Appl. Math. J. Chinese Univ.
2019, 34(4): 480-502

A Novel Bat Algorithm based on Cross Boundary Learning and Uniform Explosion Strategy

YONG Jia-shi[1] HE Fa-zhi[1,2,*] LI Hao-ran[1] ZHOU Wei-qing[1]

Abstract. Population-based algorithms have been used in many real-world problems. Bat algorithm (BA) is one of the states of the art of these approaches. Because of the super bat, on the one hand, BA can converge quickly; on the other hand, it is easy to fall into local optimum. Therefore, for typical BA algorithms, the ability of exploration and exploitation is not strong enough and it is hard to find a precise result. In this paper, we propose a novel bat algorithm based on cross boundary learning (CBL) and uniform explosion strategy (UES), namely BABLUE in short, to avoid the above contradiction and achieve both fast convergence and high quality. Different from previous opposition-based learning, the proposed CBL can expand the search area of population and then maintain the ability of global exploration in the process of fast convergence. In order to enhance the ability of local exploitation of the proposed algorithm, we propose UES, which can achieve almost the same search precise as that of firework explosion algorithm but consume less computation resource. BABLUE is tested with numerous experiments on unimodal, multimodal, one-dimensional, high-dimensional and discrete problems, and then compared with other typical intelligent optimization algorithms. The results show that the proposed algorithm outperforms other algorithms.

§1 Introduction

Optimization computation is an important topic in applied mathematics and computation mathematics [1-3]. For some complex optimization problems, in which the explicit information about optimization object function is unavailable or untrustworthy, or optimization function is nondifferentiable, derivative-free optimization methods become the most suitable way to solve this problem [4-8].

Received: 2019-01-14. Revised: 2019-07-05.
MR Subject Classification: 35B35, 65L15, 60G40.
Keywords: Optimization, Bat algorithm, Cross boundary learning, Uniform explosion strategy.
Digital Object Identifier(DOI): https://doi.org/10.1007/s11766-019-3714-1.
Supported by the National Natural Science Foundation of China (61472289) and the Open Project Program of the State Key Laboratory of Digital Manufacturing Equipment and Technology (DMETKF2017016).
*Corresponding author.

Therefore, the researchers have to invent new derivative-free optimization methods by abstracting and modeling some natural phenomena to solve complex optimization problems in applied mathematics and computation mathematics [9-12].

Typical derivative-free optimization methods are population-based algorithms, such as Ant colony algorithm [13-14], particle swarm optimization algorithm [15], simulated annealing algorithm and others [16-20]. Population-based algorithms usually generate a set of initial solutions randomly, and then approximate the optimal solution to the problem until the termination condition is satisfied.

Bat algorithm (BA) is a novel population-based algorithm that simulates biological characteristics about the ultrasonic searching and predation of prey of bats in the natural world. It was first proposed by Yang X. S [21] in 2010. BA has the advantages of simple model, fast convergence, potential parallelism and distributed characteristics. However, the BA also has the problems of falling into the local optimal and missing the exact global solution.

In order to balance the exploration and exploitation of BA, we present a novel bat algorithm based on cross boundary learning (CBL) and uniform explosion strategy (UES) to achieve both fast convergence and solution quality with less computation resource.

The paper is organized as follows: Section 2 describes related works about bat algorithm. Detailed explanation about BABLUE is in section 3. Experimental results are in section 4. Final conclusion of the work is in section 5.

§2 Related works

2.1 Bat algorithm

BA is a random search algorithm constructed by simulating bats in the natural world to search, locate and prey by means of ultrasonic waves. It simulates the using of ultrasound of bats for basic detection and localization and connects them with the optimized target function.

The bionic principle is that the bats are mapped to points in the search space. The search and optimization process is simulated as bat individuals searching for prey and moving process. The objective function of solving the problem is measured as the bat's location. In the iterative process, bat that has a better location will replace bat with poor location, which is the same as survival of the fittest.

The optimization mechanism of the original bat algorithm is defined as:

The frequency of the pulse used when bats search for prey is:

$$f_i = f_{min} + rand(f_{max} - f_{min}). \tag{1}$$

Where f_i is the pulse frequency used by the bat i. f_{min} and f_{max} is the pulse frequency range, and $rand$ is a random factor that is uniformly distributed on [0,1].

The speed at which a bat searches for prey is determined by equation 2:

$$v_{id}^{t} = v_{id}^{(t-1)} + f_i(x_{id}^{(t-1)} - x_d^*) \tag{2}$$

Where subscript d denotes the dimension of the search space, v_{id}^{t} and $v_{id}^{(t-1)}$ denote the flight speeds of the bat i at $t-1$ and t respectively, $x_{id}^{(t-1)}$ denotes the position of bat i at time $t-1$, and x_{d}^{*} represents the best position in the bat population up to the t iterations so far.

The bats' space location is updated as follows:

$$x_{id}^{t} = x_{id}^{(t-1)} + v_{id}^{t}. \tag{3}$$

The frequencies and intensities of the pulses emitted during the bat searching for a prey are:

$$r_{i}^{(t+1)} = r_{i}^{0}[1 - e^{-\gamma t}]. \tag{4}$$

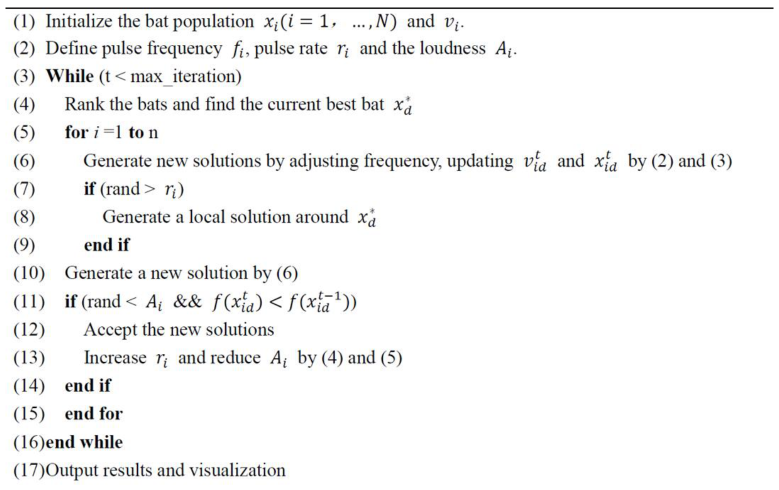
$$A_{i}^{(t+1)} = \alpha A_{i}^{t}. \tag{5}$$

Local search: a new solution for each bat is generated locally using random walk:

$$x_{new} = x_{old} + \varepsilon A^{t}. \tag{6}$$

Where $\varepsilon \in$[-1, 1] is a random number, while A^{t} is the average loudness of all the bats at this time.

When searching for preys, bats launch the pulse with lower frequency and larger sound intensity. When the prey is discovered, it gradually reduces the pulse sound intensity and increases the pulse emission frequency. The pulse frequency r and the pulse intensity A simulate the search characteristics of bats. Where r_{i}^{0} denotes the initial pulse frequency, $r_{i}^{(t+1)}$ denotes the pulse frequency of the bat at time $t+1$, γ is the pulse frequency increasing coefficient, which is a constant larger than zero, $A_{i}^{(t+1)}$ represents the sound intensity of the bat firing pulse at time t, and α is the attenuation coefficient of the pulse sound intensity, which is a constant on [0,1]. The process of BA is given in Figure 1.



```
(1)  Initialize the bat population $x_i(i = 1, ..., N)$ and $v_i$.
(2)  Define pulse frequency $f_i$, pulse rate $r_i$ and the loudness $A_i$.
(3)  **While** (t < max_iteration)
(4)    Rank the bats and find the current best bat $x_d^*$
(5)    **for** $i$ =1 **to** n
(6)      Generate new solutions by adjusting frequency, updating $v_{id}^t$ and $x_{id}^t$ by (2) and (3)
(7)      **if** (rand > $r_i$)
(8)        Generate a local solution around $x_d^*$
(9)      **end if**
(10)   Generate a new solution by (6)
(11)   **if** (rand < $A_i$ && $f(x_{id}^t) < f(x_{id}^{t-1})$)
(12)     Accept the new solutions
(13)     Increase $r_i$ and reduce $A_i$ by (4) and (5)
(14)   **end if**
(15)   **end for**
(16)**end while**
(17)Output results and visualization
```

Figure 1: **The process of BA**

2.2 Improved algorithms

One of the key advantages of standard BA is that it can provide very quick convergence at an early stage by switching from exploration to exploitation [22]. This makes it an efficient algorithm for applications when a quick solution is needed.

However, if we allow the algorithm to switch to exploitation stage too quickly by varying A and r too quickly, it may lead to stagnation after some initial stage and will fall into local optimal.

In order to improve the performance, many methods and strategies, such as Lévy flights, chaotic maps and differential operator, have been attempted to increase the diversity of the solution and thus to enhance the performance, which produced a few good variants of bat algorithm [23-27].

Yilmaz S extended the bat algorithm with a good combination of subtle variations of loudness and pulse emission rates [28]. Exploration and exploitation mechanisms of BA are improved by three modifications, inertia weight factor modification, adaptive frequency modification and scout bee modification.

Mirjalili S developed a discrete version of bat algorithm to solve discrete problems [29]. They proposed a v-shaped transfer function and in addition, there is a real application of the proposed method in optical engineering called optical buffer design at the end of the paper.

Xie J presented DLBA using Lévy flights and differential operator [30]. A differential operator is introduced to accelerate the convergence speed of proposed algorithm and Lévy flights trajectory can ensure the diversity of the population against premature convergence and make the algorithm effectively jump out of local minima.

Afrabandpey H presented a chaotic bat algorithm using chaotic maps for optimization tasks [31] and their approach is based on the substitution of the random number generator (RNG) with chaotic sequences for parameter initialization.

These above works have solved the problem of falling into the local optimal easily; however, it costs lots of resource to get a precise result. So, in this paper, we present new methods called cross boundary learning (CBL) and uniform explosion strategy (UES). CBL is used to increase the diversity of the solution and UES is used to enhance the performance on exploitation with less computation resource.

2.3 Fireworks explosion algorithm

Our work is also related to the Fireworks Explosion Algorithm (FEA).

FEA is a meta-heuristic algorithm that simulates the explosion of a bubble in a fireworks explosion by Tan [32]. When the fireworks explode, the sparks released are scattered in a circular neighborhood with the explosion point as the center. If the neighborhood is regarded as a local region of the problem, and the sparks produced by explosion are regarded as the points in the region, then the explosion is similar to the exploration of the local area. This kind of exploration is equivalent to local search of the area near the explosion point in the solution space.

FEA algorithm balances the ability of exploration and exploitation adaptively according to the fitness value of the current fireworks population [33]. However, the proposed algorithm needs to enhance the ability of exploitation by exploding. We propose a different uniform explosion strategy to maximize the local search capability of the algorithm under the low consumption of resources.

2.4 Motivation and contribution of this work

For typical bat algorithms, the ability of exploration and exploitation is not strong enough and it is hard to find a precise result. This paper presents a novel bat algorithm based on cross boundary learning (CBL) and uniform explosion strategy (UES) to avoid the above contradiction and achieve both fast convergence and high quality.

Difference from previous opposite-based learning, Cross boundary learning (CBL) expands the search area, and therefore makes the population more diverse. In this way, the proposed algorithm (BABLUE) has a higher possibility to find a better solution in each iteration than previous opposite-based learning.

Furthermore, the computing cost of the proposed Cross boundary learning (CBL) as shown in equation 8 is the same level as previous opposite-based learning as shown in equation 7. Eventually, the proposed algorithm (BABLUE) has better search ability while keeps the fast search speed.

The proposed Uniform explosion strategy (UES) has two characteristics or advantages. The first one is that UES does not simplify brow the idea of FEA. In FEA, there are a number of fireworks. Some of the fireworks have large radius and some of them have small radius, which means the population can jump out the local optimum. The proposed UES adopts FEA operation with one firework to enhance the random flying process, in which the flying center is super bat and the flying radius is the explosion radius of the single-firework. Because our UES adopts only a single- firework, it is easy to fall into local optimum. To solve this problem, we proposed UES to adaptively change the explosion radius of the single-firework, as shown in equation 12, 13 and 14.

The second one is computation cost. If we simply use the original FEA explosion strategy, as shown in equation 9, the number of newly generated sparks after explosion is very large, which result in very high computational overhead. The original FEA explosion strategy may result in a deviated distribution. In order to reduce the cost, a new explosion strategy is proposed, as shown in equation 10, in which the generated sparks number is determined by the dimensions of problem and the sparks positions are uniformly distributed. With UES, BABLUE can stably find better solution in each iteration.

§3 The proposed BABLUE Algorithm

In view of the strong local exploration capability of fireworks algorithm, we present a new USE into the bat algorithm, so that the bat algorithm can still maintain the convergence efficiency in the late stage of convergence. At the same time, in order to solve the problem of premature convergence of the bat algorithm, we add a new CBL mechanism to the bat algorithm, generate cross boundary population according the center of the best and worst bat in the generation, select the elite individuals from the original population and the opposite population, increase the diversity of the algorithm and make the algorithm converge to the global optimum faster.

Based on this, this paper proposes a novel bat algorithm based on cross boundary learning and uniform explosion strategy (BABLUE). The algorithm uses the CBL and UES to make the new algorithm perform well both in global and local search. The experiments show that the accuracy and speed of the algorithm are improved.

3.1 Cross boundary learning

In 2005, Professor Tizhoosh first proposed the concept of opposition-based learning [34]. He believes that the intelligent algorithm is based on the value of the random guess as the initial population, and then get close to the optimal solution gradually, and eventually find or close to the optimal solution. So the random guess value has a great influence on the algorithm. If the random guess value is close to the optimal solution, the algorithm may converge quickly, but if it is far or far opposite, the algorithm is very tricky and will cost more time. In the process of searching, if we search for the current solution and the opposite solution at the same time and choose a better solution as a guess solution, will greatly improve the efficiency of the algorithm. In fact, according to the probability theorem, there is a probability of 0.5 that the current solution is far more than the optimal solution than its opposite solution.

The formula for opposite-based learning is as follows:

$$X_{ij}^{*} = a_j(t) + b_j(t) - X_{ij}(t) \tag{7}$$

where $a_j(t) = \min(X_ij(t))$, $b_j(t) = \max(X_ij(t))$, i=1,2,...,ps, j=1,2,...,D. $X_{ij}(t)$ is a population of i solutions in the j dimension of the component, X_{ij}^{*} is the opposite solution corresponding to $X_{ij}(t)$, $a_j(t)$ and $b_j(t)$ are the minimum value and the maximum value of the j-th dimension of the current search area, ps is the size of the population, D is the dimension of the problem, t is the iteration.

For bat algorithm with opposite learning, in the beginning of evaluation, the opposite learning works well. However, because of the existence of super bat, after ten or dozens of generations, the population is clustered around the super bat and the search boundary narrows drastically.

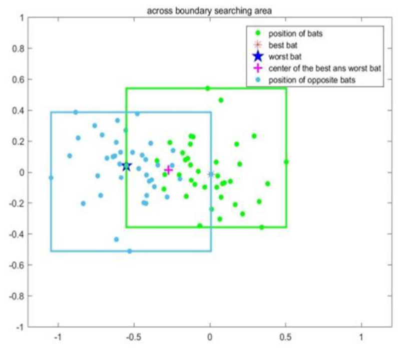
In previous algorithms [35-38], the search boundary narrows as the number of iterations increases. Based on equation 7, the boundary of the opposite population is the same as original population; therefore the range that can be searched by the opposite population also decreases with the number of iterations increasing. As we have mentioned before, the super bat makes the search boundary narrows drastically, that means the search range of opposition learning also decreases drastically. After a few iterations, the algorithm loses the diversity and is easy to fall into the local optimum.

In the proposed new Cross Boundary Learning model, we present a new Cross Boundary equation 8 which makes the search area of the opposite population and original population symmetrical about the center of the best and worst bats of the current generation. Therefore, the opposite population still has enough difference from the original population. When the super bat makes the search boundary narrows drastically, this model can prevent other bats from coming near the super bat too early. So the whole bat algorithm keeps diversity, avoids precocity and avoids the local optimum.

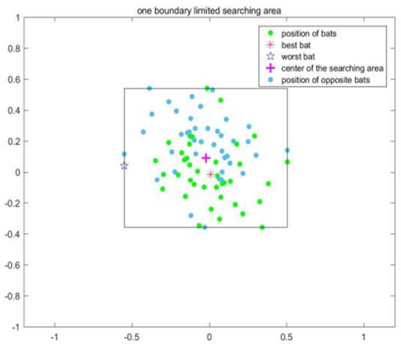
$$X_{ij}^{*} = k(X_{best} + X_{worst}) - X_{ij}(t) \tag{8}$$

$X_{ij}^{*}$is the opposite solution corresponding to $X_{ij}(t)$, X_{best} and X_{worst} are the best position and the worst position in the current generation. In our previous experiments on the opposition-based learning bat algorithm, we found that for most test functions, k=1/2 is better, therefore, at equation 8 we choose the opposition based learning model with k=1/2.

From Figure 2, we can find that when equation 8 is used, the bat can be allowed to jump out of the current search boundary and the cross boundary Learning search area can be expanded to avoid falling into the local optimum.



(a)



(b)

Figure 2: **methods (a) The method proposed in this paper:across boundary searching area, (b)previous method:one boundary limited searching area**

3.2 Uniform explosion strategy

In the local search process of bat algorithm, the average loudness of all bats is used as perturbation step in equation 6. For each bat, the search direction is certain, which means the ability of local search is not strong enough. Thus, it is hard for bat algorithm to find the optimal solution. To solve this problem, we looked for inspiration from other meta-heuristic algorithms and found that the explosion direction of the FEA was random in the whole search space, and that this type of explosion was very suitable for localized searches on a small scale. So we introduced the explosive strategy of the fireworks explosion algorithm into the local search of bat algorithm.

In this paper, we use the explosion strategy for local search during the random walk of the bats. However, we found that the explosion radius and the number of explosive sparks is adjusted adaptively according to the fitness value of the current fireworks population, which was used to balance the global search capabilities and local search capabilities of the algorithm. In this essay, we only need to use the explosion strategy to improve the local search ability of the algorithm, so the explosion strategy and explosion radius of the original FEA will not be applicable. In order to solve this problem, we propose a new uniform sampling-based (grid) explosive strategy and a new adaptive adjusted explosion radius formula in sections 2.3 and 2.4.

In the new algorithm, the explosion is around the best bat. We can see from formula 9 that the number of explosion sparks S_i is large when $f(x_i)$ is small. This results in increasing the computational complexity of the algorithm and reducing the speed of the algorithm.

$$S_i = M \times \frac{y_{max} - f(x_i) + \varepsilon}{\sum_{i=1}^{N} y_{max} - f(x_i) + \varepsilon} \tag{9}$$

Therefore, in order to avoid complicating the process of random walking, and to allow the algorithm to find the similar solution in a limited time, we propose a new uniform sampling based (grid) explosion strategy, which divides the search space into multiple regions, and create a spark in each region. Then, we find the best spark from these sparks and compare it to the best bat and get the new best bat.

We specify that the explosive sparks in each area are on the axis of a dimension, so in fact we can produce sparks from the explosion point along the axis of each dimension. Figure 3 shows the sparks generated in the two-dimensional search space and the regional division of the search space. At the same time, in order to reduce the computational complexity, we adopt different strategies for searching the high and low latitudes. The specific methods of explosion are as figure 3.

The distance between the explosion center and sparks is divided into three cases: r/3, 2r/3 and r. For the explosion point i, the positions of sparks are:

$$x_i^* = x_i + r_i \cdot \overrightarrow{b_k} (k = 1, 2, ..., m) \tag{10}$$

Where x_i is the current position of the explosion point, x_i^* is the position of sparks, r_i (j = 1,2,3) is the explosion radius and $r_1 = r, r_2 = r/3, r_3 = 2r/3$, r is the maximum radius of the current generation of explosions, $\overrightarrow{b_k}$ indicates the direction of the explosion, and m is the total number of all directions after the explosion of point i. For the low-dimensional search space (D$\leq$5), we choose to explode along the positive and negative directions of the coordinate axis, that is, the total number of the explosion directions is 2D. For the high-dimensional search space (D>5), in order to reduce the resource consumption of the algorithm, we randomly select three groups of D / 5 directions different from each other in the direction of D dimensions, and choose the opposite direction to join the group, constitute 6D / 5 directions. For the first

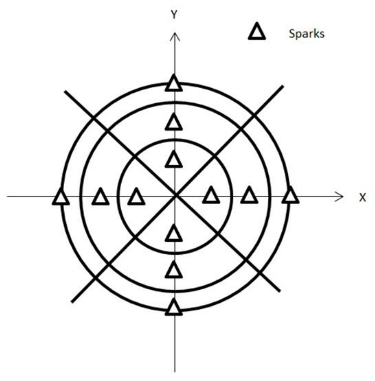


Figure 3: **Sparks and search area division**

group, we use r_1 as the explosion radius to generate sparks. Similarly, for the other two groups, we use r_2 and r_3 as their explosion radius.

Compared to random walk searching in a single direction, UES can effectively search the neighborhood area of bat and find a better solution by uniform exploding in multiple directions. UES enhances the local search ability of bat algorithm and avoids the late stagnation of the algorithm.

3.3 Adaptive radius based on bat population error

In FEA, by setting different explosion radius to maintain the balance between global search and local search, the fireworks population is exploratory and productive. The algorithm proposed in this paper is to add the explosion strategy to the random flight of bats to enhance the local search capability of the algorithm. However, the random walk of a bat is based on best bat of the population, this leads to the fitness is small so that the explosion radius is close to 0 by using (11) to calculate the radius of the explosion, which can't achieve the purpose of enhancing the local search.

$$A_i = \hat{A} \times \frac{f(x_i) - y_{min} + \varepsilon}{\sum_{i=1}^{N} f(x_i) - y_{min} + \varepsilon} \tag{11}$$

Therefore, the formula of the explosion radius of the traditional FEA algorithm is not applicable to the algorithm in this paper. Instead, we propose a new explosion radius formula using the sigmoid function to adjust the explosion radius adaptively:

$$r = \frac{1}{100(1 + e^{\frac{(\ln 9+10)*t}{N} - 10})} \tag{12}$$

Where N is the total iteration number of the algorithm and t is the current iteration number of the algorithm. After testing the typical test functions, we find that the performances of some functions are not as good as expected by adjusting the explosion radius according to the number of iterations. We found from the data that different test functions have different degrees of convergence at different iterations, but according to our formula, the radius of the explosion is the same for the same iterations, which leads to a problem that the explosion radius is too large for functions that has fast convergence rate. Therefore, we make the following adjustments to the explosion radius formula:

$$r = \frac{1}{1 + e^{-(f_{max} - f_{min})}} - 0.5 \tag{13}$$

$$r = \begin{cases} 0.01, & r > 0.01 \\ r, & r \leq 0.01 \end{cases} \tag{14}$$

Where f_{max} and f_{min} respectively represent the fitness value of the optimal location and the worst location of the current population. We add a threshold of 0.01 for r because the gap between the optimal position and the worst position is huge for the first few generations, which leads to a larger r calculated by the explosion radius formula. Therefore, after manually adjusting the threshold and comparing the experimental results, we chose a more appropriate threshold of 0.01.

The process of obtaining a new local solution using UES for local search is as follows:

Firstly, the explosion radius is calculated according to equation 13 and 14, then the direction vector $\overrightarrow{b_k}$ is obtained by using the explosion method proposed in section 3.2; secondly, the position of the sparks are calculated according to equation 10; Finally, the optimal spark is selected as the new position of the bat.

3.4 BABLUE algorithm

The idea in the article is to present a Cross Boundary Learning of each bat population of generations to obtain the cross boundary population, in order to increase the diversity of the population. Then, different from the original bat population and the opposite population, elite selection strategy is adopted to obtain a new and better population. Finally, local search is conducted by random perturbations of the explosion strategy to improve the quality and accuracy of the solution. The implementation process of the algorithm is as Figure 4.

§4 Experiment and Analysis

BABLUE is compared with other typical intelligent optimization algorithms with numerous experiments. In order to clearly present the experiments, this section is divided into 6 subsections as follows.

Section 4.1 describes the experimental parameters for the proposed algorithm. Section 4.2 describes the benchmark functions for experiment and the execution times. Section 4.3 shows the experimental results on unimodal problems. Section 4.4 shows the experimental results on high dimensions problem, Section 4.5 is the comparison of two different explosion radius we proposed as shown in equation 12, 13, and 14. Finally, in Section 4.6, we apply the proposed algorithm for 0-1 knapsack problem to demonstrate that BABLUE is also valid on discrete problems.

```
(1)  Initialize the bat population $x_i(i = 1, ..., N)$ and $v_i$.
(2)  Define pulse frequency $f_i$, pulse rate $r_i$ and the loudness $A_i$.
(3)  While (t < max_iteration)
(4)      Generate the cross boundary population of $x_i(i = 1, ..., N)$ by (8)
(5)      Choose the best N bats from cross boundary population and original population
(6)      Rank the bats and find the current best bat $x_d^*$
(7)      for i =1 to n
(8)          Generate new solutions by adjusting frequency, updating $v_{id}^t$ and $x_{id}^t$ by (2) and (3)
(9)          if (rand > $r_i$)
(10)             Generate sparks around $x_d^*$ by (10), (13) and (14), choose the best as new solution
(11)         end if
(12)         Generate sparks around $x_i$ by (10), (13) and (14), choose the best as new solution
(13)         if (rand < $A_i$ && $f(x_{id}^t) < f(x_{id}^{t-1})$)
(14)             Accept the new solutions
(15)             Increase $r_i$ and reduce $A_i$ by (4) and (5)
(16)         end if
(17)     end for
(18) end while
(19) Output results and visualization
```

Figure 4: **The process of BABLUE**

4.1 Experimental parameters

In order to verify the performance of the algorithm proposed in this paper, twelve standard test functions are selected for testing. There is no exact theoretical standard for the various parameters involved in bat algorithm. The parameter values set in this paper are determined by the experience gained from repeated tests, and the specific parameters are shown in Table 1. The termination condition of the algorithm is the maximum number of the fitness function estimates (FEs). The FEs is the number of iterations multiplied by the population size. For different test functions, the population size is different.

Table 1: **Parameters of simulation experiment.**

Name	N	f_{min}	f_{max}	A_i^0	r_i^0	α	γ
BA	40	0	100	(1,2)	(0,1)	0.9	0.9
BABLUE	40	0	100	(1,2)	(0,1)	0.9	0.05

4.2 Benchmark function and execution times

The experimental environment is as follows, system: window10; CPU: Intel(R) Core(TM) i5-5287U 2.90GHz; RAM: 8GB; programming tool: Matlab R2016a.

12 standard benchmark functions are selected in the experiment, where f1-f10 has the uniquely determined optimal solution, and f11 and f12 has multiple global optimal solutions. Specific parameters of benchmark function are shown in Table 2. Function f1-f8 uses fixed precision (Tol=10e-5) as the termination condition to compare with DLBA, and f9-f12 uses FEs

Table 2: **Parameters of Benchmark Function.**

Funs	*Name*	*x*	*d*	*Global Optimal*
f_1	Sphere	[-10,10]	30	0
f_2	Schwefel 2.22	[-10,10]	20	0
f_3	Eggcrate	[-2π,2π]	2	0
f_4	Ackley	[-30,30]	5	0
f_5	Griewank	[-600,600]	5	0
f_6	Salomon	[-5,5]	5	0
f_7	Rastrigin	[-5.12,5.12]	5	0
f_8	Zakharov	[-10,10]	5	0
f_9	Schaffer	[-100,100]	2	-1
f_{10}	Rosenbrock	[-2.048,2.048]	16	0
f_{11}	Shubert	[-10,10]	2	-186.7309
f_{12}	Branin	[-10,10]	2	0.3979

as the termination condition.

Figure 5 shows the execution time of f1-f12 benchmark function under Tol. Because of the complex terrain of f10 and f11, it takes more time and more iterations to calculate than other benchmark functions. For the other functions, BABLUE can find the optimal solution of the problem quickly (execution time less than 50ms).

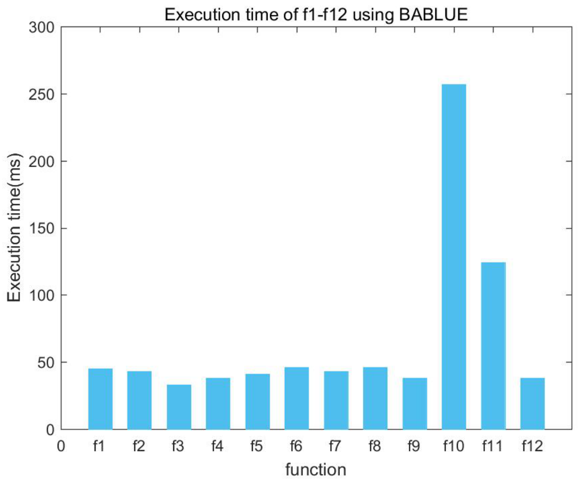


Figure 5: **Execution time of f1-f12 using BABLUE**

4.3 Comparison of experimental results

Four improved algorithms (IBA [28], DLBA [30], DGOBA [39], MBA [40]) based on bat algorithm were selected for comparison experiments.

IBA uses inertia weigh factor and adaptive frequency to improve exploration and exploitation mechanisms. MBA improves the random walk by changing the values of some but not all

Table 3: **Comparison between IBA, MBA, DLBA and BABLUE on f1 to f8 under Tol.**

Function	Method	Fitness				Iteration		
		Min	Mean	Max	std	Min	Mean	Max
f_1	MBA	0.0457	0.0928	0.1991	0.0296	200	200	200
	IBA	0.1216	0.2270	0.4229	0.0588	200	200	200
	DGOBL	2.0469e−06	5.9867e−06	9.7871e−06	2.2377e−06	7	11.4	30
	DLBA	4.3057e−08	4.8036e−06	9.9616e−06	3.0206e−06	10	17.6	26
	BABLUE	**9.1425e−32**	**2.8675e−06**	**9.8524e−06**	**2.8143e−06**	**4**	**5.6**	**17**
f_2	MBA	0.3217	10.6952	43.3804	11.7661	200	200	200
	IBA	0.6836	9.3901	517.3636	53.3170	200	200	200
	DGOBL	4.7646e−21	0.5555	55.5416	5.5541	20	68.24	200
	DLBA	5.6676e−07	6.3309e−06	9.8893e−06	2.4887e − 06	21	29.4	**37**
	BABLUE	**0**	**1.1951e−06**	**9.8005e−06**	**2.3883e−06**	**4**	**11.54**	42
f_3	MBA	1.5069e−06	0.5695	9.4884	2.2645	35	188	200
	IBA	4.2949e−05	1.9009	9.5041	3.8143	200	200	200
	DGOBL	1.7542e−08	0.1205	9.4926	0.9784	4	25.87	200
	DLBA	6.2347e−09	3.9335e−06	9.9702e−06	2.7721e−06	6	10	**20**
	BABLUE	**0**	**3.3481e−07**	**8.6771e−06**	**1.2404e−06**	**2**	**5.02**	99
f_4	MBA	0.0151	8.2271	15.1285	3.9637	200	200	200
	IBA	0.1331	0.5062	0.8400	0.1522	200	200	200
	DGOBL	0	8.8047e−06	1.1924e−04	1.3589e−05	16	49.33	200
	DLBA	4.7174e−07	6.4097e−06	9.9609e−06	2.7119e−06	16	25.6	**39**
	BABLUE	**0**	**5.4327e−07**	**9.9253e−06**	**1.7881e−06**	**3**	**8.32**	70
f_5	MBA	0.7759	6.6939	20.7067	4.0156	200	200	200
	IBA	0.0717	1.6400	6.0631	1.1971	200	200	200
	DGOBL	6.5512e−08	5.0445e−07	4.4773e−06	6.5777e−07	4	6.12	13
	DLBA	1.3818e−07	5.0103e−06	9.9300e−06	2.7734e−06	11	18.4	56
	BABLUE	**0**	**3.5668e−07**	**1.8532e−06**	**5.4339e−07**	**3**	**4.67**	**9**
f_6	MBA	0.0999	0.2479	0.3999	0.0759	200	200	200
	IBA	0.0326	0.0992	0.0999	0.0067	200	200	200
	DGOBL	0	0.0180	0.3999	0.0687	11	50.8	200
	DLBA	7.7849e−07	6.5729e−06	9.9749e−06	2.7119e−06	18	46	**114**
	BABLUE	**0**	**3.13607e−07**	**4.3495e−06**	**9.6401e−07**	**3**	**16.2**	173
f_7	MBA	0.1287	13.3862	37.0362	6.5959	200	200	200
	IBA	1.9512	9.1300	17.7209	3.3527	200	200	200
	DGOBL	0	0.9646	32.8143	4.2927	11	45.12	200
	DLBA	5.9998e−08	4.4705e−06	9.8569e−06	2.8144e−06	15	33	**87**
	BABLUE	**0**	**3.6086e−09**	**1.0776e−07**	**1.4023e−08**	**3**	**14.17**	119
f_8	MBA	1.9797e-04	0.0022	0.0145	0.0023	200	200	200
	IBA	0.0087	0.0496	0.0946	0.0202	200	200	200
	DGOBL	2.3859e−07	3.3729e−06	9.9865e−06	2.4986e−06	4	6	14
	DLBA	6.6507e−09	3.8869e−06	9.6984e−06	2.7544e−06	10	15.1	23
	BABLUE	**0**	**1.6883e−06**	**8.3333e−06**	**2.2891e−06**	**3**	**4.92**	**11**

dimensions of the candidate solution. DGOBA applies the dynamic opposite learning to avoid falling into the local optimum. DLBA uses Lévy flights and differential operator to accelerate the convergence speed and jump out of local minima. BABLUE presents CBL and UES to avoid local optimum and enhance the exploitation ability.

Table 3 is a comparison of the experimental results between IBA, MBA, DGOBL, DLBA and the proposed algorithm on the f1-f8 test function. Each function runs 100 times independently. The data of DLBA comes from [30]. Bold and underline indicate that BABLUE is better.

As can be seen from Table 3 and Figure 6, BABLUE iterates at least twice for a given search

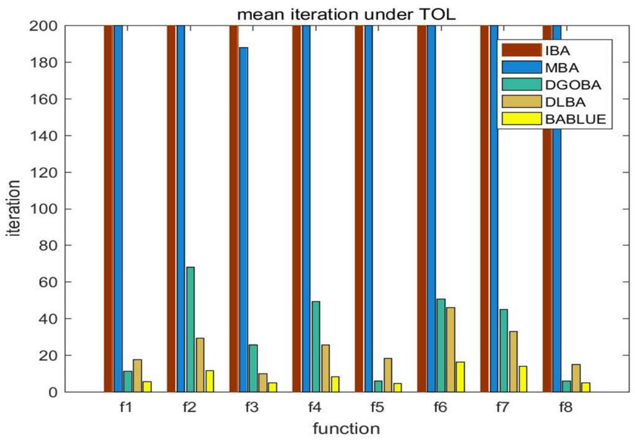


Figure 6: **mean iteration of IBA, MBA, DGOBA, DLBA and BABLUE under TOL**

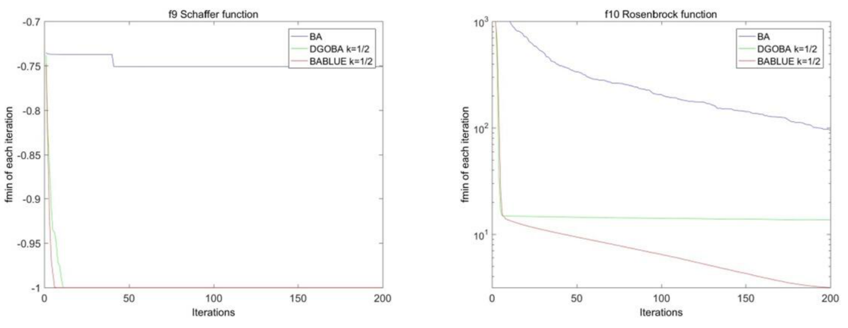
Table 4: **Comparison between BA, DGOBA and BABLUE on F9 and F10 under FEs.**

Function	Method	Fitness			
		Min	Mean	Max	std
f_9	BA	-0.99028	-0.51499	-0.75061	0.14281
	DGOBA(K=1/2)	**<u>-1</u>**	**<u>-1</u>**	**<u>-1</u>**	**<u>0</u>**
	BABLUE	**<u>-1</u>**	**<u>-1</u>**	**<u>-1</u>**	**<u>0</u>**
f_{10}	BA	39.6319	245.5212	95.3728	47.0705
	DGOBA(K=1/2)	13.3684	13.6007	13.8818	0.11311
	BABLUE	**<u>0.3356</u>**	**<u>0.64723</u>**	**<u>2.3822</u>**	**<u>0.39925</u>**

accuracy, while DLBA iterates at least four times. From the mean of the data, BABLUE performes better than DLBA, IBA and MBA on the six test functions. For the f1 and f4 functions, BABLUE performes slightly worse than the DLBA, but its optimal solution performs better than DLBA. As for the number of iterations, the average number of iterations of BABLUE is less than that of DLBA for eight functions, indicating that the convergence speed of BABLUE is faster than that of DLBA. From the average number of iterations, we calculate that the convergence rate of BABLUE is 28% - 304% faster than DLBA, with an average increase of 187%. However, comparing the maximum number of iterations, we find that BABLUE is greater than DLBA over the four functions. Therefore, we analyzed the data in detail and found that in 100 independent experiments, only for one time the iteration is large. Considering the random characteristics of the algorithm, we think this situation is reasonable, and will not affect the algorithm optimization results.

For the test function f9-f12, we compare BABLUE with BA and DGOBA. Each function runs 30 times independently, and the termination condition is FEs=8000. The results are shown in Table 4 and Table 5.

From Table 4, we can see that for f9 function, the performance of BABLUE and DGOBA is



(a) (b)

Figure 7: **Optimization curve (a) Optimization curve of f9 function (b) Optimization curve of f10 function**

flat and better than BA, but for f10 rosenbrock function, the search results and the accuracy of BABLUE are far higher than DGOBA. This shows that BABLUE still has a good optimization result on the function of narrow valley terrain similar to f10. Figure 7 respectively represents the optimization results of f9 and f10 on three algorithms. From the figure we can see that BABLUE is obviously faster than BA in search speed, and slightly faster than DGOBA.

The f11 and f12 function has multiple global optimal solutions and local optimal solutions in the solution space and is extremely easy to fall into local optima. Table 5 shows the optimal results on f11 and f12 for BABLUE, BA and DGOBA respectively. Success rate indicates the success rate of the algorithm in finding the global optimal solution (the error does not exceed 1.0e-05). From Table 5 we can see that for different opposite learning strategies (different k values), the success rate of their optimization is different. When k = rand, the performance is best. However, the success rate of BABLUE is obviously higher than that of DGOBA, and even reaches 100% on the f12 function. It proves that our method can perfectly search the global optimal solution. The experimental results show that BABLUE has better searching ability on the function with multiple global optimal solutions and local optimal solutions.

Table 5: **Comparison between BA, DGOBA and BABLUE on f11 and f12 under FEs.**

Function	*Method*	*Min*	*Mean*	*Max*	*std*
f_{11}	BA	-177.167	-186.729	34.9748	36.6%
	DGOBA(K=1/2)	-155.8219	-186.7309	45.0706	63.3%
	DGOBA(K=RAND)	-179.1524	-186.7309	20.7310	86%
	BABLUE	**-182.5203**	**-186.7309**	**16.0226**	**93.3%**
f_{12}	BA	0.47681	0.39793	0.43019	33.3%
	DGOBA(K=1/2)	0.68392	0.39789	0.71343	66.7%
	DGOBA(K=RAND)	0.5025	0.39789	0.0148	96%
	BABLUE	**0.39789**	**0.39789**	**3.6688e−06**	**100%**

4.4 Performance on high dimensions

Table 6: **Comparison between BA, DLBA and BABLUE under High Dimensions.**

Function	Method	Fitness			
		Min	Mean	Max	std
$f_1(D = 1024)$	BA	18463.4895	19752.2669	21239.2311	567.2789
	DLBA	2.6379e−109	4.7374e−87	2.0386e−85	2.8903e−86
	BABLUE	**0**	**0**	**0**	**0**
$f_5(D = 128)$	BA	2587.3876	2811.4065	2966.5672	93.9686
	DLBA	**0**	**0**	**0**	**0**
	BABLUE	**0**	**0**	**0**	**0**
$f_7(D = 320)$	BA	4468.5904	4723.4462	5008.5711	131.1882
	DLBA	**0**	**0**	**0**	**0**
	BABLUE	0	0	0	0
$f_8(D = 256)$	BA	7503.3931	8256.6130	11506.7920	613.3756
	DLBA	2.6689e−94	5.1209e−69	2.5602e−67	3.6207e−68
	BABLUE	**0**	**0**	**0**	**0**

In order to test the effect of the proposed algorithm at high latitudes, we selected four test functions f1, f5, f7 and f8 to perform the experiment with a termination condition of FEs = 8000 and compared with DLBA in [30]. The data of BA and DLBA data from [30] and the experimental results are shown in Table 6. The bold and underline indicate that the experimental results are optimal. From the table we can see that BABLUE still has good search ability even at high latitudes, the accuracy is far higher than BA. Compared with DLBA, BABLUE can find the optimal solution more accurately on the f1 and f8 functions. The experimental results show that the proposed algorithm BABLUE possesses strong exploration capability both in low latitude and high latitude, and the precision of finding the optimal result is very high.

4.5 Experiment of radius

To verify that formula 13, 14 used in 2.4 is superior to formula 12, we carried out experiments on two step length methods on F4, F7 and F10 under the same experimental parameters, and operated 50 times respectively. Success rate represents the probability of searching for extreme point 0 on F4 and F7. The results of the experiment are as follows:

As can be seen from Table 7, it is better to use formula 13 and 14 to calculate the explosion

Table 7: **Comparison between different radius.**

Function	Method	Fitness				Success rate
		Min	Mean	Max	std	
f_4	Radius in (12)	0	6.0226e−05	0.0030	4.2580e−04	90%
	Radius in (13)(14)	**0**	**0**	**0**	**0**	**100%**
f_7	Radius in (12)	0	1.4431e−13	5.7394e−12	8.2808e−13	92%
	Radius in (13)(14)	**0**	**0**	**0**	**0**	**100%**
f_{10}	Radius in (12)	0.3175	0.7050	1.9503	0.3836	-
	Radius in (13)(14)	**0.2829**	**0.6177**	**1.8226**	**0.3163**	-

radius. It proves that it is better to dynamically change the radius with the difference of bat populations than to change the radius dynamically with the number of iterations.

4.6 0-1 knapsack problem

In order to verify whether the proposed algorithm is equally valid in discrete problems, we selected several 0-1 knapsack questions to experiment with, so we discretized the bat positions according to the sigmoid function.

$$x_{id}^t = \begin{cases} 1, & ifrand < sig(x_{id}^t) \\ 0, & else \end{cases} \tag{15}$$

$$x_{id}^t = \frac{1}{1+e^{-x_{id}^t}} \tag{16}$$

Since the 0-1 knapsack problem is discrete, the explosion along the blast radius is no longer applicable. We divide the sparks generated by the explosion into three groups, taking d / 5, 2d / 5, 3d / 5 dimensions respectively from d dimensions. For the first group, each spark has an inverse of the value of one dimension, resulting in d / 5 sparks, for the second group, each spark reverses the value of the two dimensions to produce d / 5 sparks, In the third group, each spark reverses the values of the three dimensions, resulting in d / 5 sparks. The process is shown as Figure 8.

Five typical backpack problem data sets are used, as shown in Table 8. The data of IBBA, BBA and BPSO come from [26] and the experimental results are shown in Table 9.

It can be seen from Table 9 that the algorithm proposed in this paper (BABLUE) still has good searching ability on the 0-1 knapsack problem, and the average value is better than IBBA,

Table 8: **Five typical backpack problem data.**

Number	Dim	Parameters (w,p,C)	Opt
K1	10	w = (95, 4, 60, 32, 23, 72, 80, 62, 65, 46), p = (55, 10, 47, 5, 4, 50, 8, 61, 85, 87), C = 269	295
K2	20	w = (92, 4, 43, 83, 84, 68, 92, 82, 6, 44, 32, 18, 56, 83, 25, 96, 70, 48, 14, 58), p = (44, 46, 90, 72, 91, 40, 75, 35, 8, 54, 78, 40, 77, 15, 61, 17, 75, 29, 75, 63), C =878	1024
K3	50	w = (80, 82, 85, 70, 72, 70, 66, 50, 55, 25, 50, 55, 40, 48, 59, 32, 22, 60, 30, 32, 40, 38, 35, 32, 25, 28, 30, 22, 50, 30, 45, 30, 60, 50, 20, 65, 20, 25, 30, 10, 20, 25,15, 10, 10, 10, 4, 4, 2, 1), p = (220, 208, 198, 192, 180, 180, 165, 162, 160, 158, 155, 130, 125, 122, 120, 118, 115, 110, 105, 101, 100, 100, 98, 96, 95, 90, 88, 82, 80, 77,75, 73, 72, 70, 69, 66, 65, 63, 60, 58, 56, 50, 30, 20, 15, 10, 8, 5, 3, 1), C = 1000	3103
K4	80	w = (40,27, 5, 21, 51, 16, 42, 18, 52, 28, 57, 34, 44, 43, 52, 55, 53, 42, 47, 56, 57, 44, 16, 2, 12, 9, 40, 23, 56, 3, 39, 16, 54, 36, 52, 5, 53, 48, 23, 47, 41, 49, 22, 42, 10, 16, 53, 58,40, 1, 43, 56, 40, 32, 44, 35, 37, 45, 52, 56, 40, 2, 23, 49, 50, 26, 11, 35, 32, 34, 58, 6, 52, 26, 31, 23, 4, 52, 53, 19), p = (199, 194, 193, 191, 189, 178, 174, 169, 164, 164, 161, 158, 157, 154, 152, 152, 149, 142, 131, 125, 124, 124, 124, 122, 119, 116, 114, 113, 111, 110, 109, 100, 97, 94, 91, 82,82, 81, 80, 80, 80, 79, 77, 76, 74, 72, 71, 70, 69, 68, 65, 65, 61, 56, 55, 54, 53, 47, 47, 46, 41, 36, 34, 32, 32, 30, 29, 29, 26, 25, 23, 22, 20, 11, 10, 9, 5, 4, 3, 1), C = 1173	5183
K5	100	w = (54, 95, 36, 18, 4, 71, 83, 16, 27, 84, 88, 45, 94, 64, 14, 80, 4, 23, 75, 36, 90, 20, 77, 32, 58, 6, 14, 86, 84, 59, 71, 21, 30, 22, 96, 49, 81, 48, 37, 28, 6, 84, 19, 55,88, 38, 51, 52, 79, 55, 70, 53, 64, 99, 61, 86, 1, 64, 32, 60, 42, 45, 34, 22, 49, 37, 33, 1, 78, 43, 85, 24, 96, 32, 99, 57, 23, 8, 10, 74, 59, 89, 95, 40, 46, 65, 6, 89, 84, 83,6, 19, 45, 59, 26, 13, 8, 26, 5, 9), p = (297, 295, 293, 292, 291, 289, 284, 284, 283, 283, 281, 280, 279, 277, 276, 275, 273, 264, 260, 257, 250, 236, 236, 235, 235,233, 232, 232, 228, 218, 217, 214, 211, 208, 205, 204, 203, 201, 196, 194, 193, 193, 192, 191, 190, 187, 187, 184, 184, 184, 181, 179, 176, 173, 172, 171, 160, 128, 123, 114,113, 107, 105, 101, 100, 100, 99, 98, 97, 94, 94, 93, 91, 80, 74, 73, 72, 63, 63, 62, 61, 60, 56, 53, 52, 50, 48, 46, 40, 40, 35, 28, 22, 22, 18, 15, 12, 11, 6, 5), C = 3818	15170

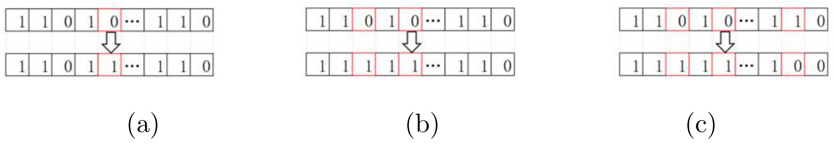


Figure 8: **(a) First Group (b) Second group (c) Third group**

Table 9: **Comparison between BPSO, BBA, IBBA and BABLUE under 0-1 knapsack problem.**

Algorithm	Metric	0-1 knapsack problem datasets				
		K1	K2	K3	K4	K5
BABLUE	Mean	**295**	**1024**	**3091.94**	**5178.72**	**15164.76**
	Std	**0**	**0**	**5.0402**	**4.6556**	**8.7308**
	Max	**295**	**1024**	**3103**	**5183**	**15170**
	Min	**295**	**1024**	**3078**	**5167**	**15141**
IBBA	Mean	**295**	**1024**	3085.7	5134.27	15051.6
	Std	**0**	**0**	7.32097	34.7453	35.6122
	Max	**295**	**1024**	**3103**	5178	**15170**
	Min	**295**	**1024**	**3073**	5063	15046
BBA	Mean	**295**	**1024**	3003.2	4410.67	12942.7
	Std	**0**	**0**	27.3526	95.5894	256.52
	Max	**295**	**1024**	3049	4663	13445
	Min	**295**	**1024**	2944	4251	12489
BPSO	Mean	**295**	956.167	2851.13	3987.77	10089.2
	Std	**0**	23.0563	37.0328	99.9616	238.369
	Max	**295**	1016	2956	4164	10491
	Min	**295**	913	2774	3812	9671

BBA and BPSO, which proves that the algorithm in this paper is still applicable to the discrete problem.

§5 Conclusion

In order to solve the problem that the bat algorithm is easy to fall into the local optimum and the late convergence rate is slow, this paper presents a Uniform Explosion strategy of the local operation. This strategy allows the bat to enhance the local search ability in the random disturbance process and can fly more accurately toward the optimal solution.

At the same time, we add the Cross Boundary Learning strategy, which increases the search range through the original population and the opposite population at the beginning of the algorithm, avoiding the premature convergence of the algorithm into a local optimum.

In addition, in the explosion strategy, because we chose a single point to explode, in order to avoid unnecessary calculation and reduce the running time of the algorithm, we did not use the explosion radius and the number of sparks formula of the original fireworks algorithm, instead, we used a new explosion along the axis, applied the sigmoid function and adjust the explosion radius according to the optimal value and the maximum difference of each generation population.

Through the simulation experiments on 12 typical benchmark functions, the results show that the proposed algorithm is effective and feasible. BABLUE is better than BA in terms of convergence speed and precision, and also shows good performance in single solution, multi-solution, high-dimensional and discrete problems. Compared to other improved bat algorithms, BABLUE also improved to a certain extent, with significant performance advantages.

In future work, we will accelerate the proposed algorithm for CAD/CAM/Graphics/Multimedia applications [41-49]. We also try to extend the proposed approach to other areas of science and engineering, such as social computing and collaboration [50-55].

References

[1] Chengjing W. *A trust region method with a conic model for nonlinearly constrained optimization*, Applied Mathematics-A Journal of Chinese Universities, 2006, 21(3): 263-275.

[2] Xiaojiao T, Shuzi Z. *A trust region algorithm for a class of nonlinear optimization*, Applied Mathematics-A Journal of Chinese Universities, 2000, 15(1): 93-98.

[3] Li H, He F, Yan X. *IBEA-SVM: An Indicator-based Evolutionary Algorithm Based on Pre-selection with Classification Guided by SVM*, Applied Mathematics-A Journal of Chinese Universities, 2019, 34(1): 1-26

[4] Zhenhai, L Yehui P. *A derivative-free algorithm for unconstrained optimization*, Applied Mathematics-A Journal of Chinese Universities, 2005, 20(4): 491-498.

[5] Pan Y, He F, Yu H. *A Novel Enhanced Collaborative Autoencoder with Knowledge Distillation for Top-N Recommender Systems*, Neurocomputing, 2019, 332: 137-148.

[6] Zhu H. *Maximizing group performance while minimizing budget*, IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2017, 99: 1-13.

[7] Sun J, He F, Chen Y, Chen X. *A multiple template approach for robust tracking of fast motion target*, Applied Mathematics-A Journal of Chinese Universities, 2016, 31(2): 177-197.

[8] Carotenuto P, Galiano G, Giordani S, et al. *A Hybrid Metaheuristic Approach for Customer Service Level in the Vehicle Routing Problem*, Working Paper, Istituto di Tecnologie Industriali e Automazione-Sezione di Roma, Italy, 2005.

[9] Chen X, He F, Yu H. *A Matting Method Based on Full Feature Coverage*, Multimedia Tools and Applications, Multimedia Tools and Applications, 2019, 78(9): 11173-11201.

[10] Garg H. *A hybrid PSO-GA algorithm for constrained optimization problems*, Applied Mathematics and Computation, 2016, 274: 292-305.

[11] Li K, He F, Yu H. *Robust Visual Tracking based on Convolutional Features with Illumination and Occlusion Handing*, Journal of Computer Science and Technology? 2018, 33(1): 223-236.

[12] Xiong X, Zhang Y C, Zhang Q D. *An improved self-adaptive PSO algorithm with detection function for multimodal function optimization problems*, Mathematical Problems in Engineering, 2013, 2013.

[13] Zhou Y, He F, Hou N, Qiu Y. *Parallel Ant Colony Optimization on Multi-core SIMD CPUs*, Future Generation Computer Systems, 2018, 79(2): 473-487.

[14] Zhou Yi, He F, Qiu Y. *Dynamic Strategy based Parallel Ant Colony Optimization on GPUs for TSPs*, Science China Information Sciences, 2017, 60(6): 068102.

[15] Yan X, He F, Hou N, Ai H. *An Efficient Particle Swarm Optimization for Large Scale Hardware/Software Co-design System*, International Journal of Cooperative Information Systems, 2018, 27(1): 1741001.

[16] Geem Z W, Kim J H, Loganathan G V. *A new heuristic optimization algorithm: harmony search*, simulation, 2001, 76(2): 60-68.

[17] Hou N, He F, Zhou Y, Chen Y. *An Efficient GPU-based Parallel Tabu Search Algorithm for Hardware/Software Co-design*, Frontiers of Computer Science. DOI: 10.1007/s11704-019-8184-3.

[18] Lewis A, Mirjalili S, Mirjalili S M. *Grey wolf optimizer*, Advances in engineering software, 2014, 69: 46-61.

[19] Li H, He F, Liang Y, Quan Q. A Dividing-based Many-objectives Evolutionary Algorithm for Large-scale Feature Selection. Soft Computing, DOI: 10.1007/s00500-019-04324-5

[20] Simon D. *Biogeography-based optimization*, IEEE transactions on evolutionary computation, 2008, 12(6): 702-713.

[21] Yang X S. *A new metaheuristic bat−inspired algorithm*, Nature inspired cooperative strategies for optimization (2010), pp.65-74.

[22] Yang X S, He X. *Bat algorithm: literature review and applications*, International Journal of Bio-Inspired Computation, 2013, 5(3): 141-149.

[23] Satapathy S C, Raja N S M, Rajinikanth V, et al. *Multi-level image thresholding using Otsu and chaotic bat algorithm*, Neural Computing and Applications, 2018: 1-23.

[24] Shan X, Liu K, Sun P L. *Modified bat algorithm based on lèvy flight and opposition based learning*, Scientific Programming, 2016, Article ID 8031560.

[25] Luo J, He F, Yong J. *An Efficient and Robust Bat Algorithm with Fusion of opposition-based learning and Whale Optimization Algorithm.* Intelligent Data Analysis. 2016, 24(3): 13-29.

[26] Huang X, Zeng X, Han R. *Dynamic inertia weight binary bat algorithm with neighborhood search*, Computational intelligence and neuroscience, 2017, Article ID 3235720.

[27] Sabba S, Chikhi S. *A discrete binary version of bat algorithm for multidimensional knapsack problem*, International Journal of Bio-Inspired Computation, 2014, 6(2): 140-152.

[28] Yilmaz S, Kucuksille E U. *Improved bat algorithm (IBA) on continuous optimization problems*, Lecture Notes on Software Engineering, 2013, 1(3): 279.

[29] Mirjalili S, Mirjalili S M, Yang X S. *Binary bat algorithm*, Neural Computing and Applications, 2014, 25(3-4): 663-681.

[30] Chen H, Xie J, Zhou Y. *A novel bat algorithm based on differential operator and Lévy flights trajectory*, Computational intelligence and neuroscience (2013).

[31] Afrabandpey H, Ghaffari M, Mirzaei A, et al. *A novel bat algorithm based on chaos for optimization tasks*, Intelligent Systems (ICIS), 2014 Iranian Conference on. IEEE, 2014: 1-6.

[32] Tan Y, Zhu Y. *Fireworks algorithm for optimization*, International Conference in Swarm Intelligence. Springer, Berlin, Heidelberg, 2010: 355-364.

[33] Janecek A, Tan Y, Zheng S. *Enhanced fireworks algorithm*, Evolutionary Computation, 2013.

[34] Tizhoosh H R. *Opposition-based learning: a new scheme for machine intelligence*, Computational intelligence for modelling, control and automation, 2005 and international conference on intelligent agents, web technologies and internet commerce, international conference on. IEEE, 2005, 1: 695-701.

[35] Ding L X, Xie D T, Wang S W, et al. *Group search optimizer applying opposition-based learning*, Computer Science, 2012, 39(9): 183-187.

[36] Wu Z J, Wang H, Zhou X Y, et al. *Elite opposition-based particle swarm optimization*, Acta Electronica Sinica, 2013, 41(8): 1647-1652.

[37] Rahnamayan S, Tizhoosh H R, Salama M M A. *Opposition-based differential evolution*, IEEE Transactions on Evolutionary computation, 2008, 12(1): 64-79.

[38] Hou N, Yan X, He F. *A Survey on Partitioning Models, Solution Algorithms and Algorithm Parallelization for Hardware/Software Co-design*, Design Automation for Embedded Systems, 2019, 23(1-2): 57-77.

[39] Yong J, He F, Li H, et al. *A Novel Bat Algorithm based on Collaborative and Dynamic Learning of Opposite Population*, 2018 IEEE 22nd International Conference on Computer Supported Cooperative Work in Design (CSCWD), IEEE, 2018: 541-546.

[40] Yilmaz S, Kucuksille E U, Cengiz Y. *Modified bat algorithm*, Elektronika ir Elektrotechnika, 2014, 20(2): 71-79.

[41] Wu Y, He F, Zhang D, Li X. *Service-Oriented Feature-Based Data Exchange for Cloud-Based Design and Manufacturing*, IEEE Transactions on Services Computing, 2018, 11(2): 341-353.

[42] Li K, He F, Yu H, Chen X. *A Parallel and Robust Object Tracking Approach Synthesizing Adaptive Bayesian Learning and Improved Incremental Subspace Learning*, Frontiers of Computer Science, 2019, 13(5): 1116-1135.

[43] Zhang Q, Nie, Y, Zhan, L, Xiao C. *Underexposed video enhancement via perception-driven progressive fusion*, IEEE Transactions on Visualization and Computer Graphics, 2015, 22(6): 1773-1785.

[44] Zhang S, He F, Ren W, Yao J. *Joint learning of image detail and transmission map for single image dehazing*, The Visual Computer, DOI: 10.1007/s00371-018-1612-9.

[45] Yu H, He F, Pan Y. *A Novel Segmentation Model for Medical Images with Intensity Inhomogeneity Based on Adaptive Perturbation*, Multimedia Tools and Applications, 2019, 78(9): 11779-11798

[46] Liotta G, Stecca G, Kaihara T. *Optimisation of freight flows and sourcing in sustainable production and transportation networks*, International Journal of Production Economics, 2015, 164: 351-365.

[47] Yu H, He F, Pan Y. *A novel region-based active contour model via local patch similarity measure for image segmentation*, Multimedia Tools and Applications, 2018, 77(18): 24097-24119.

[48] Liotta G, Kaihara T, Stecca G. *Optimization and simulation of collaborative networks for sustainable production and transportation*, IEEE Transactions on Industrial Informatics, 2016, 12(1): 417-424.

[49] Zhang J, He F, Chen Y. *A new haze removal approach for sky/river alike scenes based on external and internal clues*, Multimedia Tools and Applications, DOI: 10.1007/s11042-019-08399-y.

[50] Zhu H. *Role-Based Collaboration and the E-CARGO: Revisiting the Developments of the Last Decade*, IEEE Systems, Man, and Cybernetics Magazine, 2015, 1(3): 27-35.

[51] Ni B, He F, Pan Y, Yuan Z. *Using Shapes Correlation for Active Contour Segmentation of Uterine Fibroid Ultrasound Images in Computer-Aided Therapy*, Applied Mathematics-A Journal of Chinese Universities, 2016, 31(1): 37 - 52.

[52] Lv X, He F, Cai W, Cheng Y. *An optimized RGA supporting selective undo for collaborative text editing systems*, Journal of Parallel and Distributed Computing, 2019, 132:310-330.

[53] Pan YT, He FZ, Yu HP. *A Correlative Denoising Autoencoder to Model Social Influence for Top-N Recommender System*, Frontiers of Computer Science, DOI: 10.1007/s11704-019-8123-3.

[54] Li K, He F, Yu H, Chen X. *A Correlative Classifiers Approach based on Particle Filter and Sample Set for Tracking Occluded Target*, Applied Mathematics-A Journal of Chinese Universities, 2017, 32(3): 294-312.

[55] Zhu, H. *Avoiding Critical Members in a Team by Redundant Assignment*, IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2018, 99: 1-12.

[1] School of Computer Science, Wuhan University, Wuhan 430072, China.

[2] State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan 430074, China.

Email: fzhe@whu.edu.cn