2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

A Novel Chaotic Artificial Bee Colony Algorithm Based on Tent Map

Fangjun Kuang, Zhong Jin, Weihong Xu
School of Computer Science and Engineering
Nanjing University of Science and Technology
Nanjing, China
zhongjin@njust.edu.cn

Fangjun Kuang, Siyang Zhang
Department of Electronic and Information Engineering
Hunan Vocational Institute of Safety & Technology
Changsha, China
kfjztb@126.com

***Abstract*—A novel self-adaptive chaotic artificial bee colony algorithm based on Tent map (STOC-ABC) is proposed to enhance the global convergence and the population diversity. In the STOC-ABC, Tent chaotic opposition-based learning initialization method is presented to diversify the initial individuals and obtain good initial solutions. Furthermore, the self-adaptive Tent chaotic searching is implemented at the zones nearby individual optimum solution to help the artificial bee colony (ABC) algorithm to escape from the local optimum effectively. Moreover, the tournament selection strategy in onlooker bee phase is employed to increase the ability of the algorithm and avoid premature convergence. Experiments on six complex benchmark functions with high-dimension, the results further demonstrate that, the STOC-ABC not only accelerates the convergence rate and improves solution precision, but also provides excellent performance in dealing with complex high-dimensional functions.**

***Keywords*—Artificial bee colony; chaotic opposition-based learning; Tent chaos search; self-adapting search; tournament selection strategy**

I. INTRODUCTION

Optimization algorithm has been applied in various fields, including function optimization, engineering design, operational research, information science and related areas. Biological-inspired optimization algorithms have been developed to be successful in tackling increasingly complex real world optimization problems in recent decades, such as genetic algorithm (GA) inspired by the Darwinian law of survival of the fittest [1], particle swarm optimization (PSO) inspired by the social behavior of bird flocking or fish schooling [2], ant colony optimization (ACO) inspired by the foraging behavior of ant colonies [3], and artificial bee colony (ABC) algorithm inspired by the foraging behavior of honey bee swarm [4], and so on. Numerical comparisons demonstrated that the performance of the ABC algorithm is competitive to other population-based algorithms with the advantage of employing fewer control parameters [5-7]. Due to its simplicity and ease of implementation, the ABC algorithm has captured much attention and has been applied to solve many practical optimization problems [8-10].

However, similar to other evolutionary algorithms, the standard ABC algorithm also faces up to some challenging problems. For example, the convergence speed of the ABC algorithm is typically slower than those of the representative population-based algorithms when dealing with the unimodal problems [7], because it cannot use the adequate information to determine the most promising search direction. What is more, the ABC algorithm can also easily get trapped in the local optima when solving complex multimodal problems [7], the reason is there is still the insufficiency in ABC regarding the solution search equation, which is used to generate new candidate solution based on the information of previous solutions, is good at exploration but poor at exploitation.

Therefore, accelerating convergence speed and avoiding the local optima have become two most important and appealing goals in the ABC research. A number of variant ABC algorithms have, hence, been proposed to achieve these goals [11-16]. Zhang et al. [11] used the ABC as a data mining technique or clustering data, and compared it with other clustering methods on different data sets. Akay and Karaboga [12] proposed a modified artificial bee colony algorithm for real parameter optimization. Karaboga and Akay [13] proposed a modified artificial bee colony algorithm for constrained optimization problems. Alatas [14] proposed an ABC model that uses chaotic maps for parameter adaptation so as to improve the convergence characteristics and prevent the ABC from getting stuck in local minimums. An application of chaotic bee colony approach to air vehicle path planning was presented in [15]. Gao and Liu [16] modified the search equation of the basic ABC by using chaotic systems and opposition-based learning methods.

The existing chaos optimization algorithms were almost based on Logistic map. However, the probability density function of chaotic sequences for Logistic map is a Chebyshev-type function, which may affect the global searching capacity and computational efficiency of chaos optimization algorithm. Comparing with Logistic map, chaotic sequences produced by Tent map behave with global ergodicity and uniform, which are insensitive to initial value. Therefore, in this study, a novel self-adaptive chaotic artificial bee colony algorithm based on Tent map (STOC-ABC) is proposed to accelerate convergence speed and escape from the local optima of ABC. In STOC-ABC, a novel population initialization method is presented, which employs the Tent Chaos map and the opposition-based learning method to generate initial population, and increase the

population diversity. The self-adaptive Tent chaotic searching is implemented at the zones nearby individual optimum solution X_{best} . The STOC-ABC algorithm, combining quick optimization property of ABC in multi-dimension space and global ergodicity of chaotic searching, can greatly improve the optimization capability and prevent the ABC plunging into the local minima. Result comparison of the algorithm with others on six benchmark functions confirms its efficiency. The results show that the STOC-ABC algorithm outperforms the other algorithms in terms of population diversity, robustness, and convergence speed.

The rest of the paper is organized as follows. Section II describes the ABC algorithm. The improved ABC algorithm called STOC-ABC algorithm is presented and analyzed in Section III. In Section IV, STOC-ABC is tested on six benchmark functions compared with several other algorithms, and the experimental results are presented and discussed. Section V presents the conclusions and the future work.

II. ARTIFICIAL BEE COLONY ALGORITHM

ABC algorithm was applied to multidimensional and multimodal function optimization in [4, 5]. The swarm is divided into employed bees, scouts and onlookers. In the initialization phase, the algorithm generates a group of food sources corresponding to the solutions in the search space.

The food sources are produced randomly within the range of the boundaries of the variables.

$$x_{i,j} = x_j^{\min} + R(x_j^{\max} - x_j^{\min}) \tag{1}$$

where $i = 1,2,\cdots,SN$, $j = 1,2,\cdots,D$. SN is the number of food sources and equals to half of the colony size. D is the dimension of the problem, representing the number of parameters to be optimized. $x_j^{\min}$ and $x_j^{\max}$ are lower and upper bounds of the *jth* parameter, respectively. The fitness of food sources will be evaluated. Additionally, counters which store the numbers of trials of each bee are set to 0 in this phase.

In the employed bees' phase, a number of employed bees, set as the number of the food sources and half the colony size, are used to find new food sources using (2)

$$v_{i,j} = x_{i,j} + \Phi_{i,j}(x_{i,j} - x_{k,j}) \tag{2}$$

where $i = 1,2,\cdots,SN$, and j is a randomly selected number in $[1, D]$, D is the number of dimensions. $\Phi_{i,j}$ is a random number uniformly distributed in the range $[-1,1]$. k is the index of a randomly chosen solution, where $k \neq i$. Both V_i and X_i are then compared against each other and the employed bee exploits the better food source.

Onlooker bees next choose a random food source according to the probability given in (3)

$$P_i(t) = \frac{fit_i(t)}{\sum_{n=1}^{SN} fit_n(t)} \tag{3}$$

where $fit_i(t)$ is the fitness of the *ith* food source. Then, each onlooker bee tries to find a better food source around the selected one using (1).

If a food source cannot be improved for a predetermined number of cycles, referred to as *Limit* , this food source is abandoned. The employed bee that was exploiting this food source becomes a scout that looks for a new food source by randomly searching the problem domain.

III. IMPROVED CHAOTIC ARTIFICIAL BEE COLONY

A. Tent Chaos Map

Similar to other evolutionary algorithms, artificial bee colony still has premature convergence phenomenon. Chaos phenomenon widely exists in nonlinear systems, which is a deterministic, random-like process, dynamical system. Moreover, it has a very sensitive dependence upon its initial condition and parameter [17]. Therefore, chaotic search strategy has been applied in the ABC algorithm to improve the ability to search global optimal solution [14, 15]. The invariant density of iterates is the uniform distribution function in the interval $[0,1]$, the Tent map shows outstanding advantages and higher iterative speed than the Logistic map [18]. In this study, the Tent-map is used in chaos optimization to generate the chaotic series. Tent map is defined as follows:

$$cx_{t+1} = \begin{cases} 2cx_t, & 0 \le cx_t \le 1/2 \\ 2(1-cx_t), & 1/2 \le cx_t \le 1 \end{cases} \tag{4}$$

where cx_t is Tent chaotic vector, and $cx_t \notin \{0.2,0.4,0.6,0.8\}$.

B. Chaotic Opposition-based Learning Initialization

Population initialization is a crucial task in evolutionary algorithms because it can affect the convergence speed and the quality of the final solution. If no information about the solution is available, then random initialization is then most commonly used method to generate initial population. Owing to the randomness and sensitivity dependence on the initial conditions of chaotic maps, the chaotic maps have been used to initialize the population so that the search space information can be extracted to increase the population diversity [14]. At the same time, according to [19], replacing the random initialization with the opposition-based population initialization can get better initial solutions and accelerate convergence speed. So this paper proposes a novel initialization approach which employs the Tent chaotic map and the opposition-based learning method to generate initial population. The chaotic opposition-based learning population initialization is described as algorithm 1.

Algorithm 1 A novel initialization method

1 Set the maximum number of chaotic iteration $C_{\max}$, the population scale SN .

2 for i=1: SN

3 for j=1: D

4 Randomly generate initialize variables $cx_{0,j} \in (0,1)$

except 0.2, 0.4, 0.6, and 0.8;

5 for k=1: $C_{\max}$

6 if $cx_{k-1,j} <= 1/2$

7 $cx_{k,j} = 2cx_{k-1,j}$

8 else

9 $cx_{k,j} = 2(1 - cx_{k-1,j})$

10 end

11 end

12 $x_{i,j} = x_j^{\min} + cx_{k,j} \times (x_j^{\max} - x_j^{\min})$

13 end

14 end

15 for i=1: SN

16 for j=1: D

17 $ox_{i,j} = x_j^{\min} + x_j^{\max} - x_{i,j}$

18 end

19 end

20 Selecting SN fittest individuals from set the $\{x\}_{i=1}^{SN} \cup \{ox\}_{i=1}^{SN}$ as initial population.

C. Self-adaptive Tent Chaos Search

The Tent chaotic search is aimed to utilize the Tent map to explore a better solution near the X_{best} . Tent chaotic local search of the STOC-ABC algorithm increases the ability to avoid local optima, and reduces the computation time. The detailed procedure of chaotic local search is described as algorithm 2.

Algorithm 2 Self-adaptive Tent Chaos Search

Step 1: Find the best solution named $X_{best} = (x_{k,1}, \cdots, x_{k,D})$, and calculate the fitness of X_{best} .

Step 2: Set the iteration $Count = 0$ and generate the initial chaotic vector distribute in (0, 1) using (5).

$$cx_{k,j}^0 = (x_{k,j} - x_j^{\min}) / (x_j^{\max} - x_j^{\min}); \ k = 1, \cdots, SN; j = 1, \cdots, D \quad (5)$$

Step 3: Calculate chaotic variables $cx_{k,j}^m (m = 1,2, \cdots, C_{\max})$ for next iteration using (4) except 0.2, 0.4, 0.6, and 0.8, where $C_{\max}$ is maximum chaotic search number.

Step 4: Convert the chaotic variables $cx_{k,j}^m$ to the decision variables and generate new solution V_k using (6)

$$v_{k,j} = x_{k,j} + \frac{(x_j^{\max} - x_j^{\min})}{2}(2cx_{k,j}^m - 1) \quad (6)$$

Step 5: Calculate the fitness of V_k and compare it to the X_{best} , if the fitness of V_k is better than the fitness of X_{best} , the solution should be selected as the new X_{best} .

Step 6: $Count = Count + 1$, if the maximum iteration cycle is not reached yet, then go to step 3. Otherwise, chaotic search is completed.

D. Tournament Selection

The proportional selection in ABC algorithm requires the fitness function greater than zero. However, tournament selection [20] is different, it's a selection process based on local competition which only refers to the relative value of individuals. In this paper, we select two individuals from the population and compare their fitness values, then assign one score to a better individual of the two, repeat such process and then the individual with the highest values wins the heaviest weight.

This method of selection offers more chances for high-fitness individuals to survive. Meanwhile, this method avoids from the influence of the super individuals since it only standardizes relative value of fitness which is not in proportion to the size of fitness. To a certain extent, it also avoids from premature convergence and stagnation. Selection probability of fitness is as follow:

$$P_i(t) = \frac{c_i(t)}{\sum_{i=1}^{N} c_i(t)} \quad (7)$$

where c_i is the score of the ith individual.

E. Main Procedure of the STOC-ABC

In this section, we mainly focus on the procedure of proposed the STOC-ABC algorithm. The detailed procedure of the STOC-ABC algorithm is given as algorithm 3.

Algorithm 3 STOC-ABC

Step 1: Initial the food sources and computation conditions include population of bee colony N , number of employed bees $SN = (N/2)$, upper and lower boundaries of every decision variable, maximum iteration $G_{\max}$, $Limit$ and chaotic local search iteration number $C_{\max}$.

Step 2: Set iteration $iter = 0$, generate SN vectors X_i with D dimensions as food sources according to algorithm 1.

Step3：Sent SN employed bees to food sources. Initialize the flag vector $trial(i) = 0$, which is recorded the cycle number of a food source.

Step 4: Produce new solutions V_i using employed bees by (2), and calculate the fitness value $fit(V_i)$.

Step 5: If $fit(V_i) > fit(X_i)$, then $X_i = V_i$, $trial(i) = 0$; Else X_i is maintained, $trial(i) = trial(i) + 1$.

Step 6: Calculate the probability values P_i of food sources by applying tournament selection using (7).

Step 7: Onlooker bees choose the food sources by probabilities P_i until all of them have a corresponding food source, and produce new solutions V_i . Calculate the fitness value $fit(V_i)$.

Step 8: If $fit(V_i) > fit(X_i)$, then $X_i = V_i$, $trial(i) = 0$; Else X_i is maintained, $trial(i) = trial(i) + 1$ 。

Step 9: If $trial(i) > Limit$, then there is an abandoned solution for the scout then replace it with a new food source V_i ,which will be reinitialized by carrying out self-adaptive Tent chaos search of algorithm 2.

Step 10: Memorize the best solution found so far.

Step 11: Update $iter = iter + 1$. If the maximum iteration cycle is not reached yet, then go to step 4. Otherwise, return best solution.

IV. EXPERIMENT AND RESULTS ANALYSIS

In order to estimate the performance and analyze the exploration and exploitation abilities of the methods, we used 6 well-known benchmark functions taken from [4-7], shown in Table I. These functions contain two unimodal (containing only one optimum) functions, four multimodal (containing many local optima, but only one global optimum) functions. Well-defined benchmark functions which are based on mathematical functions can be used as objective functions to test and evaluate the performance of optimization methods. The nature, complexity and other properties of these functions can be easily obtained from their definitions. The difficulty levels of most benchmark functions are adjustable by setting their parameters.

TABLE I. THE BENCHMARK FUNCTIONS USED IN THE EXPERIMENTS

Function	Formulation and range	Global minimum	Dimensions	Property
Sphere	$f_1(X)=\sum_{i=1}^{D} x_i^2, \quad x_i \in [-100,100]$	$x_i = 0, \quad f_1(X) = 0$	$D = 50$	Unimodal
Rosenbrock	$f_2(X)=\sum_{i=1}^{D-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2), \quad x_i \in [-30,30]$	$x_i = 1, \quad f_2(X) = 0$	$D = 50$	Unimodal
Rastrigin	$f_3(X)=\sum_{i=1}^{D} (100(x_i^2 - 10\cos(2\pi x_i) + 10)), \quad x_i \in [-5.12,5.12]$	$x_i = 0, \quad f_3(X) = 0$	$D = 50$	multimodal
Griewank	$f_4(X)=\frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1, \quad x_i \in [-600,600]$	$x_i = 0, \quad f_4(X) = 0$	$D = 50$	multimodal
Ackley	$f_5(X)=-20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{D} x_i^2}) - \exp(\frac{1}{n}\sum_{i=1}^{D} \cos(2\pi x_i) + 20 + e), \quad x_i \in [-32,32]$	$x_i = 0, \quad f_5(X) = 0$	$D = 50$	multimodal
Schwefel	$f_6(X)=-\sum_{i=1}^{D} (x_i \sin(\sqrt{\lvert x_i \rvert})), \quad x_i \in [-500,500]$	$x_i = 420.9687, \quad f_6(X) = -418.9829D$	$D = 50$	multimodal

A. Parameters Settings

In order to confirm the effectiveness of the STOC-ABC algorithm, ABC and ABC based on Logistic chaos algorithm (CABC) have been applied to resolve functions optimization problem. All ABC was initialized in regions that include the global optimum for a fair evaluation, which is shown as follows:

$$fit_i(t) = \begin{cases} \dfrac{1}{1 + f_i(t)} & if\,(f_i(t) \geq 0) \\ 1 + abs(f_i(t)) & otherwise \end{cases} \tag{8}$$

where $fit_i(t)$ is the fitness value of the ith food source, and $f_i(t)$ is the objective function value specific for the optimization problem.

The population size of all algorithms is set to 100, the number of food sources, employed bees and onlooker bees is half of the population size and the number of scout bees is selected as one. Limit time of food source cannot be improved is 100; maximum iteration cycle number $G_{\max}$ is 3000; chaotic local search iteration number $C_{\max}$ is 300. All algorithms are coded in Matlab 2012a using computer with Intel(R) Core (TM) i3-3120M 2.5 GHz CPU, 2 GB RAM. The operating system of the computer is Windows 7 ultimate.

B. Performance of Chaotic Opposition-based Learning

In order to estimate the performance of chaotic opposition-based learning initialization, the random initialization and opposition-based learning initialization have been tested on the six functions in terms of the fitness value and the population diversity. In this paper, the population diversity is a measurement of the cover degree, which is defined as follows:

$$Diversity = \frac{1}{SN}\sum_{i=1}^{SN}\sqrt{\frac{1}{D}\sum_{j=1}^{D}(x_{i,j} - \overline{x_j})^2} \tag{9}$$

where SN denotes the number of food sources, which is equal to the number of employed bees or onlooker bees. D is the number of variables or the dimension of the problem, and $\overline{x}$ is

the center position of the colony.

All functions were tested with 50 dimensions and all algorithms were run 30 times. Because all test function are minimization problems, the smaller the fitness, the better it is, while the population diversity is the opposite. The results are shown in Table II.

From Table II it can be seen that the fitness value and the population diversity of chaotic opposition-based learning initialization are better than these of the two other initialization method for all the test functions. In a word, by combining the advantages of Tent chaotic map and opposition-based learning method, the chaotic opposition-based learning initialization can increase the population diversity and obtain good initial solutions.

TABLE II. PERFORMANCE COMPARISON OF POPULATION INITIALIZATION METHOD

Fun \ Method	Random initialization		Opposition-based initialization		Chaotic opposition-based initialization	
	Fitness	Diversity	Fitness	Diversity	Fitness	Diversity
f_1	1.2947e+05	58.8765	1.1870e+05	65.5079	8.0050e+04	118.4524
f_2	8.6994e+08	17.3647	6.7075e+08	17.3808	3.7083e+08	35.2190
f_3	750.5709	2.98227	886.7954	3.44844	715.3098	5.76384
f_4	2.1049e+03	344.6124	1.7049e+03	349.4727	1.0121e+03	681.1092
f_5	21.1516	18.7528	21.0568	19.5132	20.4393	39.1969
f_6	-3.5631e+03	289.4924	-3.1012e+03	290.7989	-2.8151e+03	617.5043

C. *Performance of STOC-ABC*

In order to verify the effectiveness of the STOC-ABC algorithm, ABC and ABC based on Logistic chaos (CABC) algorithms have been applied to minimize the six benchmark functions. All functions were tested with 50 dimensions. For each function, all the algorithms were run 30 times. The best, worst, mean and standard deviations of function fitness values were obtained by implementing STOC-ABC, ABC, and CABC algorithms, respectively. The comparison results are shown in Table III, in which the results of winner algorithms are marked as bold. The convergence processes of the different ABCs on test functions are shown in Fig.1.

TABLE III. OPTIMIZATION RESULTS COMPARISON OF BENCHMARK FUNCTIONS

Function	Algorithm	Best	Worst	Mean	Std
Sphere	ABC	1.16921e-15	2.30472e-15	1.59341e-15	2.43195e-16
	CABC	9.68606e-16	1.86614e-15	1.50491e-15	2.09925e-16
	STOC-ABC	**9.56249e-20**	**9.76645e-18**	**6.17839e-19**	**7.92835e-31**
Rosenbrock	ABC	3.48953e-02	2.44179e+00	4.98511e-01	5.92527e-01
	CABC	4.83917e-03	7.16931e-01	1.69646e-01	2.00503e-01
	STOC-ABC	**2.97214e-06**	**8.87529e-02**	**2.83241e-04**	**1.22748e-15**
Rastrigin	ABC	1.13687e-13	9.48717e-11	7.40859e-12	2.04839e-11
	CABC	**0**	2.27374e-13	7.57912e-14	4.56052e-14
	STOC-ABC	**0**	**1.52761e-15**	**3.28702e-17**	**1.62783e-26**
Griewank	ABC	9.99201e-16	8.53762e-14	1.19978e-14	2.07400e-14
	CABC	**0**	**5.55112e-16**	1.25825e-16	1.22778e-16
	STOC-ABC	**0**	**1.66454e-16**	**1.40346e-18**	**1.32227e-27**
Ackley	ABC	3.00249e-11	1.94881e-10	7.55055e-11	3.34588e-11
	CABC	2.29594e-12	1.24105e-11	7.04130e-12	2.66576e-12
	STOC-ABC	**2.99961e-15**	**2.99961e-15**	**2.99961e-15**	**0**
Schwefel	ABC	-2.02027e+04	-2.07123e+04	-2.08738e+04	7.28559e+01
	CABC	**-2.09491e+04**	**-2.09491e+04**	**-2.09491e+04**	**1.88551e-11**
	STOC-ABC	**-2.09491e+04**	-2.09487e+04	**-2.09491e+04**	9.90947e-08

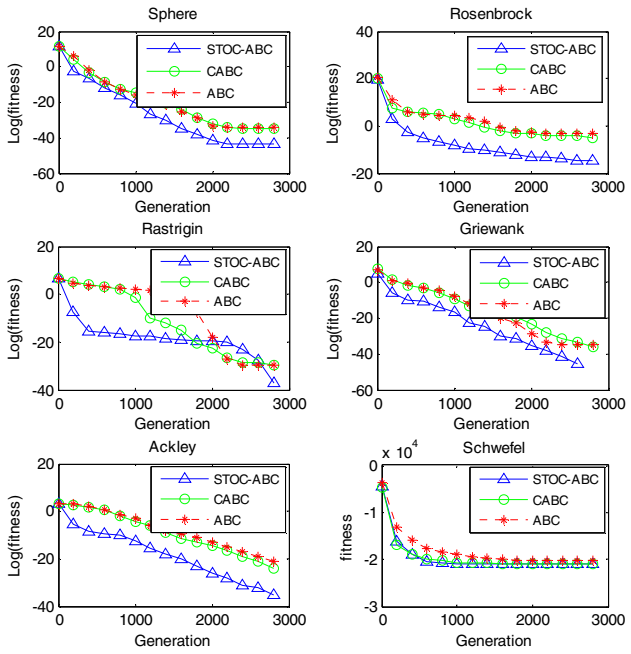

Fig.1. Convergence process of the different ABCs on test functions

As it is seen in Table III, the CABC and STOC-ABC have equal best fitness values on Rastrigin, Griewank function. The performance of CABC is slightly better than that of STOC-ABC on Schwefel function, the reason is that the minimum value of Schwefel is at the boundaries position. And, the boundary number is many and the middle number is small, which are produced by Logistic chaos sequence, however, Tent chaos sequence is uniformly distributed. For the rest of the benchmark functions, the performance of STOC-ABC is better than the CABC. For all functions, the results of the STOC-ABC and CABC are better than those of the ABC. According to Fig.1, the convergence performance of the STOC-ABC is better than the CABC and ABC, except the CABC is slightly better than the STOC-ABC on Schwefel function.

In order to further confirm the effectiveness of STOC-ABC, the performances of the STOC-ABC algorithm and the CABC algorithm were tested on Sphere, Rosenbrock, Rastrigin and Griewank functions with 10, 30, 60, 100 and 200 dimensions. The maximum iteration number used for termination condition for the algorithms is tuned according to the problem dimension and 100 iterations are executed for each dimension. Other parameters settings of the algorithms are as mentioned earlier. Each of experiments was repeated run 30 times, and the mean values and standard deviations are given in Table IV, which the best results were marked as bold.

TABLE IV. COMPARISONS OF STOC-ABC AND CABC ON SPHERE, ROSENBROCK, RASTRIGIN AND GRIEWANK FUNCTIONS

Function	Algorithm	performance	Dimensions				
			10	30	60	100	200
Sphere	CABC	Mean	7.90706e-17	4.7927e-16	1.20774e-15	2.15732e-15	4.64775e-15
		Std	1.57383e-17	6.81776e-17	1.26539e-16	1.45943e-16	2.82456e-16
	STOC-ABC	Mean	**5.25350e-22**	**5.11746e-19**	**6.43667e-19**	**2.11799e-17**	**1.04723e-16**
		Std	**8.67163e-32**	**9.91044e-31**	**8.94635e-31**	**4.75701e-30**	**1.18925e-29**
Rosenbrock	CABC	Mean	7.90276e-02	2.41492e-01	1.45429e-01	2.64516e-01	**3.05366e-01**
		Std	1.55123e-02	3.57534e-01	2.20976e-01	2.38743e-01	**3.41979e-01**
	STOC-ABC	Mean	**1.85723e-02**	**2.84572e-02**	**8.69321e-02**	**1.06451e-01**	3.15635e-01
		Std	**1.08257e-02**	**2.96273e-02**	**3.75486e-02**	**9.53723e-02**	3.67654e-01
Rastrigin	CABC	Mean	**0**	**0**	7.57912e-15	7.57912e-15	7.57912e-15
		Std	**0**	**0**	2.88433e-14	2.88433e-14	2.88433e-14
	STOC-ABC	Mean	**0**	**0**	**5.28702e-17**	**3.53328e-16**	**1.45276e-15**
		Std	**0**	**0**	**6.62783e-25**	**5.57216e-19**	**2.27374e-15**
Griewank	CABC	Mean	1.80778e-05	5.18104e-17	1.51730e-16	3.10862e-16	1.09542e-15
		Std	3.82565e-06	1.08047e-16	1.85509e-16	3.47921e-16	4.12052e-16
	STOC-ABC	Mean	**1.11022e-16**	**0**	**0**	**2.52332e-18**	**2.67335e-17**
		Std	**0**	**0**	**0**	**1.57327e-26**	**1.22213e-22**

As shown in Table IV, the STOC-ABC and CABC have equal performance on Rastrigin function with 10 and 30 dimensions and the CABC is better than the STOC-ABC on the 200-dimensional Rosenbrock function. For the rest of functions and dimensionalities, the STOC-ABC is superior to the CABC. In addition, with the dimension increasing, the performances of the two algorithms are decreased, but the STOC-ABC shows good performance although dimensionalities of the functions are increased.

Summarizing the earlier statements, the ability of STOC-ABC is that it can prevent bees from falling into the local minimum, reduce evolution process significantly and convergence faster, compute with more efficiency, and

improve the searching abilities of algorithm.

V. CONCLUSION

A novel chaotic artificial bee colony algorithm based on Tent map is proposed to enhance the population diversity, and prevent the ABC plunging into local solutions. In the STOC-ABC, by combining the advantages of the Tent chaotic map and the opposition-based learning method, the chaotic opposition-based learning initialization can increase the population diversity and obtain good initial solutions. The self-adaptive Tent chaotic search is applied to help the artificial bee colony (ABC) algorithm to escape from local optimum effectively. Moreover, the tournament selection strategy in onlooker bee phase is employed to increase the ability of the algorithm to avoid premature convergence. The simulation comparison results show that the STOC-ABC not only accelerates the convergence rate and improves solution precision, but also increases the population diversity and avoids premature convergence.

For future work, the STOC-ABC will be performed for solving different real optimization problems and hybrid search strategies based on swarm-based methods such as the PSO, ABC, bacterial foraging optimization and fruit fly optimization.

ACKNOWLEDGMENT

This work is partially supported by National Natural Science Foundation of China under Grant Nos. 61373063, 61233011, Science and Technology Department of Hunan Province of China under Grant 2012SK4046, 2012FJ3005 and 2013FJ4217, Research Foundation of Education Bureau of Hunan Province of China under Grant 13C086.

REFERENCES

[1] K. S. Tang, K.F. Man, S. Kwong, Q. He, "Genetic algorithms and their applications," IEEE Signal Process. Mag., vol. 13, no. 6, pp. 22-37, Nov. 1996.

[2] J. Kennedy, R. Eberhart, "Particle swarm optimization," in: Proc. IEEE Congr. Evol. Comput. Australia, vol. 4, pp. 1942–1948, Nov. 1995.

[3] M. Dorigo, L.M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," IEEE Trans. Evol. Comput., vol. 1, no. 1, pp. 53-66, Jan. 1997.

[4] D. Karaboga, "An idea based on honeybee swarm for numerical optimization," Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department, 2005.

[5] D. Karaboga, B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," J. Global Optim., vol. 39, pp. 459-471, Apr. 2007.

[6] D. Karaboga, B. Gorkemli, C. Ozturk, N. Karaboga, "A comprehensive survey: artificial bee colony (ABC) algorithm and applications," Artif. Intell. Rev., DOI: 10.1007/s10462-012-9328-0, Mar. 2012.

[7] D. Karaboga, B. Basturk, "A comparative study of artificial bee colony algorithm," Appl. Math. Comput., vol. 214, pp. 108-132, Aug. 2009.

[8] A. Singh, "An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem," Appl. Soft Comput., vol. 9, pp. 625-631, Mar. 2009.

[9] F. Kang, J.J. Li, Q. Xu, "Structural inverse analysis by hybrid simplex artificial bee colony algorithms," Comput. Struct., vol. 87, pp. 861-870, Jul. 2009.

[10] L. Samrat, S. Udgata, A. Abraham, "Artificial bee colony algorithm for small signal model parameter extraction of MESFET," Eng. Appl. Artif. Intell., vol. 11, pp. 1573–2916, Aug. 2010.

[11] C. Zhang, D. Ouyang, J. Ning, "An artificial bee colony approach for clustering," Expert Syst. Appl. vol. 37, pp. 4761-4767, Jul. 2010.

[12] B. Akay, D. Karaboga, "A modified artificial bee colony algorithm for real parameter optimization," Inform. Sciences, vol. 192, pp. 120-142, Jun. 2012.

[13] D. Karaboga, B. Akay, "A modified artificial bee colony (ABC) algorithm for constrained optimization problems," Appl. Soft Comput., vol. 11, no. 3, pp. 3021-3031, Apr. 2011.

[14] B. Alatas, "Chaotic bee colony algorithms for global numerical optimization," Expert Syst. Appl., vol. 37, pp. 5682-5687, Aug. 2010.

[15] C. Xu, H. Duan, F. Liu, "Chaotic artificial bee colony approach to uninhabited combat air vehicle (UCAV) path planning," Aerosp. Sci. Technol., vol. 14, pp. 535-541, Dec. 2010.

[16] W. F. Gao, S.Y. Liu, "A modified artificial bee colony algorithm," Comput. Oper. Res., vol. 39, pp. 687–697, Mar. 2012.

[17] L. S. Coelho, V. C. Mariani, "Use of chaotic sequences in a biologically inspired algorithm for engineering design optimization," Expert Syst. Appl., vol. 34, pp. 1905-1913, Apr. 2008.

[18] L. Shan, H. Qiang, J. Li, et al. "Chaotic optimization algorithm based on Tent map," Control and Decision, vol. 20, no. 2, pp. 179-182, Feb. 2005.

[19] S. Rahnamayan, H. R. Tizhoosh, M. M. A. Salama, "Opposition-based differential evolution," IEEE Trans. Evol. Comput., vol. 12, pp. 64-79, Feb. 2008.

[20] T. Blickle, and L. Thiele, "A comparison of selection schemes used in genetic algorithms (2th Edition). TIK Report No. 11, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH) Zürich, Switzerland, 1995