IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 15, 2019, accepted February 15, 2019, date of publication March 1, 2019, date of current version March 18, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2901849*

# Improved Biogeography-Based Optimization Algorithm and Its Application to Clustering Optimization and Medical Image Segmentation

**XINMING ZHANG[1], DOUDOU WANG[1], AND HAIYAN CHEN[2]**
[1]College of Computer and Information Engineering, Henan Normal University, Xinxiang 453007, China
[2]Department of Gynecological Tumor, Hubei Cancer Hospital, Wuhan 430079, China

Corresponding author: Haiyan Chen (cleverbear88@126.com)

This work was supported by the Key Research Projects of Higher Education Institutions of Henan Province, China, under Grant 19A520026.

**ABSTRACT** In order to improve the optimization efficiency of the biogeography-based optimization (BBO) algorithm, an improved BBO algorithm, that is, worst opposition learning and random-scaled differential mutation BBO (WRBBO), is presented in this paper. First, BBO's mutation operator is deleted to reduce the computational complexity and a more efficient random-scaled differential mutation operator is merged into BBO's migration operator to obtain global search ability. Second, in order to balance exploration and exploitation, the BBO's migration operator is replaced with a dynamic heuristic crossover to enhance the local search ability. Finally, a worst opposition learning is merged into the improved algorithm to avoid trapping into local optima. A large number of experiments are made on 18 various kinds of classic benchmark functions and some complex functions from the CEC-2013 test set. In addition, WRBBO is applied to clustering optimization and medical image segmentation. The experimental results show that WRBBO has better optimization efficiency on benchmark function optimization, clustering optimization, and medical image segmentation than quite a few state-of-the-art BBO variants and other algorithms.

**INDEX TERMS** Evolutionary algorithm, biogeography-based optimization, heuristic crossover, opposition learning, differential mutation, clustering, medical image segmentation.

## I. INTRODUCTION

Many problems in the real world can be considered as optimization problems. In the past, empirical analysis and mathematical methods are used to solve the problems. However, with the development of science, technology and society, more and more optimization problems have become more diversified and complex, and it is difficult for the previous optimization techniques to solve them. In recent decades, many algorithms inspired by swarm intelligence have been proposed. The swarm-based algorithm is a kind of Intelligent Optimization Algorithms (IOAs) that simulates natural phenomena, etc. IOAs have higher efficiency than traditional optimization methods have and they are widely adopted to deal with many optimization problems in science and engineering fields [1]–[3]. Well-known IOAs include Particle Swarm Optimization (PSO) [2], [4], Artificial Bee Colony (ABC) [5], Differential Evolution (DE) [6], [7], Gray Wolf Optimizer (GWO) [8], Flower Pollination Algorithm (FPA) [9], Cuckoo Search (CS) [10], Biogeography-Based Optimization (BBO) [11] and so forth.

BBO is one of IOAs, proposed by Simon [11]. In BBO, there are two main operators, migration and mutation. BBO mainly uses migration to realize information sharing between habitats, thereby obtaining exploitation capability; and the mutation operator mimics the mutation of habitats so as to obtain exploration ability and finally BBO may find approximately optimal solution. The mechanism of BBO is unique, simple and easy to implement, and BBO has good exploitation ability, so it has received extensive attention. BBO has achieved great success in numerical optimization problems [12] and is also widely used in many other fields [12]–[14]. However, BBO has some drawbacks, such as low optimization efficiency when solving some optimization problems.

The associate editor coordinating the review of this manuscript and approving it for publication was Jagdish Chand Bansal.

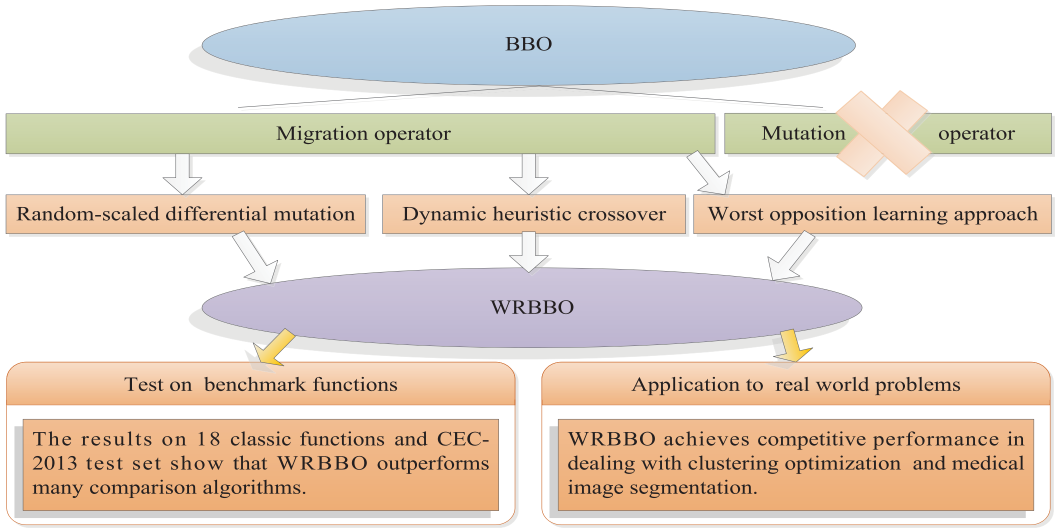



**FIGURE 1.** **Graphical abstract of this paper.**

The optimization efficiency of BBO should be comprehensively improved from the two main aspects: (1) BBO can get better optimization performance by improving the optimization ability and enhancing the balance between exploration and exploitation; (2) The optimization process is less time-consuming by reducing computational complex and accelerating convergence speed.

To enhance the optimization performance of BBO, many researchers have done a lot of work. Zheng *et al.* [15] proposed an Ecogeography-Based Optimization (EBO) which regarded the population of islands as an ecological system with a local topology and the two novel migration operators were designed to perform effective exploration and exploitation to improve the optimization performance of BBO. Ma and Simon [16] presented a blended migration operator which used the two habitats' features to replace the features of the immigration habitat to improve the optimization ability. Niu *et al.* [17] combined the mutation motivated from DE and chaos theory into the BBO structure to enhance the optimization performance. Zhang *et al.* [18] came up with a novel hybrid algorithm based on BBO and GWO. The two algorithms were improved respectively, and then combined together, the experimental results show that the proposed algorithm enhances the optimization performance. Khademi *et al.* [19] put forward a hybrid BBO, which embedded the migration operator of BBO into Invasive Weed Optimization (IWO) to improve the optimization ability.

To reduce the computational complexity of BBO, Gong *et al.* [20] proposed a hybrid BBO with mutation operator to improve the optimization performance and reduce the computational complexity. Zhang *et al.* [21] presented an efficient and merged BBO to reduce the computational complexity. Guo *et al.* [22] proposed three novel migration operators to enhance the optimization performance and reduce the computational complexity.

Although these BBO variants reviewed above have improved the optimization performance or reduced the computational complexity of BBO, they still have not high optimization efficiency when dealing with some optimization problems. In order to deal with the optimization problems of various types more efficiently, it is necessary to further improve the optimization efficiency of BBO, so the improvement research on BBO is still meaningful and valuable.

This paper presents a Worst opposition learning and Random-scaled differential mutation BBO (WRBBO) to obtain higher optimization efficiency. The contributions are described as follows.

1. A dynamic heuristic crossover is designed to improve the local search ability and accelerate the convergence speed, where the example learning selection is employed to reduce the computational complexity and improve the local search ability, too.

2. The original mutation operator is moved out and a random-scaled differential mutation is embedded into the migration operator to obtain stronger exploration ability.

3. A worst opposition learning approach is employed to prevent the algorithm from falling into local optima.

4. WRBBO is tested on 18 classic functions and some complex functions from CEC-2013 test set. The Wilcoxon signed-rank test is employed to verify the optimization performance of WRBBO. The results indicate that WRBBO performs more effectively than many state-of-the-art BBO variants and other IOAs.

5. Besides, WRBBO is applied to clustering optimization and medical image segmentation. The results also show that WRBBO outperforms most of the competitive algorithms.

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

The rest of this paper is organized below. Section II gives a brief of BBO. The proposed algorithm, WRBBO, is elaborated in Section III. The experimental results are reported and analyzed in Section IV. Section V provides conclusions and future work.

## II. BBO

BBO was proposed in 2008, based on biogeography [11]. A population consists of several habitats. Each habitat estimates its survival environment through Habitat Suitability Index (HSI). The factor which affects HSI is called Suitability Index Variables (SIVs), including temperature, humidity, illumination, etc. Assuming that there are $N$ habitats in the solution space of the $D$-dimensional optimization problem, each habitat can be regarded as a $D$-dimensional candidate solution, the habitat's SIVs are equivalent to their corresponding candidate solution components and the HSI of the habitat is equivalent to the fitness value of its candidate solution.

In BBO, the migration operator achieves information sharing between habitats. The number of species in a habitat is inversely related to the immigration rate and positively related to the emigration rate, which are calculated by the linear migration model in Figure 2. The calculations are expressed by Eqs. (1) and (2) .

$$\lambda_i = I\,(1 - S_i/S_{\max}) \tag{1}$$

$$\mu_i = E(S_i/S_{\max}) \tag{2}$$

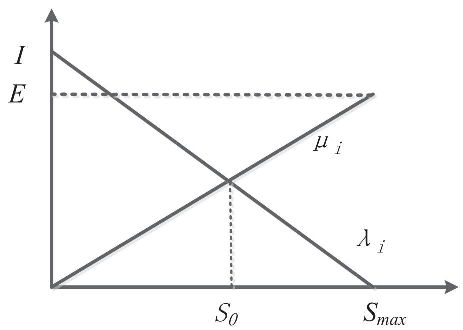



**FIGURE 2.** Linear migration model.

where $I$ and $E$ are the maximum immigration and emigration rates, respectively. $S_i$ is the current number of species of the habitat $H_i$, $S_{max}$ is the maximum number of species and $S_0$ is the balance point of immigration and emigration. The information sharing between habitats is completed by the migration operator as shown by Eq. (3).

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

where $H_i$ is the immigration habitat, $H_k$ is the emigration habitat and SIV is the emigration component. In addition, the mutation operator is used to change SIVs of the habitat randomly according to the habitat's mutation probability. The mutation rate is calculated by Eq. (4).

$$m_i = m_{\max}\,(1 - P_i/P_{\max}) \tag{4}$$

where $m_{max}$ represents the maximum mutation probability which is a user-defined parameter, and $P_i$ is the probability of species [8]. $P_{max}$ represents the maximum species number probability. The mutation operator is expressed by Eq. (5).

$$H_i(SIV_j) \leftarrow lb_j + rand(ub_j - lb_j) \tag{5}$$

where $H_i$ is the mutation habitat, $lb_j$ and $ub_j$ are the lower and upper boundary values of the $j$th SIV of $H_i$, respectively, and *rand* is a uniformly distributed random real number between 0 and 1.

In BBO, the elitism strategy is adopted to keep several best habitats. At the initial stage of each iteration, several best habitats in the current population are reserved, and several worst habitats are replaced by several best habitats in the final stage of this iteration. The pseudo-code of BBO is shown in Algorithm 1, where $t$ is the current iteration number and *MaxDT* is the maximum number of iterations.

**Algorithm 1** BBO

```
Begin
 Set parameters and randomly initialize the population (N)
 Calculate the HSI of each habitat and sort the population
 from the best  to the worst according to their HSIs
 for t = 1 to MaxDT do
   Calculate the immigration, emigration and mutation
   rate, and keep some elitist habitats
    for i = 1 to N do
     for j = 1 to D do
       if rand < λi then
       Select habitat Hk (k = 1 to N) by the roulette wheel
         selection
         Update Hi (SIVj) by by Eq. (3)
       end if
     end for
    end for
    for i = 1 to N do
     for j = 1 to D do
       if rand < mi then
         Update Hi (SIVj) by by Eq. (5)
       end if
     end for
    end for
   Calculate the HSI and sort the population from the best
   to the worst by their HSIs
   Replace the worst habitats with the best habitats
   Sort the population from the best to the worst by their
   HSIs
 end for
End
```

## III. PROPOSED BBO (WRBBO)

From Algorithm 1, BBO has the following defects. The mutation operator of BBO has some global search ability to prevent BBO from falling into local optima to some degree. However, the exploration ability of the mutation operator is weak, and it is difficult to keep balance with the exploitation ability of the migration operator. The mutation direction is at random, therefore, it may destroy some better solutions and slow down convergence speed in the later search phase. What's more, there is high computational complexity in the mutation operator. The migration operator of BBO adopts the direct-copying-based migration shown by Eq. (3). Although it can achieve some local search ability, the searchable positions of this simple migration model are limited and the local search capability is limited, too. In addition, the calculation steps of BBO are tedious, such as calculating the immigration rate, emigration rate and mutation rate, and sorting twice at each iteration. Thus, the computational complexity is very high. So WRBBO is proposed.

### A. RANDOM-SCALED DIFFERENTIAL MUTATION

DE is a well-known IOAs with strong robustness. According to the distance and direction information between individuals in the current population, it guides the population to search for the optimal solution through differential calculation [6]. The mutation strategy of DE has excellent global search ability, which is adopted by many other improved algorithms to enhance the optimization performance [23]–[25].

In this paper, the mutation operator of BBO is removed to avoid the destruction of the better solutions and also reduce the computational complexity. In order to make up the global search ability owing to the lack of the mutation operator, a more efficient random-scaled differential mutation operator is incorporated into the migration operator of BBO inspired by the DE algorithm. It is expressed by Eq. (6) almost the same as [18, eq. (15)].

$$H_i(SIV_j) \leftarrow H_i(SIV_j) + \alpha_d * (H_b(SIV_j) - H_i(SIV_j) + H_m(SIV_j) - H_n(SIV_j)) \quad (6)$$

where $H_b$ is the best habitat in the current population, $H_m$ and $H_n$ are two habitats selected randomly, which satisfy $m$, $n$, $i \in [1,N]$ and $m \neq n \neq i$, $\alpha_d$ is the differential scaling factor. From Eq. (6), the $j$th SIV of $H_i$ is affected by the corresponding SIV of itself, the differential results of the best habitat and two other habitats in the current population; it can receive more diverse information to increase population diversity and improve global search ability.

Compared with the differential mutation operation of [18], although two approaches adopt almost the same formula in form, two use different strategies of the scale factor, the scale factor of [18] is given to a random number which is uniformly distributed between 0 and 1, while that in this paper is changed with random scaled dynamic adjustment and its calculation is shown by Eq. (7).

$$\alpha_d = rand^{\beta}, \quad \beta = 4\sqrt{t/MaxDT} \quad (7)$$

From Eq. (7), the differential scaling factor is composed of the exponential form of *random*, the randomness is strong and the ability of global search is improved. What's more, the iteration number $t$ increases, the value of $\beta$ gradually increases, and *rand* is a uniformly distributed random real number between 0 and 1, so the value of $\alpha_d$ has a decreasing trend. It can enhance the global search ability in the earlier phase and the local search ability in the later search phase.

The random-scaled differential mutation has the following characteristics: (1) The habitat selected is affected by itself, the best habitat and two randomly habitats (different information merged) to improve the global search ability. (2) The best habitat is selected to ensure that the population is moving in a good direction. (3) The random-scaled dynamic adjustment approach is adopted to improve the population diversity. (4) It can further enhance the global search ability in the earlier phase and the local search ability in the later search ability. (5) The random-scaled dynamic adjustment scaling can avoid the parameter setting and improve the operability of the algorithm.

### B. DYNAMIC HEURISTIC CROSSOVER

In order to enhance the search ability further, a dynamic heuristic crossover operator is embedded into the migration operator instead of Eq. (3). It is expressed by Eq. (8) almost the same as in [18] and [21].

$$H_i(SIV_j) \leftarrow H_e(SIV_j) + \alpha_h * (1 - 2 * rand) * (H_i(SIV_j) - H_e(SIV_j)) \quad (8)$$

where $H_e$ is selected by the example learning selection [18] instead of the roulette wheel selection.

Compared with the multi-migration operation of [18], two approaches adopt almost the same formula in form, but two use different strategies in the coefficient, the coefficient of [18] is $2\sqrt{t/MaxDT} * (0.5 - rand)$. Compared with the sharing operator of [21], although two approaches adopt almost the same formula in form, there are two differences, from Eq. (8), the features of immigration habitat minus the features of emigration habitat, while the features of emigration habitat minus the features of immigration habitat in [21]. Another difference is the coefficient, the coefficient of [21] is (*rand*-0.5), while that of this paper is changed with the cross-scaling factor. Inspired by the coefficient in GWO [7], the cross-scaling factor ($\alpha_h$) is employed which adopts the dynamic adjustment strategy and it is expressed by Eq. (9).

$$\alpha_h = 2 * (1 - t/MaxDT) \quad (9)$$

From Eq. (9), the value of $\alpha_h$ is 2 when $t$ is 0 and the value of $\alpha_h$ is 0 when $t$ is $MaxDT$. $\alpha_h$ is linearly decreased from 2 to 0 over the course of iterations. From Eq. (8), ($H_i(SIV_j)$-$H_e(SIV_j)$) can obtain a value whose disturbance direction and amplitude are dynamically changed by the random value obtained by (1-2∗*rand*) and $\alpha_h$, respectively, and adding the disturbance value to $H_e(SIV_j)$ to realize dynamically local search around $H_e(SIV_j)$. The value of (1-2∗*rand*) ranges from −1 to 1. In the early stage of the algorithm, the value of $\alpha_h$∗(1-2∗*rand*) and the disturbance amplitude are large. It can search around a wide range to enhance the global search ability. In the later stages of the algorithm, the value of $\alpha_h$∗(1-2∗*rand*) and the disturbance amplitude is small and

$H_e$ is better than $H_i$, the search direction is around $H_e$ to improve the accuracy of the solution and the local search ability. Besides, the dynamic cross-scaling factor is used to avoid setting the parameter to improve the operability of the algorithm too.

Compared with the migration operator shown by Eq. (3) of BBO, the dynamic heuristic crossover operator has the following differences: (1) The emigration habitat is selected by the roulette wheel selection in BBO, while the emigration habitat is selected by the example learning selection in WRBBO. This change reduces the computational complexity and makes the population move to better positions [18], so the best solution can be found as quickly as possible. (2) In BBO, the features of the emigration habitat are copied directly into the immigration habitat. The dynamic heuristic crossover operator uses not only the characteristics of the emigration habitat, but also the disturbance values of the immigration and the emigration habitats. This increases the searchable range and enhances the population diversity.

### C. WORST OPPOSITION LEARNING APPROACH

For an algorithm, it is easy to fall into local optima if the population diversity decreases under a certain condition. The opposition learning approach is introduced by Tizhoosh [26], and it widely used to prevent the algorithm from falling into local optima [18], [25], [27]. A gray wolf randomly was used to perform the opposition learning approach to jump out of the local optima [18]. The opposition learning approach was used to improve the chaotic population through computing the opposite direction for each solution [25]. The last half of the population performed the opposition learning approach in [27]. In this paper, in order to further avoid falling into local optima, a worst opposition learning approach is proposed, that is, at each iteration, only the worst habitat adopts the opposition learning approach in the current population, and the habitat can jump greatly in the search space to improve the population diversity. It is expressed by Eq. (10).

$$H_w(SIV_j) \leftarrow lb_j + (ub_j - H_b(SIV_j)) \tag{10}$$

where $H_w$ is the worst habitat in the current population.

The opposition learning in WRBBO has the following characteristics. In WRBBO, the worst habitat in the current population is selected to adopt the opposition learning approach. From Eq. (10), the best habitat is used to perform the opposition learning approach. The reverse point of the best solution is acted as a new solution of the worst habitat. If the new solution is better than the original worst solution, the worst habitat is improved largely and the whole population may jump out of the local optima. If the new solution is worse than the original worst solution, it will only affect the poor solution and will not affect the quality of other solutions and the whole population.

### D. OTHER IMPROVEMENTS

In addition to the above improvements, the elitist strategy [11], [28] is replaced by the greedy selection [29]. This reduces one sorting step and doesn't need to set parameters of the elitist strategy. Furthermore, since habitats' HSIs are always effective in a lot of applications, the immigration rate calculation step is moved outside of the iteration loop and the objective functions use parallel computing, these improvements further reduce the computational complexity from multiple aspects. The pseudo code of WRBBO is shown in Algorithm 2 and its flowchart is given in Figure 3.

There are the following differences between BBO and WRBBO: (1) BBO generates new solutions by the migration operator and the mutation operator, while WRBBO only uses the improved migration operator to update the solutions and there is no mutation operator standalone in WRBBO. (2) The mutation operator is used to enhance the global search ability in BBO, while the worst opposition learning approach and random-scaled differential mutation and so on are utilized to

**Algorithm 2** WRBBO

```
Begin
  Set the parameters and initialize a random set of N
  habitats
  Calculate each habitat's HSI and sort the population from
  the best to the worst by their HSIs
  Calculate the immigration rate
  for t = 1 to MaxDT do
    for i = 1 to N do
      if i == N then
        for j = 1 to D do
          Perform the opposition learning approach
          by Eq. (10)
        end for
      else
        for j = 1 to D do
          if rand < λ_i then
          Select the emigration habitat with the example
          learning selection
            Update H_i(SIV_j) by by Eq. (8)
          else
            Update H_i(SIV_j) by by Eq. (9)
          end if
        end for
      end if
    end for
    Boundary constraints
    Parallel calculate each habitat's HSI
    Perform the greed selection
    Sort the population from the best to the worst by their
    HSIs
  end for
End
```

**TABLE 1. Benchmark functions used in our experimental tests.**

| Name | Function | Search Range | Min |
|---|---|---|---|
| Sphere | $f_1(x) = \sum_{i=1}^{D} x_i^2$ | $[-100, 100]^D$ | 0 |
| Tablet | $f_2(x) = 10^6 x_1^2 + \sum_{i=2}^{D} x_i^2$ | $[-100, 100]^D$ | 0 |
| Schwefel 2.22 | $f_3(x) = \sum_{i=1}^{D} \lvert x_i\rvert + \prod_{i=1}^{D} \lvert x_i\rvert$ | $[-10, 10]^D$ | 0 |
| Schwefel 1.2 | $f_4(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j\right)^2$ | $[-100, 100]^D$ | 0 |
| Zakharow | $f_5(x) = \sum_{i=1}^{D} x_i^2 + \left(\sum_{i=1}^{D} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{D} 0.5ix_i\right)^4$ | $[-5, 10]^D$ | 0 |
| Griewank | $f_6(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$ | $[-600, 600]^D$ | 0 |
| Ackley | $f_7(x) = 20 + e - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D} \cos 2\pi x_i\right)$ | $[-32, 32]^D$ | 0 |
| Rastrigin | $f_8(x) = \sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$ | $[-5.12, 5.12]^D$ | 0 |
| Sum Power | $f_9(x) = \sum_{i=1}^{D} \lvert x_i\rvert^{(i+1)}$ | $[-1, 1]^D$ | 0 |
| Alpine | $f_{10}(x) = \sum_{i=1}^{D} \lvert x_i \sin(x_i + 0.1x_i)\rvert$ | $[-10, 10]^D$ | 0 |
| Shifted Sphere | $f_{11}(x) = \sum_{i=1}^{D} z_i^2 - 450, z = x - o$ | $[-100, 100]^D$ | -450 |
| Shifted Rosenbrock | $f_{12}(x) = \sum_{i=1}^{D-1} \left[100\left(z_i^2 - z_{i+1}\right)^2 + (z_i - 1)^2\right] + 390, z = x - o + 1$ | $[-100, 100]^D$ | -390 |
| Shifted Schwefel 2.21 | $f_{13}(x) = \max_i \{\lvert z_i\rvert, 1 \leq i \leq D\} - 450, z = x - o$ | $[-100, 100]^D$ | -450 |
| Shifted Griewank | $f_{14}(x) = 1 + \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos(\frac{z_i}{\sqrt{i}}) - 180, z = x - o$ | $[-600, 600]^D$ | -180 |
| Shifted Ackley | $f_{15}(x) = 20 + e - 20\exp[-\frac{1}{5}\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}] - \exp[\frac{1}{D}\sum_{i=1}^{D} \cos(2\pi z_i)] - 140, z = x - o$ | $[-32, 32]^D$ | -140 |
| Rotated Sphere | $f_{16}(x) = \sum_{i=1}^{D} z_i^2, z = x * M$ | $[-100, 100]^D$ | 0 |
| Rotated Rastrigin | $f_{17}(x) = \sum_{i=1}^{D} \left[z_i^2 - 10\cos(2\pi z_i) + 10\right], z = x * M$ | $[-5.12, 5.12]^D$ | 0 |
| Roteted Griewank | $f_{18}(x) = \frac{1}{4000}\sum_{i=1}^{D} z_i^2 - \prod_{i=1}^{D} \cos\left(\frac{z_i}{\sqrt{i}}\right) + 1, z = x * M$ | $[-600, 600]^D$ | 0 |

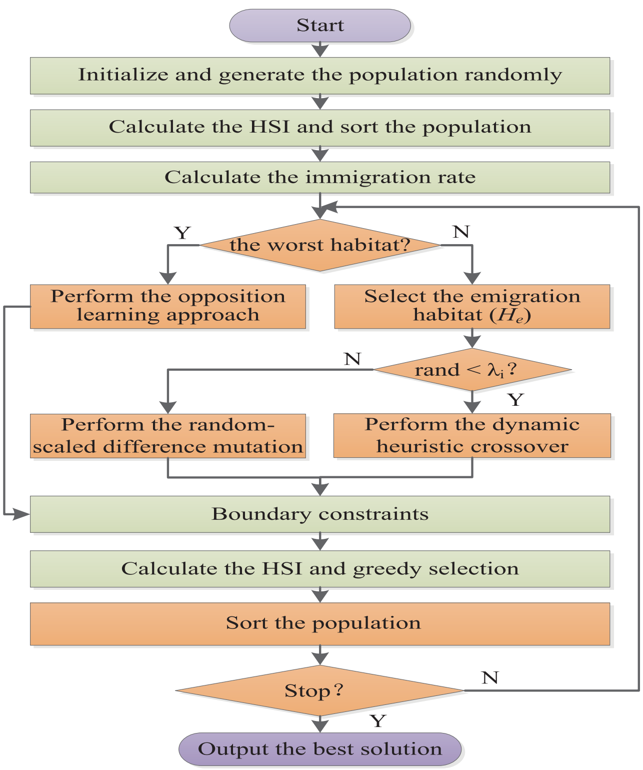



**FIGURE 3. Flowchart of WRBBO.**

strengthen the global search ability in WRBBO. (3) BBO uses the elitist strategy to keep the best habitats, while WRBBO adopts the greedy selection to update the population to omit one sorting step. (4) BBO uses the roulette wheel selection to select the emigration habitat, while the example learning selection is used and the immigration rate calculation step is moved outside of the whole loop in WRBBO. In general, WRBBO enhances the search ability and reduces the computational complexity to obtain higher optimization efficiency.

## IV. EXPERIMENT RESULTS AND ANALYSIS

This section is employed to verify the optimization efficiency of WRBBO. A lot of experiments are conducted to verify WRBBO. Subsection IV-A is the experimental setting. From Subsection IV-B to Subsection IV-D, the optimization performance of WRBBO is investigated and the running speed is tested in Subsection IV-E. The applications of WRBBO to clustering optimization and medical image segmentation are shown in Subsection IV-F. The experimental results are listed in Tables 2-6 and 8-9, they are all from our experiments, and the best are in bold.

### A. EXPERIMENTAL SETTING

In order to verify WRBBO, a large number of experiments are made on some complex functions from CEC-2013 test set [30] and 18 different types of high-dimensional classic benchmark functions, including the unimodal functions ($f_1$–$f_5$), the multimodal functions ($f_6$–$f_{10}$), the shifted functions ($f_{11}$–$f_{15}$) and the rotated functions ($f_{16}$–$f_{18}$). The detailed information of these classic functions is listed in Table 1. The unimodal function is used to evaluate the exploitation ability of the algorithm, the multimodal function is used to evaluate the exploration ability of the algorithm and the shifted and rotated functions are used to evaluate the ability of the algorithm to solve the complex optimization problem. To be fair, the common parameters of WRBBO and the other comparison algorithms are all set to the same. For the classic benchmark functions, the population size (*N*) is 20, the independent run number (*Num*) is 30. On the 30-dimensional functions, *MaxDT* is 2500 and Maximum Number of Function Evaluation (*MNFE*) is *MaxDT*∗*N*. On the 50-dimensional functions, *MaxDT* is 3500 and *MNFE* is *MaxDT*∗*N*. For CEC-2013 test set, according to the recommendation of [30], *Num* is 51, *MaxDT* is 3000 on the 30-dimensional functions, *N* is 100 and *MNFE* is *MaxDT*∗*N*. For WRBBO, the maximum immigration rate *I* is 1, and the other parameters of the comparison algorithms are referred to the corresponding reference. All experiments are implemented on PC with 3.1 GHz CPU and 4GB RAM memory under a Microsoft Windows 7 operating system. The programming language is MATLAB R2014a.

### B. COMPARISON EXPERIMENTS ON 18 CLASSIC FUNCTIONS

#### 1) COMPARISON WITH BBO VARIANTS

In this experiment group, WRBBO is compared with other BBO variants on the 30-dimensional functions. We select more competitive BBO variants as comparison algorithms to compare WRBBO, and the comparison algorithms include EBO [15], Biogeography-Based Optimization algorithm with Mutation strategies (BBOM) [17], Biogeography-based Learning Particle Swarm Optimization (BLPSO) [31], Blended BBO (BIBBO) [16], Laplacian BBO (LxBBO) [32] and hybrid DE with BBO (DEBBO) [20] with WRBBO. Although WRBBO is an improved algorithm of BBO, the original BBO is used to solve discrete problems. The improved algorithms of BBO have been proved to be more effective than BBO. Therefore, the comparison algorithms do not include BBO. The common parameters of the 6 algorithms are referred to Subsection IV-A, the other parameters are referred to the corresponding references. The results are shown in Table 2.

From Table 2, WRBBO ranks the first on 16 functions, ranks the second and third on 1 function, respectively. BBOM ranks the first on 2 functions and other algorithms don't rank the first. What's more, WRBBO obtains the optima value (0) in all the unimodal functions, that shows the dynamic heuristic crossover enhances the exploitation ability of the algorithm. On the multimodal functions ($f_6$, $f_7$-$f_9$), WRBBO obtains the optima value (0) to verify. This shows that the random-scaled differential mutation and the worst opposition learning approach make WRBBO obtain better global search ability. On the shifted functions ($f_{16}$, $f_{18}$), WRBBO also obtains the optima value (0). This shows that WRBBO has the stronger ability to solve the complex functions. From the Ave.Rank, WRBBO obtains the first, followed by LxBBO, BBOM and BLPSO, EBO, DEBBO and BIBBO. In general, WRBBO has the better performance compared with the six BBO variants.

#### 2) COMPARISON WITH OTHER STATE-OF-THE-ART IOAs

In this experiment group, WRBBO is compared with the other state-of-the-art IOAs on the 30-dimensional and the 50-dimensional functions. The selected comparison algorithms are Heterogeneous Comprehensive Learning Particle Swarm Optimization (HCLPSO) [33], Self Regulating Particle Swarm Optimization (SRPSO) [34] which are PSO variants, Ensemble of mutation strategies and control Parameters with the DE (EPSDE) [35] Sinusoidal Differential Evolution (SinDE) [36] which are DE variants, Adaptive Cuckoo search (ACS) [37] which is a CS variant, and hybrid the standard FPA with the Clonal Selection Algorithm (MFPA) [38] which is a FPA variant. They are highly competitive and have certain representativeness in their same algorithm types. For MFPA, according to the corresponding references, *N* is 50, while *MNFE* is set fairly to the same as those of the other comparison algorithms, 50,000 on the 30-dimensional functions and 70,000 on the 50-dimensional functions. The common parameter settings of these algorithms are referred to Subsection IV-A. The other parameters setting of these algorithms are associated with their corresponding references. The results are shown in Tables 3 and 4.

From Table 3, WRBBO ranks the first in all the unimodal functions and obtains the optimal value (0). On the multimodal functions, the results of WRBBO are better than those of the 6 comparison algorithms. On the shifted and the rotated functions, WRBBO has the better results compared with the 6 comparison algorithms, except for $f_{12}$. In addition, WRBBO obtains the optima value (0) on $f_6$, $f_8$, $f_9$, $f_{16}$ and $f_{18}$. From the ranking, WRBBO obtains 17 times ranking the first and 1 time ranking the second. EPSDE obtains 1 time ranking the first. The other algorithms don't obtain ranking the first. The average ranking of WRBBO is 1.06 also ranking the first, followed by EPSDE, SRPSO, SinDE, ACS, MFPA and HCLPSO. The average ranking graph is shown in Figure 4(a), and the average ranking difference between the comparison algorithms is sharply clear.

From Table 4, WRBBO ranks the first on 17 functions except for $f_{12}$ and it obtains the optima value (0) on $f_1$–$f_6$, $f_8$, $f_9$, $f_{16}$ and $f_{18}$ for the Mean and Std values. This also shows that WRBBO gets better global search ability and local search ability. On ranking, WRBBO obtains 17 times ranking the first and one time ranking the second, the same as in the case of the 30-dimensional functions. The average ranking is shown in Figure 4(b), WRBBO obtains the first again on the 50-dimensional classic functions.

On Std, from Table 2, WRBBO obtains the best value of 0 on $f_1$-$f_6$, $f_8$-$f_9$, $f_{16}$ and $f_{18}$, on $f_7$, $f_{10}$-$f_{11}$, $f_{13}$, $f_{15}$ and $f_{17}$, WRBBO is also better than other comparison algorithms. From Tables 3 and 4, On the value of Std, WRBBO has the better value than the comparison algorithms on 17 functions except for $f_{12}$. This all proves WRBBO outperforms the comparison algorithms in robustness. In addition, from

**TABLE 2.** Comparison results between WRBBO and BBO variants on 18 classic functions ($D = 30$).

| Function | Value | EBO | BBOM | BLPSO | BIBBO | LxBBO | DEBBO | WRBBO |
|---|---|---|---|---|---|---|---|---|
| $f_1$ | Mean | 6.7273e-16 | 4.5543e-07 | 6.9398e-19 | 2.432e+01 | 5.6984e-12 | 2.4279e-04 | **0** |
| | Std | 3.6686e-15 | 2.4075e-07 | 1.7478e-18 | 7.5546e+00 | 2.0911e-11 | 8.5277e-05 | **0** |
| | Rank | 3 | 5 | 2 | 7 | 4 | 6 | **1** |
| $f_2$ | Mean | 1.4512e-15 | 7.9724e-07 | 4.1928e-18 | 7.0665e+05 | 2.4281e-10 | 4.5209e-04 | **0** |
| | Std | 6.4873e-15 | 4.3647e-07 | 1.8906e-17 | 1.4128e+06 | 1.255e-09 | 1.1153e-04 | **0** |
| | Rank | 3 | 5 | 2 | 7 | 4 | 6 | **1** |
| $f_3$ | Mean | 4.0248e-12 | 1.2573e-03 | 1.7873e-09 | 1.863e+00 | 2.711e-06 | 1.4119e-03 | **0** |
| | Std | 2.1249e-11 | 3.2540e-04 | 9.1506e-09 | 3.1722e-01 | 1.2662e-05 | 2.513e-04 | **0** |
| | Rank | 2 | 5 | 3 | 7 | 4 | 6 | **1** |
| $f_4$ | Mean | 3.2417e+02 | 6.1286e+03 | 2.9084e+00 | 1.7346e+04 | 8.7641e+01 | 1.6842e+04 | **0** |
| | Std | 2.3495e+02 | 2.1032e+03 | 4.3547e+00 | 3.3656e+03 | 4.5154e+01 | 2.2604e+03 | **0** |
| | Rank | 4 | 5 | 2 | 7 | 3 | 6 | **1** |
| $f_5$ | Mean | 7.7756e-02 | 1.6728e+02 | 3.4956e-03 | 6.5192e+02 | 1.6150e+01 | 3.6960e+02 | **0** |
| | Std | 1.0468e-01 | 3.0850e+01 | 4.4481e-03 | 1.1534e+02 | 1.2793e+01 | 4.3919e+01 | **0** |
| | Rank | 3 | 5 | 2 | 7 | 4 | 6 | **1** |
| $f_6$ | Mean | 1.0801e-02 | 3.5515e-04 | 7.5497e-03 | 1.2189e+00 | 3.9997e-02 | 2.8801e-03 | **0** |
| | Std | 1.5908e-02 | 1.0882e-03 | 9.0431e-03 | 6.798e-02 | 3.8574e-02 | 4.6875e-03 | **0** |
| | Rank | 5 | 2 | 4 | 7 | 6 | 3 | **1** |
| $f_7$ | Mean | 6.9548e-02 | 3.7956e-04 | 2.8542e-10 | 2.5125e+00 | 3.8548e-06 | 3.8406e-03 | **1.1842e-15** |
| | Std | 2.6630e-01 | 6.8988e-05 | 6.0596e-10 | 2.8342e-01 | 1.2565e-05 | 6.7917e-04 | **1.7034e-15** |
| | Rank | 6 | 4 | 2 | 7 | 3 | 5 | **1** |
| $f_8$ | Mean | 8.3577e+00 | 7.3310e+00 | 3.3663e+01 | 8.5164e+00 | 9.0899e-01 | 3.4400e+01 | **0** |
| | Std | 4.1692e+00 | 2.0651e+00 | 1.2591e+01 | 2.0467e+00 | 1.0277e+00 | 3.252e+00 | **0** |
| | Rank | 4 | 3 | 6 | 5 | 2 | 7 | **1** |
| $f_9$ | Mean | 1.0620e-26 | 3.0262e-27 | 3.0543e-56 | 7.985e-05 | 1.6402e-49 | 1.0125e-20 | **0** |
| | Std | 5.7405e-26 | 4.2913e-27 | 1.6673e-55 | 1.4278e-04 | 6.1588e-49 | 1.0786e-20 | **0** |
| | Rank | 5 | 4 | 2 | 7 | 3 | 6 | **1** |
| $f_{10}$ | Mean | 2.8397e-12 | 3.6526e-01 | 3.1787e-10 | 1.6947e-01 | 4.2133e-05 | 7.1927e-02 | **2.7906e-13** |
| | Std | 1.3356e-11 | 2.0763e-01 | 1.0379e-09 | 5.3338e-02 | 1.5552e-04 | 1.1726e-01 | **1.2193e-12** |
| | Rank | 2 | 7 | 3 | 6 | 4 | 5 | **1** |
| $f_{11}$ | Mean | -4.5000e+02 | -4.5000e+02 | 2.5302e+02 | 2.9142e+01 | -4.5000e+02 | -4.5000e-02 | **-4.5000e+02** |
| | Std | 3.9368e-13 | 3.1870e-07 | 3.5270e+02 | 1.2712e+01 | 1.5028e-09 | 2.1542e-05 | **2.5856e-14** |
| | Rank | 2 | 4 | 7 | 6 | 3 | 5 | **1** |
| $f_{12}$ | Mean | 6.4674e+02 | **4.6539e+02** | 2.3252e+05 | 1.8268e+04 | 2.6560e+03 | 5.2894e+02 | 5.2456e+02 |
| | Std | 3.1469e+02 | **5.5722e+01** | 7.0916e+05 | 1.1028e+04 | 3.2797e+03 | 9.7124e+01 | 2.8166e+02 |
| | Rank | 4 | **1** | 7 | 6 | 5 | 3 | 2 |
| $f_{13}$ | Mean | -4.2958e+02 | -4.4851e+02 | 4.5416e+00 | 1.3665e+01 | -4.4667e+02 | -4.4459e+02 | **-4.4988e+02** |
| | Std | 7.2525e+00 | 1.4572e+00 | 1.6047e+00 | 2.8121e+00 | 8.5259e-01 | 7.0022e-01 | **8.4570e-02** |
| | Rank | 5 | 2 | 6 | 7 | 3 | 4 | **1** |
| $f_{14}$ | Mean | -1.7999e+02 | **-1.8000e+02** | 1.2477e+00 | 1.2107e+00 | -1.7996e+02 | -1.8000e+02 | -1.8000e+02 |
| | Std | 1.9594e-02 | **1.4558e-03** | 1.5055e+00 | 9.2924e-02 | 4.088e-02 | 3.3101e-04 | 2.2544e-03 |
| | Rank | 4 | **1** | 7 | 6 | 5 | 2 | 3 |
| $f_{15}$ | Mean | -1.3961e+02 | -1.4000e+02 | 1.9513e+01 | 2.5426e+00 | -1.4000e+02 | -1.4000e+02 | **-1.4000e+02** |
| | Std | 8.4721e-01 | 1.8143e-04 | 2.2921e-01 | 2.4487e-01 | 1.3809e-05 | 3.3375e-04 | **5.0623e-14** |
| | Rank | 5 | 3 | 7 | 6 | 2 | 4 | **1** |
| $f_{16}$ | Mean | 2.7250e-02 | 3.2663e-04 | 1.2068e-08 | 3.1532e+02 | 1.0199e-04 | 1.0145e+00 | **0** |
| | Std | 1.1436e-01 | 1.9066e-04 | 2.2073e-08 | 1.0524e+02 | 2.2026e-04 | 3.7996e-01 | **0** |
| | Rank | 5 | 4 | 2 | 7 | 3 | 6 | **1** |
| $f_{17}$ | Mean | 6.9973e+01 | 9.4505e+01 | 6.2815e+01 | 1.8853e+02 | 1.4066e+02 | 1.8455e+02 | **2.1365e+01** |
| | Std | 2.4329e+01 | 1.5416e+01 | 1.9960e+01 | 3.1215e+01 | 3.9155e+01 | 1.2299e+01 | **2.6668e+01** |
| | Rank | 3 | 4 | 2 | 7 | 5 | 6 | **1** |
| $f_{18}$ | Mean | 7.0912e-01 | 1.6662e-01 | 1.8651e-02 | 7.0869e+00 | 3.6653e-02 | 1.073e+00 | **0** |
| | Std | 3.3833e-01 | 6.2223e-02 | 1.5793e-02 | 3.3579e+00 | 1.8679e-02 | 2.1558e-02 | **0** |
| | Rank | 5 | 4 | 2 | 7 | 3 | 6 | **1** |
| Count | | 0 | 2 | 0 | 0 | 0 | 0 | 16 |
| Ave.Rank | | 3.89 | 3.78 | 3.78 | 6.61 | 3.67 | 5.11 | 1.17 |
| Total.Rank | | 5 | 3 | 3 | 7 | 2 | 6 | 1 |

Tables 3 and 4, with the increase of dimensions from 30 to 5o, WRBBO is still better than the comparison algorithms on both Mean and Std. It shows that WRBBO has better scalability compared with the comparison algorithms.

### *C. COMPARISON ON CEC-2013 TEST SET*

To further verify the optimization ability of WRBBO to cope with the complex problems, many experiments are made on some CEC-2013 benchmark functions, where $F_1$ and $F_2$ are unimodal functions, $F_6$, $F_7$, $F_{10}$, $F_{11}$, $F_{14}$ and $F_{17}$ are basic multimodal functions, and $F_{22}$, $F_{24}$, $F_{26}$ and $F_{27}$ are composition functions. The comparison algorithms are as follows: BIBBO [16], DEBBO [20], BLPSO [31], BBOM [17] and Efficient and Merged BBO (EMBBO) [21]. The parameters of these algorithms are set as Subsection IV-A. The results are shown in Table 5. From Table 5, on 7 of all the selected functions, WRBBO are better than the 5 comparison algorithms. The average ranking of these algorithms is shown in Figure 5(a) and the ranking statistics of each algorithms is shown in Figure 5(b), it uses different colors to describe the number of various rankings. From Figure 5(a), WRBBO obtains the first on the average ranking (2.00), followed by DEBBO, EMBBO, BBOM, BIBBO and BLPSO. From Figure 5(b), WRBBO obtains 7 times ranking the first, EMBBO

**TABLE 3.** Comparison results between WRBBO and other IOAs on 18 classic functions ($D = 30$).

| Function | Value | HCLPSO | SRPSO | EPSDE | SinDE | ACS | HFPSO | WRBBO |
|---|---|---|---|---|---|---|---|---|
| $f_1$ | Mean | 3.7233e-06 | 5.8762e-09 | 1.5119e-43 | 3.2361e-07 | 1.2046e-09 | 9.7422e-10 | **0** |
| | Std | 2.5825e-06 | 1.7016e-08 | 6.7493e-43 | 1.7724e-06 | 2.7422e-09 | 1.3373e-09 | **0** |
| | Rank | 7 | 5 | 2 | 6 | 4 | 3 | **1** |
| $f_2$ | Mean | 1.7290e-05 | 7.1621e-09 | 2.5462e-42 | 9.0297e-11 | 2.5418e-04 | 3.2910e-09 | **0** |
| | Std | 1.3774e-05 | 1.1260e-08 | 1.2499e-41 | 4.9457e-10 | 3.8273e-04 | 6.0966e-09 | **0** |
| | Rank | 6 | 5 | 2 | 3 | 7 | 4 | **1** |
| $f_3$ | Mean | 2.6243e-04 | 2.1567e-06 | 1.3362e-27 | 5.6683e-13 | 5.7686e-05 | 2.2739e-01 | **0** |
| | Std | 1.0370e-04 | 2.0006e-06 | 3.0141e-27 | 3.1033e-12 | 7.5298e-05 | 1.2454e+00 | **0** |
| | Rank | 6 | 4 | 2 | 3 | 5 | 7 | **1** |
| $f_4$ | Mean | 2.0063e+03 | 5.7105e+01 | 1.6552e+00 | 4.5641e+02 | 1.8088e+00 | 1.3208e+01 | **0** |
| | Std | 8.3423e+02 | 2.9831e+01 | 8.2231e+00 | 2.6338e+02 | 7.6778e-01 | 1.3325e+01 | **0** |
| | Rank | 7 | 5 | 2 | 6 | 3 | 4 | **1** |
| $f_5$ | Mean | 7.9891e+00 | 2.5370e-01 | 2.3488e+00 | 8.7617e+00 | 3.6970e-01 | 1.0603e-02 | **0** |
| | Std | 4.2024e+00 | 1.2028e-01 | 8.5515e+00 | 3.6766e+00 | 1.9140e-01 | 1.5278e-02 | **0** |
| | Rank | 6 | 3 | 5 | 7 | 4 | 2 | **1** |
| $f_6$ | Mean | 1.1474e-02 | 8.6138e-03 | 1.8721e-02 | 1.9719e-03 | 5.4146e-03 | 9.0269e-03 | **0** |
| | Std | 1.9655e-02 | 1.1501e-02 | 3.3418e-02 | 3.7408e-03 | 1.0866e-02 | 1.4159e-02 | **0** |
| | Rank | 6 | 3 | 7 | 5 | 2 | 4 | **1** |
| $f_7$ | Mean | 6.4782e-02 | 5.3000e-06 | 8.2753e-01 | 3.8836e-09 | 1.1382e+00 | 1.0670e+01 | **1.1842e-15** |
| | Std | 1.3346e-01 | 5.6139e-06 | 8.2099e-01 | 1.4243e-08 | 6.6258e-01 | 8.8332e+00 | **1.7034e-15** |
| | Rank | 4 | 3 | 5 | 2 | 6 | 7 | **1** |
| $f_8$ | Mean | 3.8711e+00 | 2.9753e+01 | 1.6583e-01 | 9.6059e+00 | 3.6314e+01 | 5.9266e+01 | **0** |
| | Std | 1.4816e+00 | 7.6762e+00 | 4.5881e-01 | 3.7650e+00 | 9.1769e+00 | 2.0333e+01 | **0** |
| | Rank | 3 | 5 | 2 | 4 | 6 | 7 | **1** |
| $f_9$ | Mean | 8.4204e-30 | 6.8334e-37 | 3.4014e-66 | 2.1474e-08 | 1.9613e-13 | 2.5485e-30 | **0** |
| | Std | 4.2155e-29 | 3.1153e-36 | 1.2355e-65 | 1.0844e-07 | 4.0265e-13 | 1.3905e-29 | **0** |
| | Rank | 5 | 3 | 2 | 7 | 6 | 4 | **1** |
| $f_{10}$ | Mean | 6.1050e-03 | 1.1335e-06 | 1.9562e-12 | 9.0809e-07 | 1.5327e+00 | 4.8974e-07 | **2.7906e-13** |
| | Std | 6.3798e-03 | 8.8858e-07 | 9.2340e-12 | 2.9177e-06 | 1.7138e+00 | 3.4218e-07 | **1.2193e-12** |
| | Rank | 6 | 5 | 2 | 3 | 7 | 4 | **1** |
| $f_{11}$ | Mean | -3.1423e+02 | -4.5000e+02 | -4.5000e+02 | -4.5000e+02 | -4.5000e+02 | 2.6823e-09 | **-4.5000e+02** |
| | Std | 1.9973e+02 | 7.7602e-08 | 8.0389e-14 | 7.6846e-14 | 3.4188e-09 | 6.9561e-09 | **2.5856e-14** |
| | Rank | 6 | 3 | 2 | 4 | 5 | 7 | **1** |
| $f_{12}$ | Mean | 1.4107e+06 | 7.0751e+02 | **4.2174e+02** | 9.2268e+02 | 6.2061e+02 | 2.8551e+03 | 5.2456e+02 |
| | Std | 5.5427e+06 | 4.3552e+02 | **2.8568e+01** | 1.6563e+03 | 2.8092e+02 | 1.3191e+04 | 2.8166e+02 |
| | Rank | 7 | 4 | **1** | 5 | 3 | 6 | 2 |
| $f_{13}$ | Mean | -4.4470e+02 | -4.4898e+02 | -4.0901e+02 | -4.2505e+02 | -4.4168e+02 | 9.9109e+00 | **-4.4988e+02** |
| | Std | 1.5409e+00 | 2.9648e-01 | 7.7460e+00 | 1.2200e+01 | 6.2846e+00 | 1.7815e-01 | **8.4570e-02** |
| | Rank | 3 | 2 | 6 | 5 | 4 | 7 | **1** |
| $f_{14}$ | Mean | -1.7780e+02 | -1.7999e+02 | -1.7990e+02 | -1.8000e+02 | -1.8000e+02 | 6.8507e-02 | **-1.8000e+02** |
| | Std | 1.8721e+00 | 1.5148e-02 | 1.9051e-02 | 2.5347e-03 | 7.4414e-03 | 1.0995e-01 | **2.2544e-03** |
| | Rank | 6 | 4 | 5 | 2 | 3 | 7 | **1** |
| $f_{15}$ | Mean | -1.3987e+02 | -1.4000e+02 | -1.3962e+02 | -1.4000e+02 | -1.2397e+02 | 8.9522e-02 | **-1.4000e+02** |
| | Std | 2.3682e-01 | 1.3839e-05 | 6.0189e-01 | 1.3977e-06 | 7.7357e+00 | 3.4003e-01 | **5.0623e-14** |
| | Rank | 4 | 3 | 5 | 2 | 6 | 7 | **1** |
| $f_{16}$ | Mean | 1.3286e+00 | 3.9762e-05 | 9.5539e-20 | 4.3061e-01 | 4.6816e-06 | 2.5901e-05 | **0** |
| | Std | 9.0898e-01 | 5.0607e-05 | 2.4508e-19 | 2.3552e+00 | 3.5533e-06 | 3.1383e-05 | **0** |
| | Rank | 7 | 5 | 2 | 6 | 4 | 3 | **1** |
| $f_{17}$ | Mean | 8.2341e+01 | 1.4857e+02 | 8.3095e+01 | 5.8979e+01 | 1.5358e+02 | 1.0278e+02 | **2.1365e+01** |
| | Std | 3.4599e+01 | 4.8575e+01 | 1.7904e+01 | 1.4860e+01 | 2.5471e+01 | 3.1351e+01 | **2.6668e+01** |
| | Rank | 3 | 6 | 4 | 2 | 7 | 5 | **1** |
| $f_{18}$ | Mean | 1.2916e+00 | 6.0813e-02 | 1.8944e-02 | 5.7977e-02 | 4.4359e-02 | 6.5643e-02 | **0** |
| | Std | 1.8344e-01 | 3.3955e-02 | 1.7596e-02 | 9.2741e-02 | 3.0715e-02 | 5.2270e-02 | **0** |
| | Rank | 7 | 5 | 2 | 4 | 3 | 6 | **1** |
| Count | | 0 | 0 | 1 | 0 | 0 | 0 | 17 |
| Ave.Rank | | 5.50 | 4.06 | 3.22 | 4.22 | 4.72 | 5.22 | 1.06 |
| Total.Rank | | 7 | 3 | 2 | 4 | 5 | 6 | 1 |

obtains 4 times ranking the first, DEBBO obtains 1 time ranking the first, and BIBBO, BLPSO and BBOM obtains no ranking the first. It can be seen that WRBBO has better optimization performance than the comparison algorithms.

### *D. WILCOXON SIGNED-RANK TEST ANALYSIS*

Wilcoxon signed-rank test is a nonparametric test method [39], and it is used to test statistically the performance of WRBBO compared with the comparison algorithms. The software is IBM SPSS Statistics 19. In this section, Wilcoxon signed-rank test is performed only on the 30-dimensional and the 50-dimensional classic functions. The data is taken from Tables 2–4. The Wilcoxon signed-rank test results are shown in Table 6. $R^+$ refers to the sum of ranks for the problems in which WRBBO outperformed the comparison algorithm and $R^-$ refers to the sum of ranks for the opposite. When WRBBO and the comparison algorithm obtain the equal optimization performance, the corresponding ranks are split evenly to $R^+$ and $R^-$. The $p$ values can be computed according to the $R^+$ and $R^-$ values. '$n/w/t/l$' means the number of the benchmark functions are $n$ and WRBBO wins on $w$ functions, ties on $t$ functions and loses on $l$ functions. The standard is as follows: the difference of both algorithms is not significant when $p > 0.05$ and the difference of both algorithms is

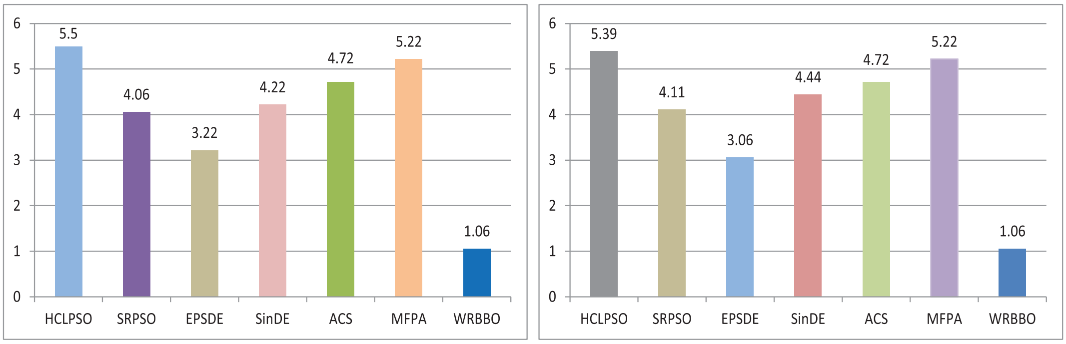
**TABLE 4. Comparison results between WRBBO and other IOAs on 18 classic functions (*D*=50).**

| Function | Value | HCLPSO | SRPSO | EPSDE | SinDE | ACS | MFPA | WRBBO |
|---|---|---|---|---|---|---|---|---|
| $f_1$ | Mean | 3.5511e-04 | 7.8558e-05 | 1.1120e-26 | 2.8668e-01 | 2.6354e-07 | 3.5922e-06 | **0** |
| | Std | 2.2702e-04 | 2.9859e-04 | 5.8106e-26 | 1.5702e+00 | 3.5144e-07 | 3.5347e-06 | **0** |
| | Rank | 6 | 5 | 2 | 7 | 3 | 4 | **1** |
| $f_2$ | Mean | 1.1382e-03 | 1.1443e-04 | 8.0691e-28 | 1.0120e-04 | 1.8591e-02 | 2.1081e-05 | **0** |
| | Std | 4.8698e-04 | 3.4107e-04 | 2.8566e-27 | 4.7840e-04 | 1.8976e-02 | 3.2890e-05 | **0** |
| | Rank | 6 | 5 | 2 | 4 | 7 | 3 | **1** |
| $f_3$ | Mean | 4.9923e-03 | 1.3907e-03 | 4.6998e-19 | 3.5703e-10 | 4.1201e-03 | 1.7205e-04 | **0** |
| | Std | 1.6862e-03 | 3.0159e-03 | 7.4935e-19 | 1.2068e-09 | 8.4970e-03 | 4.6505e-04 | **0** |
| | Rank | 7 | 5 | 2 | 3 | 6 | 4 | **1** |
| $f_4$ | Mean | 1.3614e+04 | 1.2135e+03 | 2.3773e+03 | 5.4904e+03 | 8.7990e+01 | 4.6403e+02 | **0** |
| | Std | 3.5726e+03 | 3.0932e+02 | 6.7686e+03 | 1.3395e+03 | 3.9872e+01 | 2.1884e+02 | **0** |
| | Rank | 5 | 4 | 6 | 7 | 2 | 3 | **1** |
| $f_5$ | Mean | 8.0981e+01 | 5.7836e+00 | 4.6494e+01 | 1.2862e+02 | 2.4299e+01 | 2.8577e+00 | **0** |
| | Std | 2.4497e+01 | 9.2695e-01 | 9.5905e+01 | 2.8700e+01 | 7.6228e+00 | 2.4687e+00 | **0** |
| | Rank | 6 | 3 | 5 | 7 | 4 | 2 | **1** |
| $f_6$ | Mean | 1.4769e-02 | 5.1104e-03 | 5.0461e-02 | 6.5220e-03 | 3.9422e-03 | 1.7635e-02 | **0** |
| | Std | 2.1406e-02 | 9.6085e-03 | 9.4428e-02 | 1.9573e-02 | 5.9125e-03 | 2.3222e-02 | **0** |
| | Rank | 6 | 4 | 3 | 5 | 2 | 7 | **1** |
| $f_7$ | Mean | 1.1204e+00 | 6.0801e-04 | 2.1274e+00 | 8.3108e-04 | 2.6196e+00 | 1.1424e+01 | **1.5395e-15** |
| | Std | 4.1035e-01 | 4.4561e-04 | 7.9232e-01 | 3.2726e-03 | 4.9352e-01 | 8.0651e+00 | **1.7906e-15** |
| | Rank | 4 | 2 | 5 | 3 | 6 | 7 | **1** |
| $f_8$ | Mean | 1.5344e+01 | 7.4648e+01 | 2.7974e+00 | 4.1385e+01 | 7.7851e+01 | 9.2200e+01 | **0** |
| | Std | 2.9598e+00 | 4.1049e+01 | 1.1784e+01 | 1.0959e+01 | 2.2986e+01 | 2.6980e+01 | **0** |
| | Rank | 3 | 5 | 2 | 4 | 6 | 7 | **1** |
| $f_9$ | Mean | 2.2158e-30 | 1.4158e-35 | 1.8778e-51 | 3.4105e-12 | 3.3796e-11 | 8.4498e-26 | **0** |
| | Std | 9.9134e-30 | 6.0118e-35 | 5.2344e-51 | 1.6796e-11 | 1.1989e-10 | 3.1872e-25 | **0** |
| | Rank | 4 | 3 | 2 | 6 | 7 | 5 | **1** |
| $f_{10}$ | Mean | 6.3661e-02 | 3.3934e-04 | 3.4480e-08 | 1.4299e-05 | 1.8752e+00 | 2.6086e-05 | **1.7354e-17** |
| | Std | 5.0398e-02 | 5.4088e-04 | 1.8681e-07 | 2.6305e-05 | 2.2133e+00 | 6.0363e-05 | **9.1711e-17** |
| | Rank | 6 | 5 | 2 | 3 | 7 | 4 | **1** |
| $f_{11}$ | Mean | -6.436e+01 | -4.5000e+02 | -4.5000e+02 | -4.4992e+02 | -4.5000e+02 | 7.9045e-06 | **-4.5000e+02** |
| | Std | 3.2607e+02 | 1.0505e-03 | 1.3458e-12 | 4.2004e-01 | 1.6741e-07 | 5.4123e-07 | **3.6566e-14** |
| | Rank | 6 | 4 | 2 | 5 | 3 | 7 | **1** |
| $f_{12}$ | Mean | 1.3635e+07 | 5.5591e+02 | **4.7449e+02** | 9.4146e+05 | 5.6864e+02 | 6.4890e+02 | 5.1392e+02 |
| | Std | 3.3479e+07 | 1.2079e+02 | **4.1930e+01** | 3.1341e+06 | 2.1931e+02 | 1.4724e+03 | 1.4534e+02 |
| | Rank | 7 | 3 | **1** | 6 | 4 | 5 | 2 |
| $f_{13}$ | Mean | -4.3737e+02 | -4.4593e+02 | -3.8536e+02 | -4.0550e+02 | -4.0823e+02 | 3.5121e+01 | **-4.4732e+02** |
| | Std | 1.8279e+00 | 5.3997e-01 | 6.3115e+00 | 1.2512e+01 | 8.4432e+00 | 5.8608e+00 | **9.3298e-01** |
| | Rank | 3 | 2 | 6 | 5 | 4 | 7 | **1** |
| $f_{14}$ | Mean | -1.7540e+02 | -1.7999e +02 | -1.7992e+02 | -1.8000e+02 | -1.8000e+02 | 3.7132e-02 | **-1.8000e+02** |
| | Std | 3.6629e+00 | 1.3579e-02 | 1.7269e-01 | 1.4484e-02 | 5.8682e-03 | 6.5251e-02 | **3.2158e-03** |
| | Rank | 6 | 4 | 5 | 3 | 2 | 7 | **1** |
| $f_{15}$ | Mean | -1.3845e+02 | -1.3996e+02 | -1.3802e+02 | -1.4000e+02 | -1.2186e+02 | 7.3471e-01 | **-1.4000e+02** |
| | Std | 5.8208e-01 | 1.8736e-01 | 9.8024e-01 | 7.2633e-06 | 5.4800e+00 | 2.2967e+00 | **4.4438e-10** |
| | Rank | 4 | 3 | 5 | 2 | 6 | 7 | **1** |
| $f_{16}$ | Mean | 2.5287e+01 | 3.0209e-02 | 5.1528e-12 | 3.7924e-03 | 1.4743e-04 | 5.3048e-02 | **0** |
| | Std | 8.8461e+00 | 4.8138e-02 | 8.9085e-12 | 6.5918e-03 | 8.6909e-05 | 8.2810e-02 | **0** |
| | Rank | 7 | 5 | 2 | 4 | 3 | 6 | **1** |
| $f_{17}$ | Mean | 1.7794e+02 | 3.3404e+02 | 2.1559e+02 | 1.3027e+02 | 2.9738e+02 | 1.7077e+02 | **3.5498e+01** |
| | Std | 4.2833e+01 | 5.7206e+01 | 2.8401e+01 | 2.6747e+01 | 4.5986e+01 | 4.3660e+01 | **5.7035e+01** |
| | Rank | 4 | 7 | 5 | 2 | 6 | 3 | **1** |
| $f_{18}$ | Mean | 3.1250e+00 | 3.7323e-01 | 1.4642e-02 | 2.4580e-01 | 8.7630e-02 | 6.2077e-01 | **0** |
| | Std | 1.0968e+00 | 2.2381e-01 | 1.3820e-02 | 7.3877e-01 | 7.6554e-02 | 2.1142e-01 | **0** |
| | Rank | 7 | 5 | 2 | 4 | 3 | 6 | **1** |
| Count | | 0 | 0 | 1 | 0 | 0 | 0 | 17 |
| Ave. Rank | | 5.39 | 4.11 | 3.06 | 4.44 | 4.72 | 5.22 | 1.06 |
| Total.Rank | | 7 | 3 | 2 | 4 | 5 | 6 | 1 |

significant when $p \leq 0.05$. From Table 6, no matter on the 30-dimensional or the 50-dimensional functions, the values of $p$ are all less than 0.05, so the optimization performance of WRBBO is significantly better than the comparison algorithms.

### *E. CPU TIME*

In this section, to investigate the runtime of WRBBO, we record the runtime of each algorithm on each function from the experiment of Subsection IV-B. Figure 6(a) shows the average runtimes obtained on the 30-dimensional functions from CEC-2013 test set, and Figure 6 (b) and (c) show the average runtimes obtained on the 30-dimensional classic functions and the 50-dimensional classic functions, respectively. The *y*-coordinate is the runtime and its unit is ‘second’(s). From Figure 6 (a), WRBBO’s average runtime is the least (0.4903s), which is EBO’s (2.1767s), BBOM’s (0.988s), BLSPO’s (1.7445s), BIBBO’s (1.1143s), LxBBO’s (1.167s) and DEBBO’s (0.9554s), 22.52%, 49.63%, 53.90%, 44%, 42.01% and 51.32%, respectively. It shows that WRBBO obtains faster speed compared with BBO variants. From Figure 6(b), WRBBO’s average runtime is the least (0.4903s), which is HCLPSO’s (1.5501s), SRPSO’s (1.6276s), EPSDE’s (1.682s), SinDE’s




(a) (b)

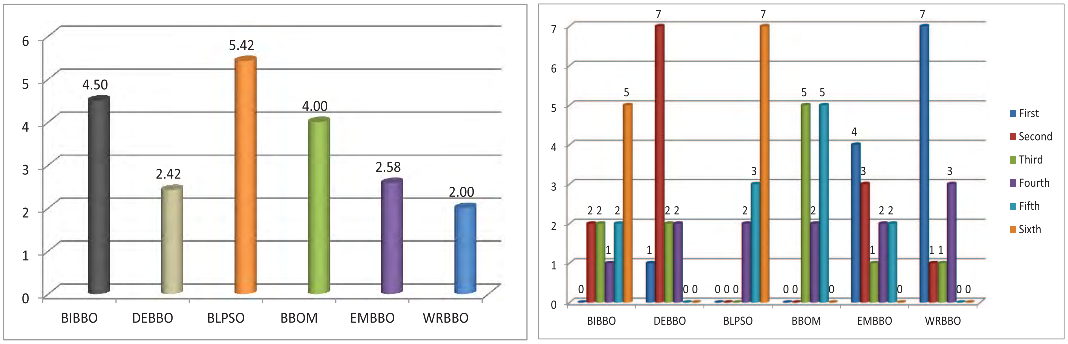
**FIGURE 4.** Average ranking chart. (a) on the 30-dimensional classic functions and. (b) on the 50-dimensional classic functions.

**TABLE 5.** Comparison results between WRBBO and BBO variants on CEC-2013 test set($D$=30).

| | Value | BIBBO | DEBBO | BLPSO | BBOM | EMBBO | WRBBO |
|---|---|---|---|---|---|---|---|
| $F_1$ | Mean | 4.8175e-01 | 2.2292e-14 | 3.5666e-13 | 9.3125e-04 | 3.1208e-14 | **0** |
| | Std | 1.9761e-01 | 6.8286e-14 | 1.4562e-13 | 1.3014e-04 | 7.9022e-14 | **0** |
| | Rank | 6 | 2 | 4 | 5 | 3 | **1** |
| $F_5$ | Mean | 3.9265e-01 | 1.1369e-13 | 2.2923e+00 | 5.9526e-03 | 1.1369e-13 | **8.6937e-14** |
| | Std | 1.1925e-01 | 0 | 1.2739e+01 | 5.4870e-14 | 0 | **4.8704e-14** |
| | Rank | 5 | 2 | 6 | 4 | 2 | **1** |
| $F_6$ | Mean | 4.9756e+01 | 1.4965e+01 | 4.8724e+01 | 1.6519e+01 | 1.9507e+01 | **1.9101e-01** |
| | Std | 2.7104e+01 | 1.4852e-01 | 2.6538e+01 | 1.9206e+00 | 3.8455e+00 | **1.1930e+01** |
| | Rank | 6 | 2 | 5 | 3 | 4 | **1** |
| $F_7$ | Mean | 1.0883e+02 | 4.7727e+00 | 9.0097e+01 | 5.9476e+01 | 1.0095e+02 | **5.3252e-01** |
| | Std | 2.4656e+01 | 1.9494e+00 | 2.4697e+01 | 4.7015e+00 | 1.1118e+01 | **4.3893e-01** |
| | Rank | 6 | 2 | 4 | 3 | 5 | **1** |
| $F_{10}$ | Mean | 1.1315e+01 | **1.2662e-02** | 1.5655e+01 | 2.8327e+00 | 6.7990e+00 | 6.1522e-02 |
| | Std | 3.9788e+00 | **9.1819e-03** | 3.4045e+01 | 6.8274e-01 | 1.9189e+00 | 2.8563e-02 |
| | Rank | 5 | **1** | 6 | 3 | 4 | 2 |
| $F_{11}$ | Mean | 9.3069e-01 | 1.1146e-14 | 2.8573e+02 | 1.4608e+01 | 1.0031e-14 | 1.7948e+00 |
| | Std | 4.5859e-01 | 2.2793e+00 | 5.0984e+01 | 1.8246e+00 | 2.1885e-14 | 1.2430e+00 |
| | Rank | 3 | 2 | 6 | 5 | **1** | 4 |
| $F_{14}$ | Mean | 5.3453e+00 | 4.6024e+01 | 2.2719e+03 | 3.5651e+02 | **4.0822e-03** | 2.6598e+02 |
| | Std | 1.7961e+00 | 1.0398e+01 | 3.7062e+02 | 8.9838e+01 | **8.3481e-03** | 1.2759e+02 |
| | Rank | 2 | 3 | 6 | 5 | **1** | 4 |
| $F_{17}$ | Mean | 3.4541e+01 | 3.3914e+01 | 2.0192e+02 | 6.1703e+01 | **3.0434e+01** | 4.9840e+01 |
| | Std | 3.9143e+01 | 2.9970e+01 | 5.7286e+02 | 7.7.52e+01 | **1.8285e-06** | 2.7792e+00 |
| | Rank | 3 | 2 | 6 | 5 | **1** | 4 |
| $F_{22}$ | Mean | 1.0961e+02 | 1.5986e+02 | 3.0370e+03 | 3.8601e+02 | **2.9925e+01** | 1.4921e+02 |
| | Std | 8.2561e-01 | 4.7040e-01 | 2.9943e+01 | 4.3523e+00 | **2.0743e+01** | 2.7042e+01 |
| | Rank | 2 | 4 | 6 | 5 | **1** | 3 |
| $F_{24}$ | Mean | 2.7577e+02 | 2.0086e+02 | 2.8540e+02 | 2.6295e+02 | 2.8325e+02 | **2.0031e+02** |
| | Std | 8.8244e+00 | 3.9522e-01 | 1.4378e+01 | 3.7728e+00 | 4.5446e+00 | **3.8832e-01** |
| | Rank | 4 | 2 | 6 | 3 | 5 | **1** |
| $F_{26}$ | Mean | 2.6207e+02 | 2.0439e+02 | 2.1850e+02 | 2.0046e+02 | 2.0079e+02 | **2.0039e+02** |
| | Std | 8.3416e+01 | 2.2492e+01 | 5.1104e+01 | 1.2927e-01 | 1.6741e-01 | **1.9934e-01** |
| | Rank | 6 | 4 | 5 | 3 | 2 | **1** |
| $F_{27}$ | Mean | 1.0839e+03 | 7.6147e+02 | 1.0428e+03 | 1.0062e+03 | 4.6674e+02 | **3.5674e+02** |
| | Std | 1.0532e+02 | 2.4090e+02 | 1.0921e+02 | 3.0754e+01 | 2.0363e+02 | **1.3790e+02** |
| | Rank | 6 | 3 | 5 | 4 | 2 | **1** |
| Count | | 0 | 1 | 0 | 0 | 4 | 7 |
| Ave. Rank | | 4.50 | 2.42 | 5.42 | 4.00 | 2.58 | 2.00 |
| Total.Rank | | 5 | 2 | 6 | 4 | 3 | 1 |

(1.3632s), ACS's (1.3691s), MFPA's (2.2376s), 31.63%, 30.12%, 29.15%, 35.97%, 35.81% and 21.91% respectively. In running speed, WRBBO is the fastest among the 7 algorithms. From Figure 6(c), On the 50-dimensional functions, WRBBO's average runtime is also the least (0.9418s), which is HCLPSO's (2.5234s), SRPSO's (2.5373s), EPSDE's (2.5581s), SinDE's (2.0635s), ACS's (2.0563s), MFPA's (3.3155s), 37.32%, 37.12%, 36.82%, 45.64%, 45.80% and 28.41% respectively. This shows that WRBBO reduce the computational complexity such as no mutation operator, example learning selection and so on, while these BBO variants still uses time-consuming approaches such as mutation operation, roulette selection, and so forth, leading to their high-computing load.

From Subsection IV-B1, it verifies that WRBBO has better optimization performance compared with BBO variants on

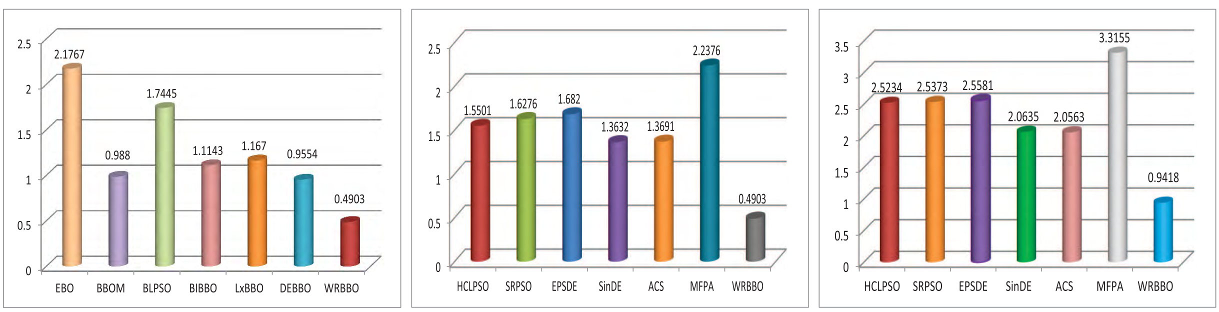



(a) (b)

**FIGURE 5.** Ranking chart on the 30-dimensional functions from CEC-2013. (a) the average ranking and. (b) the ranking statistics.

**TABLE 6.** Wilcoxon signed-rank test results on 18 classic functions.

| | 30-dimensional functions | | | |
|---|---|---|---|---|
| | *p* value | $R^+$ | $R^-$ | *n/w/t/l* |
| WRBBO versus EBO | 1.5259e-05 | 162 | 9 | 18/17/1/0 |
| WRBBO versus BBOM | 4.2725e-03 | 133.5 | 37.5 | 18/14/3/1 |
| WRBBO versus BLPSO | 7.6294e-06 | 171 | 0 | 18/18/0/0 |
| WRBBO versus BIBBO | 7.6294e-06 | 171 | 0 | 18/18/0/0 |
| WRBBO versus LxBBO | 3.0518e-05 | 153.5 | 17.5 | 18/16/2/0 |
| WRBBO versus DEBBO | 3.0518e-05 | 153.5 | 17.5 | 18/16/2/0 |
| WRBBO versus HCLPSO | 1.5793e-03 | 154 | 17 | 18/17/0/1 |
| WRBBO versus SRPSO | 3.0518e-05 | 153.5 | 17.5 | 18/16/2/0 |
| WRBBO versus EPSDE | 3.1586e-03 | 145 | 26 | 18/16/1/1 |
| WRBBO versus SinDE | 6.1035e-05 | 145.5 | 25.5 | 18/15/3/0 |
| WRBBO versus ACS | 3.0518e-05 | 153.5 | 17.5 | 18/16/2/0 |
| WRBBO versus MFPA | 7.6294e-06 | 171 | 0 | 18/18/0/0 |
| | 50-dimensional functions | | | |
| | *p* value | $R^+$ | $R^-$ | *n/w/t/l* |
| WRBBO versus HCLPSO | 7.6294e-06 | 171 | 0 | 18/18/0/0 |
| WRBBO versus SRPSO | 1.5259e-05 | 162 | 9 | 18/17/1/0 |
| WRBBO versus EPSDE | 1.3428e-03 | 149 | 22 | 18/16/1/1 |
| WRBBO versus SinDE | 3.0518e-05 | 153.5 | 17.5 | 18/16/2/0 |
| WRBBO versus ACS | 3.0518e-05 | 153.5 | 17.5 | 18/16/2/0 |
| WRBBO versus MFPA | 7.6294e-06 | 171 | 0 | 18/18/0/0 |




(a) (b) (c)

**FIGURE 6.** the average runtime. (a) on the 30-dimensional functions from CEC-2013, (b) and. (c) on the 30-dimensional and the 50-dimensional classic functions.

the 30-dimensional classic functions. And it verifies that WRBBO has also better optimization performance compared with other algorithms on the 30-dimensional and the 50-dimensional classic functions in Subsection IV-B2. WRBBO outperforms other algorithms on some complex functions from CEC-2013 test set in Subsection IV-C. What's more, the Wilcoxon signed-rank test shows that the optimization performance of WRBBO is significantly better than the those of comparison algorithms on the classic functions. From Subsection IV-E, WRBBO has the least runtime, so those prove that WRBBO has better optimization efficiency.

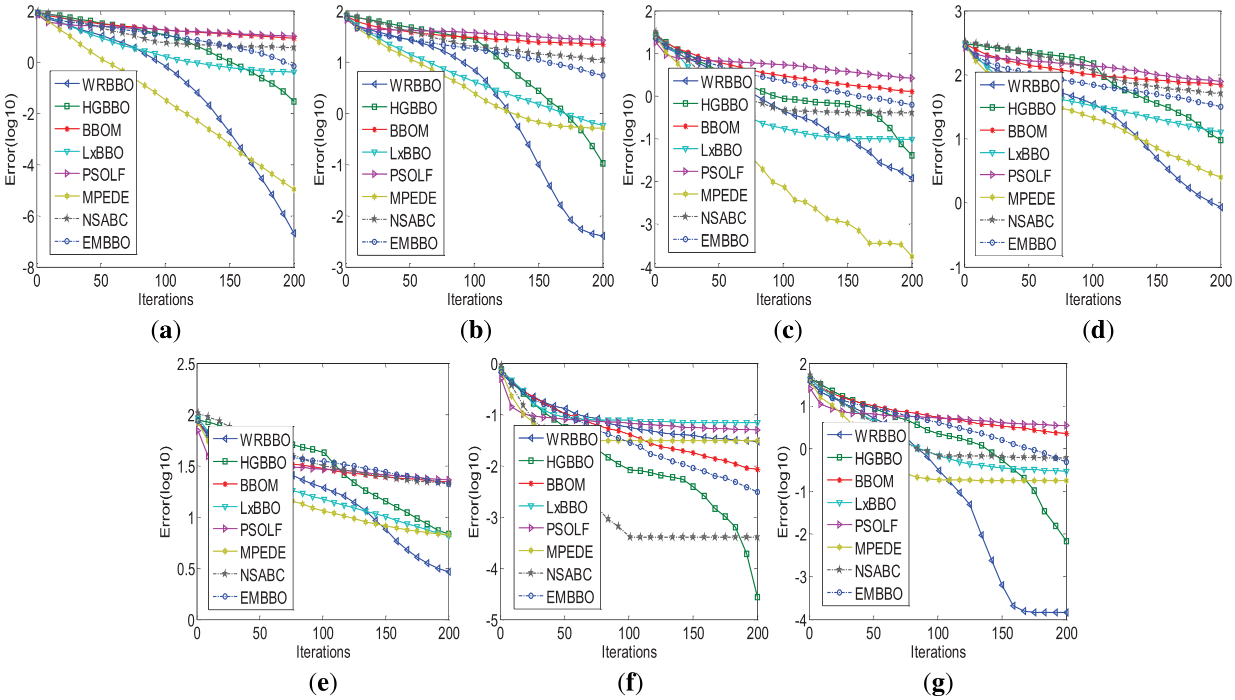



**FIGURE 7.** Convergence curves of the 8 algorithms on UCI datasets. (a) Heart. (b) Wine. (c) Iris. (d) Lonosphere. (e) Glass. (f) Baloon. (g) Newthyroid.

**TABLE 7.** Specifications of the seven datasets.

| Dataset | Number of Samples | Number of Attributes | Number of Clusters |
|---|---|---|---|
| Heart | 270 | 13 | 2 |
| Wine | 178 | 13 | 3 |
| Iris | 150 | 4 | 3 |
| Lonosphere | 351 | 34 | 2 |
| Glass | 214 | 9 | 6 |
| Baloon | 20 | 4 | 2 |
| Newthyroid | 215 | 5 | 3 |

## *F. APPLICATION OF WRBBO TO CLUSTERING OPTIMIZATION*

Clustering optimization plays an important role in many fields. By analyzing the data to be clustered, the specific distribution of data can be obtained. In recent years, many researchers have applied IOAs to the clustering optimization problems to enhance the clustering effect of the algorithm. However, in the face of complex clustering optimization problems [23], a more powerful IOA is needed to deal with it. $K$-means is a classic clustering algorithm with the advantages of simple principles, good scalability and high efficiency. But there are also the numbers of $K$ that cannot be determined and are sensitive to the initial point. Therefore, it has great research value to apply the proposed algorithm to $K$-means clustering optimization problems.

Each individual of WRBBO is considered as a candidate solution cluster center for the clustering optimization problems. In cluster optimization problems, Eq. (11) is used as the objective function. The solution with the minimum value of the objective function can be obtained as the best solution output:

$$f = \sum_{i=1}^{K} \sum_{x \in C_i} \|x - v_i\|_2^2 \tag{11}$$

where $K$ is the cluster number, $x$ is a sample which belongs to $C_i$, $C_i$ is the $i$th cluster and $v_i$ is the $i$th clustering center.

### 1) WRBBO FOR CLUSTERING OPTIMIZATION ON UCI DATASETS

In order to investigate WRBBO on $K$-means cluster optimization, this section uses the University of California at Irvine (UCI) datasets to conduct many experiments. The datasets (including Heart, Wine, Iris, Lonosphere, Glass and Baloon) are adopted for illustration in experiment group, and these datasets are taken from the UCI Machine Learning Repository [40]. The specifications of the datasets are given in Table 7.

The comparison algorithms include HBBOG [18], LxBBO [32], modified artificial bee colony with novel search equation and improved dimension selection strategy (NSABC) [41], modified PSO with Levy Flight (PSOLF) [42], Multi-Population Ensemble DE (MPEDE) [43], EMBBO [21] and BBOM [17]. The common parameters of these algorithms are set as follows: $N$ is 50, $MaxDT$ is 200 and $Run$ is 30. The results are shown in Table 8. The convergence curves of the 8 algorithms on UCI datasets are shown in Figure 7.

From Table 8, WRBBO obtains 5 times ranking the first (on Heart, Wine, Lonosphere, Glass and Newthyroid),

**TABLE 8.** Comparison results of clustering optimization on UCI datasets.

| Dataset | Value | WRBBO | HBBOG | LxBBO | NSABC | PSOLF | MPEDE | EMBBO | BBOM |
|---|---|---|---|---|---|---|---|---|---|
| Heart | Mean | **2.8377e+02** | 2.8379e+02 | 2.8416e+02 | 2.8733e+02 | 2.9392e+02 | 2.8377e+02 | 2.8445e+02 | 2.9216e+02 |
| | Std | **2.1477e-07** | 2.2970e-02 | 1.8151e+00 | 1.2027e+00 | 7.6851e+00 | 1.3923e-05 | 6.2531e-01 | 1.6189e+00 |
| | Rank | **1** | 3 | 4 | 6 | 8 | 2 | 5 | 7 |
| Wine | Mean | **8.8576e+01** | 8.8676e+01 | 8.9140e+01 | 9.9676e+01 | 1.1530e+02 | 8.9076e+01 | 9.3933e+01 | 1.1043e+02 |
| | Std | **5.0730e-03** | 4.1986e-02 | 1.0922e+00 | 1.9880e+00 | 7.9725e+00 | 2.6583e+00 | 3.9796e+00 | 2.3685e+00 |
| | Rank | **1** | 2 | 4 | 6 | 8 | 3 | 5 | 7 |
| Iris | Mean | 2.9161e+01 | 2.9190e+01 | 2.9246e+01 | 2.9556e+01 | 3.1740e+01 | **2.9149e+01** | 2.9761e+01 | 3.0426e+01 |
| | Std | 3.4250e-02 | 1.2706e-01 | 9.2944e-02 | 2.4784e-01 | 2.6583e+00 | **9.5477e-04** | 2.4472e-01 | 3.3194e-01 |
| | Rank | 2 | 3 | 4 | 5 | 8 | **1** | 6 | 7 |
| Lonosphere | Mean | **4.5248e+02** | 4.6120e+02 | 4.6432e+02 | 5.0290e+02 | 5.2983e+02 | 4.5410e+02 | 4.8279e+02 | 5.2171e+02 |
| | Std | **4.7125e-01** | 4.8943e+00 | 4.0212e+00 | 5.8535e+00 | 2.5110e+01 | 2.0606e+00 | 2.4608e+00 | 3.9793e+00 |
| | Rank | **1** | 3 | 4 | 6 | 8 | 2 | 5 | 7 |
| Glass | Mean | **5.4107e+01** | 5.77983e+01 | 5.7662e+01 | 7.2394e+01 | 7.3993e+01 | 5.7849e+01 | 7.2020e+01 | 7.3088e+01 |
| | Std | **2.5471e+00** | 2.1509e+00 | 2.4868e+00 | 2.7449e+00 | 3.6553e+00 | 3.9018e+00 | 3.5887e+00 | 2.0542e+00 |
| | Rank | **1** | 4 | 2 | 6 | 8 | 3 | 5 | 7 |
| Baloon | Mean | 1.6975e+01 | **1.6946e+01** | 1.7015e+01 | 1.6946e+01 | 1.6996e+01 | 1.6977e+01 | 1.6949e+01 | 1.6954e+01 |
| | Std | 4.5251e-02 | **5.4243e-05** | 9.3911e-02 | 6.9452e-04 | 5.1994e-02 | 4.7974e-02 | 3.5161e-03 | 3.7940e-03 |
| | Rank | 5 | **1** | 8 | 2 | 7 | 6 | 3 | 4 |
| Newthyroid | Mean | **4.0052e+01** | 4.0058e+01 | 4.0350e+01 | 4.0650e+01 | 4.3490e+01 | 4.0232e+01 | 4.0545e+01 | 4.2289e+01 |
| | Std | **2.7056e-04** | 1.1608e-02 | 1.0607e+00 | 4.0512e-01 | 2.5962e+00 | 9.8778e-01 | 6.3777e-01 | 5.7205e-01 |
| | Rank | **1** | 2 | 4 | 6 | 8 | 3 | 5 | 7 |
| Count | | 5 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| Ave.Rank | | 1.71 | 2.57 | 4.29 | 5.29 | 7.86 | 2.86 | 4.86 | 6.57 |
| Total.Rank | | 1 | 2 | 4 | 6 | 8 | 3 | 5 | 7 |

**TABLE 9.** Comparison results of clustering optimization on image segmentation.

| | WRBBO | HBBOG | LxBBO | EBO | BBO | BBOM |
|---|---|---|---|---|---|---|
| Mean | **1.1989e+03** | 1.1991e+03 | 1.2093e+03 | 1.1990e+03 | 1.3450e+03 | 1.1992e+03 |
| Std | **2.5795e-02** | 4.7727e-01 | 2.3703e+01 | 1.0986e-01 | 1.1907e+02 | 2.5741e-01 |
| Max | **1.1990e+03** | 1.2004e+03 | 1.2700e+03 | 1.1992e+03 | 1.6230e+03 | 1.1999e+03 |
| Min | **1.1989e+03** | **1.1989e+03** | **1.1989e+03** | **1.1989e+03** | 1.223e+03 | 1.1990e+03 |

HBBOG and MPEDE obtain 1 time ranking the first respectively on seven data sets, and WRBBO obtains the first on the average ranking (1.71). In general, compared with the other algorithms, WRBBO has the highest optimization performance in solving clustering optimization problems.

From Figure 7, On Heart, Wine, Lonosphere, Glass and Newthyroid datasets, WRBBO's convergence speed is much faster than other algorithms's. On Iris and Baloon datasets, HGBBO obtains the fastest convergence speed. Generally, WRBBO obtains the better convergence speed compared with the comparison algorithms.

### 2) APPLICATION OF WRBBO TO MEDICAL IMAGE SEGMENTATION

The underlying objective of medical image segmentation is to partition it into different anatomical structures, thereby separating the components of interest, such as liver tumors, from their background. We also use the clustering optimization algorithm to solve the problem of medical image segmentation and select a CT liver tumor image from many experimental images as a illustration to explain concisely.

In this section, the comparison algorithms include HBBOG [23], LxBBO [32], EBO [29], BBO [8] and BBOM [30]. *N* is 50, *MaxDT* is 200 and *Run* is 10. The results are shown in Table 9. The smaller the values are, the better the algorithm works. From Table 9, WRBBO obtains better results than the other algorithms on the values of Mean, Std and Max. WRBBO is smaller or equal than some other algorithms on the value of Min. In general, the clustering optimization performance of WRBBO is much better.

Then, WRBBO is used for image segmentation of the CT liver tumor. The result is shown in Figure 8. In Figure 8, (a) is the original image in which the shadow is the tumor, (b) and (e) are the results by the clustering optimization, and (d) is the images of the liver and tumor after removing the background. In order to facilitate the results of the segmentation, (c) shows the result of segmenting the benign liver and (f) shows the segmented results of the tumor, respectively, by the fast level set evolution which is used for only clear segmentation results. It can be seen that clustering optimization can be more helpful to medical image segmentation.

From all the above experimental results, WRBBO has the following advantages in general: (1) on optimization performance, whether on the 30-dimensional classic benchmark functions or on the 50-dimensional ones, the optimization performance of WRBBO is better than those of BBO variants and some other IOAs. (2) From on classical functions, CEC-2013 test set and the clustering to on image segmentation, the results show that WRBBO has the better ability to deal with these optimization problems, so the universality of WRBBO is better. (3) WRBBO gets the least average running time compared with quite a few state-of-the-art BBO variants and other IOAs. Thus, the optimization efficiency of WRBBO is better. (4) For most of the parameters of WRBBO, it adopts

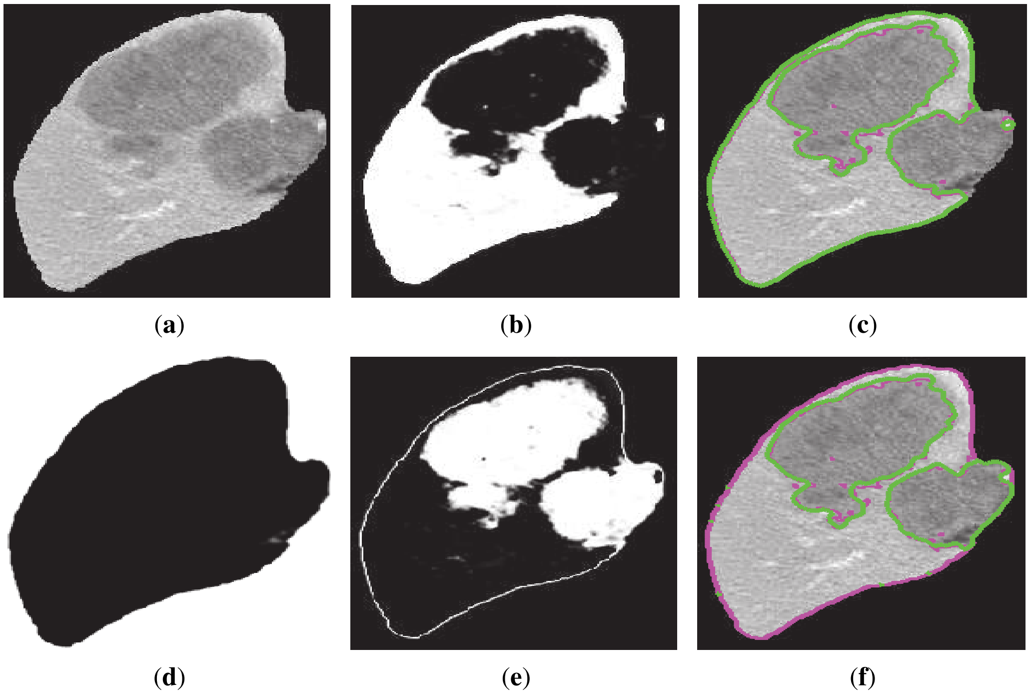



**FIGURE 8.** Original image, segmented images based on clustering and final segmented images.

the dynamic or random parameter adjustment approach to make WRBBO's operability stronger.

## V. CONCLUSIONS

In view of some drawbacks of BBO, in order to obtain an efficient optimization algorithm, a BBO algorithm with Worst opposition learning and Random-scaled differential mutation (WRBBO) is proposed. Firstly, the mutation operator of BBO is removed, which reduces computational load, and a more efficient random-scaled differential mutation operator is incorporated into the migration operator to obtain global search ability. Secondly, a dynamic heuristic crossover is used to replace the original migration operation of BBO, which overcomes the shortcomings in the migration operator of BBO and enhances the local search ability. Finally, in order to avoid falling into local optima, a worst opposition learning strategy is used. In addition, the example learning instead of the roulette wheel selection and the greedy selection instead of the elitist strategy are adopted, which reduce computational complexity largely, the immigration rate calculation step is moved outside of the iteration loop to further reduce computational complexity. In order to verify WRBBO, a large number of experiments are made on 18 various kinds of classic benchmark functions and some complex functions from CEC-2013 test set. WRBBO is also applied to clustering optimization and medical image segmentation. The experimental results show that WRBBO gets better optimization performance, and less runtime, which lead to higher efficiency, more universality and stronger ability to solve clustering optimization problems compared with quite a few BBO variants and other state-of-the-art IOAs. In the future, WRBBO may be improved further and combined with other optimization algorithms to solve more complex optimization problems, and it is expected to apply more engineering fields, such as pattern recognition, economic dispatch and so on.

## REFERENCES

[1] S. S. Reddy, P. R. Bijwe, and A. R. Abhyankar, "Faster evolutionary algorithm based optimal power flow using incremental variables," *Int. J. Elec. Power.*, vol. 54, pp. 198–210, Sep. 2014.

[2] Y. Liu, C. H. Mu, W. D. Kou, and J. Liu, "Modified particle swarm optimization-based multilevel thresholding for image segmentation," *Soft Comput.*, vol. 19, no. 5, pp. 1311–1327, Sep. 2015.

[3] H. Nazif and L. S. Lee, "Optimised crossover genetic algorithm for capacitated vehicle routing problem," *Appl. Math. Model.*, vol. 36, no. 5, pp. 2110–2117, Sep. 2012.

[4] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 4. Nov. 1995, pp. 1942–1948.

[5] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Erciyes Univ., Kayseri, Turkey, Tech. Rep., TR06, Sep. 2005.

[6] L. B. Deng, S. Wang, L. Y. Qiao, and B. Q. Zhang, "DE-RCO: Rotating crossover operator with multiangle searching strategy for adaptive differential evolution," *IEEE Access.*, vol. 6, pp. 2970–2983, 2017.

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

[8] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.

[9] X. S. Yang, "Flower pollination algorithm for global optimization," In *Unconventional Computation and Natural Computation* Berlin, Germany: Springer, Sep. 2012, pp. 240–249.

[10] X. S. Yang and S. Deb, "Cuckoo search via levy flights," In *Proc. World Congr. Nature Biologically Inspired Comput., NaBIC*, Sep. 2009, pp. 210–214.

[11] D. Simon, "Biogeography-based optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 6, pp. 702–713, Dec. 2008.

[12] W. Guo, M. Chen, L. Wang, Y. Mao, and Q. Wu, "A survey of biogeography-based optimization," *Neural Comput. Appl.*, vol. 28, no. 8, pp. 1909–1926, Sep. 2017.

[13] D. F. Wu, F. K. Ren, L. Qiao, and W. D. Zhang, "Active disturbance rejection controller design for dynamically positioned vessels based on adaptive hybrid biogeography-based optimization and differential evolution," *ISA Trans.*, vol. 78, pp. 56–65, Jul. 2017.

[14] J. C. Bansal, P. Farswan, and A. N. Atulya, "Design of wind farm layout with non-uniform turbines using fitness difference based BBO," *Eng. Appl. Artif. Intell.*, vol. 71, pp. 45–59, May 2018.

[15] Y.-J. Zheng, H.-F. Ling, and J.-Y. Xue, "Ecogeography-based optimization: Enhancing biogeography-based optimization with ecogeographic barriers and differentiations," *Comput. Oper. Res.*, vol. 50, pp. 115–127, Oct. 2014.

[16] H. P. Ma and D. Simon, "Blended biogeography-based optimization for constrained optimization," *Eng. Appl. Artif. Intell.*, vol. 24, no. 3, pp. 517–525, Apr. 2011.

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

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

[19] G. Khademi, H. Mohammadi, and D. Simon, "Hybrid invasive weed/biogeography-based optimization," *Eng. Appl. Artif. Intell.*, vol. 64, pp. 213–231, Sep. 2017.

[20] 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 Comput.*, vol. 15, no. 4, pp. 645–665, Apr. 2011.

[21] 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 Comput.*, vol. 8, pp. 1–20, Sep. 2018.

[22] W. A. Guo, L. Wang, C. Si, Y. Zhang, H. Tian, and J. Hu, "Novel migration operators of biogeography-based optimization and Markov analysis," *Soft Comput.*, vol. 21, no. 22, pp. 1–28, Jun. 2016.

[23] Q. Zhang, J. Gao, H. Dong, and Y. Mao, "WPD and DE/BBO-RBFNN for solution of rolling bearing fault diagnosis," *Neurocomputing*, vol. 312, pp. 27–33, Oct. 2018.

[24] S. S. Reddy and P. R. Bijwe, "Differential evolution-based efficient multi-objective optimal power flow," *Neural Comput. Appl.*, vol. 31, pp. 1–14, Sep. 2017.

[25] R. A. Ibrahim, M. A. Elaziz, and S. Lu, "Chaotic opposition-based grey-wolf optimization algorithm based on differential evolution and disruption operator for global optimization," *Expert Syst. Appl.*, vol. 108, pp. 1–27, Oct. 2018.

[26] H. Tizhoosh, "Opposition-based learning: A new scheme for machine intelligence," in *Proc. Int. Conf. Comput. Intell. Model., Control Automat. Int. Conf. Intell. Agents, Web Technol. Internet Commerce*, Sep. 2005, pp. 695–701.

[27] 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, Feb. 2016.

[28] J. Y. Long, Z. Zheng, X. Q. Gao, and P. M. Pardalos, "Scheduling a realistic hybrid flow shop with stage skipping and adjustable processing time in steel plants," *Appl. Soft Comput.*, vol. 64, pp. 536–549, Mar. 2018.

[29] Q. X. Feng, S. Liu, J. Zhang, G. Yang, and L. Yong, "Improved biogeography-based optimization with random ring topology and Powell's method," *Appl. Math. Model.*, vol. 41, pp. 630–649, Jan. 2016.

[30] J. Liang, B. Y. Qu, P. N. Suganthan, and A. G. Hernandez-Diaz, "Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization," Technological University, Singapore, Tech. Rep., Sep. 2013.

[31] X. Chen, H. Tianfield, C. L. Mei, W. Du, and G. Liu, "Biogeography-based learning particle swarm optimization," *Soft Comput.*, vol. 21, no. 24, pp. 7519–7541, Dec. 2016.

[32] V. Garg and K. Deep, "Performance of laplacian biogeography-based optimization algorithm on CEC 2014 continuous optimization benchmarks and camera calibration problem," *Swarm Evol. Comput.*, vol. 27, pp. 132–144, Apr. 2016.

[33] N. Lynn and P. N. Suganthan, "Heterogeneous comprehensive learning particle swarm optimization with enhanced exploration and exploitation," *Swarm Evol. Comput.*, vol. 24, pp. 11–24, Aug. 2015.

[34] M. R. Tanweer, S. Suresh, and S. N. Undararajan, "Self regulating particle swarm optimization algorithm," *Inf. Sci.*, vol. 294, 182–202, Sep. 2015.

[35] R. Mallipeddi, P. N. Suganthan, Q. K. Pan, and M. F. Tasgetiren, "Differential evolution algorithm with ensemble of parameters and mutation strategies," *Appl. Soft Comput.*, vol. 11, no. 2, pp. 1679–1696, 2011.

[36] A. Draa, S. Bouzoubia, and I. Boukhalfa, "A sinusoidal differential evolution algorithm for numerical optimisation," *Appl. Soft Comput.*, vol. 27, pp. 99–126, Feb. 2015.

[37] M. Naik, M. R. Nath, A. Wunnava, S. Sanhany, and R. Panda, "A new adaptive Cuckoo search algorithm," in *Proc. IEEE Int. Conf. Recent Trends Inf. Syst., Veer Surendra Sai Univ. Technol.*, Sep. 2015, pp. 9–11.

[38] E. Nabil, "A modified flower pollination algorithm for global optimization," *Expert Syst. Appl.*, vol. 57, pp. 192–203, Sep. 2016.

[39] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.

[40] C. L. Blake and C. J. Merz. (1998). *UCI Repository of Machine Learning Databases*. [Online]. Available: http://www.ics.uci.edu/mlearn/

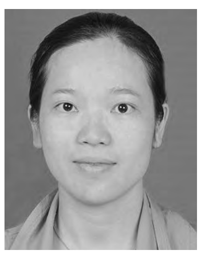
[41] Y. Shi, C. M. Pun, H. D. Hu, and H. Gao, "An improved artificial bee colony and its application," *Knowl.-Based Syst.*, vol. 107, pp. 14–31, Sep. 2016.

[42] R. Jensi and G. W. Jiji, "An enhanced particle swarm optimization with levy flight for global optimization," *Appl. Soft Comput.*, vol. 43, pp. 248–261, Jun. 2016.

[43] G. Wu, R. Mallipeddi, P. N. Suganthanc, R. Wang, and H. Chen, "Differential evolution with multi-population based ensemble of mutation strategies," *Inf. Sci.*, vol. 329, pp. 329–345, 2016.

**XINMING ZHANG** is currently a Full Professor and a Master's Supervisor with the College of Computer and Information Engineering, Henan Normal University, Xinxiang, China. He has published over 100 papers in important journals and conferences such as *Information Sciences* and *Applied Soft Computing*. His current research interests include swarm intelligence, evolutionary computation, and digital image processing.

**DOUDOU WANG** received the B.Sc. degree from the College of Software, Henan Normal University, China, in 2017, where she is currently pursuing the M.Sc. degree with the College of Computer and Information Engineering. Her research interests include intelligent optimization algorithm and image segmentation.



**HAIYAN CHEN** is currently a Doctor with the Department of Gynecological Tumor, Hubei Cancer Hospital, Wuhan, China. Her research interests include cancer identification and medical image processing.