A Modified Bacterial Foraging Optimizer with Adaptive Chemotactic Step in Dynamic Search Region

Yibo Yong[1], Lianbo Ma[1](✉), Junfeng Zhao[2], and Xiaolong Shen[2]

[1] College of Software, Northeastern University, Shenyang, China
malb@swc.neu.edu.cn
[2] 2012 Laboratories, Huawei Technologies Co., Ltd., Shenzhen, China

Abstract. Bacterial foraging optimization (BFO), inspired from the foraging process of bacterium called *E.coli*, has been applied successfully to a variety of real world optimization problems. However, BFO easily encounters the issue of poor convergence when dealing with complex landscapes of optimization problems due to its inherent fixed chemotactic strategy. Aiming at the above issue, an adaptive bacterial foraging optimizer is presented in this paper, which is able to obtain a good balance between exploration and exploitation during the search. In this approach, the chemotactic step-length is adjusted dynamically, that is a larger chemotactic step is for global search and a smaller chemotactic step is conducive to local search. Moreover, the outstanding swarming pattern is incorporated to perform information sharing in population during the evolution, aiming to maintain diversity and convergence. Simulation results on a set of benchmark functions validate the effectiveness of the proposed algorithm.

Keywords: Bacteria foraging optimization · Adaptive chemotactic step · Swarm intelligence

1 Introduction

As an important computation paradigm in artificial intelligence, swarm intelligence algorithms, which are usually based on bio-inspired computation paradigms, have received a surge of attentions recently. The promising examples include Genetic Algorithm (GA) [1], Artificial Bee Colony (ABC) [2, 3], Particle Swarm Optimization (PSO) [4], Brain Storm Optimization (BSO) [5], Ant Colony Optimization [6], etc. Such biologically inspired algorithms are frequently used to optimize many-objective problems [7]. Among them, bacterial foraging optimization (BFO) algorithm [8] was firstly proposed by Passino in 2002 [9], based on the principle of foraging behaviors of *E.coli* bacteria. The bacterial swarm is essentially a complex adaptive system, where the foraging behaviors show the features of self-organization and self-adaptation. These interesting patterns inspire to develop an optimization paradigm based on bacterial foraging behaviors. Up to now, many improved BFO algorithms have been proposed such as citations [10–14]. BFO algorithm has been applied to many real-world problems, such as harmonic signal estimation [15], PID controller design [16], optimal power flow [17] and optimal power system stabilizers design [18], load forecasting [19], stock market prediction [20], optimum economic dispatch [21].

BFO algorithm solves single-objective the problem, which is different from multi-objective algorithm [22]. When dealing with complex optimization problems, however, BFO suffers from low convergence speed and being trapped into local optima easily. According, this paradigm can't obtain convincing results in a certain range of optimization problems, due to a chemotactic step-length is always set as a constant no matter which phase the optimization is. For this weakness, a modified adaptive bacterial foraging optimization (MABFO) algorithm is devised by incorporating a modified adaptive chemotactic strategy including adaptive step-length and adaptive tumbling in this paper.

The remainder of this paper is organized as follows: Sect. 2 provides a brief introduction to BFO. Section 3 introduces the proposed MABFO. The simulation results of evaluating MABFO on nine benchmark functions and discussions are shown in Sect. 4. The conclusion is drawn in Sect. 5.

2 Bacterial Foraging Optimization

In principle, bacteria tend to gather towards the nutrient-rich areas in the living environment. In such dynamical environment, the most adapted bacteria survive into the next generation and have a change to reproduce via natural selection, passing on their genetic traits to their offspring. At the same time, those that can't survive are eliminated. Some bacteria may migrate to other places, leading to extinction and migration operations in real bacterial population. In the search process, bacteria in population share information via cell-to-cell communication. This foraging activity can inspire the researchers to use it as optimization process.

The classical BFO algorithm comprises four processes, namely, chemotaxis, swarming, reproduction and elimination-dispersal. The following is brief description of these four processes.

2.1 Chemotaxis

This process simulates swimming behavior and tumbling behavior behavior of *E.coli* bacteria via flagella. The movement ways of bacteria include two types: swimming and tumbling. Biologically, bacteria can move in the above two ways alternatively, that is, swimming in the same direction or possibly tumbling for a period of time. Suppose $\theta^i(j,k,l)$ represents the position of *ith* bacterial after the *jth* chemotaxis operation, *kth* reproduction operation and *lth* extinction and migration operation. $C(i)$ represents the swimming step-length of the *ith* individual in random selected direction, $\angle\phi(j)$ represents a vector of random direction. The mathematical expression of bacterial chemotaxis phase is defined as follows:

$$\theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\angle\phi(j) \quad (1)$$

2.2 Swarming

In this process, a cell-to-cell communication is simulated by releasing different signals. As all bacterium are on the move, they release signals to attract other bacteria swimming toward it. At the same time, each bacterium releases a rejection signal to warn other bacteria maintaining a safe distance in face of harmful substances. Bacterial foraging optimization algorithm simulates the social behavior of information sharing and cooperation among bacteria. In the search process, bacteria should not only remember their own information, but also consider the information of their companions to transmit to each other. The numerical sequence expression of the influence value of swarming can be formulated as follows:

$$\begin{aligned} J_{cc}(\theta, P(j,k,l)) &= \sum_{i=1}^{s} J_{cc}^{i}(\theta, \theta^{i}(j,k,l)) \\ &= \sum_{i=1}^{s} [h_{repelent} \exp(-w_{repelent} \sum_{m=1}^{p} (\theta_m - \theta_m^i)^2)] \\ &+ \sum_{i=1}^{s} [-d_{attract} \exp(-w_{attract} \sum_{m=1}^{p} (\theta_m - \theta_m^i)^2)] \end{aligned} \tag{2}$$

where s is the total number of bacteria. p is the number of optimized parameters for each bacterium, that is, the position of individuals in the popultion. $d_{attract}$, $w_{attract}$, $h_{repelent}$, $w_{repelent}$ are different coefficients.

2.3 Reproduction

The evolution and development of bacteria follow principle of 'The survival of the fittest in natural selection'. After the chemotaxis process, some selections are obviously failed, and the individuals with weak foraging ability are eliminated. The individuals with strong foraging ability maintain stability of population quantity and improve quality of population through reproduction.

Health status of each bacterium is represented by sum of adaptive values of each chemotactic step-length in its life cycle:

$$J_{health}^{i} = \sum_{j=1}^{Nc} J(i,j,k,l) \tag{3}$$

In order to avoid the disappearance of good feasible solutions in the search process, this strategy eliminates the inferior ones and retain the excellent ones. All bacteria are ordered according to the fitness function. Half of these bacteria with poor fitness will be eliminated, and the remaining bacteria will replicate, so as to keep the number of bacterial population unchanged.

2.4 Elimination-Dispersal

E.coli bacterium may trap into local optimum. In order to improve the global search ability, the bacteria will die or migrate after N_{re} reproduction operation. With the probability of P_{ed}, bacteria are dispersed to any location within the optimized region. This elimination-dispersal operation can reduce the possibility of bacteria falling into local optimal solution. The number of elimination-dispersal operation is denoted as N_{ed}.

3 Modified Adaptive Bacterial Foraging Optimization

During the search process of BFO, the search region size of population at different phases is shown in Fig. 1 and Fig. 2. Figure 1 shows distribution of population at the initial phase, and Fig. 2 demonstrates distribution of population at the later period. As the algorithm runs, the search region becomes smaller. If a chemotactic step-length is too small, the convergence speed will be slow. On the contrary, it may occur oscillation phenomenon, as shown in Fig. 3, causing the bacteria to trap into local optimum easily. Therefore, it is difficult to balance exploration and exploitation with a fixed chemotactic step-length, which affects both the accuracy and speed of algorithm search. To summarize, a proper chemotactic step-length is critical to the performance of BFO.

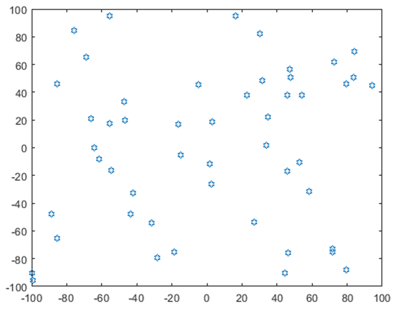

Fig. 1. Population distribution-1

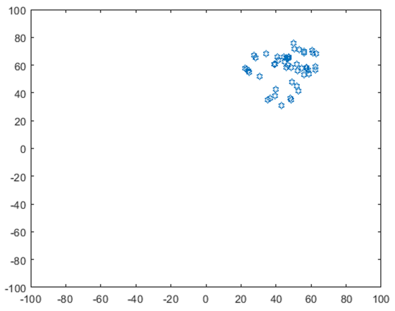

Fig. 2. Population distribution-2

The size of chemotactic step-length is dynamically adjusted in the reproduction process, which ensures the bacteria moving towards the global optimum quickly at the beginning of whole search process, and converging to the global optimum accurately in the end. In this way, it could balance exploration and exploitation of BFO.

Based on the above-mentioned analysis about changing chemotactic step-length, we proposed an adaptive step-length based on dynamic search region in which at each

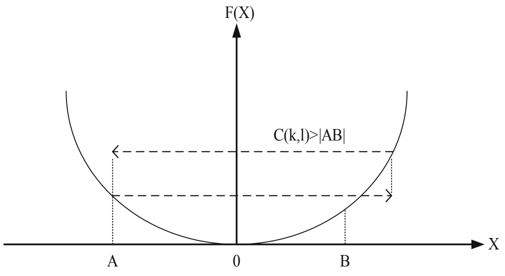

Fig. 3. Oscillation phenomenon of solution

reproduction and elimination-dispersal process of algorithm. The novel adaptive chemotactic step-length formula is shown as follows:

$$\begin{cases} C^i(k,l) = \lambda * R_f^{\frac{T_i}{T_{\max}}} \\ R_f = \frac{R(k,l)}{R(1,1)} \\ R(k,l) = \frac{1}{n} * \sum_{i=1}^{n} |x_i - x_{best}| \end{cases} \tag{4}$$

where λ is a constant controlling the step-leng that the beginning of whole search process. x_i is position of the *ith* bacterium x_{best} is the best position in current population. T_i is total number of motion steps of *ith* bacterium in current reproduction phase. $T_{\max}$ is the ideally maximal number of motion steps in a reproduction phase. If T_i is larger, the area *ith* bacterium searches is more likely to get the optimal value. Thus, a large value C^i is needed to increase ability of exploitation. On the contrary, the value of C^i for *ith* bacterium is less than other bacteria. This strategy is inspired by non-uniform mutation [23]. This means that step-length of each bacterium in population may be different at the same time, which greatly increases the local search ability of population.

Moreover, MABFO also introduced a mechanism that new position of each bacterium is calculated according to Eq. (5), which makes all bacteria moving towards the global optimum as far as possible at the beginning of the iteration and maintaining sufficient diversity. In this way, it can obtain expected convergence and diversity in the whole search process.

$$\begin{cases} \theta^i(j+1,k,l) = \theta^i(j,k,l) + C(i)\angle\phi(i,j) \\ \angle\phi(i,j) = (1-R_f) * \angle\phi_{rand} + R_f * D_i \\ D_i = x_{best} - x_i \end{cases} \tag{5}$$

where $\angle\phi(i,j)$ is direction of movement (swimming or tumbling) of *ith* bacterium in the *jth* chemotaxis. $\angle\phi_{rand}$ is a random direction.

The pseudocode of MABFO is given in Table 1. *pop* represents the number of bacteria. N_s represents the maximum number of swimming, N_c represents the maximum number of chemotaxis, N_{re} represents the maximum number of reproduction, N_{ed} represents the maximum number of elimination and dispersal, P_{ed} represents the maximum number of elimination-dispersal. The main process include: first, the initialization procedure generates *pop* initial solutions. It performs chemotactic process after initialization. Then it executes reproduction process. Finally, it carries out elimination-dispersal process. The computational complexity of MABFO isn't significantly different compared with classic BFO, because the size of chemotactic step-length is only dynamically adjusted in the reproduction process.

Table 1. Pseudocode of MABFO.

```
Randomly initialize positions of bacteria in the domain
FOR (Elimination-dispersal $l$=1: $N_{ed}$ )
  FOR (Reproduction loop $k$=1: $N_{re}$ )
    FOR (Chemotaxis loop $j$ =1: $N_c$ )
      FOR (each bacterium $i$ =1: $pop$)
        Calculate: Calculate $J^i(j,k,l)$ and set as $J_{last}$ ;
        Tumble: Generate a random angle represented by $\angle\phi_{rand}$ . Move to a random
                direction $\angle\phi(i,j)$ by a unit walk, the new position is calculated
                by equation (3-2).
        Run:    For bacterium $i$, calculate the step fitness as $J_i$ . If $J_i(j,k) < J_{last}$ ,
                continue to swim until $N_s$ steps. Otherwise, start another chemotac-
                tic step.
      END FOR (each bacterium)
    END FOR (Chemotaxis loop)
    Calculate adaptive chemotactic step-length $C(k,l)$ by equation (3-1).
  END FOR (Reproduction loop)
  Perform elimination-dispersal with a probability $P_{ed}$ for each bacterium.
END FOR (Elimination-dispersal loop)
```

4 Simulation Studies and Discussions

4.1 The Benchmark Function

To evaluate the performance of MABFO, nine benchmark functions selected from citation [24] as follows (Table 2):

Table 2. Test function formula.

Functions	Formula
Shifted and Rotated Zakharov Function	$F_1(x) = f_3(M(x - o_3)) + 300$
Shifted and Rotated Expanded Scaffer's Function	$F_2(x) = f_{20}(M(\frac{0.5(x-o_6)}{100})) + 600$
Shifted and Rotated Lunacek Bi_Rastrig-in Function	$F_3(x) = f_7(M(\frac{600(x-o_7)}{100})) + 700$
Hybrid Function 1 (N = 3)	$F_4(x) = f_3(M_1z_1) + f_4(M_2z_2) + f_5(M_3z_3) + 1100$
Hybrid Function 5 (N = 4)	$F_5(x) = f_1(M_1z_1) + f_{18}(M_2z_2) + f_5(M_3z_3) + f_4(M_3z_3) + 1500$
Hybrid Function 6 (N = 5)	$F_6(x) = f_{16}(M_1z_1) + f_{13}(M_2z_2) + f_{19}(M_3z_3) + f_{10}(M_2z_2) + f_5(M_3z_3) + 1700$
Composition Function 3 (N = 4)	$F_7(x) = \sum_{i=1}^{N} \{\omega_i * [\lambda_i g_i(x) + bias_i]\} + 2300$
Composition Function 6 (N = 5)	$F_8(x) = \sum_{i=1}^{N} \{\omega_i * [\lambda_i g_i(x) + bias_i]\} + 2600$
Composition Function 10 (N = 3)	$F_9(x) = \sum_{i=1}^{N} \{\omega_i * [\lambda_i g_i(x) + bias_i]\} + 2900$

The test suite includes nine benchmark functions in this paper, encompassing uni-modal, simple multi-modal, hybrid and composition problems. Function F1 is uni-modal function. Functions F2–F3, which are more complex optimization problems, are simple multi-modal functions. Functions F4–F6 are hybrid functions, and functions F7–F9 are composition functions. Both hybrid function and composition functions consist of a variety of basic functions, where the number of local minimum increases exponentially with the problem dimension. So they are considered to be the most difficult class of optimization problems. More detailed function definition is given in citation [24].

4.2 Parameter Settings

The performance of the proposed MABFO is evaluated on the benchmark functions in comparison with PPBSO [25], ABFO [13] and BFO [8]. Here, the total number of iterations for each algorithm is set as 2000. Additionally, as for other specific parameters of each algorithm, as shown in Table 3:

It can be known from the description in Sect. 3, λ is important for the performance for the proposed algorithm. We should choose an appropriate value of λ to further improve the performance of the algorithm. Table 4 shows the stimulation results of nine functions with different values of λ. There are the largest number of functions that can get the most accurate value when coefficient (λ) is set as 1.0.

Table 3. Parameter settings.

Algorithm	Parameter
MABFO	$\lambda = 1.0,\ C\_\min = 0.01,\ N_s = 4,\ N_c = 250,\ N_{re} = 4,\ N_{ed} = 2,\ P_{ed} = 0.25$
PPBSO	$m = 30, Nc_{\max} = 200, K = 3, p_{5a} = 0.2, p_{6b} = 0.8, p_{6b3} = 0.4, p_{6c} = 0.5,$ $w_{predator} = 0.05, p_{prey} = 0.1$
ABFO	$C\_\min = 0.01,\ C\_\max = 0.1,\ a = 4,\ n = 4,\ N_s = 4,\ N_c = 250,$ $N_{re} = 4,\ N_{ed} = 2,\ P_{ed} = 0.25$
BFO	$N_s = 4,\ N_c = 250,\ N_{re} = 4,\ N_{ed} = 2,\ P_{ed} = 0.25$

Table 4. Results with different values of coefficient λ on test functions

Functions		$\lambda = 0.5$	$\lambda = 0.6$	$\lambda = 0.7$	$\lambda = 0.8$	$\lambda = 0.9$	$\lambda = 1.0$
F1	10D	**3.02e + 02**	3.02e + 02	3.03e + 02	3.05e + 02	3.04e + 02	3.06e + 02
	30D	3.62e + 04	3.36e + 04	2.93e + 04	2.38e + 04	2.23e + 04	**1.79e + 04**
F2	10D	6.14e + 02	6.13e + 02	6.11e + 02	6.10e + 02	6.08e + 02	**6.07e + 02**
	30D	6.49e + 02	6.48e + 02	6.49e + 02	6.46e + 02	6.48e + 02	**6.45e + 02**
F3	10D	7.34e + 02	7.29e + 02	**7.28e + 02**	7.30e + 02	7.28e + 02	7.30e + 02
	30D	1.16e + 03	1.11e + 03	1.06e + 03	1.06e + 03	1.04e + 03	**1.03e + 03**
F4	10D	1.11e + 03	1.12e + 03	1.11e + 03	1.11e + 03	1.11e + 03	**1.11e + 03**
	30D	1.39e + 03	1.36e + 03	**1.34e + 03**	1.37e + 03	1.34e + 03	1.35e + 03
F5	10D	1.53e + 03	1.53e + 03	1.53e + 03	1.54e + 03	**1.53e + 03**	1.54e + 03
	30D	1.94e + 04	2.02e + 04	2.59e + 04	**1.83e + 04**	2.56e + 04	2.86e + 04
F6	10D	1.75e + 03	1.74e + 03	**1.74e + 03**	1.74e + 03	1.74e + 03	1.75e + 03
	30D	2.19e + 03	2.16e + 03	2.17e + 03	2.10e + 03	2.13e + 03	**2.10e + 03**
F7	10D	2.79e + 03	2.77e + 03	2.78e + 03	**2.68e + 03**	2.69e + 03	2.74e + 03
	30D	4.57e + 03	4.41e + 03	4.42e + 03	4.39e + 03	**4.25e + 03**	4.32e + 03
F8	10D	2.97e + 03	2.86e + 03	2.89e + 03	2.86e + 03	2.95e + 03	**2.85e + 03**
	30D	**2.88e + 03**	3.17e + 03	2.90e + 03	2.90e + 03	2.91e + 03	2.92e + 03
F9	10D	3.18e + 03	3.17e + 03	3.17e + 03	3.17e + 03	3.17e + 03	**3.16e + 03**
	30D	3.88e + 03	3.83e + 03	3.91e + 03	3.72e + 03	**3.72e + 03**	3.75e + 03

4.3 Simulation Results

The merits and characteristics of MABFO are discussed in comparison with PPBSO, ABFO and BFO as follows. Table 5 demonstrates simulation results obtained by MABFO, PPBSO, ABFO and BFO applied on the nine benchmark functions respectively. The simulation results for these benchmark functions are shown in Fig. 1, 2, 3 and 4 which describe the convergence process of MABFO, PPBSO, ABFO and BFO respectively.

Precision of Optimization: The proposed MABFO algorithm is evaluated on the 10-dimension and 30-dimension benchmark functions in comparison with PPBSO, ABFO and BFO respectively. Each algorithm runs 20 times to give a mean value of the best solutions and a standard deviation.

Table 5. The performance comparison of MABFO, PPBSO, ABFO and BFO.

Functions			MABFO	PPBSO	ABFO	BFO
F1	10D	Mean	3.0584e + 02	**3.0000e + 02**	1.3749e + 03	8.1190e + 03
		SD	2.6261e + 00	**4.5829e − 14**	4.0948e + 02	3.7756e + 03
	30D	Mean	1.7774e + 04	**8.8504e + 03**	1.2257e + 05	1.4437e + 05
		SD	5.9123e + 03	**9.3753e + 03**	1.5786e + 04	1.9211e + 04
F2	10D	Mean	**6.0759e + 02**	6.2205e + 02	6.2526e + 02	6.4604e + 02
		SD	**4.0422e + 00**	9.5718e + 00	6.2246e + 00	5.9501e + 00
	30D	Mean	**6.4593e + 02**	6.5011e + 02	6.7454e + 02	6.6973e + 02
		SD	**7.7749e + 00**	7.7364e + 00	5.4036e + 00	4.4806e + 00
F3	10D	Mean	**7.2814e + 02**	7.8390e + 02	7.6519e + 02	1.0776e + 03
		SD	**5.1883e + 00**	2.9834e + 01	1.1577e + 01	9.2470e + 01
	30D	Mean	**1.0340e + 03**	1.3067e + 03	1.7438e + 03	3.1204e + 03
		SD	**4.8066e + 01**	1.1808e + 02	1.2740e + 02	2.7588e + 02
F4	10D	Mean	**1.1105e + 03**	1.1245e + 03	1.1108e + 03	1.1512e + 03
		SD	**5.1915e + 00**	1.4632e + 01	3.9058e + 00	2.9547e + 01
	30D	Mean	**1.3347e + 03**	1.3363e + 03	1.5922e + 03	1.8618e + 04
		SD	**4.7908e + 01**	8.1675e + 01	6.6374e + 01	7.6777e + 03
F5	10D	Mean	**1.5373e + 03**	1.8244e + 03	1.6795e + 03	1.8381e + 03
		SD	**1.8705e + 01**	4.2506e + 02	7.3928e + 01	1.4335e + 02
	30D	Mean	2.8687e + 04	**1.2391e + 04**	6.3020e + 04	1.0596e + 08
		SD	1.8467e + 04	**8.9803e + 03**	2.1868e + 04	1.1227e + 08
F6	10D	Mean	**1.7471e + 03**	1.7490e + 03	1.7506e + 03	1.8156e + 03
		SD	**1.3037e + 01**	1.2734e + 01	1.7534e + 01	8.1156e + 01
	30D	Mean	**2.0732e + 03**	2.5271e + 03	2.2469e + 03	2.9394e + 03
		SD	**1.6597e + 02**	3.3938e + 02	1.4962e + 02	3.1378e + 02
F7	10D	Mean	**2.7222e + 03**	2.7292e + 03	2.7269e + 03	2.9380e + 03
		SD	**1.0978e + 02**	5.0220e + 01	3.8044e + 01	6.8809e + 01
	30D	Mean	4.2918e + 03	**3.5391e + 03**	3.9840e + 03	4.5136e + 03
		SD	3.2712e + 02	**2.0798e + 02**	2.3874e + 02	2.3819e + 02
F8	10D	Mean	2.8836e + 03	2.9564e + 03	**2.8490e + 03**	3.2559e + 03
		SD	1.7279e + 02	3.4298e + 02	**5.6717e + 01**	4.9001e + 02
	30D	Mean	**2.9342e + 03**	3.3389e + 03	3.3761e + 03	1.2707e + 04
		SD	**1.1595e + 01**	1.6167e + 03	2.7501e + 02	1.3268e + 03
F9	10D	Mean	**3.1692e + 03**	3.2022e + 03	3.1800e + 03	3.3064e + 03
		SD	**1.8905e + 01**	8.2237e + 01	2.2377e + 01	9.0455e + 01
	30D	Mean	**3.7376e + 03**	4.0253e + 03	4.0250e + 03	4.9156e + 03
		SD	**2.0469e + 02**	3.0944e + 02	1.9343e + 02	2.9212e + 02

From Table 5, it can be seen clearly that the average convergence value of the proposed MABFO algorithm ranks first for 13 times in total among the nine functions, and has small deviation at the same time. This demonstrates that the search accuracy of MABFO algorithm is better than the other three algorithms. This is because that the modified adaptive chemotactic step strategy of MABFO can help escape from local optimum and improve its capability of searching global optimum.

Speed of Convergence: From the results presented in Fig. 4, 5, 6 and 7, comparing MABFO and classic BFO, the adaptive chemotactic step-length mechanism of MABFO can balance exploration and exploitation on the whole phase. Because MABFO can use big chemotactic step-length for fast searching feasible solution region in exploration phase, and for local search with small chemotactic step-length in exploitation phase quickly. On the contrary, the constant chemotactic step-length of classic BFO unable to maintain high-speed search ability in different environments.

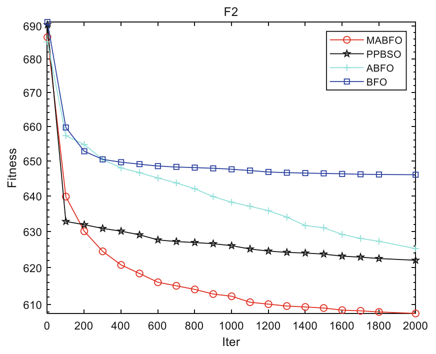

Fig. 4. Convergence curve of F2 10-D

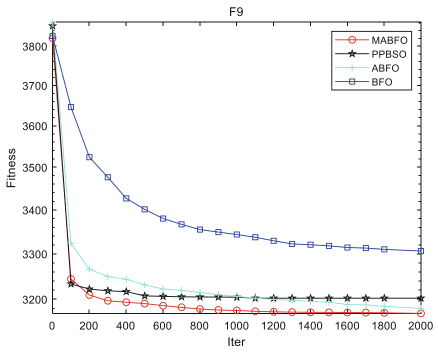

Fig. 5. Convergence curve of F9 10-D

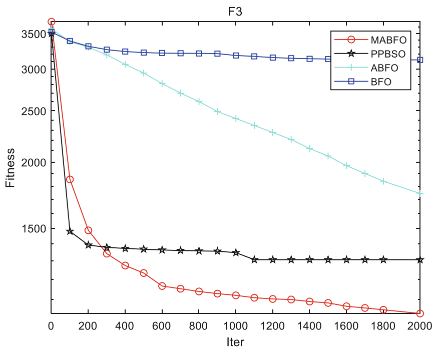

Fig. 6. Convergence curve of F3 30-D

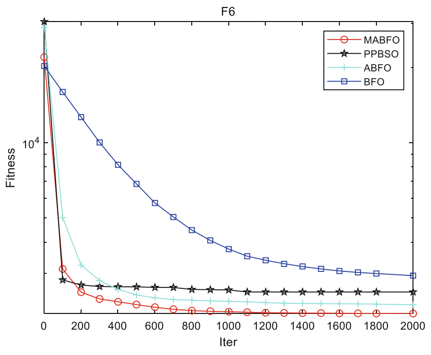

Fig. 7. Convergence curve of F6 30-D

From Fig. 4, 5, 6 and 7, we can find that the convergence speed of MABFO is better than PPBSO, ABFO and BFO. The convergence speed of MABFO and PPBSO at the same level roughly in the initial search phase. Because MABFO not only adds mechanism of adaptive chemotactic step-length, but also introduces the adaptive tumbling mode that makes full use of the information of the global best individual.

A larger chemotactic step-length and more suitable movement direction can obtain a relatively better convergence speed in the initial search phase. Moreover, MABFO also has faster convergence speed than other three algorithms in the middle and late stage of whole search process, because it has a desired chemotactic step-length and direction in the whole dynamic search region.

5 Conclusions

In this paper, a modified adaptive chemotaxis strategy in dynamic search region is proposed to improve the BFO algorithm. Nine benchmark functions were then used to test effectiveness of the proposed improvement. Compared with PPBSO, ABFO and BFO, the proposed MABFO algorithm has fast search performance according to its adaptive character in whole search phase. In a large number of scientific research and engineering application practice, people found that biologically inspired algorithm has some problems, mainly premature convergence, and easy to fall into the local optimal. Evaluation results obtained on benchmark functions have proved the effectiveness of the proposed MABFO algorithm in solving problems of premature convergence and easy to fall into the local optimal. But the performance on the 30-dimension is worse than that on the 10-dimension. This encourages us to study further to solve higher dimensions optimization problems.

Acknowledgment. This work was supported in part by National Natural Science Foundation of China under Grant No. 61773103 and Huawei HIRP project under No. HO2019085002.

References

1. Whitley, D.: A genetic algorithm tutorial. Stat. Comput. **4**(2), 65–85 (1994)
2. Karaboga, D., Akay, B.: A comparative study of artificial bee colony algorithm. Appl. Math. Comput. **214**(1), 108–132 (2009)
3. Ma, L., Hu, K., Zhu, Y., Chen, H.: Cooperative artificial bee colony algorithm for multi-objective RFID network planning. J. Network Comput. Appl. **42**, 143–162 (2014)
4. Shi, Y.: Particle swarm optimization: developments, applications and resources. In: Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546), vol. 1, pp. 81–86. IEEE (2001)
5. Ma, L., Cheng, S., Shi, Y.: Enhancing learning efficiency of brain storm optimization via orthogonal learning design. IEEE Trans. Syst. Man Cybern. Syst. (2020). https://doi.org/10.1109/tsmc.2020.2963943
6. Dorigo, M., Birattari, M., Stutzle, T., et al.: Ant colony optimization: artificial ants as a computational intelligence technique. IEEE Comput. Intell. Mag. **1**(4), 28–39 (2006)
7. Ma, L., Wang, R., Chen, M., Wang, X., Cheng, S., Shi, Y.: A novel many-objective evolutionary algorithm based on transfer learning with kriging model. Inf. Sci. **509**, 437–456 (2020)
8. Das, S., Biswas, A., Dasgupta, S., et al.: Bacterial foraging optimization algorithm: theoretical foundations, analysis, and applications. In: Abraham, A., Hassanien, A.E., Siarry, P., Engelbrecht, A. (eds.) Foundations of Computational Intelligence Volume 3, vol. 203, pp. 23–55. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-01085-9_2

9. Passino, K.M.: Biomimicry of bacterial foraging for distributed optimization and control. IEEE Control Syst. Mag. **22**(3), 52–67 (2002)
10. Tang, W.J., Wu, Q.H., Saunders, J.R.: Bacterial foraging algorithm for dynamic environments. In: 2006 IEEE International Conference on Evolutionary Computation, pp. 1324–1330. IEEE (2006)
11. Chu, Y., Mi, H., Liao, H., et al.: A fast bacterial swarming algorithm for high-dimensional function optimization. In: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), pp. 3135–3140. IEEE (2008)
12. Dasgupta, S., Das, S., Abraham, A., et al.: Adaptive computational chemotaxis in bacterial foraging optimization: an analysis. IEEE Trans. Evol. Comput. **13**(4), 919–941 (2009)
13. Niu, B., Wang, H., Tan, L., et al.: Improved BFO with adaptive chemotaxis step for global optimization. In: 2011 Seventh International Conference on Computational Intelligence and Security, pp. 76–80. IEEE (2011)
14. Chen, H., Zhu, Y., Hu, K.: Adaptive bacterial foraging optimization. Abstract and Applied Analysis, Hindawi (2011)
15. Mishra, S.: A hybrid least square-fuzzy bacterial foraging strategy for harmonic estimation. IEEE Trans. Evol. Comput. **9**(1), 61–73 (2005)
16. Kim, D.H., Cho, J.H.: Adaptive tuning of PID controller for multivariable system using bacterial foraging based optimization. In: Szczepaniak, Piotr S., Kacprzyk, J., Niewiadomski, A. (eds.) AWIC 2005. LNCS (LNAI), vol. 3528, pp. 231–235. Springer, Heidelberg (2005). https://doi.org/10.1007/11495772_36
17. Tang, W.J., Li, M.S., He, S., et al.: Optimal power flow with dynamic loads using bacterial foraging algorithm. In: 2006 International Conference on Power System Technology, pp. 1–5. IEEE (2006)
18. Das, T.K., Venayagamoorthy, G.K., Aliyu, U.O.: Bio-inspired algorithms for the design of multiple optimal power system stabilizers: SPPSO and BFA. In: Industry Applications Conference, pp. 1445–1457. IEEE (2006)
19. Ulagammai, M., Venkatesh, P., Kannan, P.S., et al.: Application of bacterial foraging technique trained artificial and wavelet neural networks in load forecasting. Neuro Comput. **70**(16–18), 2659–2667 (2007)
20. Majhi, R., Panda, G., Majhi, B., et al.: Efficient prediction of stock market indices using adaptive bacterial foraging optimization (ABFO) and BFO based techniques. Expert Syst. Appl. **36**(6), 10097–10104 (2009)
21. Farhat, I.A., Elhawary, M.E.: Dynamic adaptive bacterial foraging algorithm for optimum economic dispatch with valve-point effects and wind power. IET Gener. Trans. Distrib. **4**(9), 989–999 (2010)
22. Ma, L., Wang, X., Huang, M., Lin, Z., Tian, L., Chen, H.: Two-level master-slave RFID networks planning via hybrid multi-objective artificial bee colony optimizer. IEEE Trans. Syst. Man Cybern. Syst. **49**(5), 861–880 (2019)
23. Deep, K., Thakur, M.: A new mutation operator for real coded genetic algorithms. Appl. Math. Comput. **193**(1), 211–230 (2007)
24. Awad, N.H., Ali, M.Z., Suganthan, P.N., Liang, J.J., Qu, B.Y.: Problem definitions and evaluation criteria for the CEC 2017 special session and competition on single objective real-parameter numerical optimization. Technical report (2016)
25. Duan, H., Li, S., Shi, Y.: Predator-prey brain storm optimization for DC brushless motor. IEEE Trans. Magn. **49**(10), 5336–5340 (2013)