprung up, and they are widely used in many areas [6, 9–11].

BBO is a biogeography-based IOA proposed by Simon [8]. The mathematical models of biogeography describe the migration and mutation of species. According to the models, two operators are created in BBO. The migration operator of BBO shares information between habitats to improve the quality of poor solutions, and it demonstrates the exploitation ability of BBO. The mutation operator, which randomly generates new feature values, can get good population diversity to reflect the exploration ability. As BBO has a good model, simple search mechanism, and excellent performance [8], not only has it achieved great success on numerical optimization problems [12] but also is being used in so many applications [13–16]. Compared with some IOAs, BBO is more competitive and has attracted more widespread attention [17]. However, BBO has some defects such as easy entrapment into local optima and weak exploration [18].

Quadratic Assignment Problem (QAP) is a mathematical model for the location of indivisible economic activities, which is one of the most difficult combinational optimization problems [19]. QAP has been studied for several years in an assignment problem that models a variety of real-world problems, such as backboard wiring problem, campus planning problem, airport gate assignment, and traveling salesman problem [20]. Hospital layout problem is a typical QAP to minimize the total travel distance of patients. It can bring better service to patients by renovating the existing hospital and optimizing the reallocation of the departments, reduce the time consumption of each patient, and improve the efficiency of hospital service for more patients [20]. IOAs are devoted to the search for the good quality solutions, which are applied to solve QAP [19].

Although BBO variants are proposed to deal with the defects of BBO, whether BBO or its variants, the migration operator plays an important role. Garg and Deep [21] proposed a novel BBO based on the Laplace migration operator (LxBBO), which is an improved migration operator to strengthen the search capability of BBO. However, LxBBO still has some drawbacks such as poor performance, weak operability, and high complexity. So, in this paper, an improved LxBBO (ILxBBO) is proposed, and it is used to solve QAP: an alternative approach for locating hospital departments.

The contributions of this paper are described as follows:

(1) A dynamic two-differential perturbing operator is proposed to update the first two best habitats and used mainly to enhance the exploration in the early search phase, and the local search ability can be improved in the late search phase

(2) A two-global-best guiding operator is presented to update the worst habitat and used mainly to enhance exploitation; at the same time, the global search ability can be improved in the early search phase

(3) An improved Laplace operator is formulated for the other habitats' updating to fasten the convergence speed and improve the operability

(4) ILxBBO is used for complex function optimization on CEC-2013 and applied to QAP; a lot of experimental results show ILxBBO obtains more performance than the comparison algorithms.

The graphical abstract of this paper is shown in Figure 1.

The rest of this paper is organized as follows: Section 2 gives the related work. The proposed ILxBBO is elaborated in Section 3. In Section 4, experimental results on the CEC-2013 test set and QAP are reported and analyzed. Section 5 gives conclusions and future work.

2. Related Works

2.1. Biogeography-Based Optimization. BBO mainly uses migration and mutation models of species in biogeography to solve optimization problems. In BBO, each solution is called a "habitat" with a Habitat Suitability Index (HSI) to measure the quality of the habitat. The factors of a habitat that characterize habitability are called Suitability Index Variables (SIVs). BBO searches the best solution mainly based on migration and mutation steps.

2.1.1. Migration Operator. In BBO, a good solution tends to have a high HSI and it is analogous to a habitat with many species, which has high emigration and low immigration rates, and vice versa. The purpose of the migration operator is to share information between different solutions. Good solutions tend to share their features with poor solutions, and poor solutions accept a lot of features from good solutions. Each habitat has its own immigration rate λ and emigration rate μ, and they are calculated as follows:

$$\lambda_k = I\left(1 - \frac{N_k}{N}\right), \tag{1}$$

$$\mu_k = E\left(\frac{N_k}{N}\right), \tag{2}$$

where I is the maximum immigration rate, E is the maximum emigration rate, N_k is the number of species of the habitat H_k, and N is the maximum number of species. From equations (1) and (2), the migration presents a simple linear model, but more often, there are complex nonlinear models [22].

The migration operator modifies the habitat's SIVs by accepting features from other good habitats, which can be expressed as follows:

$$H_i(\text{SIV}) \,\%\!\leftarrow H_k(\text{SIV}), \tag{3}$$

where H_i is the immigration habitat and H_k is the emigration habitat, which is selected by the roulette wheel selection [8].

2.1.2. Mutation Operator. Sudden events may drastically alter certain characteristics of a habitat, thereby changing its HSI and causing a significant change in the number of species. The mutation rate of the habitat is inversely proportional to the species probability in BBO. The mutation rate m_i is calculated by the probability p_i of the species number, and its expression is as follows:

$$m_i = m_{\max}\left(1 - \frac{p_i}{p_{\max}}\right), \tag{4}$$

where $m_{\max}$ is the maximum mutation rate, which is a user-defined parameter, the computation way of p_i refers to [8], and $p_{\max} = \max\,(p_i)$. The mutation can be conducted below:

$$H_i\left(\text{SIV}_j\right) \,\%\!\leftarrow \text{lb}_j + \text{rand} * \left(\text{ub}_j - \text{lb}_j\right), \tag{5}$$

where H_i is the mutation habitat, $j \in [1, D]$ (where D is the decision variable number), lb_j and ub_j are the lower and the upper boundary values of the jth SIV of H_i, respectively, and rand is a uniformly distributed random real number between 0 and 1.

In order to save the best solutions in the search process, the elitism strategy is used to keep some best solutions. At each iteration, after operations such as migration and mutation are carried out, the population is sorted. The

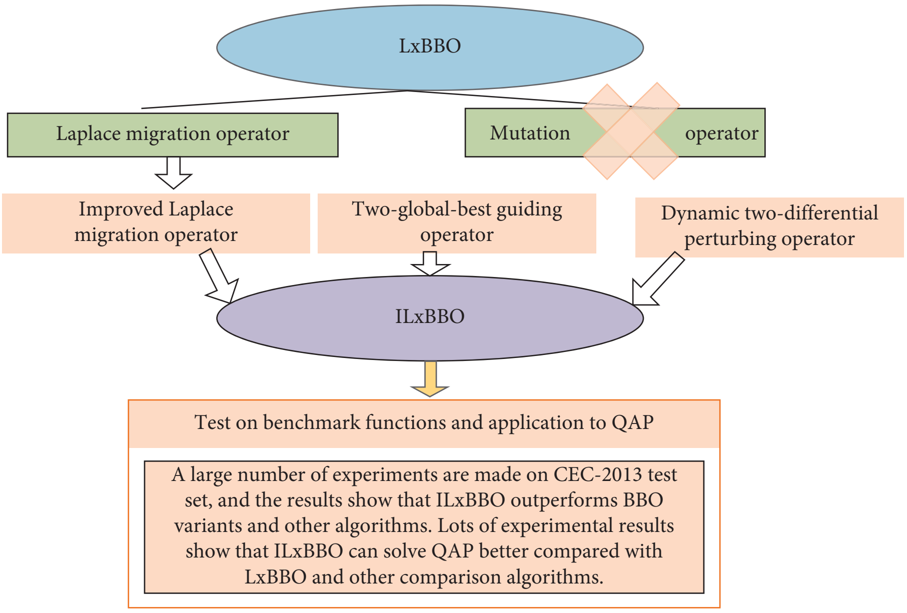

LxBBO
Laplace migration operator
Mutation operator
Improved Laplace migration operator
Two-global-best guiding operator
Dynamic two-differential perturbing operator
ILxBBO
Test on benchmark functions and application to QAP
A large number of experiments are made on CEC-2013 test set, and the results show that ILxBBO outperforms BBO variants and other algorithms. Lots of experimental results show that ILxBBO can solve QAP better compared with LxBBO and other comparison algorithms.

Figure 1: Graphical abstract of this paper.

several worst habitats are replaced with some elitists kept before, and the population is sorted again. The steps of BBO are given as follows:

Step 1: set the parameters and initialize the population randomly

Step 2: evaluate each habitat and sort the population from the best to the worst by their HSIs

Step 3: calculate the immigration, emigration, and mutation rates and keep the elitists

Step 4: perform the migration operator by equation (3)

Step 5: perform the mutation operator by equation (5)

Step 6: limit each new solution's boundary

Step 7: calculate each habitat's HSI and sort the population from the best to the worst

Step 8: replace some worst habitats with the elitists

Step 9: sort the population again from the best to the worst

Step 10: decide whether the stopping criterion is met

Step 11: if it is so, output the best solution; otherwise, return to Step 3

According to the steps of BBO, BBO can get strong local search capability through migration and global search ability by mutation. However, there are some drawbacks in BBO. For example, the migration operator simply provides new solutions through directly copying some features of good solutions and cannot generate new features from new promising areas in the search space. The mutation operator affects the accuracy of the solution in the later search of algorithm and so on.

To remedy BBO's defects, great efforts have been made to improve BBO by many researchers. The improvements are mainly divided into the following aspects. (1) Topology. Zheng et al. [23] used three different topologies, namely, ring topology, square topology, and random topology, to enhance the exploration ability of BBO. Feng et al. [24] presented a hybrid migration operator with random ring topology to enhance the potential population diversity of BBO. (2) Migration operator. Ma and Simon [25] proposed a blended migration operator, in which a new solution consists of two parts, the features of other solutions and its own features. Xiong et al. [18] presented a polyphyletic migration operator to raise the population diversity of BBO, and an orthogonal learning strategy was used to make a systematic and elaborate search. Li et al. [22] presented a perturbed migration in order to enhance the exploration ability and integrated the Gaussian mutation into BBO to improve the exploration. Chen et al. [26] put forward the covariance matrix for reducing the dependence on a coordinate system and enhancing BBO's rotational invariance. Zhang et al. [27] presented an Efficient and Merged BBO (EMBBO) to enhance the optimization efficiency. (3) Mutation operator. Gong et al. [28] embedded Gaussian, Cauchy, and Levy mutations into BBO, respectively. Lohokare et al. [29] proposed a mutation operator which combined two individuals to generate a new feasible solution to improve the exploration ability. (4) Hybrid BBO with other IOAs. Gong et al. [30] combined the exploration of DE with the exploitation of BBO to enhance the performance. Savsani et al. [31] integrated BBO with Artificial Immune Algorithm (AIA) and ACO, respectively, and proposed four mixed BBOs. Khademi et al. [32] combined the Invasive Weed Optimization (IWO) with BBO to enhance the performance of BBO. Zhang et al. [33] combined BBO and GWO to obtain a BBO with strong universal applicability. In addition, many improvements are from not only one way but many ones to maximize the performance of BBO at present. Therefore, further improvements to the BBO variant are necessary.

2.2. Laplacian Biogeography-Based Optimization. Garg and Deep proposed LxBBO based on the Laplace crossover to improve the optimization performance of BBO [21]. The Laplace crossover is described as follows. There are two parents, x_1 the immigration habitat,x_2 the emigration habitat that is selected by the roulette wheel selection. A random number (β) is generated, and it follows Laplace distribution, which is given by the following equation:

$$\beta = \begin{cases} a - b * \log(u), & u \le 0.5, \\ a + b * \log(u), & u > 0.5, \end{cases} \tag{6}$$

where $a \in R$ is called the location parameter and $b > 0$ is called the scale parameter. Then, two new offsprings are generated as

$$y_1 = x_1 + \beta(x_1 - x_2), \tag{7}$$

$$y_2 = x_2 + \beta(x_1 - x_2). \tag{8}$$

The two new habitats are blended to make a new habitat y (see equation (9)) with a blended parameter γ given by equation (10):

$$y = \gamma * y_1 + (1 - \gamma) * y_2, \tag{9}$$

$$\gamma = \gamma_{\min} + (\gamma_{\max} - \gamma_{\min})^{k^t}, \tag{10}$$

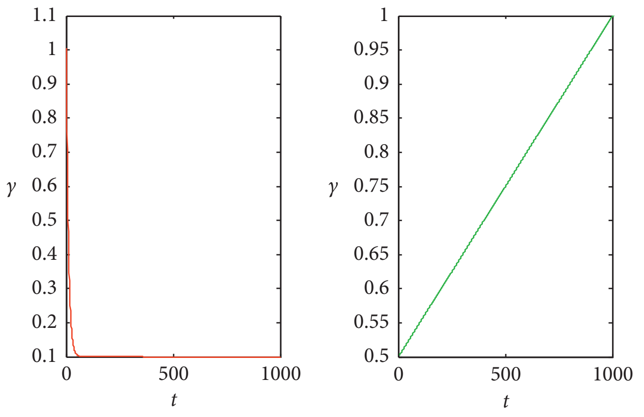
where $\gamma_{\min}$ and $\gamma_{\max}$ are two parameters which are the minimum and maximum values of γ, respectively, and these two parameters lie in [0, 1], and t is the current iteration number. k is a user-defined parameter and less than 1.

From equations (7) and (8), the difference between the two equations lies in the first item of them; the first term of equation (7) is x_1, while the first one of equation (8) is x_2. The value of γ is getting smaller as t increases in equation (10) (see Figure 2(a)). From equation (10), in the earlier search phase, the value of γ is larger, so y is mostly affected by the offspring y_1; the difference of x_1 and x_2 is larger, and the search range around x_1 is larger, so the algorithm has stronger exploration. In the later search phase, the value of γ is smaller, so y is mostly affected by the offspring y_2; the difference between x_1 and x_2 is smaller, the search range around x_2 (good position) is smaller, and the algorithm has stronger exploitation.

In LxBBO, except the migration operator is replaced by the Laplace operator, the rest is the same as in BBO, for example, both use the mutation and the elitist strategy. LxBBO uses the immigration and emigration habitats as a parent to generate two offsprings and then get a new habitat. This way can generate new positions from new promising areas in the search space to enhance the optimization performance of BBO.

3. Improved Laplacian Biogeography-Based Optimization

3.1. Defects of Laplacian Biogeography-Based Optimization. Although LxBBO enhances the performance of BBO largely, there are some drawbacks. (1) In the Laplacian operator, there are many parameters. From equations (6) to (10), a, b, $\gamma_{\min}$, $\gamma_{\max}$, and k need tedious tuning in various applications. In the experiment, according to [21], k is set to 0.95, and the power of the difference between the maximum value and the minimum value is calculated at each iteration, which results in some computation cost. (2) LxBBO and BBO share some of the drawbacks. For example, when the emigration habitats are selected by the roulette wheel selection, a poor habitat may be chosen to share its information with a good habitat to reduce the quality of the population and the roulette wheel selection has high calculation complexity. (3) Both BBO and LxBBO use the mutation operation. Although the mutation operator can enhance the global exploration ability, it is possible to mutate some better habitats and destroy them, causing population degradation and affecting the convergence quality especially in the late search phase. (4) The mutation operator needs to compute the mutation rate of each habitat and complete the mutation operation. (5) The elitist strategy is used, and the population needs to sort twice at each iteration, which results in high computation complexity. In order to solve the above drawbacks, several creative improvements are brought up in this paper.



(a)

(b)

Figure 2: γ with the iteration number t.

3.2. Improved Laplace Operator. To enhance the performance and operability of LxBBO, an improved Laplace operator is proposed.

Inspired by the optimal idea of [34], when an individual (namely, H_e) with better fitness subtracts an individual with poor fitness (namely, H_k) (see equations (11) and (12)), it will search toward the good individual to accelerate the convergence speed. The random number β is calculated by equation (13), and the two habitats H_1 and H_2 are generated by equations (11) and (12). Compared with equations (7) and (8), the difference is that the emigration habitats of equations (11) and (12) are selected by the example learning selection [33]. H_e has a better fitness value than H_k does, using the difference between H_e and H_k to ensure that the search direction is closer to the better solution, and the convergence quality is improved:

$$H_1 = H_k + \beta * (H_e - H_k), \tag{11}$$

$$H_2 = H_e + \beta * (H_e - H_k), \tag{12}$$

$$\beta = \begin{cases} -0.5 * \log(u), & u \leq 0.5, \\ +0.5 * \log(u), & u > 0.5. \end{cases} \tag{13}$$

There are many parameters to be set and much complexity in equation (10), so a new dynamic weight parameter γ is adopted. It is expressed as equation (14), and the difference of γ in equations (10) and (14) is shown in Figure 2. Figure 2(a) represents the curve of r in LxBBO, and Figure 2(b) represents the curve of r in ILxBBO. From Figure 2, γ is getting smaller as t increases, and it keeps an almost constant value (0.1) after about 100 iterations in LxBBO. However, γ increases linearly with t in ILxBBO, when $t = 0$ and $\gamma = 0.5$, and when $t = \text{MaxDT}$, $\gamma = 1$, so γ is a linear number between 0.5 and 1. No parameters need tuning and that makes ILxBBO get stronger operability by using the dynamic weight γ:

$$\gamma = 0.5 + 0.5 * \left(\frac{t}{\text{MaxDT}}\right), \tag{14}$$

where MaxDT is the maximum iteration number. H is given by the following equation:

$$H = \gamma * H_1 + (1 - \gamma) * H_2. \tag{15}$$

From equation (15), in the earlier stage, H accepts the information from H_1 and H_2 to increase the diversity. In the later stage, H is more affected by H_2 to enhance the exploitation and the convergence speed.

3.3. Improved Migration Operator. In LxBBO, the emigration habitats are selected by the roulette wheel selection like BBO, and good solutions may get the features from poor solutions, so the population may be degenerated. In order to overcome the drawbacks of the roulette wheel selection, the example learning selection [33] is adopted to replace the roulette wheel selection. The selection approach is described as follows. When the population is sorted from the best to the worst, the k-th habitat has a higher value than the one behind the k-th habitat does. When H_k is selected for migration, there are $k - 1$ habitats for it to learn from. The emigration habitat H_e is calculated as follows:

$$H_e = \text{ceil}((k - 1) * \text{rand}, \tag{16}$$

where ceil () is the function which rounds toward positive infinity. From equation (16), H_e is not worse than H_k, which improves the quality of the solution by sharing features from good solutions. Furthermore, ILxBBO only needs to calculate the immigration rate and does not need to calculate the emigration rate, which reduces the computation complexity further. The emigration habitat can be selected by only equation (16). The example learning selection overcomes the defects of the roulette wheel selection, and its calculation is simple to reduce the computational complexity. The improved Laplace operator and the improved migration operator together form an improved Laplace migration operator.

3.4. Dynamic Two-Differential Perturbing Operator. DE, proposed by Storn and Price in 1997 [3], is a popular optimization algorithm. It generates temporary individuals based on the degree of individual difference in the population and realizes evolution by random recombination. In the earlier search period, the individuals have more difference in the population, and the algorithm can search in a large range, thus obtaining global exploration ability. In the later search period, the algorithm searches in the vicinity of the individual and obtains local search ability [35]. A dynamic differential evolution algorithm is proposed by Wu et al. [36], and the experiments show that the dynamic method has better performance.

LxBBO can get a global exploration ability by its mutation operator. However, the mutation operation is at random, and it is easy to destroy the better solutions, especially in the late search stage. In addition, the mutation operation needs to calculate the mutation rate of each habitat, along with its own calculation, and those increase the computation complexity. So the mutation operator is removed in ILxBBO. In Section 3.3, the emigration habitats are selected by the example learning approach and poor habitats can accept their features from good habitats. However, some best habitats cannot be updated because there are few examples for them to learn from leading to low search efficiency. Although the best habitat may be the example of the second habitat, there is little difference between the two habitats in many cases, the second one is almost unchanged, and the search efficiency is also low. So a dynamic two-differential perturbing operator is adopted in the best habitat and the second best habitat to enhance the search ability. The value of the scaling factor (w) of the dynamic two-differential perturbing operator decreases with the increase in the current iteration number, which is expressed as follows:

$$w = 1 - \frac{t}{\text{MaxDT}}. \tag{17}$$

The dynamic two-differential perturbing operator is expressed as follows:

$$H_k = H_k + w * (H_b - H_k + H_m - H_r), \tag{18}$$

where H_k is the best habitat or the second best habitat, H_r and H_m are two habitats selected randomly from the current population, and they satisfy $m \neq k \neq r$. In the early earlier stage, the values of $(H_m - H_r)$ and $(H_b - H_k)$ and the value of w are all comparatively large, so H_k searches in a larger range to enhance the global search ability. The values of $(H_m - H_r)$ and $(H_b - H_k)$ and the value of w are comparatively small in the late stage, so H_k searches in a smaller range to enhance the local search ability. From equation (18), H_k is affected by itself, the dynamic coefficient factor, and the two differences. Therefore, it is called a dynamic two-differential perturbing operator.

3.5. Two-Global-Best Guiding Operator. In order to further enhance the local search ability, the best and subbest habitats are used to guide the worst habitat to update. In the first half of the search, the mathematical equation is expressed as follows:

$$\begin{aligned} H_w = H_w + 2 * (\text{rand} - 0.5) * \left(H_b - H_w\right) \\ + 2 * (0.5 - \text{rand}) * \left(H_s - H_w\right), \end{aligned} \tag{19}$$

where H_s is the subbest habitat. In the second half of the search, equations (20) and (21) are used to update the worst habitat based on the same probability:

$$\begin{aligned} H_w = H_w + 2 * (0.5 - \text{rand}) * \left(H_b - H_w\right) \\ + \text{rand} * \left(H_s - H_w\right), \end{aligned} \tag{20}$$

$$H_w = H_w + (0.5 + 0.5 * \text{rand})\left(H_b - H_w + H_s - X_w\right). \tag{21}$$

From equation (19), the worst habitat is affected by three aspects: the worst habitat itself, the random weighted (−1~1) difference between the best habitat and the worst habitat, and the random weighted (−1~1) difference between the subbest habitat and the worst. In the early search stage, the difference between the best habitat or the subbest habitat and the worst habitat is bigger. The range of $2 * (\text{rand} - 0.5)$ is from −1 to 1. So the worst habitat searches around the wider range of itself to obtain some global search abilities. In the late stage, the worst habitat adopts equation (20) or equation (21) to update. They are similar to equation (19) but with different random weights. Under the guidance of the best and the subbest habitats, the worst habitat can obtain a local search ability by searching in a small range around itself. From equations (19)–(21), H_w is all affected by the best habitat and the subbest habitat, so this operator is called two-global-best guiding operator.

3.6. Other Improvements. In addition, the following improvements are also adopted. First, the greedy selection [24, 37] replaces the elitism strategy [8]. So, on the one hand, the population is just needed to sort once at each iteration to reduce computing complex; on the other hand, the greedy selection avoids setting the elitist parameter. Second, the immigration rate calculation step is moved outside of the iteration loop. It means the immigration rate is calculated once in the whole iterations to reduce the computation complexity. The flowchart of ILxBBO is given in Figure 3.

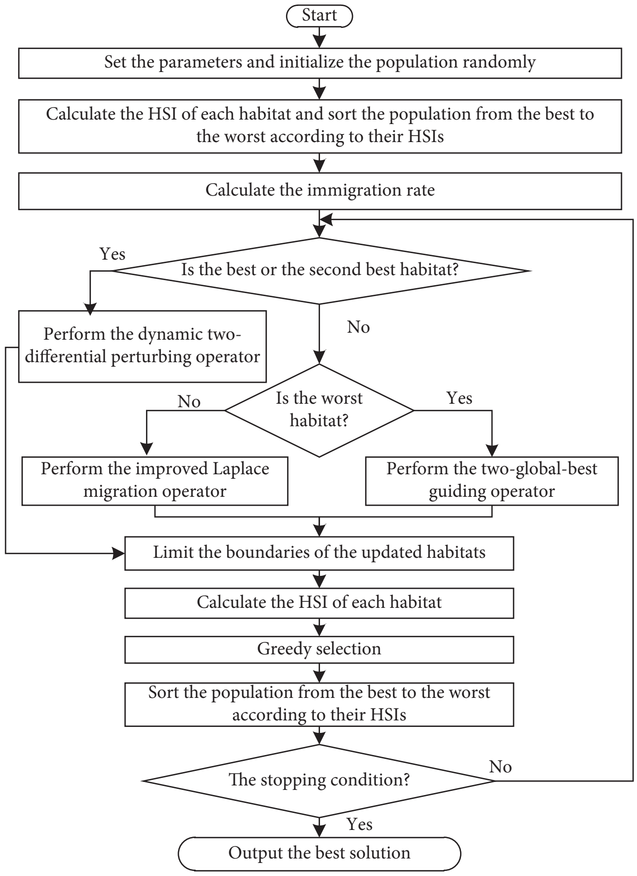

Start
Set the parameters and initialize the population randomly
Calculate the HSI of each habitat and sort the population from the best to the worst according to their HSIs
Calculate the immigration rate
Yes
Is the best or the second best habitat?
Perform the dynamic two-differential perturbing operator
No
No
Is the worst habitat?
Yes
Perform the improved Laplace migration operator
Perform the two-global-best guiding operator
Limit the boundaries of the updated habitats
Calculate the HSI of each habitat
Greedy selection
Sort the population from the best to the worst according to their HSIs
The stopping condition?
No
Yes
Output the best solution

Figure 3: Flowchart of ILxBBO.

From the above description, there are the following differences between ILxBBO and LxBBO. (1) In ILxBBO, the mutation operator is removed to omit calculating the mutation probability and the mutation operation to reduce the computation complexity. (2) The best habitat and the subbest habitat adopt a dynamic two-differential perturbing operator to update in ILxBBO, while the first two best habitats use the Laplace operator to update in LxBBO. (3) The worst habitat uses a two-global-best guiding operator to update, while the worst habitat also uses the Laplace operator in LxBBO. (4) The remaining habitats use an improved Laplacian migration operator in ILxBBO, while the Laplace operator is also used in LxBBO. (5) There are few parameters to be tuned in ILxBBO, while LxBBO has many parameters to be tuned in various applications. (6) LxBBO uses the roulette wheel selection, while the emigration habitat is selected by the example learning approach in ILxBBO. The population tends to be in the best direction, the example learning approach does not require calculating the migration rate, and the emigration habitat can be selected with only a small amount of calculation. (7) The elitist strategy is adopted in LxBBO, while the greedy selection is used instead of the elitist strategy in ILxBBO. It saves one sorting step to further reduce the computation complexity.

4. Experiment Results and Analysis

4.1. Experiment Preparation. In order to verify ILxBBO, a large number of experiments are made on the complex functions from the CEC-2013 test set [38], where f_1–f_5 are unimodal functions, f_6–f_{20} are basic multimodal functions, and f_{21}–f_{28} are composition functions. All the experiments are implemented on PC with 3.1 GHz CPU and 4 GB RAM under a Microsoft Windows 7 operating system. The programming language is MATLAB R2014a.

To be fair, in all the experiments on the CEC-2013 test set, according to [38], the independent run number (Run) is 51, MaxDT dynamically adjusts based on the dimension of the problem, with 30-D and 50-D, MaxDT is 3000 and 5000, respectively, and maximum number of function evaluations ($\mathrm{MaxNFE_S}$) is set to $D * 10{,}000$. The population size (N) is 100 in LxBBO and ILxBBO. The maximum immigration rate I of ILxBBO is 1 like LxBBO. According to [21], the parameters used in LxBBO are as follows: the maximum emigration/immigration rate $(E/I) = 1$, $m_{\max} = 0.005$, $\gamma_{\min} = 0.1$, $\gamma_{\max} = 1$, $a = 0$, $b = 0.5$, and $k = 0.95$.

We evaluate the statistical average (Mean) and standard deviation (Std) of these algorithms after some independent runs. For an algorithm, Mean represents its optimization ability and Std embodies its stability. The ranking criteria are described below: first, we compare each algorithm's mean value on each function; the better the mean value is, and the better the ranking is. If some algorithms obtain the same mean values, then we compare their Std values; the better the Std value is, the better the ranking is. If some algorithms obtain the same Mean and Std values, their rankings are considered to be the same. In addition, the best values are in bold in all the result tables.

4.2. Comparison of ILxBBO with Its Incomplete Variants. To illustrate the effectiveness of each component of ILxBBO, ILxBBO is only compared with its incomplete variants and LxBBO on the 30-dimensional functions. These incomplete variants are described as follows:

GLxBBO is an incomplete variant of ILxBBO, which is LxBBO with only the two-global-best guiding operator, without the dynamic two-differential perturbing operator and improved Laplace migration operator

DLxBBO is an incomplete variant of ILxBBO, which is LxBBO with only the dynamic two-differential perturbing operator without the improved Laplace migration operator and two-global-best guiding operator

OLxBBO is an incomplete variant of ILxBBO, which contains the improved Laplace migration operator without the dynamic two-differential perturbing operator and two-global-best guiding operator

The experimental results are shown in Table 1. Form Table 1, ILxBBO obtains 14 times ranking the first. LxBBO obtains 4 times ranking the first. GLxBBO obtains 9 times ranking the first. DLxBBO and OLxBBO obtain 1 time ranking the first. ILxBBO's, LxBBO's, GLxBBO's, DLxBBO's and OLxBBO's average rankings are 1.79, 4.11, 2.32, 3.14, and 3.36, respectively. ILxBBO's average ranking is significantly higher than those of its incomplete algorithms. The average ranking of LxBBO is the last. It shows that each improvement on LxBBO is effective and the two-global-best guiding strategy contributes most to ILxBBO. The average ranking of ILxBBO is the first, and it shows that every improvement in LxBBO is essential.

4.3. Comparison with BBO's Variants. In this experiment group, ILxBBO is compared with quite a few state-of-the-art BBO's variants on the 30-dimensional and 50-dimensional functions from the CEC-2013 test set. The comparison algorithms include TDBBO [39], BIBBO [25], BBOM [40], DEBBO [30], BLPSO [41], PRBBO [24], WRBBO [37], EMBBO [27], and BHCS [42]. These algorithms are all BBO variants proposed in recent years, with much comparability. Their common parameter settings are the same as those of ILxBBO, and other parameter settings are referred to their corresponding references. The experimental results on the 30-dimensional functions and 50-dimensional functions are shown in Tables 2 and 3, respectively.

From Table 2, ILxBBO obtains 9 times ranking the first and the optimal value (0) on f_1 is obtained by ILxBBO. TDBBO obtains 1 time ranking the first. BIBBO, BLSPO, and BHCS obtain 0 times ranking the first. DEBBO obtains 4 times ranking the first, and both WRBBO and EMBBO obtain 5 times ranking the first. On the 5 unimodal functions, ILxBBO obtains 2 times ranking the first and 2 times ranking the second, and it shows that the improved migration operator and the two-global-best perturbing operator enhance the local search ability. On the 15 basic multimodal functions, ILxBBO obtains 6 times ranking the first and 3 times ranking the second, and it shows that the dynamic two-differential perturbing operator enhances the global search ability. On the 8 composition functions, ILxBBO obtains 1 time ranking the first. On average ranking, ILxBBO's average ranking (2.36) is significantly higher than those of the comparison algorithms. These comparison results show that, in general, ILxBBO obtains the most significant optimization performance among these 9 optimization algorithms.

From Table 3, ILxBBO obtains 10 times ranking the first and gets the first average ranking (2.14). Its performance is just as good as that on the 30-dimensional functions.

4.4. Comparison with Other IOAs. To further verify the effectiveness of ILxBBO, ILxBBO is compared with other IOAs on the 30-dimensional functions. These algorithms include YYPO [43], DPCABC [44], DFnABC [45], FMPSO [46], GLPSO [47], HFPSO [48], and MEGWO [49], which are the latest and most famous IOAs and are competitive and representative. DPCABC and DFnABC are ABC variants, and MEGWO is a GWO variant. FMPSO is a PSO variant, and GLPSO and HFPSO are PSO hybrid algorithms with Genetic Algorithm (GA) and Firefly Algorithm (FA), respectively. For comparison algorithms, on the 30-dimensional functions, MaxFEs is 300,000, and Run is 51(except FMPSO is 30). The other parameters settings are referred to their corresponding references. The data of these comparison algorithms are directly from their corresponding to references. The experimental results are shown in Table 4.

From Table 4, ILxBBO obtains 9 times ranking the first, YYPO obtains 2 times ranking the first, DPCABC obtains 4 times ranking the first, DFnABC and GLPSO obtain 7 times ranking the first, FMPSO obtains 3 times ranking the first, HFPSO obtains 1 time ranking the first, and MEGWO obtains 0 times ranking the first. The average rankings of IlxBBO (2.71) is the first; the next is GLPSO, DFnABC, YYPO, FMPSO, DPCABC, HFPSO, and HFPSO. Therefore,

Table 1: Comparison results with incomplete variants and LxBBO on the 30-dimensional functions.

Function	ILxBBO			LxBBO			GLxBBO			DLxBBO			OLxBBO		
	Mean	Std	Rank	Mean	Std	Rank	Mean	Std	Rank	Mean	Std	Rank	Mean	Std	Rank
f_1	**0**	**0**	**1**	1.68e−06	8.52e−07	5	**0**	**0**	**1**	**0**	**0**	**1**	**0**	**0**	**1**
f_2	**2.00e+05**	**1.04e+05**	**1**	3.04e+06	1.09e+06	4	1.22e+06	7.68e+05	3	3.99e+05	2.05e+05	2	5.81e+06	2.53e+06	5
f_3	**1.09e+06**	**1.58e+06**	**1**	3.04e+08	3.52e+08	5	4.23e+06	4.07e+06	3	2.07e+06	2.15e+06	2	9.41e+06	9.30e+06	4
f_4	**5.44e+00**	**3.83e+00**	**1**	2.26e+03	1.27e+03	4	2.09e+03	1.03e+03	3	1.42e+01	5.01e+01	2	2.56e+04	2.70e+03	5
f_5	**9.14e−14**	**4.56e−14**	**1**	3.10e−04	1.50e−04	5	1.09e−13	2.23e−14	3	1.09e−13	2.23e−14	3	1.03e−13	3.41e−14	2
f_6	**1.61e+01**	**1.40e+01**	**1**	5.73e+01	2.64e+01	5	2.43e+01	2.14e+01	3	2.16e+01	1.73e+01	2	3.21e+01	2.52e+01	4
f_7	**1.10e+01**	**6.89e+00**	**1**	6.54e+01	1.90e+01	5	1.56e+01	6.84e+00	3	1.24e+01	6.04e+00	2	1.72e+01	7.01e+00	4
f_8	2.09e+01	5.64e−02	4	2.09e+01	6.83e−02	5	**2.09e+01**	**3.83e−02**	**1**	2.09e+01	5.05e−02	3	2.09e+01	4.51e−02	2
f_9	1.87e+01	3.76e+00	2	2.40e+01	4.06e+00	3	**1.78e+01**	**4.28e+00**	**1**	2.73e+01	1.51e+00	4	2.78e+01	1.09e+00	5
f_{10}	2.22e−01	1.35e−01	2	2.71e−01	1.06e−01	3	3.22e−01	1.39e−01	4	**2.16e−01**	**9.46e−02**	**1**	3.53e−01	1.59e−01	5
f_{11}	6.05e−01	6.92e−01	3	**1.98e−02**	**1.39e−01**	**1**	7.61e−01	1.01e+00	5	5.46e−01	6.39e−01	2	7.41e−01	7.67e−01	4
f_{12}	2.73e+01	7.36e+00	2	7.13e+01	1.77e+01	5	**2.67e+01**	**6.49e+00**	**1**	3.57e+01	4.86e+00	4	3.57e+01	3.94e+00	3
f_{13}	**5.51e+01**	**1.86e+01**	**1**	1.36e+02	3.45e+01	5	5.77e+01	1.63e+01	2	6.54e+01	1.03e+01	3	6.62e+01	7.84e+00	4
f_{14}	4.38e+01	4.06e+01	2	**1.46e+01**	**6.32e+00**	**1**	4.70e+01	5.12e+01	4	5.09e+01	4.15e+01	5	4.55e+01	4.26e+01	3
f_{15}	3.03e+03	5.70e+02	2	4.00e+03	7.73e+02	3	**2.95e+03**	**5.10e+02**	**1**	4.17e+03	2.16e+02	5	4.04e+03	2.19e+02	4
f_{16}	1.66e+00	2.17e−01	4	**1.42e+00**	**3.58e−01**	**1**	1.51e+00	2.40e−01	2	1.70e+00	2.22e−01	5	1.64e+00	1.59e−01	3
f_{17}	3.28e+01	5.61e−01	3	3.83e+01	2.31e+00	5	**3.24e+01**	**5.04e−01**	**1**	3.32e+01	6.57e−01	4	3.28e+01	4.94e−01	2
f_{18}	7.19e+01	1.61e+01	2	9.78e+01	2.16e+01	5	**6.55e+01**	**1.25e+01**	**1**	1.03e+02	6.06e+00	4	1.01e+02	6.33e+00	3
f_{19}	1.63e+00	2.81e−01	2	2.51e+00	6.66e−01	5	**1.57e+00**	**2.54e−01**	**1**	2.05e+00	1.63e−01	4	1.89e+00	1.89e−01	3
f_{20}	**1.02e+01**	**4.98e−01**	**1**	1.37e+01	8.10e−01	5	1.03e+01	4.84e−01	2	1.08e+01	3.48e−01	3	1.09e+01	3.88e−01	4
f_{21}	**3.12e+02**	**4.59e+01**	**1**	3.23e+02	6.91e+01	5	3.20e+02	6.71e+01	3	3.21e+02	6.03e+01	4	3.17e+02	6.01e+01	2
f_{22}	1.24e+02	7.47e+00	2	1.29e+02	4.48e+01	5	**1.22e+02**	**6.60e+00**	**1**	1.27e+02	7.08e+00	4	1.26e+02	8.69e+00	3
f_{23}	**3.28e+03**	**5.44e+02**	**1**	5.03e+03	8.30e+02	5	3.39e+03	5.72e+02	2	4.43e+03	3.72e+02	4	4.22e+03	3.72e+02	3
f_{24}	2.08e+02	3.55e+00	3	2.43e+02	1.23e+01	5	2.09e+02	3.76e+00	4	2.08e+02	3.29e+00	2	**2.09e+02**	**2.09e+02**	**1**
f_{25}	**2.59e+02**	**1.30e+01**	**1**	2.87e+02	1.15e+01	5	2.60e+02	1.09e+01	2	2.68e+02	1.55e+01	3	2.71e+02	1.55e+01	4
f_{26}	**2.00e+02**	**1.76e−02**	**1**	2.09e+02	3.41e+01	4	2.02e+02	1.48e+01	3	2.00e+02	3.84e−01	2	2.09e+02	2.26e+02	5
f_{27}	**3.76e+02**	**3.54e+01**	**1**	8.30e+02	1.34e+02	5	3.80e+02	3.24+01	2	3.82e+02	3.05e+01	3	3.92e+02	2.95e+01	4
f_{28}	3.00e+02	1.77e−13	3	**2.96e+02**	**2.80e+01**	**1**	3.00e+02	1.77e−13	3	3.00e+02	1.72e−13	5	3.00e+02	2.42e−13	2
Count		14			4			9			2			2	
Ave. Rank		1.79			4.11			2.32			3.14			3.36	
Total. Rank		1			5			2			3			4	

Table 2: Comparison results among ILxBBO and BBO's variants on the 30-dimensional functions.

Function	Value	TDBBO	BIBBO	BBOM	DEBBO	BLPSO	WRBBO	EMBBO	BHCS	ILxBBO
f_1	Mean	1.83e − 13	4.82e − 01	9.31e − 04	2.23e − 14	3.57e − 13	**0**	3.12e − 14	3.21e − 13	**0**
	Std	9.12e − 14	1.98e − 01	1.30e − 04	6.83e − 14	1.46e − 13	**0**	7.90e − 14	1.30e − 13	**0**
	Rank	5	9	8	3	7	**1**	4	6	**1**
f_2	Mean	**1.13e + 05**	2.50e + 07	8.02e + 06	2.08e + 07	7.40e + 05	1.01e + 07	1.42e + 07	2.22e + 05	2.00e + 05
	Std	**6.23e + 04**	8.99e + 06	2.30e + 06	6.79e + 06	3.65e + 05	3.76e + 06	3.57e + 06	1.29e + 05	1.04e + 05
	Rank	**1**	9	5	8	4	6	7	3	2
f_3	Mean	1.94e + 06	2.46e + e + 09	2.76e + 08	**1.60e + 05**	7.16e + 08	8.85e + 05	1.29e + 09	1.02e + 08	1.09e + 06
	Std	4.80e + 06	2.65e + 09	1.25e + 08	**2.83e + 05**	1.37e + 09	1.56e + 06	5.99e + 08	1.89e + 08	1.58e + 06
	Rank	4	9	6	**1**	7	2	8	5	3
f_4	Mean	2.22e + 01	7.42e + 04	5.57e + 04	3.15e + 04	3.13e + 03	9.49e + 03	5.21e + 04	1.71e + 02	**5.44e + 00**
	Std	1.45e + 01	1.96e + 04	5.30e + 03	4.49e + 03	1.04e + 03	1.76e + 03	6.49e + 03	4.97e − 13	**3.83e + 00**
	Rank	2	9	8	6	4	5	7	3	**1**
f_5	Mean	3.46e − 03	3.93e − 01	5.95e − 03	1.14e − 13	2.29e + 00	**8.69e − 14**	1.14e − 13	4.97e − 13	9.14e − 14
	Std	2.47e − 02	1.19e − 01	6.85e − 04	0	1.27e + 01	**4.87e − 14**	0	1.22e − 13	4.56e − 14
	Rank	6	8	7	3	9	**1**	3	5	2
f_6	Mean	3.88e + 01	4.98e + 01	1.65e + 01	**1.50e + 01**	4.87e + 01	1.91e + 01	1.95e + 01	1.50e + 01	1.61e + 01
	Std	2.71e + 01	2.71e + 01	1.92e + 00	**1.49e − 01**	2.65e + 01	1.19e + 01	3.85e + 00	1.65e + 01	1.40e + 01
	Rank	7	9	4	**1**	8	5	6	2	3
f_7	Mean	5.48e + 00	1.09e + 02	5.95e + 01	4.77e + 00	9.10e + 01	**5.33e − 01**	1.01e + 02	9.53e + 01	1.10e + 01
	Std	4.90e + 00	2.47e + 01	4.70e + 00	1.95e + 00	2.47e + 01	**4.39e − 01**	1.11e + 01	4.36e + 01	6.89e + 00
	Rank	3	9	5	2	6	**1**	8	7	4
f_8	Mean	2.10e + 01	2.10e + 01	2.09e + 01	2.09e + 01	2.09e + 01	2.09e + 01	2.09e + 01	2.10e + 01	**2.09e + 01**
	Std	4.42e − 02	5.48e − 02	5.20e − 02	5.09e − 02	6.75e − 02	4.85e − 02	4.57e − 02	5.45e − 02	**5.64e − 02**
	Rank	7	9	5	4	6	3	2	8	**1**
f_9	Mean	2.82e + 01	3.22e + 01	2.72e + 01	2.98e + 01	2.67e + 01	3.75e + 01	2.89e + 01	2.91e + 01	**1.87e + 01**
	Std	1.44e + 00	3.53e + 00	1.18e + 00	1.61e + 00	2.84e + 00	1.54e + 00	1.73e + 00	4.66e + 00	**3.76e + 00**
	Rank	4	8	3	7	2	9	5	6	**1**
f_{10}	Mean	1.92e − 01	1.13e + 01	2.83e + 00	**1.27e − 02**	1.57e + 01	6.15e − 02	6.80e + 00	5.74e − 02	2.22e − 01
	Std	1.08e − 01	3.98e + 00	6.83e − 01	**9.18e − 03**	3.40e + 01	2.86e − 02	1.92e + 00	3.82e − 02	1.35e − 01
	Rank	4	8	6	**1**	9	3	7	2	5
f_{11}	Mean	1.88e + 01	9.31e − 01	1.46e + 01	**1.11e − 14**	2.86e + 02	1.79e + 00	1.45e − 14	6.92e + 01	6.05e − 01
	Std	5.50e + 00	4.59e − 01	1.82e + 00	**2.28e + 00**	5.10e + 01	1.24e + 00	2.50e − 14	2.61e + 01	6.92e − 01
	Rank	7	4	6	**1**	9	5	2	8	3
f_{12}	Mean	4.20e + 01	6.94e + 01	6.04e + 01	8.87e + 01	3.12e + 02	1.64e + 02	1.45e + 02	8.39e + 01	**2.73e + 01**
	Std	9.44e + 00	2.21e + 01	8.89e + 00	1.07e + 01	5.74e + 01	9.65e + 00	1.93e + 01	2.62e + 01	**7.36e + 00**
	Rank	2	4	3	6	9	8	7	5	**1**
f_{13}	Mean	8.10e + 01	1.47e + 02	1.19e + 02	1.19e + 02	4.47e + 02	1.60e + 02	1.77e + 02	1.64e + 02	**5.51e + 01**
	Std	2.15e + 01	3.37e + 01	1.80e + 01	1.22e + 01	7.13e + 01	1.07e + 01	2.27e + 01	3.98e + 01	**1.86e + 01**
	Rank	2	5	4	3	9	6	8	7	**1**
f_{14}	Mean	7.89e + 02	5.35e + 00	3.57e + 02	4.60e + 01	2.27e + 03	2.66e + 02	**4.08e − 03**	5.27e + 03	4.38e + 01
	Std	2.59e + 02	1.80e + 00	8.98e + 01	1.04e + 01	3.71e + 02	1.28e + 02	**8.35e − 03**	7.53e + 02	4.06e + 01
	Rank	7	2	6	4	8	5	**1**	9	3
f_{15}	Mean	3.32e + 03	4.56e + 03	3.21e + 03	5.47e + 03	3.25e + 03	7.23e + 03	3.84e + 03	4.69e + 03	**3.03e + 03**
	Std	4.45e + 02	7.12e + 02	4.26e + 02	3.15e + 02	4.52e + 02	2.48e + 02	2.78e + 02	1.50e + 03	**5.70e + 02**
	Rank	4	6	2	8	3	9	5	7	**1**
f_{16}	Mean	1.04e + 00	1.87e + 00	8.23e − 01	2.17e + 00	2.15e − 01	2.43e + 00	1.21e + 00	2.40e + 00	1.66e + 00
	Std	2.25e − 01	4.84e − 01	1.84e − 01	3.05e − 01	8.60e − 02	2.72e − 01	1.89e − 01	5.00e − 01	2.17e − 01
	Rank	3	6	2	7	1	9	4	8	5
f_{17}	Mean	5.75e + 01	3.45e + 01	6.17e + 02	3.39e + 01	2.02e + 02	4.98e + 01	**3.04e + 01**	1.44e + 02	3.28e + 01
	Std	6.34e + 00	8.26e − 01	4.35e + 00	4.70e − 01	2.99e + 01	2.78e + 00	**1.83e − 06**	3.69e + 01	5.61e − 01
	Rank	6	4	9	3	8	5	**1**	7	2
f_{18}	Mean	8.38e + 01	1.90e + 02	1.33e + 02	1.77e + 02	2.42e + 02	1.95e + 02	2.05e + 02	1.49e + 02	**7.19e + 01**
	Std	8.07e + 00	2.64e + 01	1.62e + 01	1.13e + 01	5.25e + 01	9.97e + 00	1.68e + 01	4.78e + 01	**1.61e + 01**
	Rank	2	6	3	5	9	7	8	4	**1**
f_{19}	Mean	4.34e + 00	1.79e + 00	3.29e + 00	2.69e + 00	4.66e + 00	1.18e + 01	**6.81e − 01**	7.40e + 00	1.63e + 00
	Std	1.11e + 00	4.59e − 01	4.38e − 01	1.77e − 01	1.20e + 00	8.87e − 01	**1.56e − 01**	3.57e + 00	2.81e − 01
	Rank	6	3	5	4	7	9	**1**	8	2
f_{20}	Mean	**1.02e + 01**	1.37e + 01	1.30e + 01	1.17e + 01	1.45e + 01	1.19e + 01	1.25e + 01	1.27e + 01	1.02e + 01
	Std	**4.29e − 01**	7.39e − 01	1.48e + 01	2.93e − 01	2.17e − 01	2.51e − 01	4.19e − 01	8.15e − 01	4.98e − 01
	Rank	**1**	8	7	3	9	4	5	6	2

Table 2: Continued.

Function	Value	TDBBO	BIBBO	BBOM	DEBBO	BLPSO	WRBBO	EMBBO	BHCS	ILxBBO
f_{21}	Mean	3.41e+02	3.14e+02	2.42e+02	2.63e+02	3.17e+02	2.92e+02	**2.18e+02**	3.14e+02	3.12e+02
	Std	8.19e+01	7.25e+01	8.10e+01	4.88e+01	9.09e+01	7.50e+01	**3.42e+01**	7.82e+01	4.59e+01
	Rank	9	6	2	3	8	4	**1**	7	5
f_{22}	Mean	6.57e+02	1.10e+02	3.86e+02	1.60e+02	3.04e+03	1.49e+02	**2.99e+01**	4.51e+03	1.24e+02
	Std	1.71e+02	3.91e+01	7.74e+01	3.00e+01	5.73e+02	2.70e+01	**2.07e+01**	1.45e+03	7.47e+00
	Rank	7	2	6	5	8	4	**1**	9	3
f_{23}	Mean	3.50e+03	5.25e+03	3.66e+03	5.63e+03	4.11e+03	7.28e+03	4.67e+03	5.70e+03	**3.28e+03**
	Std	5.13e+02	1.15e+03	4.74e+02	3.67e+02	5.74e+02	2.79e+02	4.21e+02	1.24e+03	**5.44e+02**
	Rank	2	6	3	7	4	9	5	8	**1**
f_{24}	Mean	2.14e+02	2.76e+02	2.63e+02	2.01e+02	2.85e+02	**2.00e+02**	2.83e+02	2.96e+02	2.08e+02
	Std	5.62e+00	8.82e+00	3.77e+00	3.95e−01	1.44e+01	**3.88e−01**	4.54e+00	1.33e+01	3.55e+00
	Rank	4	6	5	2	8	**1**	7	9	3
f_{25}	Mean	**2.44e+02**	2.94e+02	2.96e+02	2.50e+02	3.31e+02	2.86e+02	2.98e+02	3.08e+02	2.59e+02
	Std	**2.47e+01**	8.21e+00	3.61e+00	1.13e+01	1.72e+01	2.48e+01	5.48e+00	1.26e+01	1.30e+01
	Rank	**1**	5	6	2	9	4	7	8	3
f_{26}	Mean	2.04e+02	2.62e+02	2.00e+02	2.04e+02	2.19e+02	2.00e+02	2.01e+02	3.34e+02	2.00e+02
	Std	2.23e+01	8.34e+01	1.29e−01	2.25e+01	5.11e+01	1.99e−01	1.67e−01	7.19e+01	1.76e−02
	Rank	5	8	1	6	7	3	4	9	2
f_{27}	Mean	4.08e+02	1.08e+03	1.01e+03	7.61e+02	1.04e+03	**3.57e+02**	4.67e+02	1.06e+03	3.76e+02
	Std	4.48e+01	1.05e+02	3.08e+01	2.41e+02	1.09e+02	**1.38e+02**	2.04e+02	1.09e+02	3.54e+01
	Rank	3	9	6	5	7	**1**	4	8	2
f_{28}	Mean	3.00e+02	4.55e+02	**1.59e+02**	3.00e+02	2.76e+03	3.00e+02	2.63e+02	3.76e+02	3.00e+02
	Std	3.10e−13	3.52e+02	**8.49e+01**	2.40e−13	9.67e+02	0	7.67e+01	4.07e+02	1.77e−13
	Rank	6	8	**1**	5	9	3	2	7	4
Count		3	0	1	4	0	5	5	0	9
Ave. Rank		4.29	6.61	4.79	4.11	6.93	4.71	4.64	6.43	2.36
Total. Rank		3	8	6	2	9	5	4	7	1

this further verifies the optimization performance of ILxBBO.

4.5. Convergence Analysis. In order to highlight the difference between ILxBBO and LxBBO on convergence, Figure 4 shows the convergence curves of ILxBBO and LxBBO only on the 10-dimensional functions. To illustrate the point concisely, some representative convergence graphs are plotted on unimodal (f_2 and f_3), multimodal (f_6, f_9, f_{11}, f_{13}, f_{14}, f_{17}, f_{19}, and f_{20}), and composition functions (f_{21}, f_{22}, f_{24}, f_{25}, f_{27}, and f_{28}), respectively. From Figure 4, on f_2, f_3, f_6, f_{11}, f_{25}, f_{27}, and f_{28}, ILxBBO's convergence speed is much faster than LxBBO's and the advantages of ILxBBO are prominent. On f_9, f_{13}, f_{20}, and f_{22}, in the early search stage, the convergence speed of ILxBBO is not as fast as LxBBO's, but in the late stage, ILxBBO' convergence speed is faster than LxBBO's. On f_{14}, f_{17}, f_{19}, f_{21}, and f_{24}, ILxBBO achieves nearly the same convergence speed as LxBBO does. On the whole, ILxBBO gets better convergence performance than LxBBO does. This verifies that the improved Laplace migration operator, example learning, and so on fasten the convergence speed of LxBBO.

4.6. CPU Runtime. In order to indicate the running time of ILxBBO, its average running time is recorded and analyzed only on the 30-dimensional functions after 51 independent runs. Figure 5 shows the average runtime comparison results between ILxBBO and 9 BBO variants. In Figure 5, the y-coordinate is the average runtime and its unit is "second"(s). From Figure 5, ILxBBO's average runtime is the least (6.0200 s), and TDBBOs (7.4356 s), BIBBOs (11.0533 s), BBOMs (10.8169 s), DEBBOs (7.5540 s), BLPSOs (15.5514 s), WRBBO (10.6473), EMBBO (9.6159), BHCS (6.9128), and LxBBOs (11.4140 s) are 80.96%, 54.46%, 55.65%, 79.69%, 38.71%, 56.54%, 65.68%, 87.08%, and 52.74%, respectively. ILxBBO obtains the fastest speed owing to the adoption of several strategies such as example learning, mutation operation removing, and greedy selection which reduces one sorting step and so on.

4.7. Application to Quadratic Assignment Problem (QAP). QAP is an NP-hard problem, which was first introduced by Koopmans and Beckmann [50], and it has been among the problems studied most in all of combinatorial optimization problems. QAP can be described as the problem of assigning a set of facilities to a set of locations with given distances between the locations and given flows between the facilities.

The approach is to have two matrices of size $N * N$ given by the following equations:

$$A = \left(f_{ij}\right), \tag{22}$$

$$B = \left(d_{ij}\right), \tag{23}$$

where $N = 1, 2, ..., n$ and f_{ij} is the flow or weight between each pair of facilities which represents the flow from facility i to facility j. d_{ij} is the distance between each pair of locations

Table 3: Comparison results among ILxBBO and BBO's variants on the 50-dimensional functions.

Function	Value	PRBBO	TDBBO	DEBBO	WRBBO	BHCS	EMBBO	ILxBBO
f_1	Mean	**0**	3.52e − 13	2.14e − 13	8.47e + 00	8.07e − 13	2.23e − 13	1.20e − 13
	Std	**0**	1.95e − 13	5.40e − 14	1.11e − 13	1.95e − 13	3.18e − 14	1.15e − 13
	Rank	**1**	5	3	7	6	4	2
f_2	Mean	6.91e + 05	5.83e + 05	5.94e + 07	1.81e + 07	**5.56e + 05**	2.80e + 07	5.58e + 05
	Std	4.50e + 05	2.19e + 05	1.44e + 07	1.10e + 07	**3.00e + 05**	4.72e + 06	1.86e + 05
	Rank	4	3	7	5	**1**	6	2
f_3	Mean	4.78e + 07	1.77e + 07	3.31e + 06	**3.23e + 06**	1.88e + 08	8.10e + 09	4.67e + 06
	Std	6.94e + 07	1.33e + 07	1.52e + 07	**3.31e + 06**	1.72e + 08	2.29e + 09	5.14e + 06
	Rank	5	4	2	**1**	6	7	3
f_4	Mean	1.27e + 04	1.31e + 02	5.21e + 04	1.23e + 04	6.84e + 02	8.30e + 04	**1.39e + 01**
	Std	2.67e + 03	6.41e + 01	5.36e + 03	1.80e + 03	5.45e + 02	8.14e + 03	**9.16e + 00**
	Rank	5	2	6	4	3	7	**1**
f_5	Mean	**1.14e − 13**	5.30e − 04	1.18e − 13	**1.14e − 13**	1.19e − 12	1.89e − 13	**1.14e − 13**
	Std	**0**	3.11e − 03	2.23e − 14	**0**	4.80e − 13	5.41e − 14	**0**
	Rank	**1**	7	4	**1**	6	5	**1**
f_6	Mean	**4.18e + 01**	6.54e + 01	4.34e + 01	4.49e + 01	4.81e + 01	4.39e + 01	4.38e + 01
	Std	**8.47e + 00**	2.47e + 01	8.48e − 03	6.68e − 01	1.71e + 01	6.53e − 01	5.19e − 01
	Rank	**1**	7	2	5	6	4	3
f_7	Mean	2.83e + 01	2.32e + 01	3.84e + 01	2.14e + 00	8.15e + 01	1.44e + 02	1.60e + 01
	Std	5.86e + 00	7.94e + 00	6.23e + 00	1.53e + 00	1.53e + 01	1.12e + 01	4.44e + 00
	Rank	4	3	5	1	6	7	2
f_8	Mean	**2.11e + 01**	2.11e + 01	2.11e + 01	2.11e + 01	2.12e + 01	2.11e + 01	2.11e + 01
	Std	**2.86e − 02**	3.28e − 02	4.52e − 02	4.33e − 02	3.82e − 02	3.52e − 02	3.71e − 02
	Rank	**1**	2	6	5	7	3	4
f_9	Mean	**3.28e + 01**	5.37e + 01	5.75e + 01	7.12e + 01	5.32e + 01	5.79e + 01	3.46e + 01
	Std	**4.72e + 00**	6.31e + 00	2.58e + 00	1.70e + 00	7.80e + 00	1.97e + 00	5.37e + 00
	Rank	**1**	4	5	7	3	6	2
f_{10}	Mean	**2.66e − 02**	2.18e − 01	8.17e − 01	1.41e − 01	4.66e − 02	2.26e + 01	2.98e − 01
	Std	**2.30e − 02**	1.16e − 01	3.68e − 01	6.79e − 02	4.05e − 02	3.96e + 00	1.78e − 01
	Rank	**1**	4	6	3	2	7	5
f_{11}	Mean	1.42e + 00	6.60e + 01	8.79e − 13	6.09e + 00	1.76e + 02	**5.68e − 14**	2.73e + 00
	Std	1.48e + 00	1.51e + 01	4.47e − 12	2.71e + 00	5.12e + 01	**0**	1.46e + 00
	Rank	3	6	2	5	7	**1**	4
f_{12}	Mean	1.21e + 02	1.04e + 02	1.99e + 02	3.40e + 02	2.35e + 02	3.90e + 02	**6.09e + 01**
	Std	3.25e + 01	2.31e + 01	2.04e + 01	1.31e + 01	5.62e + 01	3.70e + 01	**1.44e + 01**
	Rank	3	2	4	6	5	7	**1**
f_{13}	Mean	1.86e + 02	2.11e + 02	2.64e + 02	3.41e + 02	3.87e + 02	4.06e + 02	**1.49e + 02**
	Std	3.38e + 01	2.94e + 01	2.75e + 01	1.46e + 01	5.65e + 01	2.68e + 01	**3.26e + 01**
	Rank	2	3	4	5	6	7	**1**
f_{14}	Mean	2.84e + 02	1.42e + 03	1.44e + 02	6.77e + 02	1.01e + 04	**9.80e − 03**	7.03e + 01
	Std	1.82e + 02	4.90e + 02	2.52e + 01	2.05e + 02	1.40e + 03	**1.07e − 02**	7.04e + 01
	Rank	4	6	3	5	7	**1**	2
f_{15}	Mean	8.06e + 03	7.24e + 03	1.06e + 04	1.41e + 04	1.09e + 04	7.88e + 03	**6.61e + 03**
	Std	1.44e + 03	5.05e + 02	5.33e + 02	3.35e + 02	2.38e + 03	5.24e + 02	**7.66e + 02**
	Rank	4	2	5	7	6	3	**1**
f_{16}	Mean	2.76e + 00	**1.51e + 00**	2.94e + 00	3.33e + 00	3.35e + 00	1.72e + 00	2.31e + 00
	Std	4.65e − 01	**2.71e − 01**	2.99e − 01	2.87e − 01	4.95e − 01	2.03e − 01	2.70e − 01
	Rank	4	**1**	5	6	7	2	3
f_{17}	Mean	6.59e + 01	1.35e + 02	5.98e + 01	8.69e + 01	2.91e + 02	**5.08e + 01**	5.72e + 01
	Std	3.94e + 00	1.64e + 01	9.42e − 01	5.09e + 00	8.64e + 01	**6.82e − 14**	1.52e + 00
	Rank	4	6	3	5	7	**1**	2
f_{18}	Mean	1.60e + 02	1.80e + 02	3.52e + 02	3.82e + 02	3.00e + 02	4.42e + 02	**1.03e + 02**
	Std	4.10e + 01	1.84e + 01	1.60e + 01	1.52e + 01	9.95e + 01	2.58e + 01	**2.13 + 01**
	Rank	2	3	5	6	4	7	**1**
f_{19}	Mean	3.32e + 00	1.34e + 01	4.96e + 00	2.54e + 01	1.77e + 01	**1.57e + 00**	3.00e + 00
	Std	5.83e − 01	3.56e + 00	4.90e − 01	1.30e + 00	7.62e + 00	**2.35e − 01**	4.80e − 01
	Rank	3	5	4	7	6	**1**	2
f_{20}	Mean	1.90e + 01	**1.89e + 01**	2.14e + 01	2.19e + 01	2.14e + 01	2.27e + 01	1.91e + 01
	Std	1.01e + 00	**7.19e − 01**	2.85e − 01	2.26e − 01	9.71e − 01	4.33e − 01	5.65e − 01
	Rank	2	**1**	4	6	5	7	3

Table 3: Continued.

Function	Value	PRBBO	TDBBO	DEBBO	WRBBO	BHCS	EMBBO	ILxBBO
f_{21}	Mean	9.27e+02	9.43e+02	3.99e+02	8.93e+02	8.85e+02	**2.00e+02**	8.92e+02
	Std	3.35e+02	2.58e+02	3.83e+02	3.27e+02	2.87e+02	**1.56e+01**	3.45e+02
	Rank	6	7	2	5	3	**1**	4
f_{22}	Mean	1.37e+02	2.26e+03	1.54e+02	1.25e+02	9.79e+03	**1.55e+01**	6.18e+01
	Std	6.97e+01	4.38e+02	4.05e+01	6.18e+01	1.73e+03	**1.90e+00**	3.91e+01
	Rank	4	6	5	3	7	**1**	2
f_{23}	Mean	7.84e+03	7.92e+03	1.05e+04	1.41e+04	1.11e+04	9.70e+03	**7.13e+03**
	Std	1.51e+03	5.54e+02	6.34e+02	4.03e+02	2.73e+03	6.98e+02	**1.03e+03**
	Rank	2	3	5	7	6	4	**1**
f_{24}	Mean	2.45e+02	2.47e+02	2.64e+02	**2.20e+02**	3.86e+02	3.63e+02	2.23e+02
	Std	7.64e+00	8.78e+00	3.64e+01	**1.84e+01**	2.49e+01	6.59e+00	6.19e+00
	Rank	3	4	5	**1**	7	6	2
f_{25}	Mean	3.24e+02	3.42e+02	3.24e+02	3.92e+02	4.05e+02	3.99e+02	**3.09e+02**
	Std	9.79e+00	3.20e+01	2.46e+01	3.03e+01	1.99e+01	7.12e+00	**1.36e+01**
	Rank	2	4	3	5	7	6	**1**
f_{26}	Mean	2.45e+02	2.61e+02	3.05e+02	2.24e+02	4.42e+02	**2.02e+02**	2.35e+02
	Std	7.01e+01	7.35e+01	1.16e+02	5.29e+01	1.56e+01	**5.22e−01**	5.46e+01
	Rank	4	5	6	2	7	**1**	3
f_{27}	Mean	9.04e+02	8.54e+02	1.61e+03	7.56e+02	1.88e+03	1.60e+03	**6.98e+02**
	Std	9.34e+01	8.89e+01	9.80e+01	3.79e+02	1.89e+02	6.02e+02	**9.95e+01**
	Rank	4	3	6	2	7	5	**1**
f_{28}	Mean	4.00e+02	4.00e+02	4.00e+02	4.58e+02	1.05e+03	4.00e+02	**4.00e+02**
	Std	3.50e−13	3.97e−13	3.56e−13	4.12e+02	1.41e+03	1.31e−07	**2.70e−13**
	Rank	2	4	3	6	7	5	**1**
Count		6	2	0	3	1	7	10
Ave. Rank		2.93	4.00	4.29	4.57	5.61	4.36	2.14
Total. Rank		2	3	4	6	7	5	1

Table 4: Comparison results with the other IOAs on the 30-dimensional functions.

Function	Value	YYPO	DPCABC	DFnABC	FMPSO	GLPSO	HFPSO	MEGWO	ILxBBO
f_1	Mean	5.07e−09	**0**	**0**	**0**	1.33e−13	3.17e−13	3.85e−07	**0**
	Std	5.38e−09	**0**	**0**	**0**	1.13e−13	1.12e−13	6.93e−07	**0**
	Rank	7	**1**	**1**	**1**	5	6	8	**1**
f_2	Mean	1.39e+06	1.65e+07	9.53e+06	6.49e+06	3.77e+05	7.82e+05	3.22e+05	**2.00e+05**
	Std	8.36e+05	4.62e+06	3.56e+06	5.73e+06	1.51e+05	5.59e+05	2.52e+05	**1.04e+05**
	Rank	5	8	7	6	3	4	2	**1**
f_3	Mean	1.21e+08	8.44e+08	2.45e+08	2.64e+07	3.88e+06	2.50e+08	2.61e+08	**1.09e+06**
	Std	2.77e+08	6.40e+08	2.90e+08	2.38e+07	6.54e+06	3.93e+08	2.55e+08	**1.58e+06**
	Rank	4	8	5	3	2	6	7	**1**
f_4	Mean	2.28e+03	1.05e+05	8.37e+04	1.24e+04	7.67e+02	7.40e+02	7.56e+02	**5.44e+00**
	Std	1.59e+03	1.39e+04	1.32e+04	5.27e+03	6.26e+02	4.81e+02	7.07e+02	**3.83e+00**
	Rank	5	8	7	6	4	2	3	**1**
f_5	Mean	5.52e−05	5.35e−14	**0**	**0**	1.25e−13	1.29e−12	1.05e−04	9.14e−14
	Std	1.90e−04	5.73e−14	**0**	**0**	3.41e−14	6.86e−13	3.27e−04	4.56e−14
	Rank	7	3	**1**	**1**	5	6	8	4
f_6	Mean	4.75e+01	2.01e+01	1.72e+01	2.05e+01	**1.42e+01**	3.09e+01	2.78e+01	1.61e+01
	Std	3.36e+01	6.66e+00	3.18e+00	1.26e+01	**2.50e+00**	2.53e+01	2.72e+01	1.40e+01
	Rank	8	4	3	5	**1**	7	6	2
f_7	Mean	6.29e+01	1.14e+02	9.57e+01	3.65e+01	**4.70e+00**	6.56e+01	3.43e+01	1.10e+01
	Std	2.52e+01	1.80e+01	1.23e+01	6.83e+00	**3.29e+00**	3.57e+01	1.03e+01	6.89e+00
	Rank	5	8	7	4	**1**	6	3	2
f_8	Mean	2.09e+01	2.09e+01	2.09e+01	**2.09e+01**	2.09e+01	2.09e+01	2.10e+01	2.09e+01
	Std	5.83e−02	6.50e−02	4.03e−02	**3.49e−02**	5.79e−02	5.98e−02	5.25e−02	5.64e−02
	Rank	5	7	2	**1**	4	6	8	3
f_9	Mean	2.19e+01	3.09e+01	2.92e+01	2.15e+01	**1.12e+01**	1.93e+01	2.02e+01	1.87e+01
	Std	4.16e+00	1.97e+00	1.83e+00	3.36e+00	**2.59e+00**	3.53e+00	3.54e+00	3.76e+00
	Rank	6	8	7	5	**1**	3	4	2
f_{10}	Mean	**3.91e−02**	2.02e+00	1.58e+00	1.02e+01	8.28e−02	1.05e−01	7.92e−02	2.22e−01
	Std	**3.97e−02**	4.22e−01	2.75e−01	4.02e−02	4.02e−02	5.02e−02	7.14e−02	1.35e−01
	Rank	**1**	7	6	8	3	4	2	5

Table 4: Continued.

Function	Value	YYPO	DPCABC	DFnABC	FMPSO	GLPSO	HFPSO	MEGWO	ILxBBO
f_{11}	Mean	3.63e − 01	**0**	**0**	5.74e + 01	3.12e − 01	4.44e + 01	4.69e + 01	6.05e − 01
	Std	6.50e − 01	**0**	**0**	1.14e + 01	4.66e − 01	1.43e + 01	1.15e + 01	6.92e − 01
	Rank	4	**1**	**1**	8	3	6	7	5
f_{12}	Mean	1.73e + 02	1.38e + 02	1.42e + 02	6.54e + 01	3.73e + 01	7.77e + 01	4.71e + 01	**2.73e + 01**
	Std	5.06e + 01	2.78e + 01	2.12e + 01	1.90e + 01	1.23e + 01	2.24e + 01	1.28e + 01	**7.36e + 00**
	Rank	8	6	7	4	2	5	3	**1**
f_{13}	Mean	1.93e + 02	2.09e + 02	2.02e + 02	1.45e + 02	7.95e + 01	1.50e + 02	1.17e + 02	**5.51e + 01**
	Std	4.53e + 01	2.37e + 01	2.70e + 01	2.65e + 01	2.35e + 01	3.81e + 01	2.93e + 01	**1.86e + 01**
	Rank	6	8	7	4	2	5	3	**1**
f_{14}	Mean	5.24e + 00	9.67e − 01	**4.43e − 01**	1.90e + 03	3.54e + 00	1.40e + 03	2.27e + 03	4.38e + 01
	Std	2.74e + 00	1.29e + 00	**5.82e − 01**	4.45e + 02	3.39e + 00	3.34e + 02	6.29e + 02	4.06e + 01
	Rank	4	2	**1**	7	3	6	8	5
f_{15}	Mean	4.04e + 03	4.13e + 03	3.72e + 03	3.42e + 03	3.08e + 03	3.66e + 03	4.09e + 03	**3.03e + 03**
	Std	7.25e + 02	3.60e + 02	3.86e + 02	4.10e + 02	7.22e + 02	7.25e + 02	9.65e + 02	**5.70e + 02**
	Rank	6	8	5	3	2	4	7	**1**
f_{16}	Mean	1.38e + 00	1.42e + 00	1.32e + 00	1.35e + 00	1.37e + 00	**1.06e + 00**	2.36e + 00	1.66e + 00
	Std	3.66e − 02	2.45e − 01	1.65e − 01	4.33e − 01	9.86e − 01	**4.08e − 01**	3.04e − 01	2.17e − 01
	Rank	5	6	2	3	4	**1**	8	7
f_{17}	Mean	3.21e + 01	3.05e + 01	**3.05e + 01**	1.02e + 02	3.35e + 01	6.87e + 01	7.30e + 01	3.28e + 01
	Std	7.45e − 01	3.15e − 02	**2.31e − 02**	2.83e + 01	1.15e + 00	1.06e + 01	1.80e + 01	5.61e − 01
	Rank	3	2	**1**	8	5	6	7	4
f_{18}	Mean	1.43e + 02	2.11e + 02	1.86e + 02	8.02e + 01	**6.68e + 01**	1.00e + 02	1.20e + 02	7.19e + 01
	Std	2.95e + 01	2.34e + 01	1.87e + 01	1.25e + 01	**1.70e + 01**	2.09e + 01	4.41e + 01	1.61e + 01
	Rank	6	8	7	3	**1**	4	5	2
f_{19}	Mean	1.62e + 00	**2.03e − 01**	2.93e − 01	5.10e + 00	1.50e + 00	5.70e + 00	7.85e + 00	1.63e + 00
	Std	4.59e − 01	**8.27e − 02**	1.06e − 01	1.18e + 00	2.75e − 01	7.15e + 00	3.58e + 00	2.81e − 01
	Rank	4	**1**	2	6	3	7	8	5
f_{20}	Mean	1.16e + 01	1.28e + 01	1.32e + 01	1.27e + 01	**9.93e + 00**	1.36e + 01	1.10e + 01	1.02e + 01
	Std	9.07e − 01	5.83e − 01	8.94e − 01	4.78e + 01	**9.12e − 01**	1.94e + 00	8.70e − 01	4.98e − 01
	Rank	4	6	7	5	**1**	8	3	2
f_{21}	Mean	3.10e + 02	**2.28e + 02**	2.36e + 02	2.40e + 02	3.29e + 02	3.15e + 02	3.13e + 02	3.12e + 02
	Std	9.31e + 01	**3.26e + 01**	4.51e + 01	4.08e + 01	7.29e + 01	1.02e + 02	6.43e + 01	4.59e + 01
	Rank	4	**1**	2	3	8	7	6	5
f_{22}	Mean	1.22e + 02	3.12e + 01	**2.60e + 01**	1.31e + 02	1.12e + 02	1.88e + 03	2.53e + 03	1.24e + 02
	Std	5.47e + 01	2.70e + 01	**2.49e + 01**	3.65e + 01	1.98e + 01	5.65e + 02	6.81e + 02	7.47e + 00
	Rank	4	2	**1**	6	3	7	8	5
f_{23}	Mean	4.71e + 03	5.12e + 03	4.74e + 03	3.98e + 03	**3.23e + 03**	4.35e + 03	4.07e + 03	3.28e + 03
	Std	6.36e + 02	4.83e + 02	4.87e + 02	6.79e + 02	**5.90e + 02**	7.11e + 02	7.52e + 02	5.44e + 02
	Rank	6	8	7	3	**1**	5	4	2
f_{24}	Mean	2.62e + 02	2.82e + 02	2.79e + 02	2.87e + 02	2.09e + 02	2.63e + 02	2.42e + 02	**2.08e + 02**
	Std	1.65e + 01	6.32e + 00	5.76e + 00	7.74e + 00	5.16e + 00	1.24e + 01	6.48e + 00	**3.55e + 00**
	Rank	4	7	6	8	2	5	3	**1**
f_{25}	Mean	2.89e + 02	3.01e + 02	3.03e + 02	2.98e + 02	**2.45e + 02**	2.91e + 02	2.80e + 02	2.59e + 02
	Std	1.23e + 01	6.25e + 00	0.00e − 00	7.12e + 00	**1.93e + 01**	1.39e + 01	1.15e + 01	1.30e + 01
	Rank	4	7	8	6	**1**	5	3	2
f_{26}	Mean	2.00e + 02	2.01e + 02	2.00e + 02	3.45e + 02	2.29e + 02	2.98e + 02	2.65e + 02	**2.00e + 02**
	Std	3.08e − 02	3.58e − 01	3.56e + 06	2.97e + 01	5.00e + 01	7.05e + 01	6.92e + 01	**1.76e − 02**
	Rank	2	4	3	8	5	7	6	**1**
f_{27}	Mean	**8.89e + 01**	7.04e + 02	4.92e + 02	6.04e + 02	4.19e + 02	8.13e + 02	6.89e + 02	3.76e + 02
	Std	**1.06e + 02**	3.50e + 02	2.90e + 08	2.37e + 02	7.61e + 01	1.05e + 02	7.3221e + 01	3.54e + 01
	Rank	**1**	7	4	5	3	8	6	2
f_{28}	Mean	3.18e + 02	2.96e + 02	**2.93e + 02**	3.00e + 02	3.00e + 02	4.48e + 02	3.70e + 02	3.00e + 02
	Std	1.63e + 02	2.35e + 01	**1.32e + 04**	8.03e − 11	2.74e − 13	3.98e + 02	3.24e + 02	1.77e − 13
	Rank	6	2	**1**	5	4	8	7	3
Count		2	4	7	3	7	1	0	9
Ave. Rank		4.79	5.29	4.21	4.82	2.93	5.50	5.46	2.71
Total. Rank		4	6	3	5	2	8	7	1

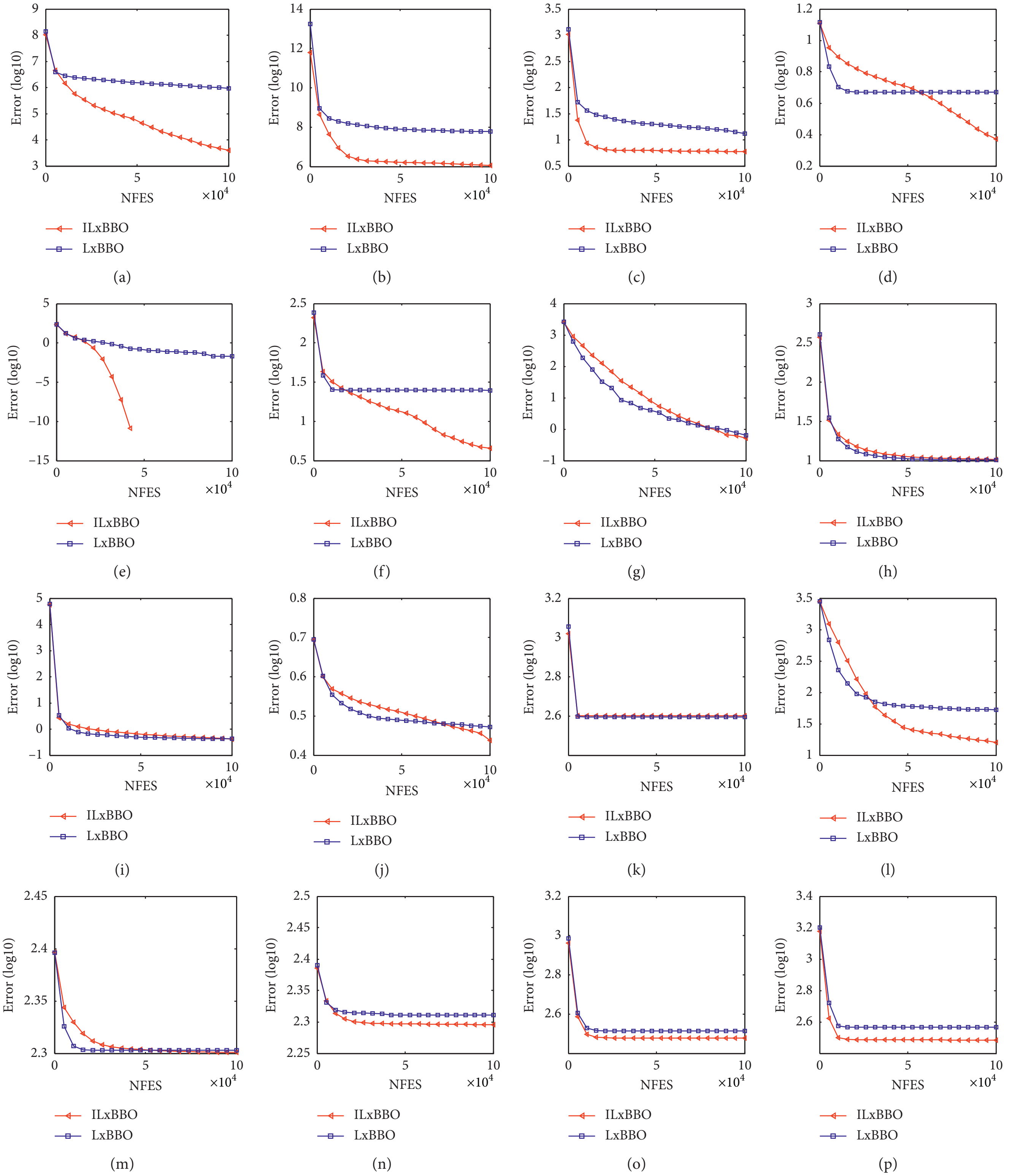


Figure 4: Convergence curves of the 2 algorithms on the 10-dimensional functions: (a) f_2; (b) f_3; (c) f_6; (d) f_9; (e) f_{11}; (f) f_{13}; (g) f_{14}; (h) f_{17}; (i) f_{19}; (j) f_{20}; (k) f_{21}; (l) f_{22}; (m) f_{24}; (n) f_{25}; (o) f_{27}; (p) f_{28}.

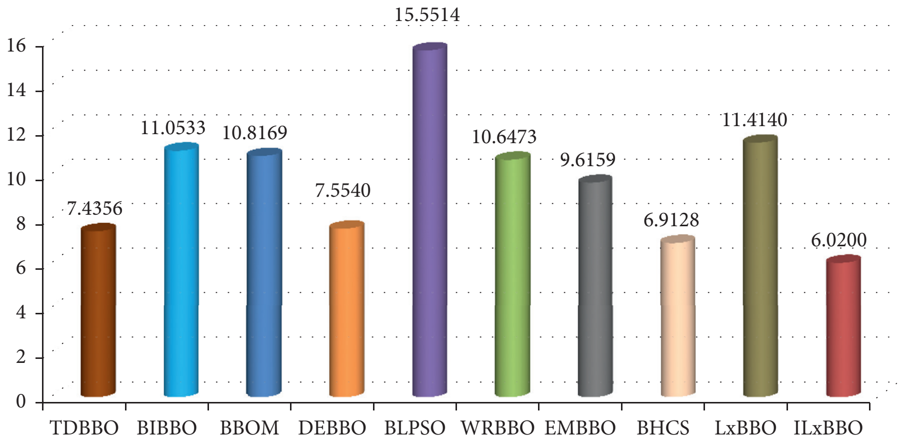


Figure 5: Average runtime comparisons of 10 algorithms on the 30-dimensional functions.

which represents the Euclidean distance from Location i to Location j. The objective function is given as follows:

$$\min = \sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij} d_{\pi(i)\pi(j)}. \tag{24}$$

QAP can be applied to many practical problems, such as modeling the location of hospital departments, optimizing the configuration of departments, reducing the time consumption of each patient, and improving the efficiency of hospital services for patients. For example, a simple example, in a single layer of a hospital, five departments (D1, D2, D3, D4, and D5) are set at five locations (1, 2, 3, 4, and 5), as shown in Figure 6.

In a single layer, the importance of each department is different, and the departments with more people flow are more important. On the contrary, the departments with less people flow are not more important, and the importance between the two departments is represented by the flow. The matrix F represents the flow between two departments, and the matrix L represents the distance between the two locations. The next step is to assign this group of departments to this group of locations to minimize the cost of patients between departments.

ILxBBO deals with continuous problems on which the search agents are represented by real values. To solve QAP with ILxBBO, it is necessary first for ILxBBO how to change from continuous optimization problems to discrete optimization problems. The changed method from [20] is adopted in this paper, and the method is explained as follows: the dimensionality of each solution represents the number of positions or facilities in QAP, and each solution represents the arrangement of different positions, corresponding to a location allocation scheme. The best allocation scheme is obtained through an IOA's optimization process. Suppose a QAP problem has 10 facility points that need to be allocated to 10 location points, as shown in Figure 7(a). Facility 1 is assigned to Location 8, Facility 2 is assigned to Location 5, and Facility 3 is assigned to Location 1, and so on. QAP is considered as a permutation discrete problem, so when IOA is used to solve QAP, it uses the maximum real value to map the real value into the permutation sequence, as shown in Figure 7(b). That is, the maximum real value 13.26 corresponds to the minimum integer 1, the minimum real value 0.85 to the maximum integer 10, and so on.

ILxBBO is used to solve QAP, and it has two modifications: the first is that each random real solution or updated real solution is changed into a permutation sequence according to real value mapping as in Figure 7; the second is that equation (24) is considered as the objective function. The comparison algorithms include IPSO and IFA, they both come from [51], and they are improved algorithms of PSO and FA, respectively. They are good at solving QAP and therefore have strong comparability.

To be fair, according to the best recommendations of common parameters from [20], MaxNFEs is 100,000, and Run is 30. So the parameters of the 4 comparison algorithms are set as follows: Run is 30, $N *$ MaxDT = 100,000, N is 100, and MaxDT is 1000. Table 5 records Mean and Std of each algorithm on the benchmark data, and the data are from [20] and have been used as a standard set for solving QAP. Best-known refers to the best solution found so far.

From Table 5, on Mean, ILxBBO obtains the best results on all the 10 data. On Std, ILxBBO gets 0 on 7 (had12, had14, had16, scr12, tal13a, tal12b, and chr12b) of 10 data, and it indicates that ILxBBO has strong stability. On had12, had14, had16, scr12, tai12a, tai12b, and chr12b, the mean value of ILxBBO is the same as those of Best-known, but on scr15, tai15a, and chr12b, the mean value of ILxBBO is slightly less than that of Best-known. Generally, it shows that ILxBBO can solve QAP better than LxBBO can.

4.8. Wilcoxon Signed Rank Test Analysis. Wilcoxon signed rank test is a nonparametric test method [52], and we use it to test statistically the performance of ILxBBO compared with the comparison algorithms on the 30-dimensional functions, the 50-dimensional functions, and on solving QAP. The software is IBM SPSS Statistics 19. The data are taken from Tables 2–5.

The Wilcoxon signed rank test results are shown in Table 6, where R^+ refers to the sum of ranks for the problems in which ILxBBO outperforms the comparison algorithm

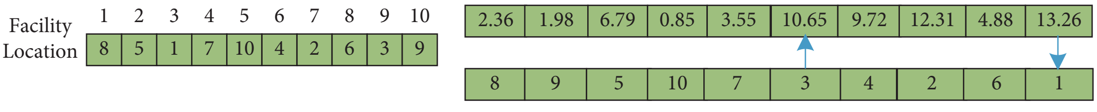
$$F = \begin{array}{c|ccccc} & D1 & D2 & D3 & D4 & D5 \\ \hline D1 & 2 & 1 & 4 & 1 & 0 \\ D2 & 1 & 5 & 3 & 3 & 5 \\ D3 & 4 & 2 & 1 & 4 & 4 \\ D4 & 0 & 0 & 5 & 2 & 3 \\ D5 & 1 & 3 & 0 & 0 & 0 \end{array}$$

(a)

$$L = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 12 & 8 & 6 & 0 & 18 \\ 2 & 15 & 4 & 13 & 17 & 14 \\ 3 & 0 & 5 & 22 & 23 & 10 \\ 4 & 9 & 16 & 2 & 5 & 7 \\ 5 & 22 & 18 & 20 & 9 & 6 \end{array}$$

(b)

Figure 6: Flow and distance matrixes of the hospital departments.



Facility	1	2	3	4	5	6	7	8	9	10
Location	8	5	1	7	10	4	2	6	3	9

(a)

2.36	1.98	6.79	0.85	3.55	10.65	9.72	12.31	4.88	13.26
8	9	5	10	7	3	4	2	6	1

(b)

Figure 7: (a) Solution representation; (b) real value mapping.

Table 5: Results of 4 algorithms on the 10 datasets.

Data	Best-known	Algorithm	Mean	Std
had12	1652	ILxBBO	**1652**	**0**
		LxBBO	1663.73	10.63
		IPSO	1656.87	6.80
		IFA	1652.87	2.44
had14	2724	ILxBBO	**2724**	**0**
		LxBBO	2736.6	13.29
		IPSO	2734.07	11.16
		IFA	2724.13	0.73
had16	3720	ILxBBO	**3720**	**0**
		LxBBO	3732.53	20.16
		IPSO	3726.87	17.93
		IFA	3720.93	1.01
scr12	31410	ILxBBO	**31410**	**0**
		LxBBO	33172.2	819.14
		IPSO	32544.07	732.37
		IFA	31633.4	360.20
scr15	51140	ILxBBO	**51358.6**	**610.78**
		LxBBO	54920.27	1810.98
		IPSO	55402.53	1177.87
		IFA	54398.27	1037.19
tal12a	224416	ILxBBO	**224416**	**0**
		LxBBO	240554.07	6099.48
		IPSO	235268.33	7034.64
		IFA	230336.13	4266.16
tal12b	39464925	ILxBBO	**39464925**	**0**
		LxBBO	40958448.6	1549935.98
		IPSO	41329078.33	1846814.90
		IFA	39552017	177161.72
tal15a	388214	ILxBBO	**390914.73**	**1916.80**
		LxBBO	404106.93	5864.4646
		IPSO	401980.2	6964.16
		IFA	397428.6	5088.67
chr12a	9552	ILxBBO	**9644**	**191.22**
		LxBBO	11111.73	884.95
		IPSO	10935.67	1078.83
		IFA	10065.07	383.58
chr12b	9742	ILxBBO	**9742**	**0**
		LxBBO	11781.07	2067.43
		IPSO	10591.13	969.71
		IFA	9917.53	387.15

Table 6: Wilcoxon signed rank test results.

Problem	ILxBBO versus	p	R^+	R^-	$n/w/t/l$
30 D	TDBBO	$1.30e-03$	324.5	81.5	28/21/5/2
	BIBBO	$9.40e-06$	379	27	28/26/0/2
	BBOM	$1.66e-04$	341.5	64.5	28/23/2/3
	DEBBO	$4.41e-02$	282	124	28/18/2/8
	BLPSO	$4.47e-08$	390	16	28/26/1/1
	WRBBO	$1.64e-02$	286	120	28/17/4/7
	EMBBO	$9.61e-03$	309	97	28/19/1/8
	BHCS	$3.20e-07$	396	10	28/26/0/2
	YYPO	$1.38e-02$	298.5	107.5	28/17/2/9
	DPCABC	$6.82e-03$	307	99	28/17/2/9
	DFnABC	$9.64e-03$	297.5	108.5	28/16/3/9
	FMPSO	$1.51e-05$	347.5	58.5	28/22/3/3
	GLPSO	$4.41e-01$	234	172	28/13/2/13
	HFPSO	$3.13e-07$	384.5	21.5	28/25/1/2
	MEGWO	$5.22e-08$	402	4	28/27/0/1
50 D	PRBBO	$7.50e-05$	338.5	67.5	28/13/3/6
	TDBBO	$2.09e-06$	366.5	39.5	28/22/2/4
	DEBBO	$1.78e-03$	321.5	84.5	28/22/2/4
	WRBBO	$9.19e-04$	327.5	78.5	28/21/2/5
	BHCS	$3.18e-05$	371	35	28/25/0/3
	EMBBO	$1.30e-02$	299.5	106.5	28/18/2/8
QAP	LxBBO	$1.95e-03$	55	0	10/0/0
	IPSO	$1.95e-03$	55	0	10/0/0
	IFA	$1.95e-03$	55	0	10/0/0

and R^- refers to the sum of ranks for the opposite. When ILxBBO and the comparison algorithm obtain the equal optimization performance, the corresponding ranks are split evenly to R^+ and R^- [33]. The p values can be computed according to the R^+ and R^- values. "$n/w/t/l$" means the number of the benchmark problems is n, and ILxBBO wins on w functions, ties on t functions, and loses on l functions. The following criterion is applied to compare the results:

(1) When $p > 0.05$, the difference of both algorithms is not significant

(2) When $p \leq 0.05$, the difference of both algorithms is significant

From Table 6, the value of p is more than 0.05 for ILxBBO versus GLPSO, so ILxBBO is not significantly better than GLPSO. However, ILxBBO wins 13 compared with GLPSO and the two functions that are equal. That is because the mean is equal, and the variance of ILxBBO is less than the variance of GLPSO from Table 4. On the 30-dimensional functions, the optimization performance of ILxBBO is significantly better than those of TDBBO, BIBBO, BBOM, DEBBO, BLPSO, WRBBO, EMBBO, BHCS, YYPO, DPCABC, DFnABC, FMPSO, and HFPSO. On the 50-dimensional functions, the optimization performance of ILxBBO is significantly better than those of PRBBO, TDBBO, DEBBO, WRBBO, BHCS, and EMBBO. On QAP, ILxBBO is significantly better than LxBBO, IPSO, and IFA. Generally, the optimization performance of ILxBBO is better. The Wilcoxon signed rank tests show again that ILxBBO obtains better optimization performance and verify the previous conclusions.

5. Conclusions and Future Work

In this paper, an improved LxBBO (ILxBBO) is proposed to improve the optimization performance of LxBBO. These improvements are as follows: the mutation operator is deleted to simplify the search process and to reduce the computational complexity. A dynamic two-differential perturbing operator is proposed to update the first two best habitats so that the global search ability and the local one can be improved in the early search phase and in the late one, respectively. The worst habitat adopts a two-global-best guiding operator to improve the search ability. An improved Laplace migration operator is used for other habitats' updating to reduce some parameters setting and fasten the convergence speed. In addition, some approaches such as example learning selection instead of roulette wheel selection, greedy selection instead of elitism strategy, and so on are adopted to reduce the computation complexity and enhance the performance. In order to verify ILxBBO, a large number of experiments are made on the CEC-2013 test set. Experiment results verify ILxBBO outperforms quite a few state-of-the-art algorithms on most cases. Moreover, the obtained results indicate how effective ILxBBO is in solving QAP. In the future, the improved Laplace operator of this paper may be applied to other BBO variants and other IOAs, and it is expected to apply ILxBBO to other more engineering problems.

Data Availability

CEC-2013 databases used in this paper are publicly available for download and, in particular, can be accessed from http://www.rforge.net/cec2013/files/, whereas QAP databases can be downloaded from http://anjos.mgi.polymtl.ca/qaplib/inst.html.

Disclosure

This article does not contain any studies with human participants performed by any of the authors.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant U1904123, in part by the Natural Science Foundation of Henan Province under Grant 162300410177, and in part by the Key Research Program of Application Foundation and Advanced Technology of Henan Province under Grant 19A520026.

References

[1] D. W. Gong, Y. Zhang, and C. L. Qi, "Environmental/economic power dispatch using a hybrid multi-objective optimization algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 32, no. 6, pp. 607–614, 2010.

[2] X. M. Zhang, Z. H. Fu, H. Y. Chen, W. T. Mao, S. W. Liu, and G. Q. Liu, "Levy flight shuffle frog leaping algorithm based on differential perturbation and quasi-Newton search," *IEEE Access*, vol. 7, pp. 116078–116093, 2019.

[3] R. Storn and K. Price, "Differential evolution–a simple and efficient adaptive scheme for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[4] G. G. Wang, S. Deb, A. H. Gandomi, and A. H. Alavi, "Opposition-based krill herd algorithm with Cauchy mutation and position clamping," *Neurocomputing*, vol. 177, pp. 147–157, 2016.

[5] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, no. 36–38, pp. 3902–3933, 2005.

[6] Z. H. Cui, B. Sun, G. G. Wang, Y. Xue, and J. J. Chen, "A novel oriented cuckoo search algorithm to improve dv-hop performance for cyber-physical systems," *Journal of Parallel and Distributed Computing*, vol. 103, pp. 42–52, 2017.

[7] Q. H. Gu, X. X. Li, and S. Jiang, "Hybrid genetic grey wolf algorithm for large-scale global optimization," *Complexity*, Article ID 2653512, 18 pages, 2019.

[8] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.

[9] Z. W. Liao, W. Y. Gong, X. S. Yan, L. Wang, and C. Y. Hu, "Solving nonlinear equations system with dynamic repulsion-based evolutionary algorithms," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2018.

[10] A. Wu and Z. L. Yang, "An elitist transposon quantum-based particle swarm optimization algorithm for economic dispatch problems," *Complexity*, Article ID 7276585, 15 pages, 2018.

[11] W. Y. Gong, Y. Wang, Z. H. 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*, pp. 1–15, 2018.

[12] H. P. Ma, "An analysis of the equilibrium of migration models for biogeography-based optimization," *Information Science*, vol. 180, no. 18, pp. 3444–3464, 2010.

[13] K. Jamuna and K. S. Swarup, "Multi-objective biogeography based optimization for optimal pmu placement," *Applied Soft Computing*, vol. 12, no. 5, pp. 1503–1510, 2012.

[14] D. F. Wu, F. k. Ren, and Q. Lei, "Active disturbance rejection controller design for dynamically positioned vessels based on adaptive hybrid biogeography-based optimization and differential evolution," *ISA Transactions*, vol. 78, pp. 56–65, 2017.

[15] K. Ambreen, R. Khalid, R. Maroof et al., "Implementing critical peak pricing in home energy management using

biography based optimization and genetic algorithm in smart grid," in *Proceeding of the International Conference on Broadband & Wireless Computing*, Springer, Cham, Switzerland, November 2017.

[16] C. Yogesh, M. Hariharan, R. Ngadiran, S. Y. Abdul Hamid Adom, C. Berkai, and K. Polat, "A new hybrid PSO assisted biogeography-based optimization for emotion and stress recognition from speech signal," *Expert Systems Applications*, vol. 69, pp. 149–158, 2016.

[17] W. A. Guo, M. Chen, L. Wang, Y. F. Mao, and Q. D. Wu, "A survey of biogeography-based optimization," *Neural Computing and Applications*, vol. 28, pp. 1909–1926, 2017.

[18] G. J. Xiong, D. Y. Shi, and X. Z. Duan, "Enhancing the performance of biogeography-based optimization using polyphyletic migration operator and orthogonal learning," *Computers & Operations Research*, vol. 41, no. 1, pp. 125–139, 2014.

[19] T. Dokeroglu, E. Sevinc, and A. Cosar, "Artificial bee colony optimization for the quadratic assignment problem," *Applied Soft Computing*, vol. 76, pp. 595–606, 2019.

[20] M. Abdel-Basset, G. Manogaran, D. El-shahat, and S. Mirjalili, "Integrating the whale algorithm with tabu search for quadratic assignment problem: a new approach for locating hospital departments," *Applied Soft Computing*, vol. 73, pp. 530–546, 2018.

[21] V. Garg and K. Deep, "Performance of Laplacian biogeography-based optimization algorithm on CEC 2014 continuous optimization benchmarks and camera calibration problem," *Swarm and Evolutionary Computation*, vol. 27, pp. 132–144, 2016.

[22] X. T. Li, J. Y. Wang, J. P. Zhou, and M. H. Yin, "A perturb biogeography based optimization with mutation for global numerical optimization," *Applied Mathematics and Computation*, vol. 218, no. 2, pp. 598–609, 2011.

[23] Y. J. Zheng, H. F. Ling, X. B. Wu, and J. Y. Xue, "Localized biogeography-based optimization," *Soft Computing*, vol. 18, no. 11, pp. 323–2334, 2014.

[24] Q. X. Feng, S. Y. Liu, J. K. Zhang, G. P. Yang, and L. Q. Yong, "Improved biogeography- based optimization with random ring topology and powell's method," *Applied Mathematical Modelling*, vol. 41, pp. 630–649, 2016.

[25] H. P. Ma and D. Simon, "Blended biogeography-based optimization for constrained optimization," *Engineering Applications of Artificial Intelligence*, vol. 24, no. 3, pp. 517–525, 2011.

[26] X. Chen, H. Tianfield, W. L. Du, and G. H. Liu, "Biogeography-based optimization with covariance matrix based migration," *Applied Soft Computing*, vol. 45, pp. 71–85, 2016.

[27] X. M. Zhang, Q. Kang, Q. Tu, J. F. Cheng, and X. Wang, "Efficient and merged biogeography-based optimization algorithm for global optimization problems," *Soft Computing*, vol. 23, no. 12, pp. 4483–4502, 2019.

[28] W. Y. Gong, Z. H. Cai, C. X. Ling, and H. Li, "A real-coded biogeography-based optimization with mutation," *Applied Mathematics and Computation*, vol. 216, no. 9, pp. 2749–2758, 2010.

[29] M. R. Lohokare, S. S. Pattnaik, B. K. Panigrahi, and S. Das, "Accelerated biogeography-based optimization with neighborhood search for optimization," *Applied Soft Computing*, vol. 13, no. 5, pp. 2318–2342, 2013.

[30] W. Y. Gong, Z. H. Cai, and C. X. Ling, "DE/BBO: A hybrid differential evolution with biogeography-based optimization for global numerical optimization," *Soft Computing*, vol. 15, no. 4, pp. 645–665, 2010.

[31] P. Savsani, R. Jhala, and V. Savsani, "Effect of hybridizing biogeography-based optimization (BBO) technique with artificial immune algorithm (AIA) and ant colony optimization (ACO)," *Applied Soft Computing*, vol. 21, no. 5, pp. 542–553, 2014.

[32] G. Khademi and H. Mohammadi, "Hybrid invasive weed/biogeography-based optimization," *Engineering Applications of Artificial Intelligence*, vol. 64, pp. 213–231, 2017.

[33] X. M. Zhang, Q. Kang, J. F. Cheng, and X. Wang, "A novel hybrid algorithm based on biogeography-based optimization and grey wolf optimizer," *Applied Soft Computing*, vol. 67, pp. 197–214, 2018.

[34] J. Brest, B. Boškovic, S. Greiner, V. Zumer, and S. M. Maucec, "Performance comparison of self-adaptive and adaptive differential evolution algorithms," *Soft Computing*, vol. 11, no. 7, pp. 617–629, 2007.

[35] W. Y. Gong, Y. Wang, Z. H. Cai, and S. X. 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.

[36] L. H. Wu, Y. N. Wang, and S. W. Zhou, "Self-adapting control parameters modified differential evolution for trajectory planning of manipulators," *Journal of Control Theory & Applications*, vol. 5, no. 4, pp. 365–373, 2007.

[37] X. M. Zhang, D. D. Wang, and H. Y. Chen, "Improved biogeography-based optimization algorithm and its application to clustering optimization and medical image segmentation," *IEEE Access*, vol. 7, pp. 28810–28825, 2019.

[38] J. Liang, P. N. Suganthan, and A. G. Hernández-Díaz, "Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization," Technical report 201212, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, 2013.

[39] F. Q. Zhao, S. Qin, Y. Zhang, W. M. Ma, C. Zhang, and H. B. Song, "A two-stage differential biogeography-based optimization algorithm and its performance analysis," *Expert Systems with Applications*, vol. 115, pp. 329–345, 2019.

[40] Q. Niu, L. T. Zhang, and K. Li, "A biogeography-based optimization algorithm with mutation strategies for model parameter estimation of solar and fuel cells," *Energy Conversion and Management*, vol. 86, pp. 1173–1185, 2014.

[41] X. Chen, B. Xu, and W. L. Du, "An improved particle swarm optimization with biogeography-based learning strategy for economic dispatch problems," *Complexity*, Article ID 7289674, 15 pages, 2018.

[42] X. Chen and K. J. Yu, "Hybridizing cuckoo search algorithm with biogeography-based optimization for estimating photovoltaic model parameters," *Solar Energy*, vol. 180, pp. 192–206, 2019.

[43] V. Punnathanam and P. Kotecha, "Yin-yang-pair optimization: a novel lightweight optimization algorithm," *Engineering Applications of Artificial Intelligence*, vol. 54, pp. 62–79, 2016.

[44] L. Z. Cui, G. G. Li, Y. L. Luo et al., "An enhanced artificial bee colony algorithm with dual-population framework," *Swarm and Evolutionary Computation*, vol. 43, pp. 184–206, 2018.

[45] L. Z. Cui, K. Zhang, G. H. Li et al., "A smart artificial bee colony algorithm with distance-fitness-based neighbor search and its application," *Future Generation Computer Systems*, vol. 89, pp. 478–493, 2018.

[46] N. Lynn and P. N. Suganthan, "A fitness-based multi-role particle swarm optimization," *Swarm and Evolutionary Computation*, vol. 44, pp. 349–364, 2019.

[47] Y. J. Gong, J. J. Li, Y. C. Zhou et al., “Genetic learning particle swarm optimization,” *IEEE Transactions on Cybernetics*, vol. 46, no. 10, pp. 2277–2290, 2015.

[48] I. B. Aydilek, “A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems,” *Applied Soft Computing*, vol. 66, pp. 232–249, 2018.

[49] Q. Tu, X. Chen, and X. Liu, “Multi-strategy ensemble grey wolf optimizer and its application to feature selection,” *Applied Soft Computing*, vol. 76, pp. 16–30, 2019.

[50] T. C. Koopmans and M. Beckmann, “Assignment problems and the location of economic activities,” *Econometrica*, vol. 25, no. 1, pp. 53–76, 1957.

[51] Yarpiz, “Quadratic assignment problem using GA, PSO and FA,” 2015, http://yarpiz.com/359/ypap104-quadratic-assignmentproblem.

[52] D. Joaquín, 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.