Journal of Computers Vol. 29 No. 1, 2018, pp. 228-241
doi:10.3966/199115992018012901020

A Modified Artificial Bee Colony Algorithm for Global Optimization Problem

Xiao-Fang Liu[1], Pei-Zhong Liu[1], Yan-Ming Luo[2], Jia-Neng Tang[1], De-Tian Huang[1], Yong-Zhao Du[1*]

[1] College of Engineering, Huaqiao University, Quanzhou, Fujian, China
406598306@qq.comp,zliu@hqu.edu.cn, tangjianeng@sina.com, huangdetian@hqu.edu.cn, yongzhaodu@126.com

[2] College of Computer Science and Technology, Huaqiao University, Xiamen, Fujian, China
lym@hqu.edu.cn

Received 7 April 2017; Revised 18 July 2017; Accepted 9 August 2017

Abstract. The artificial bee colony algorithm (ABC) is a kind of stochastic optimization algorithm, which is used to solve optimization problems. In view of the shortcomings of basic ABC with slow convergence and easily falling into local optimum, a modified artificial bee colony algorithm (MABC) is proposed. First, a high dimension chaotic system is employed for the sake of improving the population diversity and enhancing the global search ability of the algorithm when the initial population is produced and scout bee stage. Second, a new search equation is proposed based on the differential evolution (DE) algorithm, which is guided by the optimal solution in the next generation of search direction to improve the local search. Finally, a learning probability (P) method is introduced, corresponding to different value with each particle. Thus, the capacity of the exploration and exploitation of each particle in the population is different, which can solve different types of problems. The performance of proposed approach was examined on well-known 10 benchmark functions, and results are compared with basic ABC and other ABCs. As documented in the experimental results, the proposed approach is very effective in solving benchmark functions, and is successful in terms of solution quality and convergence to global optimum.

Keywords: artificial bee colony algorithm, high dimension chaotic system, learning probability, numerical optimization, search equation

1 Introduction

Inspired by the intelligent behavior of the insects, swarm intelligence(SI) have caused concern in the natural computing field recently. It regards foraging behavior as optimization process, using the idea of “survival of the fittest in nature, the selection mechanism of the survival of the fittest” to make the population evolve continuously. The most famous algorithms include genetic algorithm (GA) [1], particle swarm optimization (PSO) [2], ant colony optimization (ACO) [3], and artificial bee colony (ABC) [4]. In this paper, we focus on the ABC algorithm [4-6], originally developed by Dervis Karaboga in 2005 [6-8]. Compared with the aforementioned algorithms [1-3], the ABC is simple, easy to implement, with fast convergence, strong robustness and obvious advantages in dealing with combined optimization problems [6]. Yet, similar to other SI algorithms, the ABC still has some similar common problems. For example, the convergence rate still has room for further improvement, the population diversity is not sufficient, and the local optimization is easy to be introduced.

In SI algorithms, the two most basic operations are exploration and exploitation closely relating with the algorithm performance. Strong exploration requires plenty of time in locating best solution, leading to slow convergence. Conversely, powerful exploitation normally results in premature convergence. Thus, it

* Corresponding Author

is vital to strike a good balance between exploration and exploitation.

To improve the ABC algorithm, many scholars have aimed at the above problems. Because solution search equation of the ABC tends to exploration as opposed to exploitation. Improved algorithms in the literature include the follows: Zhu and Kwong [7] proposed the GABC based on PSO algorithm, which are both the global optimal and personal optimal information to join the neighborhood search strategy, and can effectively enhance the convergence characteristics of the algorithm. However, "oscillation" phenomenon may occur since the guidance of the two terms may be in opposite directions. This situation causes inefficiency to the search ability and delays the convergence speed. Gao and Liu [8], inspired by differential evolution (DE), proposed two improved solution search equations, namely "ABC/best/1" and "ABC/rand/1". Subsequently, a selective probability P is introduced to control the frequent introduction of "ABC/rand/1" and "ABC/best/1". Gao and Liu [9] proposed an improved solution search equation based on the DE, considering that the bee searches only around the best solution of the previous iteration to improve the exploitation. In the literature [10], the Powell's method is to enhance the local search with large computation. K_1 ran and F_1 nd_1 k [11] developed a directed search strategy by adding direction information for each dimension, leading to slow convergence. Kiran et al. [12] launched the idea of mixing five various search strategies to update solutions via different update rules on employed or onlooker phrase. Thus, it is more robust and effective than that of the basic ABC. Yet, no other measures were taken to improve performance while improving the search equation in the literature [8-12]. In addition, numerous scholars modified ABC by changing its initial method to obtain good initial solutions. For example, Alatas [13] adopted seven different chaotic maps to apply to ABC algorithm and analyzed performance with each improved ABC. Kuang et al. [14] introduced two strategies: Tent chaotic opposition-based learning initialization method and the self-adaptive tent chaotic searching. Sharma and Pant [15] suggested the Halton chaos system to generate initial distribution, which is done using Halton point sets for enhancing the performance of basic ABC. The common flaw of the three literature is the lack of selectivity.

Compared with previous works, the main contributions are proposed to address the exploration and exploitation trade-off issue: a high dimension chaotic initial method is employed for the sake of improving the population diversity. Subsequently, a new solution search equation based on DE is incorporated to ABC to avoid "oscillation" phenomenon" caused by GABC. In addition, the learning probability (P) is proposed to handle different values of the particles, and thus the capacity of the exploration and exploitation of each particle is different. This enables a general solution for different types of problems.

The rest of this paper is organized as follows: Basic ABC is introduced in Section 2, and the proposed MABC is presented in Section 3. In Section 4, the performance of the proposed MABC algorithm is evaluated using the benchmark problems and compared with the state-of-art ABC algorithms. Finally, Section 5 draws the conclusion, research limitation, and future works..

2 Basic Artificial Bee Colony Algorithm

In the ABC algorithm, the bees are divided into three types, namely employed bees, onlooker bees and scout bees. Half of the population consists of employed bees, and the other half includes onlooker ones. The colony of bees moves around in a multidimensional search space, and employed bees are in charge of searching food sources according to the experience of themselves and then telling the onlooker bees about the information; onlooker bees select food sources in line with the materials shared by employed bees. The scout bees work when the times of foraging is beyond a certain limit but no better food source appears. For solving optimization problem, the food source represents a possible solution of optimization problem, and the process of collecting honey (food source) is the process of searching the optimal solution. The merits of the food source depend on the fitness value (or function) of optimization problem. The main steps of the ABC are as follows:

2.1 Initialization

The search scope is corresponding to the nest which is formed by the food source location, and the algorithm begins to generate a set of random food source locations, which corresponds to the search space of the solution. The initial sites are produced via random contribution within the parameter range.

$$x_i^j = x_{\min}^j + rand(0,1)(x_{\max}^j - x_{\min}^j) . \quad \textbf{(1)}$$

where i = 1, 2, ... , SN, j = 1, 2, ... , D; SN denotes the number of food sources (solutions); D is the dimensions of solutions; $x_{\min}^j$ and $x_{\max}^j$ are lower and upper bounds, respectively.

2.2 Employed Bees Phase

For employed bees, the duty is to search the location of the food source around the neighborhood on the basis of their memory, looking for better food source near the food .When an employed bee found a food source, assessing the fitness will be processed. Herein, the following formula is employed to determine the neighbor food source.

$$v_i^j = x_i^j + \varphi_i^j (x_i^j - x_k^j) \ . \quad \textbf{(2)}$$

where x_i^j is the food source which is randomly selected; φ_i^j is the random number in the range [-1, 1], $i \in [1, SN]$, $j \in [1, D]$; k is the location index in between 1 and SN by random selection and not equal to i. When a new food source is produced, its fitness is calculated, and the greedy method is utilized to make a choice between V_i and X_i based on their fitness. For the value exceeding its scope, it is replaced by its upper or lower bounds. Specifically, if $x_i < x_i^{\min}$, then $x_i = x_i^{\min}$; if $x_i > x_i^{\max}$, then $x_i = x_i^{\max}$.

For the minimization problem, the fitness value of the solution can be solved as follows.

$$fit_i = \begin{cases} 1/(1+f_i), & f_i \geq 0; \\ 1 + abs(f_i), & f_i \leq 0. \end{cases} \ . \quad \textbf{(3)}$$

where f_i is a cost value that belongs to V_i.

2.3 Onlooker Bees Phase

For the onlooker bees phase, randomizing the selection is conducted based on the information by sharing their food source information with the bees waiting in the hive on employed bees phase. Consequently, a selection method based on the fitness value is needed, such as a roulette wheel. The probability of being selected can be formulated as follows.

$$p_i = \frac{fit_i}{\sum_{i=1}^{SN} fit_i} \ . \quad \textbf{(4)}$$

When a food source is selected by an onlooker bee, a neighbor food source is generated from Eq.(2), and the fitness value is then calculated. During the period of employed bees, the choice is made and the greedy method is adopted. Thus, more onlooker bees will choose high quality food source and feedback to positive signal.

2.4 Scout Bees Phase

If the employed bees still cannot improve the quality of the solution via a given number of attempts running, the bees will become scout bees, and the solution will be abandoned. After converting the scout bee, they began to randomly search new solutions. For example, if the bee is abandoned, the original owner of the employed bees adopts Eq.(2) to generate stochastic new solutions. Thus, those are randomly generated or in the near future will be abandoned, resulting in negative signals to balance the positive signal.

3 Modified Artificial Bee Colony Algorithm

For the basic ABC, the bees explore the new nectar relying on the large sharing stochastic and non-scheduled individual information, and thus it has a strong global search ability. Yet, the current optimal solution is not fully developed, resulting in poor local search ability. Specifically, no balance is achieved between the global and local search abilities. In addition, when the search is carried out in the later period, the population diversity is reduced. Subsequently, the search efficiency decreases significantly, leading to slow convergence speed. In light of these issues, three strategies are proposed to improve the performance of the algorithm as follows.

3.1 Initial Population

The quality of initial solution has certain influence on the accuracy and convergence rate of the final solution. Consequently, we need to design a method to improve the diversity of population and obtain some high quality initial solutions which may improve the performance of the algorithm.

Chaos is a type of nonlinear phenomena widely existing in nature with the characteristics of randomness, ergodicity and boundness. In a certain range, it can traverse all states according to their own laws without repeating. It has been proved that the chaotic map are effective for searching the entire solution space [13], and most of chaos system applying to bionic intelligence algorithm are one dimensional. Yet, one dimensional chaotic system has the following disadvantages: (1) low space, (2) iterative operation produces single sequence, and (3) weak in uniformity [16]. Consequently, a high dimensional Lorenz chaotic system, generated in each of the three different random chaotic sequence iteration, is adopted. The Lorenz system, with a faster speed, is more complex than that of the one-dimensional chaotic systems, which cannot be predicted with more sequences. Eq.(5) formulates the exploitation operation for improving the population diversity.

$$\begin{cases} \dot{x} = \delta(y - x) \\ \dot{y} = \gamma x - y - xz \\ \dot{z} = xy - bz \end{cases} . \tag{5}$$

where $x(0), y(0), z(0)$ denote initial value, and δ, γ, b are parameters of Lorenz system. The system is chaotic for $\delta = 10$, $\beta = 8/3, \gamma > 24.74$. From Fig. 1 and Fig. 2, it is obvious that Lorenz system is more uniform than that of the Logistic system (one dimension chaos system). Subsequently, we obtain a set of number from three different random chaotic sequence, recorded as φ. By putting it into the Eq.(1), a new equation can be obtained as:

$$x_i^j = x_{\min}^j + \varphi(x_{\max}^j - x_{\min}^j) \ . \tag{6}$$

where i = 1, 2, ... , SN, j = 1, 2, ... , D; SN denotes the number of food sources (solutions); D is the dimension of solutions; φ is a series of data obtained by Lorenz system. $x_{\min}^j$ and $x_{\max}^j$ are lower and upper bounds, respectively. φ, gained by Lorenz system, can avoid the blindness caused by random parameters $rand(0,1)$ to a great extent.

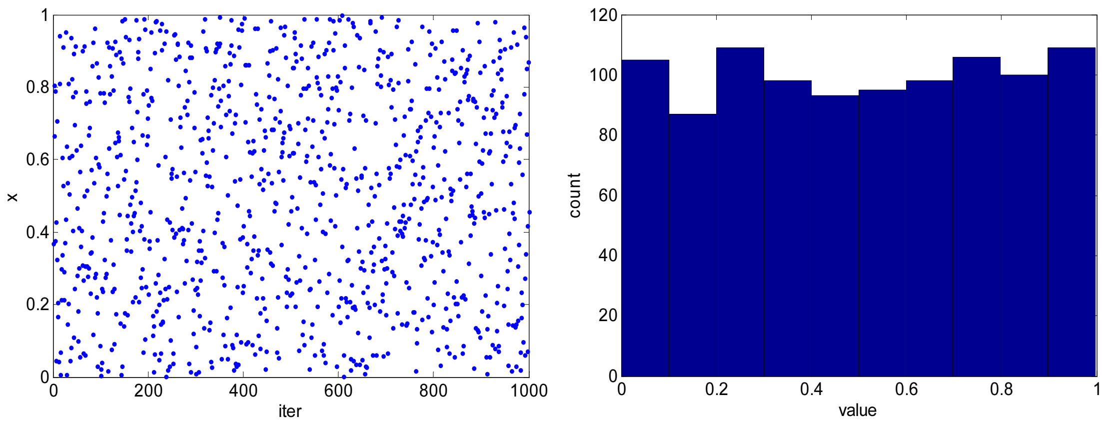

Fig. 1. Distribution analysis of Lorenz system

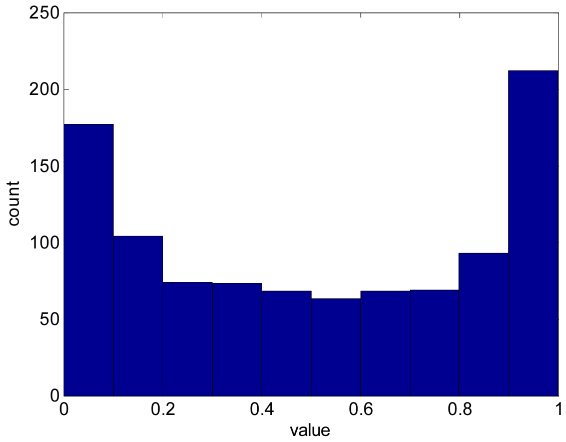

Fig. 2. Distribution analysis of Logistic system

3.2 New Search Mechanism

It is well known that the optimization performance of the algorithm is decreased due to a very low balance between global search and local search. In Eq.(2) of ABC algorithm, the coefficient φ_i^j is randomly obtained in between [-1,1], and the parameters j and k are random numbers in [1,D]. The search results are too stochastic to maintain good exploitation. In a nutshell, the basic ABC specializes in exploration, but presents weakness in exploitation. Thus, a new search strategy which can advance the property of exploitation is urgently needed.

Inspired by DE [17-18], we propose a new search strategy as follows:

$$v_i^j = x_{best}^j + \varphi_i^j (x_{best}^j - x_r^j) \ . \tag{7}$$

where the index r is a random integer which belongs to {1, 2, ... , SN}, and varies with i. The implication of i and j is as in the above case of Eq.(2). The coefficient φ_i^j is chosen from the range of [-1, 1]. The variable x_r^j is the j^{th} dimension of r^{th} particle. The variable x_{best}^j refers to the j^{th} dimension of best particle, remembered by the algorithm in the current population. Thus, the old solution will be guided by the current best particle, which can enhance the ability of exploitation and make solution more accurate.

3.3 Learning Probability P

To improve the performance of the PSO algorithm, the former work [19] concluded that different results were calculated by adopting different learning probability values for the same optimal function when the same P was used for all the particles in the population. On the non-rotating problem, the smaller value can achieve a better effect. Yet, for the rotation problem, the best effect of the probability value is also varied in different problems. For the simple unimodal problem, the results obtained from different learning probability values are close, however, for the complex multimodal problem, the performance of the algorithm tends to be sensitive to the value of probability. To deal with this situation, the former work [19] suggests that each particle corresponds to a different value. Consequently, the capacity of the exploration and exploitation of each particle in the population is different, which can solve different types of problems. As a result, a new expression is developed to set a value for each particle empirically, and the expression as follows is used in the ABC.

$$P_i = m + n \times \frac{\exp(10(i-1)/(SN-1))-1}{\exp(10)-1} \ . \tag{8}$$

where the index i represents the i^{th} particle, and SN refers to the number of employed bees, with half of the population size. In Fig. 3, we set up an example when the number of bees is 150 and m = 0.01, n = 0.99. Each particle's value is differentiated.

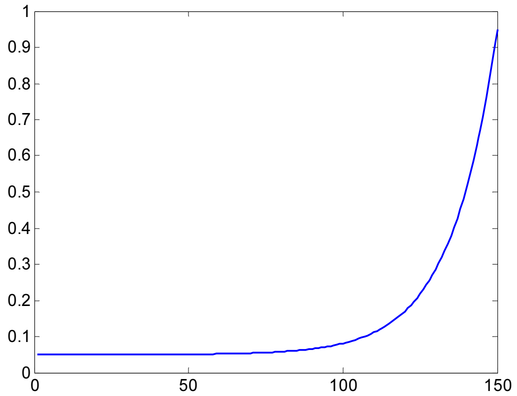

Fig. 3. Values of the each particle

3.4 The Modified Algorithm

Through the above analysis, it is concluded that the above strategies can improve the performance of the ABC algorithm. Fig. 4 shows the flowchart of the modified ABC.

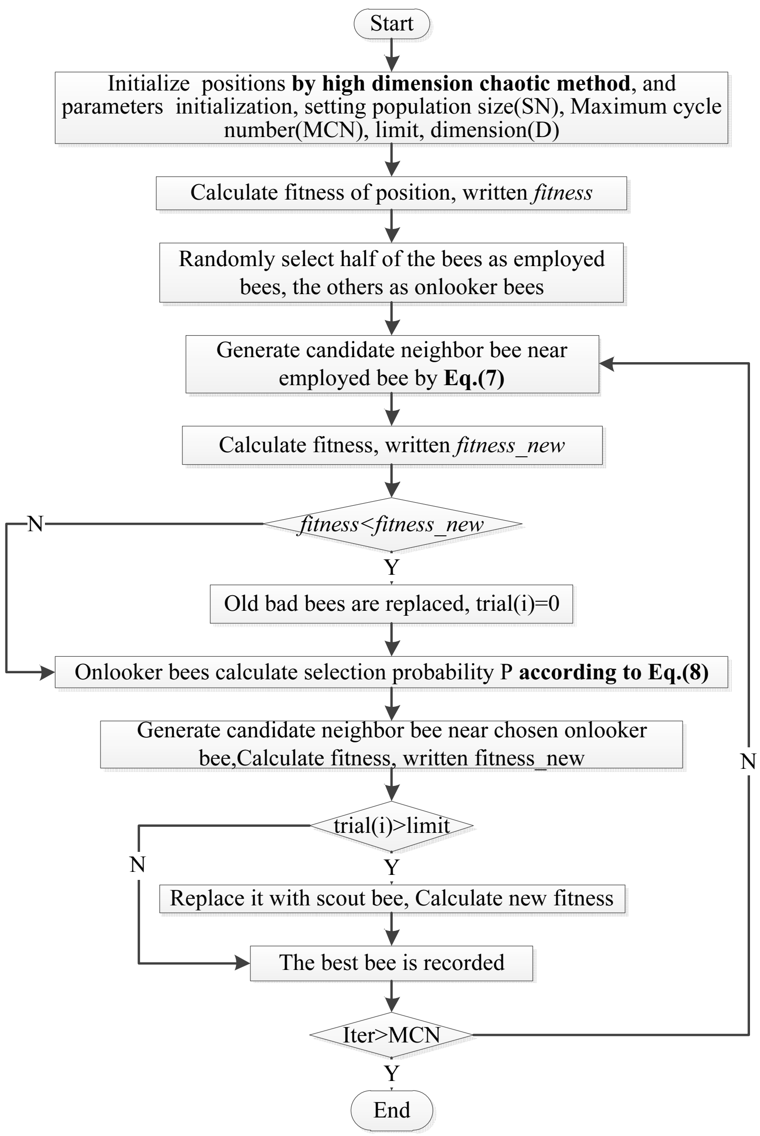

Fig. 4. Flowchart of the modified ABC

4 Experiments

4.1 Benchmark Functions and Parameter Settings

To verify the performance of the modified artificial bee colony algorithm (MABC), 10 classical benchmark functions are adopted to carry out the experiment. The name of the benchmark function, the theoretical optimal value and the search space are given in Table 1. f_1 and f_2 are unimodal functions; f_3 - f_8 are multimodal functions; f_9 is a noisy function, and f_{10} is a complex unimodal function with an offset. The specific definitions of functions can be referred to [20]. The parameter values are organized in Table 2.

Table 1. Benchmark function

Number	Function Name	Min	Search Range
f_1	Sphere function	0	$[-100, 100]^D$
f_2	Ellipitic function	0	$[-100, 100]^D$
f_3	Rastrigin's function	0	$[-5.12, 5.12]^D$
f_4	Non-continuous Rastrigin's function	0	$[-5.12, 5.12]^D$
f_5	Griewanks's function	0	$[-600, 600]^D$
f_6	Ackley's function	0	$[-32.768, 32.768]^D$
f_7	Weierstrass function	0	$[-0.5, 0.5]^D$
f_8	Schwefel's function	0	$[-500, 500]^D$
f_9	Sphere_noise function	0	$[-100, 100]^D$
f_{10}	Shifted sphere function	f_{bias}	$[-100, 100]^D$

Table 2. Parameter settings

Variable Name	Value
Population size	150
SN	75
limit	200
D	10, 30 & 60
MCN	1000

4.2 Effect of The Parameter P

In Eq.(8), two parameters, m and n, are considered. The effects of these two parameters on the MABC algorithm are evaluated by a multimodal function (rastrigin) when D equals to 10. The mean of error is drawn in Fig. 5. It is clear that the parameter settings of m and n can lead to different performance. By adjusting the two parameters and other functions (take rastrigin as an example, other effect please see the experiment results in subsection 4.3), the optimal values of m and n are 0.05 and 0.95, respectively.

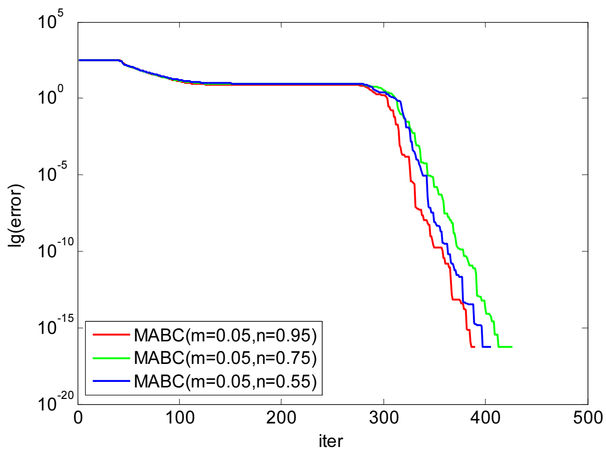

Fig. 5. Errors under various m & n setting (rastrigin)

4.3 Comparative Study

In the experiment, the MABC is compared with ABC [6] and GABC [7]. Each algorithm runs independently on each function for 30 times, and the best value, the median value, the worst value, mean and the standard deviation of errors are shown in Tables 4-6, corresponding to 10, 30 and 60 dimensions. The mean reflects the algorithm's accuracy on solving different problems within the maximum number of evaluations; the standard deviation reflects the stability of the algorithm; the best value, the median value and the worst value associate to quality of the solution from different perspectives. For more intuitive analysis, we plot the evolution curves of each algorithm for each test function in Fig. 6 to Fig. 8.

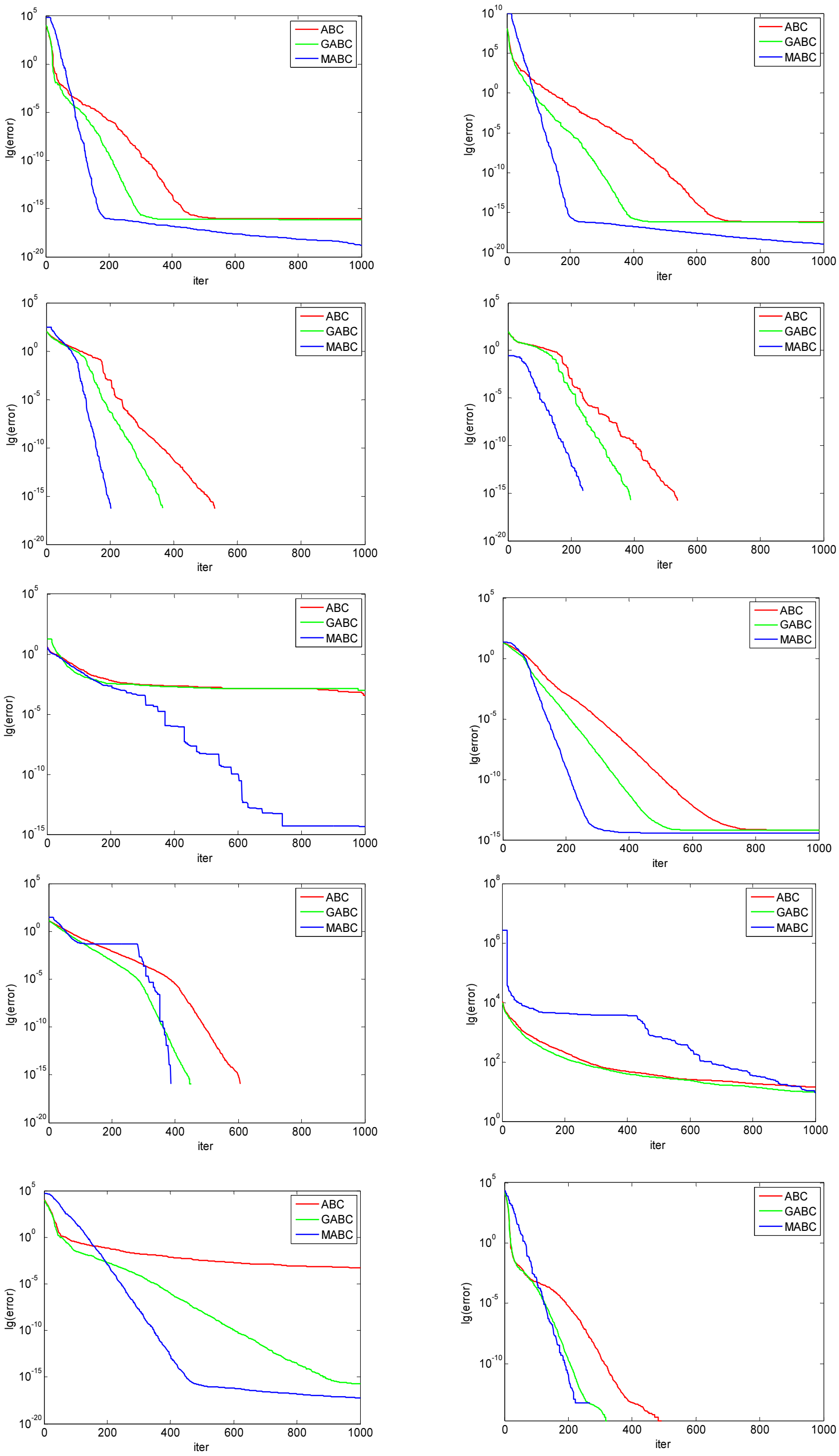

Fig. 6. Performance comparison with dimension of 10

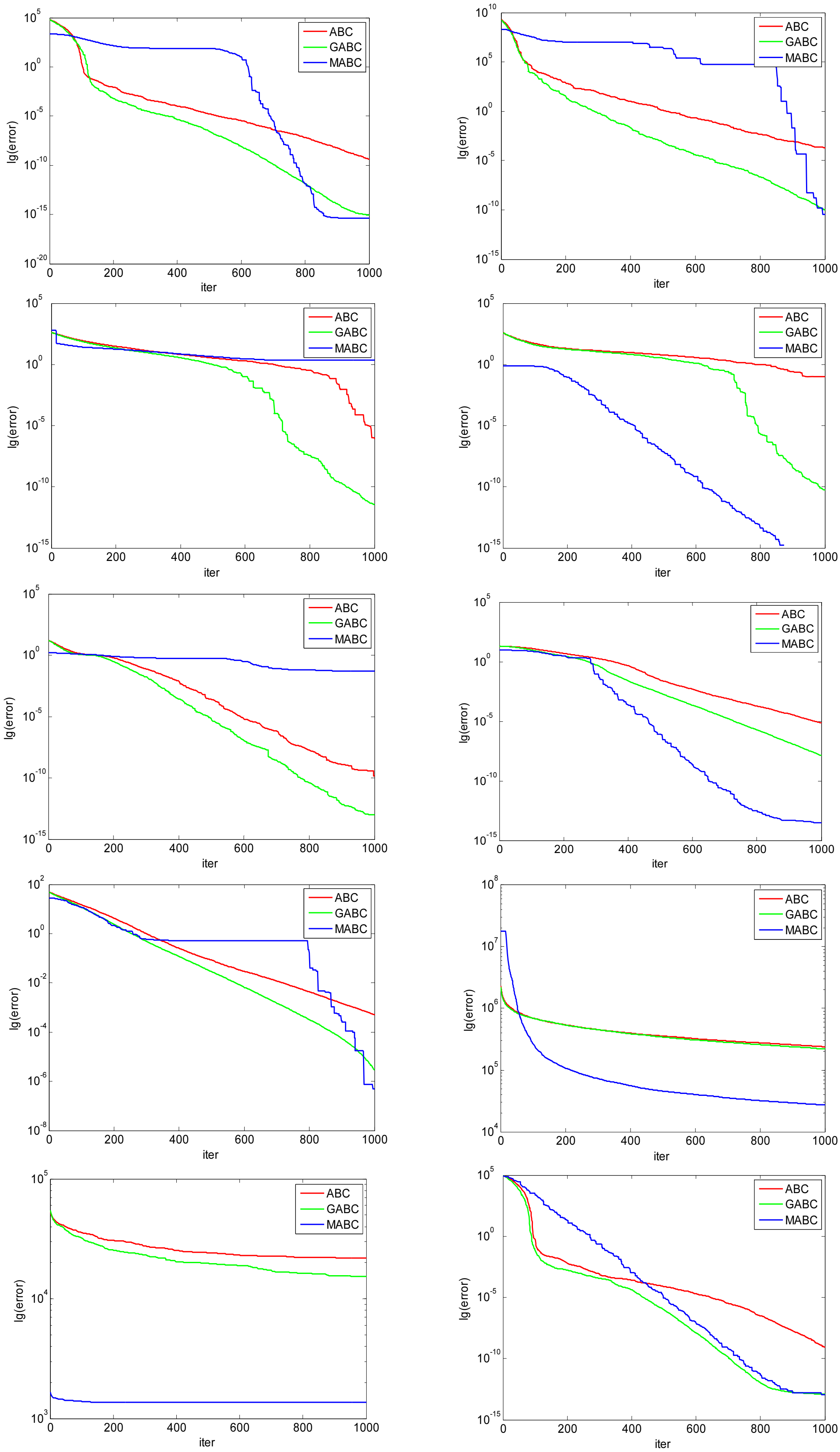

Fig. 7. Performance comparison with dimension of 30

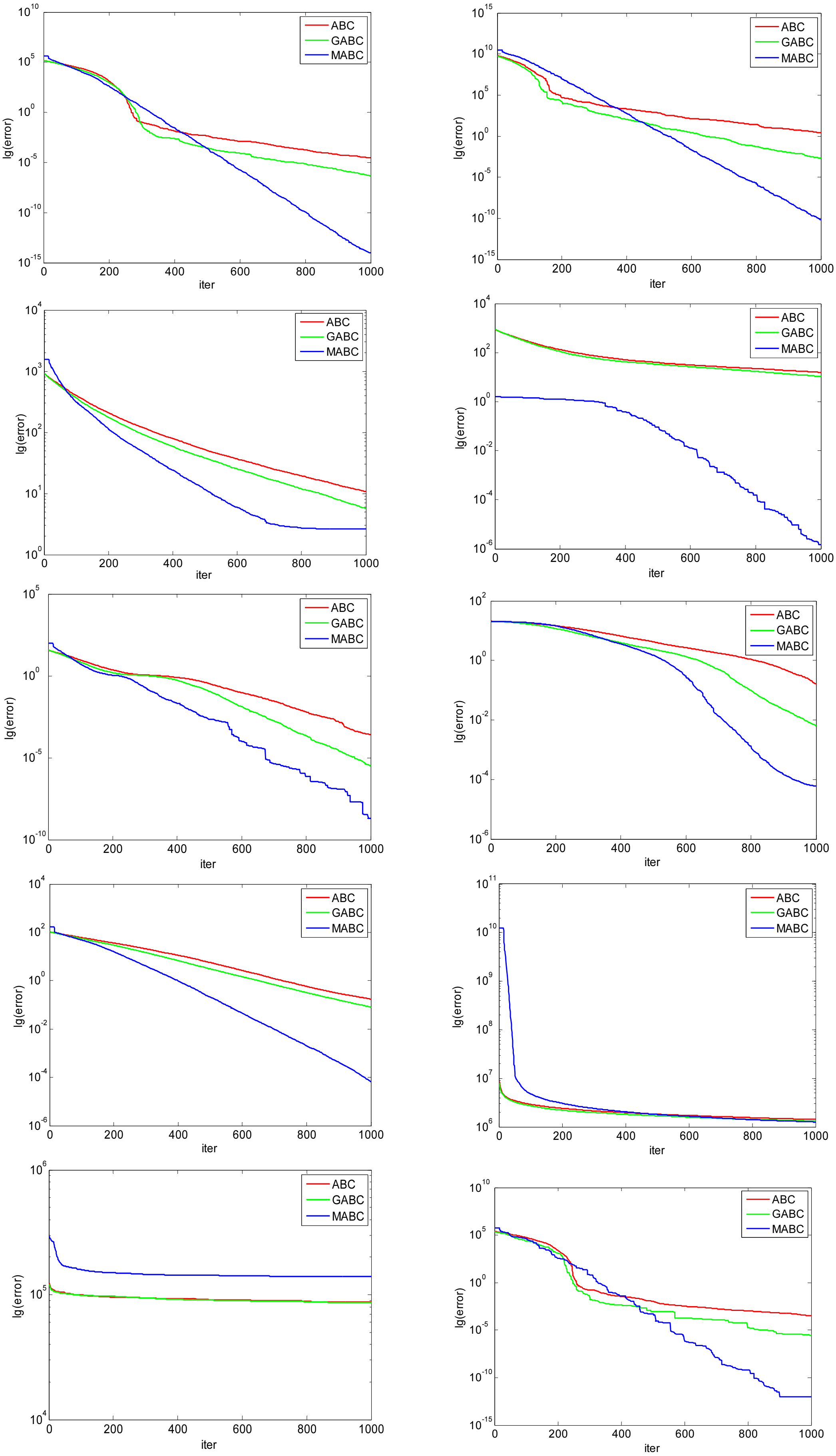

Fig. 8. Performance comparison with dimension of 60

As shown in Table 3(10-D), the accuracy of MABC is superior to that of the ABC and GABC on f_1, f_2, f_5, f_6, f_9, and the stability of proposed algorithm is also higher than the contrast algorithm. The three algorithms can achieve the theoretical optimal value on f_3, f_4, f_7, f_{10}. Although the methods cannot achieve the ideal optimal result on f_8, the result obtained by proposed algorithm is still the best among the three. For clarity, the results of the best algorithms are marked in boldface. Besides the accuracy and stability, the convergence curve are another essential measure of the performance shown in Fig. 6. With a reasonable agreement to the fact that MABC is slower than the contrast algorithm at early stage and faster than them at this late stage on f_1 - f_3, f_8, f_{10}. This is because MABC finds a balance between exploration and exploitation to accelerate the convergence speed. And MABC is always faster on f_4 - f_7. For f_9, MABC is worse than ABC and GABC because of noise characteristics.

Table 3. Results of 10-*D* Test Problems

Function Number	Algorithms	Best	Mid	Worst	Mean	SD
	ABC	4.856e-17	7.485e-17	1.940e-16	8.797e-17	4.129e-17
f_1	GABC	4.796e-17	6.546e-17	8.845e-17	6.616e-17	1.123e-17
	MABC	**1.446e-20**	**8.756e-20**	**4.999e-19**	**1.505e-19**	**1.678e-19**
	ABC	3.648e-17	7.137e-17	1.005e-16	7.188e-17	1.230e-17
f_2	GABC	2.202e-17	6.413e-17	8.786e-17	5.904e-17	1.422e-17
	MABC	**6.883e-21**	**6.040e-20**	**4.975e-19**	**1.124e-19**	**1.298e-19**
	ABC	**0**	**0**	**0**	**0**	**0**
f_3	GABC	**0**	**0**	**0**	**0**	**0**
	MABC	**0**	**0**	**0**	**0**	**0**
	ABC	**0**	**0**	**0**	**0**	**0**
f_4	GABC	**0**	**0**	**0**	**0**	**0**
	MABC	**0**	**0**	**0**	**0**	**0**
	ABC	0	4.996e-16	0.008451	0.000348	0.001557
f_5	GABC	0	0	0.012321	0.000994	0.003074
	MABC	**0**	**1.110e-16**	**8.659e-14**	**4.278e-15**	**1.591e-14**
	ABC	3.552e-15	7.105e-15	7.105e-15	6.631e-15	1.228e-15
f_6	GABC	3.552e-15	7.105e-15	7.105e-15	6.394e-15	1.445e-15
	MABC	**3.552e-15**	**3.552e-15**	**7.105e-15**	**3.671e-15**	**6.486e-16**
	ABC	**0**	**0**	**0**	**0**	**0**
f_7	GABC	**0**	**0**	**0**	**0**	**0**
	MABC	**0**	**0**	**0**	**0**	**0**
	ABC	3.33769	13.1584	43.7609	14.011	8.69506
f_8	GABC	3.06603	9.03397	28.5006	9.74873	4.88215
	MABC	**2.39604**	**6.09246**	**23.5503**	**7.37528**	**3.67201**
	ABC	6.289e-06	0.00027	0.00225	0.00054	0.00062
f_9	GABC	5.731e-17	1.957e-16	4.525e-16	1.844e-16	8.677e-17
	MABC	**2.681e-19**	**3.822e-18**	**3.275e-17**	**5.940e-18**	**6.870e-18**
	ABC	**-450**	**-450**	**-450**	**-450**	**0**
f_{10}	GABC	**-450**	**-450**	**-450**	**-450**	**0**
	MABC	**-450**	**-450**	**-450**	**-450**	**0**

In Table 4(30-D), the proposed algorithm can achieve promising results and the performance is superior to that of the contrast algorithm on f_1, f_2, f_4, f_6, f_7, f_{10}. On f_3 and f_5, the average performance of MABC is inferior to that of the ABC and GABC, but only MABC can achieve the optimal value, the other two algorithms have certain errors, suggesting that MABC has the ability to achieve optimal solution. In f_8 and f_9, the three algorithms obtain poor results. In Fig. 7, it is clear that the convergence speed or the accuracy can be improved with the proposed MABC on the 8 functions, except for f_3 and f_5.

In Table 5(60-D), MABC can achieve better performance on functions f_1, f_2, f_4 - f_7, f_{10} comparing to ABC and GABC. In general, the average performance on f_3 is poor, but MABC is the closest approach to the theoretical optimum from the aspects of optimal value and median. In Fig. 8, convergence speed or accuracy has improved on the 8 functions, except for f_6 and f_8. The analysis in detail is similar to Fig. 6.

According to the above experimental results, the proposed algorithm outperforms ABC and GABC in most cases. Thus, it can be a very competitive candidate to handle practical applications.

Table 4. Results of 30-*D* Test Problems

Function Number	Algorithms	Best	Mid	Worst	Mean	SD
f_1	ABC	5.982e-11	2.937e-10	1.263e-09	4.237e-10	3.342e-10
	GABC	5.106e-16	7.655e-16	1.136e-15	8.339e-16	1.504e-16
	MABC	**1.799e-16**	**3.956e-16**	**5.276e-16**	**3.898e-16**	**9.439e-17**
f_2	ABC	1.372e-05	0.000109	0.000735	0.000183	0.000190
	GABC	1.014e-11	5.906e-11	4.168e-10	1.054e-10	1.070e-10
	MABC	**2.225e-16**	**4.429e-16**	**1.060e-09**	**3.536e-11**	**1.937e-10**
f_3	ABC	7.580e-11	1.635e-09	2.142e-05	8.343e-07	3.928e-06
	GABC	**1.154e-13**	**1.023e-12**	**4.498e-11**	**3.672e-12**	**8.766e-12**
	MABC	0	1.49244	5.96975	2.15574	2.10793
f_4	ABC	6.763e-10	4.410e-08	1.00001	0.100005	0.316229
	GABC	**1.390e-11**	**3.651e-11**	**1.448e-10**	**5.407e-11**	**4.106e-11**
	MABC	0	0	0	0	0
f_5	ABC	3.672e-13	7.196e-12	2.885e-09	1.483e-10	5.501e-10
	GABC	**4.440e-16**	**3.996e-15**	**1.440e-12**	**9.513e-14**	**3.062e-13**
	MABC	0	0.052190	0.124204	0.051713	0.039992
f_6	ABC	2.228e-06	6.917e-06	1.271e-05	7.339e-06	2.305e-06
	GABC	7.878e-09	1.357e-08	2.300e-08	1.344e-08	4.074e-09
	MABC	**3.197e-14**	**7.411e-12**	**4.297e-08**	**0.916e-11**	**1.472e-010**
f_7	ABC	0.0003141	0.000497	0.000658	0.000501	9.354e-05
	GABC	1.227e-06	2.565e-06	5.688e-06	2.968e-06	1.310e-06
	MABC	**0**	**1.841e-07**	**3.441e-05**	**3.564e-09**	**7.602e-08**
f_8	ABC	5161.72	9238.93	12639.1	8992.6	1961.37
	GABC	5255.71	9525.37	13974.1	9200.3	1946.95
	MABC	**409.68**	**1077.8**	**1785.12**	**1053.66**	**324.114**
f_9	ABC	12837.5	19323	31657.2	21807.1	5111.51
	GABC	5700.85	14705.7	26814.9	15347.4	4952.84
	MABC	**1275.13**	**1365.13**	**1445.58**	**1365.17**	**47.9801**
f_{10}	ABC	**-450**	**-450**	**-450**	**-450**	**0**
	GABC	**-450**	**-450**	**-450**	**-450**	**0**
	MABC	**-450**	**-450**	**-450**	**-450**	**0**

Table 5. Results of 60-*D* Test Problems

Function Number	Algorithms	Best	Mid	Worst	Mean	SD
f_1	ABC	7.546e-06	3.424e-05	6.080e-05	3.158e-05	1.727e-05
	GABC	2.25e-07	3.367e-07	2.815e-06	8.055e-07	8.530e-07
	MABC	**3.142e-15**	**9.058e-15**	**3.517e-14**	**1.294e-14**	**1.122e-14**
f_2	ABC	0.332474	2.0378	10.1978	2.71525	2.47424
	GABC	0.000230	0.001991	0.007471	0.002142	0.001763
	MABC	**2.786e-12**	**3.043e-11**	**3.797e-10**	**6.784e-11**	**9.385e-11**
f_3	ABC	5.23149	10.7761	15.8448	10.7121	2.69938
	GABC	2.23034	6.03793	9.74992	5.73108	1.80661
	MABC	**1.394e-11**	**3.163e-09**	**63.677**	**2.65321**	**11.8859**
f_4	ABC	10.5085	15.7926	19.624	15.7279	2.81814
	GABC	7.02719	10.8874	13.338	10.7091	1.48200
	MABC	**6.849e-07**	**1.439e-06**	**2.687e-06**	**1.565e-06**	**5.070e-07**
f_5	ABC	3.333e-05	0.000186	0.000922	0.000257	0.000221
	GABC	3.842e-07	1.721e-06	2.025e-05	3.303e-06	4.085e-06
	MABC	**2.331e-15**	**4.767e-12**	**4.602e-08**	**1.932e-09**	**8.428e-09**
f_6	ABC	0.021912	0.112670	0.501245	0.164958	0.141153
	GABC	0.003670	0.006453	0.012301	0.006521	0.001661
	MABC	**7.057e-07**	**5.745e-06**	**0.001549**	**5.933e-05**	**0.000281**
f_7	ABC	0.108236	0.163293	0.282263	0.173832	0.044598
	GABC	0.062701	0.081004	0.106711	0.080760	0.009182
	MABC	**2.477e-05**	**5.982e-05**	**0.000161**	**6.747e-05**	**3.078e-05**

Table 5. Results of 60-*D* Test Problems (continu)

Function Number	Algorithms	Best	Mid	Worst	Mean	SD
f_8	ABC	33078.1	57251.2	68381.1	56378.9	6885.68
	GABC	39895	56587.5	68127.1	56391.9	7337.31
	MABC	**35878**	**50387.7**	**70599.4**	**49808.3**	**8379.15**
f_9	ABC	86744.6	97146.9	87530.1	3966.7	66316.7
	GABC	66316.7	86756.7	94641.1	86081.2	6435.65
	MABC	**125593**	**140153**	**151502**	**139519**	**7315.48**
f_{10}	ABC	-450	-450	-449.99	-450	0.000210
	GABC	**-450**	**-450**	**-450**	**-450**	**0**
	MABC	**-450**	**-450**	**-450**	**-450**	**0**

5 Conclusions

To improve the accuracy and convergence rate of the ABC algorithm, a modified artificial bee colony algorithm (MABC) is proposed. The original ABC algorithm is improved in threefold: (1) The Lorenz system (a kind of three-dimensional chaotic system) is utilized on the population initialization and scout bee stage to obtain the initial solution with high quality; (2) a novel search strategy is proposed based on DE, which is used to improve the local search ability, and (3) the learning probability P is introduced to address different types of problems. The experimental results on 10 benchmark functions show that the modified algorithm can reduce error or speed up the convergence rate compared to ABC and GABC in most cases. However, it is worth considering that the proposed algorithm has not yet received acceptable optimization results for the noisy function. In the future, the endeavor can be put to handle the complex multimodal function and noisy function.

Acknowledgements

The authors would like to thank the Fundamental Research Funds for the Central Universities for support (JB-ZR1202).

References

[1] M. Mitchell, An introduction to genetic algorithms, An Introduction to Genetic Algorithms 24(4-5)(1996) 325-336.

[2] H. Xu, Y. Yang, L. Mao, Study and improvement on particle swarm algorithm, Journal of Computers 8(4)(2013) 937-942.

[3] V. Maniezzo, A. Carbonaro, Ant Colony Optimization: An Overview, in: C. Ribeiro, P. Hansen (Eds.), Essays and Survey in Metaheuristics, Springer, New York, NY, 2001, pp. 469-492.

[4] D. Karaboga, An idea based on honey bee swarm fornumerical optimization, Technical Report - TR06, 2005.

[5] B. Akay, D. Karaboga, A modified artificial bee colony algorithm for real-parameter optimization, Information Sciences 192(6)(2012) 120-142.

[6] D. Karaboga, B. Basturk, On the performance of artificial bee colony (abc) algorithm, Applied Soft Computing 8(1)(2008) 687-697.

[7] G. Zhu, S. Kwong, Gbest-guided artificial bee colony algorithm for numerical function optimization, Applied Mathematics & Computation 217(7)(2010) 3166-3173.

[8] W. Gao, S. Liu, Improved artificial bee colony algorithm for global optimization, Information Processing Letters 111(17)(2011) 871-882.

[9] W.-F. Gao, S.-Y. Liu, A modified artificial bee colony algorithm, Computers & Operations Research 39(3)(2012) 687-697.

[10] W.-F. Gao, S.-Y. Liu, L.-L. Huang, A novel artificial bee colony algorithm with powell's method, Applied Soft Computing

13(9)(2013) 3763-3775.

[11] M.S. Kiran, O. Findik, A directed artificial bee colony algorithm, Applied Soft Computing 26(2015) 454-462.

[12] M.S. Kiran, H. Hakli, M. Gunduz, H. Uguz, Artificial bee colony algorithm with variable search strategy for continuous optimization, Information Sciences 300(2015) 140-157.

[13] B. Alatas, Chaotic bee colony algorithms for global numerical optimization, Expert Systems with Applications 37(8)(2010) 5682-5687.

[14] F. Kuang, Z. Jin, W. Xu, S. Zhang, A novel chaotic artificial bee colony algorithm based on tent map, in: Proc. IEEE Congress on Evolutionary Computation, 2014.

[15] T. K. Sharma, M. Pant, Halton based initial distribution in artificial bee colony algorithm and its application in software effort estimation, in: Proc. International Conference on Bio-Inspired Computing: Theories & Applications, 2011.

[16] M. Ahmad, B. Alam, O. Farooq, Chaos based mixed keystream generation for voice data encryption, International Journal on Cryptography & Information Security 2(1)(2012) 36-45.

[17] K. Price, R.M. Storn, J.A. Lampinen, Differential Evolution: A Practical Approach to Global Optimization (natural computing series), Springer, New York, NY, 2005.

[18] R. Storn, K. Price, Differential evolution c a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11(4)(1997) 341-359.

[19] J.-J. Liang, A.-K. Qin, P.N. Suganthan, S. Baskar, Evaluation of comprehensive learning particle swarm optimizer, Lecture Notes in Computer Science 3316(2004) 230-235.

[20] J.-J. Liang, T.P. Runarsson, E. Mezura-Montes, M. Clerc, P.N. Suganthan, C.A.C. Coello, K. Deb, Problem definitions and evaluation criteria for the cec 2006 special session on constrained real-parameter optimization, Technical Report, 2006.