The Artificial Bee Colony Algorithm Improved with Simplex Method

Xufang Zhao[1,*], Ximing Liang[1], Long Wen[2]

[1] School of Science, Beijing University of Civil Engineering and Architecture, Beijing 102600,China.

[2] Key Laboratory of Economics System Simulation, Guizhou University of Finance & Economics, Guiyang 550025, PR China.

[*] Corresponding author. Tel.:18811169212; email: 81426904@qq.com
Manuscript submitted March 16, 2018; accepted June 5, 2018.
doi: 10.17706/jsw.13.6.350-359

Abstract: Artificial bee colony (ABC) is one of very effective intelligence optimization algorithms, which has good performance in solving global optimization problems(GOPs). However, the ABC algorithm performs relatively poorly in complex GOPs for some weaknesses, such as slow convergence and poor exploitation. An improved artificial bee colony algorithm named SABC algorithm is proposed, which combines the ABC algorithm and the simplex method for GOPs. In SABC algorithm, the simplex method is employed to update the food sources which cannot be improved after limit updates in the ABC algorithm. The proposed SABC algorithm takes not only the advantages of directness, simpleness and small calculation amount of simplex method, but also the strong global search ability of ABC algorithm. The accurate and rapid local research ability of the simplex method can guarantee to avoid some blind meaningless iteration in the ABC algorithm. The validity has been verified by the numerical experiments on 9 benchmark GOPs. The numerical result show that the proposed SABC algorithm has faster convergence speed and better convergence accuracy compared to the basic ABC algorithm.

Keywords: Artificial bee colony algorithm, simplex method, numerical experiments.

1. Introduction

Artificial bee colony algorithm is a swarm intelligence optimization algorithm proposed by Karaboga [1]-[3]. It has good ability for global optimization and has some advantages such as simple operation, easy realization, less control parameters, simple calculation, wide application, etc. Like other swarm intelligence algorithms, however, the artificial bee colony algorithm has the disadvantages of low convergence precision and slow convergence speed in solving the problems of function optimization [4]. Therefore, scholars have conducted in-depth research and proposed some improved artificial bee colony algorithms, which have achieved good numerical results [5], [6]. Wei, *et. al*. [7] substituted the traditional roulette model with pheromone and sensitivity model in the free search algorithm, and used OBL strategy to produce new food source to replace the worst food source in each iteration in the onlooker bee stage, which improves the convergence accuracy and convergence rate of ABC algorithm. To accelerate the convergence speed of ABC algorithm, Yang, et. al.[8] proposed a strategy of adjustable pressure sorting and adjusted the onlooker bee stage to a dynamic strategy set by comparing the qualities of food sources in each evolution. Du, et. al.[9] used the chaos operator in the local search strategy on the current optimal solution, and endowed the onlooker bee with the bacterial chemotaxis behavior to improve the local search ability of the ABC algorithm. Banhamsakun, *et. al*. [10] proposed an adaptive greedy search method which was used in the onlooker bee stage and made full use of successful search experience to reduce the blindness of the ABC algorithm. It is noted that these improved ABC algorithms can enhance the convergence performance of the ABC algorithm in a large part. However, no one algorithm can get all ideal global minimum point for all optimization problems. Therefore, it is of high research value to modify the ABC algorithm to improve its optimization effect and expand its adaptation range.

An improved artificial bee colony algorithm (SABC) is proposed, where the simplex method is employed to improve the convergence performance of ABC algorithm. The SABC algorithm makes full use of the fast local search ability of the simplex method and the global optimization ability of ABC algorithm. The faster convergence speed and higher convergence precision of SABC algorithm has been verified by the numerical experiments on 9 benchmark GOPs.

2. Basic Artificial bee Colony Algorithm and Simplex Method

The unconstrained continuous optimization problems can be expressed as follows.

$$\min f(X), X \in R^n \tag{1}$$

If $X^* \in R^n$ satisfies that:

$$f(X^*) \le f(X), \forall X \in R^n \tag{2}$$

X^* is called the global minimum point of $f(X)$ in the whole space R^n.

2.1. Basic Artificial Bee Colony Algorithm

The ABC algorithm is a swarm intelligence algorithm based on the intelligent forge behavior of honey bee colony, which can achieve the search goal through the cooperation and information communication among individuals. The unique mechanism of ABC algorithm is role transformation, by which the employed bee, onlooker bee and scout bee collaborate on seeking for high quality food source. In the process of searching for optimization in ABC algorithm, the roles of the three kinds of bees are different: the employed bees are used to maintain excellent solutions; the onlooker bees are used to improve convergence speed; the scout bees are used to get out of local optimal solutions. When ABC algorithm is employed to solve the optimization problem, the location of the food source are abstracted into the points in the solution space,which represent the potential solutions of the problem.The swarm is made up of the employed bees and scout bees. The amounts of each kind of bees are equal and equal to the amount of honey.

The initial population contains NP food source, each of which is represented by an n dimensional vector $X_i^t = [x_{i1}^t, x_{i2}^t, ..., x_{in}^t]$, where $x_{id}^t \in (L_d, U_d)$, $d = 1,2,\cdots,n$ and t is the number of iterations.The vectors $L = [L_1, ..., L_n]$ and $U = [U_1, ..., U_n]$ represent the lower bound and upper bound of the search space, respectively.

2.1.1. Initialization stage

In the initialization stage $(t = 0)$, set $trail_i = 0$ as a count. The position of each food source is generated in the search space according to Eq.(3),

$$x_{id}^0 = x_d^a + rand(0,1)(x_d^b - x_d^a) \tag{3}$$

where $i = 1,2,\ldots,NP$, $d = 1,2,\ldots,n$, x_d^a and x_d^b present the lower bound and upper bound of the i th dimension, respectively. The fitness value of each food source is calculated by Eq (4),

$$fit_i = \begin{cases} \dfrac{1}{1+f_i}, & f_i \geq 0 \\ 1+abs(f_i), & f_i < 0 \end{cases} \tag{4}$$

where f_i and fit_i represent the objective function value of the point $X_i^0 = [x_1^0, x_2^0, ..., x_n^0]$ and fitness value of the i th food source, respectively.

2.1.2. Employed bee stage

In the search stage, each employed bee makes the neighborhood search around the current food source and a new candidate food source position is produced by Eq (5),(6),

$$v_{id} = x_{id}^0 + \varphi(x_{id}^0 - x_{jd}^0) \tag{5}$$

$$v_{im} = x_{im}^0 \tag{6}$$

where d is randomly chosen from $\{1, \ldots, n\}$,which means the employed bee select one dimension randomly to search, j is randomly picked up from $\{1, 2, \cdots, NP\}$ and $j \neq i$, φ is a random real number in the range[-1,1], which determines the amplitude of disturbance, m is one dimension of vector and $m \neq d$. If the new solution V_i is better than the original solution X_i^{t-1}, according to the fitness value, X_i^{t-1} is replaced by V_i and the counter $trail_i$ is set to 0. Otherwise, X_i^{t-1} is kept to the next generation and the counter increases by 1. The process is as follows,

$$Y_i = \begin{cases} V_i, & fit(V_i) > fit(X_i^{t-1}) \\ X_i^{t-1}, & fit(V_i) \leq fit(X_i^{t-1}) \end{cases} \tag{7}$$

$$trail_i = \begin{cases} 0, & fit(V_i) > fit(X_i^{t-1}) \\ trail_i + 1, & fit(V_i) \leq fit(X_i^{t-1}) \end{cases} \tag{8}$$

2.1.3. Onlooker bee stage

All the employed bees complete a search and fly back to the exchange area to share their information. Onlooker bees exploit a selected food source, whose position information is shared by the employed bees. The probability of the i th food source position selected by onlooker bees is calculated as follows,

$$p_i = fit(Y_i) / \sum_{i=1}^{NP} fit_i(Y_i) \tag{9}$$

A random number r in [0,1] is produced for selection. If $p_i > r$, the onlooker bee produces a new solution near the selected food source, according to Eq (10),(11),

$$v_{id} = y_{id} + \varphi(y_{id} - y_{jd}) \tag{10}$$

$$v_{im} = y_{im} \tag{11}$$

Use the same greedy choice method to determine the new food source as that in employed bee stage. That is, if the new food source V_i is better than the original one Y_i, according to the fitness value, Y_i is replaced by V_i and the counter $trail_i$ is set to 0. Otherwise, Y_i is kept to the next generation and the counter increases by 1.

The process is as follows,

$$Z_i = \begin{cases} V_i, & fit(V_i) > fit(Y_i) \\ Y_i, & fit(V_i) \le fit(Y_i) \end{cases} \tag{12}$$

$$trail_i = \begin{cases} 0, & fit(V_i) > fit(Y_i) \\ trail_i + 1, & fit(V_i) \le fit(Y_i) \end{cases} \tag{13}$$

2.1.4. Scout bee stage

In the scout bee stage, if a food source cannot be improved during a predefined $limit$ times of iteration, the current food source is abandoned and the onlooker bee change role into a scout bee to search for a new food source to replace X_i^t, as show in Eq (14),

$$X_i^t = \begin{cases} L_d + rand(0,1)(U_d - L_d), & trail_i \ge limit \\ Z_i, & trail_i < limit \end{cases} \tag{14}$$

The above is the three core parts in ABC algorithm, that is, the employed bee searches for food source, the onlooker bee choices food source to search by roulette wheel select scheme, and the scout bee surveys in search space randomly.

2.2. Simplex Method

Simplex method [11]-[15]is a traditional optimization method, which was proposed by Spendly, Hest and Himswaorth. Simplex is the convex hull which has $n+1$ vertices in the R^n. Simplex method searches a point with smaller function values in the existing simplex. For example, we select a set of points $P_1, P_2, ..., P_n, P_{n+1}$ in the R^n. Let P_k be the minimum point of all vertices and $\overline{P}$ be the center of gravity of all vertices except the minimum vertex. The new points after reflection,expansion and contraction are written as P', P'', P''', respectively.

Simplex method finds the best point, the worst point and the the nearly worst point among the points $P_1, P_2, ..., P_n, P_{n+1}$, and then replaces the vertices with maximum function values by the vertices with smaller function value. A series of operations of reflection, expansion and contraction is done by comparison of function values of each vertex. The operations are described as follows.

(1)Reflection operation. $P' = 2\overline{P} - P_k$, the point with maximum function values is moved to the center of gravity in opposite direction randomly, which helps to explore various possible points in the space.

(2)Expansion operation. $P'' = \overline{P} + \alpha(P' - \overline{P})$, the new point with maximum fitness values are used to continue to expand further from the maximum vertex. If the current minimum point is the local minimum point, the expansion operation may make the point cross out the local minimum value region.

(3)Contraction operation. $P''' = \overline{P} + \gamma(P_k - \overline{P})$, if the reflection operation produces a bad point, the bad point can be adjusted by contraction.

We select $n+1$ points $P_1, P_2, ..., P_n, P_{n+1}$ in the R^n from a simplex. Comparing the fitness values of each point, we suppose that $fit(P_1) > ... > fit(P_l) > fit(P_k)$, which means P_1 is the best point, P_k is the worst point, P_l is the nearly worst point. Firstly, we calculate the center of gravity point $\overline{P}$, then use the reflection operation to get point P' of the worst point P_k. P' will be used to get point P'' by expansion operation if $fit(P') > fit(P_1)$. Replace X_k with P'' if $fit(P'') > fit(P')$, otherwise, still use P' to replace P_k. If $fit(P_1) > fit(P') > fit(P_l)$, replace P_k with P'. If $fit(P_l) > fit(P') > fit(P_k)$, which means P' has gone too far and must compress $P_m = \overline{P} + \beta(P' - \overline{P})$, and then replace P_k with P_m. If $fit(P_k) > fit(P')$, there must compress more and use contraction operation to get point

P''', at the meanwhile, if $f(P''') > f(P_k)$, replace P_k with P''', otherwise, we think that all the fitness values in the direction from P_k to $\overline{P}$ are smaller than $fit(P_k)$ and we will search not in this orientation next, let P_1 be the center, compress P_kP_1 and P_lP_1 to the half of the simplex. Take the $n+1$ points obtained above as a new simplex and continue the operations of reflection, expansion and contraction till some termination criterion is satisfied. The simplex method has a strong local search ability and can search the solution space more comprehensively through the operations of reflection, expansion and contraction.

3. The Proposed Algorithm

In the scout bee stage of ABC algorithm, when a food source position cannot improved after a given $limit$ times of iteration, it is replaced by some food source selected randomly. As the probabilities of random selection from good or bad food source are equal, the adjust scheme(14) is likely to lead a poor local search ability. Coupling simplex method into the ABC algorithm, we will get a hybrid ABC algorithm(SABC), which takes account of the stronger global optimization ability of ABC algorithm and local search ability of the simplex method.

In contrast to the detection strategy in ABC algorithm, SABC algorithm will search the food source without being updated by simplex method if counter is larger than the predefined $limit$. The rule of updating is as follows. Select randomly a set of points $X, X+G^{(1)}(0,1), X+G^{(2)}(0,1),..., X+G^{(n)}(0,1)$, where X is the current best point, $G^{(m)}(0,1)$ is a gaussian distribution with expected value of 0 and variance of 1, and then get better point X' by the operations of reflection, expansion and contraction. If $fit(X') > fit(X_i^t)$, replace X_i^t with X', otherwise, use X to replace X_i^t. Coupling the simplex method in ABC algorithm will effectively help the ABC algorithm to overcome the blindness of search food source, which results in the proposed SABC algorithm to find the best solution more quickly. The process of SABC algorithm is as follows.

Step1 Initialize NP food sources and calculated their fitness values of each food source. Given parameters $limit$ and set maximum iteration number to be max.

Step2 Produce new food source. Calculate fitness value of the new food sources and compare with the original fitness value. If the new food source is better than the original one according to their fitness values, then use the new food source to replace the original one and the counter $trail_i$ is set to 0, otherwise, the original food source is still kept and the counter $trail_i$ increases by 1.

Step3 Calculate the probability of food source selected by Eq(7), select food source to update by roulette wheel selection method, namely produce $r \in [0,1]$ randomly and compare p_i and r.

Step4 If $p_i > r$, the onlooker bee update the position of the food source by Eq(8).

Step5 Compare the fitness value of new food source with the original food source, if the new food source is better than the original one according to their fitness values, then use the new food source to replace the original food source and the counter $trail_i$ is set to 0, otherwise, the old food source is still kept and the counter $trail_i$ increases by 1.

Step6 If the position of some food source cannot be improved over a predefined $limit$ times of iteration, the associated employed bee discards its current food source and becomes a scout bee to search for a new food source by simplex method, and the counter $trail_i$ is set to 0.

Step7 Calculate the fitness values of all food sources, find and record the current best solution .

Step8 If the specified precision is satisfied or maximum number of iteration is reached, output the best solution obtained so far; otherwise, turn to Step2.

4. Experimental Results and Analysis

The following nine typical unconstrained optimization problems[16]are used to test the performance of the proposed SABC algorithm by comparing among ABC algorithm, GABC algorithm[17]and SABC algorithm. The objective function of these test problems are shown in Table 1.

Table 1. The Benchmark Problems Used in the Experiments

Function	Formulation	Minimum	Range
Sphere	$f_1(x)=\sum_{i=1}^{n}x_i^2$	0	[-100,100]
Griewank	$f_2(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$	0	[-600,600]
Rosenbrock	$f_3(x)=\sum_{i=1}^{n}[100(x_{i+1}-x_i^2)+(x_i-1)^2]$	0	[-30,30]
Rastrigin	$f_4(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	0	[-5.12,5.12]
Schwefel	$f_5(x)=\sum_{i=1}^{n}(\sum_{j=1}^{i}x_j^2)$	0	[-10,10]
Ackley	$f_6(x)=-20\exp(-0.2\sqrt{\sum_{i=1}^{n}x_i^2/n})-\exp(\sum_{i=1}^{n}\cos(2\pi x_i/n)+20+e$	0	[-32,32]
Shaffer	$f_7(x)=(x_1+x_2)^{0.25}[\sin^2(50(x_1^2+x_2^2)^{0.1})+1]$	0	[-100,100]
Six-hum pcame back	$f_8(x)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	-1.0316	[-5,5]
Extended Beale	$f_9(x)=[1.5-x_1(1-x_2)]^2+[2.25-x_1(1-x_2^2)]^2+[2.625-x_1(1-x_2^3)]^2$	0	[-5,5]

4.1. Convergence Accuracy of Fixed Iteration Times

In order to further compare the performance of the algorithms and to reduce the influence of contingency, the experiment on each benchmark problems is repeated 30 times independently, and the average values of the relative results of the experiment are used to compare. To make a fair comparison, for the test functions $f_1 \sim f_9$, the same parameters are set in the three algorithms. The population size is set to be 100, the parameter $limit$is set to be 20, and the maximum number of iteration is set to be 3000. In the simplex method, parameters α , β and γ are set to be 1.5, 0.5 and 0.4,respectively. For the test functions $f_1 \sim f_6$, the dimensions are set to be 10, 20 and 30,respectively, for the test functions $f_7 \sim f_9$, the dimension is set to be 2. The means, standard deviations, the worst solution and the best solution of objective function values from the experimental data are shown in Table 2.

Table 2. Statistical Experimental Results on Three Algorithms

Function	Dimensions	Algorithm	Means	Standard Deviations	Worst Solution	Best Solution
f_1	10	ABC	5.66e-08	1.15e-07	5.85e-07	1.76e-16
		GABC	4.82e-17	1.28e-17	7.22e-17	1.97e-17
		SABC	6.94e-17	1.91e-17	1.10e-16	3.42e-17
	20	ABC	1.74e-04	7.97e-05	3.15e-04	4.15e-05
		GABC	2.39e-16	3.30e-17	2.89e-16	1.62e-16

		SABC	2.60e-16	3.41e-17	3.16e-16	1.84e-16
		ABC	0.001128	5.76e-04	0.002557	2.23e-04
	30	GABC	3.52e-16	8.17e-17	5.18e-16	2.12e-16
		SABC	4.61e-16	7.93e-17	5.44e-16	2.95e-16
		ABC	0.003873	0.003580	0.012668	7.46e-05
	10	GABC	0.011179	0.007109	0.024267	5.01e-04
		SABC	8.58e-04	0.002243	0.007411	0
		ABC	0.008699	0.008404	0.033245	4.68e-05
f_2	20	GABC	0.048722	0.032962	0.140056	0.001090
		SABC	4.20e-07	2.25e-06	1.23e-05	0
		ABC	0.021992	0.017971	0.060621	1.36e-04
	30	GABC	0.205398	0.122984	0.560844	0.061345
		SABC	5.85e-09	3.05e-08	1.67e-07	0
		ABC	26.53104	10.68712	44.24029	10.60602
	10	GABC	5.027279	1.027279	2.694529	0.311053
		SABC	0.684424	0.760418	3.922391	0.011089
		ABC	6.075012	2.835028	17.53667	2.401123
f_3	20	GABC	6.024572	3.591280	16.79524	0.633680
		SABC	1.907482	2.760622	9.807784	0.001465
		ABC	26.66079	17.46751	39.40539	14.44351
	30	GABC	10.72740	14.12652	20.12905	4.017299
		SABC	2.059488	3.060753	11.37704	9.27e-04
		ABC	5.10e-04	3.79e-04	0.001412	2.47e-05
	10	GABC	2.23e-14	1.07e-13	5.90e-13	0
		SABC	9.24e-10	3.90e-09	2.03e-08	0
		ABC	2.130611	0.790072	3.892076	0.679964
f_4	20	GABC	1.61e-07	5.51e-07	2.49e-06	1.78E-15
		SABC	0.314124	0.587971	2.081519	0
		ABC	10.62522	1.884068	13.60882	6.032060
	30	GABC	6.09e-03	1.42e-02	5.50e-02	5.17e-10
		SABC	4.274558	2.463931	8.017724	0
		ABC	5.83e-16	1.69e-15	9.50e-15	9.55e-17
	10	GABC	5.21e-17	1.31e-17	7.99e-17	2.24e-17
f_5		SABC	6.87e-17	1.76e-17	9.90e-17	2.03e-17
	20	ABC	2.93e-05	2.61e-05	9.07e-05	2.33e-07
		GABC	3.07e-16	7.47e-17	4.73e-16	1.98e-16
		SABC	2.63e-16	3.37e-17	3.19e-16	2.09e-16
		ABC	4.60e-04	2.72e-04	1.12e-03	8.49e-05
	30	GABC	7.59e-07	2.46e-06	1.10e-05	7.19e-16
		SABC	4.70e-16	6.77e-17	5.40e-16	3.10e-16
		ABC	1.27e-04	1.78e-04	6.39e-04	8.01e-12
	10	GABC	2.59e-05	6.78e-05	3.60e-04	1.47e-13
		SABC	5.86e-15	1.77e-15	7.99e-15	4.44e-15
		ABC	3.72e-02	2.54e-02	9.12e-02	4.40e-03
f_6	20	GABC	1.90e-02	1.86e-02	7.89e-02	1.75e-03
		SABC	1.94e-14	3.29e-15	2.22e-14	1.51e-14
		ABC	0.444777	0.248911	1.199117	0.070959
	30	GABC	0.302073	0.268158	1.020752	0.029738
		SABC	3.94e-14	5.20e-15	5.06e-14	2.93e-14
f_7	2	ABC	4.12e-05	6.22e-05	2.58e-04	4.63e-08
		GABC	8.05e-05	1.89e-04	6.93e-04	1.93e-16

		SABC	1.42e-16	2.94e-16	1.65e-15	1.26e-17
		ABC	-1.031628	6.45e-16	-1.031628	-1.031628
f_8	2	GABC	-1.031628	6.78e-16	-1.031628	-1.031628
		SABC	-1.031628	6.78e-16	-1.031628	-1.031628
		ABC	1.59e-06	2.20e-06	8.86e-06	2.52e-09
f_9	2	GABC	1.64e-10	4.98e-10	2.68e-09	6.19e-15
		SABC	4.79e-06	1.56e-05	7.85e-06	4.18e-16

Table 2 shows that, under the different dimensions, the SABC algorithm performs well in both the quality of solution and the stability, compared to ABC algorithm and GABC algorithm. For test functions f_1, f_4, f_5 and f_6 with dimension $D=10$, SABC algorithm found the optimal solution with a high precision, and the results are more stable than that of ABC algorithm. For problem f_2, the optimization accuracy is poor, but the best solution obtained by SABC algorithm is almost the optimal solution. For problem f_3, the result of SABC algorithm has been improved in accuracy, compare to ABC algorithm and GABC algorithm. With $D=20$ and $D=30$, the SABC algorithm found the optimal solution with a high precision. For test functions f_1, f_2, f_5 and f_6, the results are more stable than those of other two algorithms. For problems f_3 and f_4, the optimization accuracy is poor. But for problem f_4 the best solution obtained by SABC algorithm is almost the optimal solution, and for problem f_3, the solution obtained by SABC algorithm has higher precision. For test function f_7, only the SABC algorithm found its optimal solution. For test function f_8, three algorithms all achieved the optimal solution. For test function f_9, SABC algorithm obtained the worst solution.

In order to further compare the performance of the three algorithms, the evolution curve of thirty-time average optimal solution for nine test problems are shown as Fig. 1-9. The vertical coordinates of the figures are taken as logarithm of the average optimal solution except Fig. 8.

Fig. 1-9 show that the proposed SABC algorithm is well performed among three algorithms. For the benchmark functions $f_1 \sim f_7$ except f_4, the numbers of iteration that the SABC algorithm needed to obtain the optimal solution are smaller than those in other two algorithms. The result in Figure8 shows that the three algorithms have the same performance. Figures 4 and 8 depict that the SABC algorithm has medium level of performance in aspect of the number of iteration among three algorithms.

4.2. The Number of Iterations under Specified Precision

In order to further compare the performance of the algorithms and to reduce the influence of contingency, the experiment on each benchmark problems is repeated 30 times independently, and the average values of the relative results of the experiment are used to compare. To make a fair comparison, for test function $f_1 \sim f_9$, the same parameters are set in the three algorithms. The success rate is the number of successful times which some algorithm obtains the solution in the specified precision divided by the number of independent running times 30. For test functions $f_1 \sim f_6$, the dimensions are set to be 20 , for test functions $f_7 \sim f_9$, the dimension is set to be 2. For test functions f_1, f_5, f_6, f_7 and f_9, the precision is set to be 10^{-6}, for test function f_2, the precision is set to be 10^{-5}, for test function f_3, the precision is set to be 1, for test function f_4, the precision is set to be 10^{-3}, for test function f_8, the precision is set to be 10^{-4}. The success rate, means, minimum number, maximum number of iterations and average time of 30 independent runs from the statistical experimental data are shown in Table 3.

Table 3 Statistical Experimental Results on Three Algorithms

Function	Precision	Algorithm	Success Rate	Means	Minimum Number	Maximum Number	Time(s)
		ABC	0	3000	3000	3000	31.361458
f_1	1.00e-06	GABC	1	636	457	1308	20.103646
		SABC	1	488	333	632	8.1682371
		ABC	0	3000	3000	3000	37.290622
f_2	1.00e-05	GABC	1	744	460	2616	27.798958
		SABC	1	528	419	692	7.5781250
		ABC	0	3000	3000	3000	44.390625
f_3	1	GABC	0	3000	3000	3000	79.984375
		SABC	0.7	699	470	1148	29.624479
		ABC	0	3000	3000	3000	41.142188
f_4	1.00e-03	GABC	1	995	674	2449	18.052083
		SABC	0.5	707	436	1282	28.417186
		ABC	0.37	444	394	515	29.104688
f_5	1.00e-06	GABC	1	482	320	783	11.301563
		SABC	1	371	289	424	5.7416667
		ABC	0	3000	3000	3000	36.451563
f_6	1.00e-06	GABC	0	3000	3000	3000	85.881250
		SABC	1	487	350	610	10.229687
		ABC	0.27	173	539	2879	28.695313
f_7	1.00e-06	GABC	0.43	1604	534	2767	67.659375
		SABC	1	510	275	821	10.684375
		ABC	1	13	6	22	0.1927080
f_8	1.00e-04	GABC	1	14	2	22	0.3692708
		SABC	1	11	3	16	0.188021
		ABC	1	625	204	2221	15.449479
f_9	1.00e-06	GABC	1	560	128	1522	13.314063
		SABC	1	362	56	655	7.5416670

It is show that, for the functions f_1, f_2, the ABC algorithm does not get the solution under the specified precision. For the functions f_3 and f_6, the solution can not be achieved by the ABC algorithm and GABC algorithm, but the success rate of SABC algorithm is 100%. For the function f_4, the result of SABC algorithm has medium level of success rate among three algorithms, but its computation time is the least. For functions f_5 and f_7, the SABC algorithm has the highest success rate and the least computation time. For functions f_7 and f_8, all the success rate of three algorithms are 100%, but the computation time in the SABC algorithm is the least.

The above results show that, compared to the ABC algorithm and GABC algorithm under the specified precision, the proposed SABC algorithm has advantages in success rate, stability and calculation efficiency.

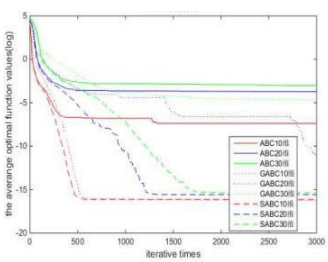

Fig. 1. Evolution curve of f_1.

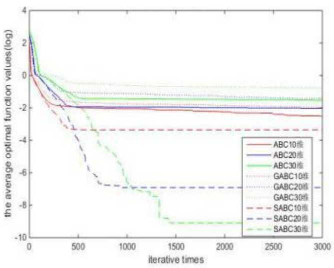

Fig. 2. Evolution curve of f_2.

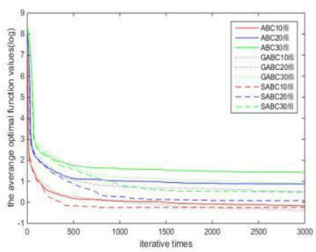

Fig. 3. Evolution curve of f_3.

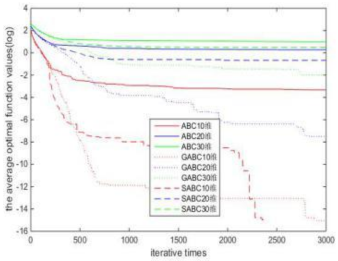

Fig. 4. Evolution curve of f_4.

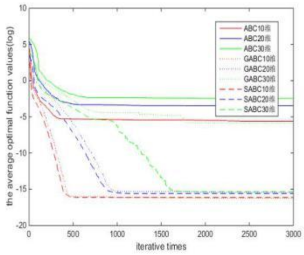

Fig. 5. Evolution curve of f_5.

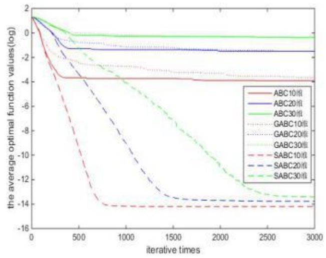

Fig. 6. Evolution curve of f_6.

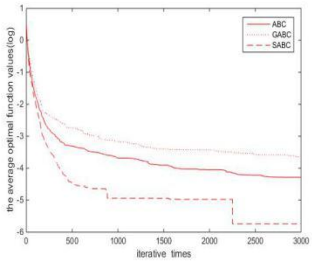

Fig. 7. Evolution curve of f_7.

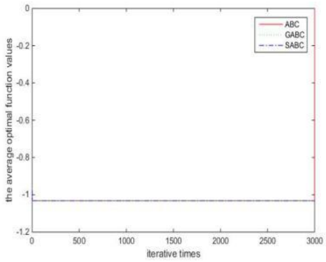

Fig. 8. Evolution curve of f_8.

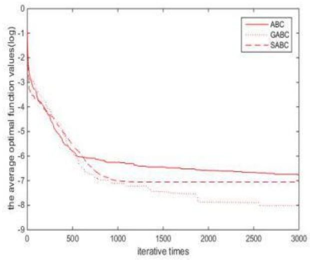

Fig. 9. Evolution curve of f_9.

5. Conclusion

An artificial bee colony algorithm(SABC) based on simplex method is proposed. The SABC algorithm employees the simplex method to update the food sources that don't improve over a predefined time period (*limit*). Compare to ABC algorithm, SABC algorithm overcomes the blindness searches, improves the search efficiency and speeds up the convergence rate of ABC algorithm. The numerical results on nine benchmark problems show that the proposed SABC algorithm performance well in convergence speed and precision compare to ABC algorithm and GABC algorithm. However, the optimization of multi-modal functions are not ideal, hence the problem needs to be further studied and discussed.

Acknowledgment

The authors thank the anonymous reviewers for their valuable comments and suggestions. This work was supported by the National Natural Science Foundation of China under Grant No. 61463009, Program for the Science and Technology Top Talents of Higher Learning Institutions of Guizhou under Grant No. KY[2017]070 , Joint Beijing Natural Science Foundation Program under Grant No.4122022, Education Department of Guizhou Province Projects under Grant No. KY[2017]004 , Central Support Local Projects under Grant No. PXM 2013-014210-0 0 0173.

Reference

[1] Karaboga, D. (2005). An idea based on honey bee swarm for numerical optimization. Kayseri: Computer Engineering Department Engineering Faculty Erciyes University, 2005.

[2] Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: Artificial Bee Colony(ABC) algorithm. *Journal of Global Optimization*, *39(3)*, 459-471.

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

[4] Fuli, Z., & Hui, L., & Shouming, Z. (2017). An improved artificial bee colony algorithm with modified-neighborhood based update operation and independent-inheriting-search strategy for global optimization. *Engineering Application of Artificial Intelligence*, *58*, 134-156.

[5] Liang, Z. P., Hu, K. F., Zhu, Q. X., & Z. Z. X. (2017). An enhanced artificial bee colony algorithm with adaptive differential operators. *Applied Soft Computing*, *58*, 480-494.

[6] Bi, X. J., & Wang, Y. J. (2012). A modified artificial bee colony algorithm and its application. *Journal of Harbin Engineering University*, *33(1)*, 117-123.

[7] Wei, F. T., Yue, M. J., & Zhen, J. M. (2017). Improved artificial bee colony algorithm based on multi-strategy fusion. *Computer Engineering and Applications*, 1-10.

[8] Yang, C. H., Qian, X. S., Gui, & Wei, H. (2011). Hybrid algorithm of chaotic differential evolution and particle swarm optimization. *Application Research of Computers*, *28(2)*, 339-441.

[9] Du, Z. X., Han, D. Z., & Liang, Z. (2017). Adaptive greedy searching artificial bee colony algorithm. *Journal of Yanshan University*, *41(2)*, 183-188.

[10] Banhamsakun, A., & Achalakul, T. The best-so-far selection in artificial bee colony algorithm. *Applied Soft Computing*, *11(2)*, 2888-2901.

[11] Spendley, W., Hext, G., & Himsworth, F. R. (1962). Sequential application of simplex designs in optimization and evolution. *Technometrica*, 441-461.

[12] Nelder, J. A., & Mead, R. A. (1965). Simplex method for function minimization. *The Computer Journal*, *7(4)*, 308-313.

[13] Jie, L., Wu, L. H., Liu, J. X. (2009). Hybrid differential evolution algorithm based on simplex operator. *Computer Engineering*, *35(13)*, 179-182.

[14] Jie, L., & Wang, Y. P. (2014). An improved central optimization based on simplex method. *Journal of zhejiang University (Engineering Science)*, *48(12)*, 2115-2123.

[15] Zhang, H. X., Yi, L., & Shi, R. F. (2011). Artificial fish swarm algorithm based on simplex method. *Journal of Computer Applications*, *31(5)*, 1321-1327.

[16] Cui, L. Z., Li, G. H., & Lin, Q. Z. *et al*. (2016). A novelartificial bee colony algorithm with depth-first search framework and elite-guided search equation. *Information Sciences*, *22(7)*, 1012-1044.

[17] Zhu, G. P., & Sam, K. (2010). Gbest-guided artificial bee colony algorithm for numerical function optimization. *Applied Mathematics and Computation*, 3166-3173.

Xufang Zhao was born in 1991, in Shuozhou, Shanxi province, China and has been studying for a master's degree since 2016 in Beijing, China. She research discretion is optimization method and application.

Ximing Liang was born in 1967, in Hunan province, is postdoctor and asa professor, and is researching in Beijing University of Civil Engineering and Architecturein. He research discretion is optimization method and application.

Long Wen was born in 1977, in Hunan province. He is a postdoctoral and a professor. He research discretion is evolutionary computation, intelligent optimization algorithm and application.