Research Article

A Multistrategy Optimization Improved Artificial Bee Colony Algorithm

Wen Liu^{1,2}

¹ The School of Computer Science and Technology, Dalian University of Technology, Dalian, China

² Department of Electrical Engineering, Xinjiang Institute of Engineering, Tianjin Road, No. 176, Urumqi 830011, China

Correspondence should be addressed to Wen Liu; liuwen_lw@126.com

Received 24 January 2014; Accepted 27 February 2014; Published 3 April 2014

Academic Editors: X. Meng, Z. Zhou, and X. Zhu

Copyright © 2014 Wen Liu. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Being prone to the shortcomings of premature and slow convergence rate of artificial bee colony algorithm, an improved algorithm was proposed. Chaotic reverse learning strategies were used to initialize swarm in order to improve the global search ability of the algorithm and keep the diversity of the algorithm; the similarity degree of individuals of the population was used to characterize the diversity of population; population diversity measure was set as an indicator to dynamically and adaptively adjust the nectar position; the premature and local convergence were avoided effectively; dual population search mechanism was introduced to the search stage of algorithm; the parallel search of dual population considerably improved the convergence rate. Through simulation experiments of 10 standard testing functions and compared with other algorithms, the results showed that the improved algorithm had faster convergence rate and the capacity of jumping out of local optimum faster.

1. Introduction

The artificial bee colony algorithm is a new heuristic optimization algorithm proposed in recent years by Karaboga [1]. References [2, 3] pointed out that by comparing the performance of optimization of differential evolution algorithm [4] and the particle swarm algorithm [5], ABC algorithm obtained more favorable test results and is one of the most outstanding function optimization methods, which has become a hot topic at the forefront of domestic and foreign heuristic algorithm researches.

However, similar to other intelligent algorithms, the standard ABC algorithm also had disadvantages of easily prematurely falling into local optima and slow convergence rate in later stage. In this regard, a number of scholars made corresponding improvement. Reference [6] proposed an artificial bee colony algorithm solving the problem of minimum spanning tree; [7] proposed an improved algorithm where combining particle swarm algorithm, to some extent, accelerated the local convergence rate of algorithm; and [8] proposed an improved algorithm combining differential evolution algorithm and artificial bee colony algorithm,

which effectively improved the searching accuracy and the convergence rate of the algorithm to some extent. However, improved algorithms above did not effectively improve the convergence rate and avoid premature convergence problem at the same time.

In order to overcome premature to improve the convergence speed and optimal accuracy, this paper proposes a new improved artificial bee colony algorithm. First, a chaos reverse learning strategy was proposed and introduced into the initialization phase of artificial bee colony algorithm, making the initial population uniformly distributed in the search space, in order to improve the quality of population solution, thus speeding up the global convergence rate of the algorithm. Secondly, according to the idea of separate optimization and survival of the fittest, two populations was filter optimized by using dual population structure, thus the optimization process was accelerated in the case of maintaining the population diversity. The introduction of the dynamic adaptive idea to the algorithm and the improvement of nectar update formula based on the comparison of population diversity measurement values, the problem of algorithm falling into a local optimum was solved. The simulation results of

optimization of 10 standard test functions that were widely used showed that, comparing with the existing two artificial bee colony algorithms, the proposed algorithm had better optimization accuracy, convergence rate, and robustness.

2. Algorithm Description

2.1. Artificial Bee Colony Algorithm. Artificial bee colony algorithm uses simulating the mechanism of bees collecting nectar to achieve the optimization processing function. In artificial bee colony algorithm, the bees are divided into three categories, that is, employed bees, onlookers, and scouts [9]. The main task of employed bees and onlookers is to search and mine nectar, and scouts are used to search and compare nectar in order to avoid few nectar species. The location and quantity of the nectar are the solution of function optimization problem and corresponding function value. The process of searching optimal nectar is as follows: employed bees find nectar and memorize and search for new nectar in the vicinity of each nectar; at the same time employed bees release information that is proportional to the mass of marked nectar to attract onlookers. Onlookers select the appropriate marked nectar under some mechanism and search for new nectar source in the vicinity and compare with selected nectar. Select excellent quality nectar as final marked nectar, looking for the best nectar in repeated cycles. If during the process of collecting nectar, after several searches, nectar is unchanged, then corresponding employed bees are changed into scouts and they randomly search for new nectar [9–11]. The function optimization problem can be expressed as follows:

$$\min \text{fit} = \text{fit}(\mathbf{x}), \quad \mathbf{x} = (x_1, x_2, \dots, x_n) \in S, \quad S = [x_{i,L}, x_{i,H}], \quad (1)$$

where fit represents the objective function, $\mathbf{x}$ is an n -dimensional variable, and $[x_{i,L}, x_{i,H}]$ is the corresponding upper and lower bounds of the i th dimensional variable. Set the number of nectar, employed bees, and onlookers to be N in the ABC algorithm. The specific steps of ABC algorithm are as follows.

Step 1. $2N$ nectar positions are generated randomly by the following formula:

$$V_{ij} = x_{j,L} + \text{rand} \times (x_{j,H} - x_{j,L}), \quad (2)$$

where V_{ij} is the corresponding search position of i th bee in j th dimension; $x_{j,L}, x_{j,H}$ are upper and lower bounds of the j th dimensional variables; select N positions with low fitness values as the position of nectar.

Step 2. Employed bees search and update nectar in the vicinity of nectar according to the following:

$$V_{ij} = x_{ij} + r_{ij} \times (x_{ij} - x_{kj}), \quad (3)$$

where V_{ij} is the position of new nectar, x_{ij} is the j th dimensional position of nectar i , x_{kj} is j th dimensional position of randomly selected nectar k , and $k \neq i$, r_{ij} is a random number of $[-1, 1]$.

Step 3. Comparing the pros and cons of before and after nectar, replace the previous nectar, if after searching nectar is superior to previous nectar.

Step 4. According to the way of roulette and nectar information released by employed bees, onlookers select nectar, the selection probability of onlookers is as follows:

$$P_i = \frac{\text{fit}(x_i)}{\sum_{i=1}^N \text{fit}(x_i)}. \quad (4)$$

Step 5. Onlookers search for new nectar in accordance with (3), and compared with the nectar quantity searched by employed bees, Set the N position of more nectar as the position of employed bees; the rest is position of onlookers.

Step 6. If some nectar is unchanged after limit cycles, the nectar is given up, corresponding employed bees are turned into onlookers, and new nectar is randomly generated according to (2).

Step 7. Record location of best nectar source and return to Step 2 until the termination condition is met.

2.2. Particle Swarm Algorithm. Mathematical description of the particle swarm intelligence algorithm [5] is as follows. Suppose in a D -dimensional target space, N particles with potential problem solutions composed a group, where i th particle is represented as a D -dimensional vector, $X_i = [X_{i1}, X_{i2}, \dots, X_{iD}]^T$ ($i = 1, 2, \dots, N$); position of the i th particle in D -dimensional search space is X_i ; flight speed is V_i ; P_i is the personal best position searched by i th particle so far; and remember P_g is the global optimal position searched by particle swarm so far; in each iteration, particles update speed and position in accordance with

$$\begin{aligned} V_i^{t+1} &= wV_i^t + c_1r_1(P_i^t - X_i^t) + c_2r_2(P_g^t - X_i^t), \\ X_i^{t+1} &= X_i^t + V_i^{t+1}, \end{aligned} \quad (5)$$

where, $i = 1, 2, 3, \dots, N$, t is the number of iterations; w is the inertia coefficient; c_1, c_2 are learning factors and suitable c_1, c_2 can speed up convergence and not easily fall into local optimum; and r_1, r_2 are random numbers in $[0, 1]$. Particles find P_g which is the global optimal solution via constantly learning and updating [5, 12–14]. Particle swarm algorithm is applied to the positioning phase of the proposed algorithm; the main steps are as follows.

Step 1. Determine parameters: the number of particles N , the inertia factor w , and the number of iterations t .

Step 2. Randomly generate a population of N particles.

Step 3. Update velocity and position of particle using (5).

Step 4. Global optimal solution P_g is obtained by comparing and calculating the fitness function values; solution is the coordinates of unknown node.

Step 5. Determine whether the condition of the loop termination was met, if it was, record coordinates of unknown node, otherwise return to Step 3.

3. Improved Artificial Bee Colony Algorithm

3.1. Chaos Reverse Learning Strategies. Population initialization is particularly important in intelligent algorithm, because initialization quality directly affects the algorithm global convergence speed and the corresponding solution quality. Under normal circumstances, due to the lack of a priori information, random initialization is often used to generate the initial solution of algorithm. Reference [15] proposed a chaotic initialization method in the process of researching particle swarm algorithm, while [16] proposed initialized method of reverse learning. On this basis, this paper proposed a chaotic reverse learning strategy by combining these two initialization methods, and the strategy was used to initialize ABC algorithm; concrete steps are as follows.

- (1) Set maximum chaotic iteration step $K \geq 400$ and the population size $2N$. The N -S charts of chaotic phase and reverse learning phase are given in Figures 1 and 2.
- (2) Select $2N$ best fitness value particles as the initial bee swarm from $\{V(2N) \cup \text{Opl}_V(2N)\}$.

3.2. Dynamic Self-Adaptive Nectar Update Strategy. In the process of searching nectar source, employed bees often choose nectar source with more nectar quantity, but when many employed bees select the same nectar, this information amount of nectar will increase in vain, which causes too many employed bees to concentrate on one nectar source, causing blockage or stagnation. When solving the optimization problem, this will manifest premature and local convergence [17–20]. In order to solve this problem, a new dynamic self-adaptive nectar update strategy was proposed. This strategy introduced the concept of population diversity measurement and it was used to the redefinition of nectar update formula, in order to improve the algorithm search capabilities.

Reference [21] pointed out that the difference between the average particle distance and particle fitness was commonly used to indicate the population diversity. On the basis of analyzing disadvantages of this approach, the similarity degree of the individuals in population was used to characterize the population diversity, which was introduced to the updated nectar formula that is formula (3). Let individual number of ABC algorithm be $2N$, and the j th individual of i th generation bee colony $\mathbf{x}_i$ is $\mathbf{x}_{i,j} = (x_{i,j(1)}, x_{i,j(2)}, \dots, x_{i,j(n)})$, where n is the number of nectar solution dimensions; the j th individual successive dynasties nectar optimum position is $\mathbf{y}_j = (y_{j(1)}, y_{j(2)}, \dots, y_{j(n)})$. Combine individual nectar position and successive dynasties optimum position together, referred to as $\mathbf{Z}_j = (x_{i,j(1)}, x_{i,j(2)}, \dots, x_{i,j(n)})$; all individuals $\mathbf{Z}_j$ in bee colony can be composed of a matrix $\mathbf{Z}$ of $2N \times 2n$ order,

normalization process $\mathbf{Z}$, and matrix $\mathbf{Z}$ of $2N \times 2n$ order can be obtained as

$$\mathbf{Z}' = \frac{\mathbf{Z}_{uv} - \min_{1 \leq g \leq 2N, 1 \leq l \leq 2n} \mathbf{Z}_{gl}}{\max_{1 \leq g \leq 2N, 1 \leq l \leq 2n} \mathbf{Z}_{gl} - \min_{1 \leq g \leq 2N, 1 \leq l \leq 2n} \mathbf{Z}_{gl}}, \quad (6)$$

where $1 \leq u \leq 2N$, $1 \leq v \leq 2n$, and each row vector can be seen as a fuzzy set, expressed as membership degrees of the nectar current location and successive position of each component searched by j th employed bee or onlooker; any similarity degree of two $\mathbf{Z}'_u, \mathbf{Z}'_v$ can be expressed by nearness; that is,

$$L(u, v) = 1 - \frac{1}{2n} \sum_{t=1}^{2n} |\mathbf{Z}'_{u,t} - \mathbf{Z}'_{v,t}|. \quad (7)$$

Bees Diversity Measurement F can be expressed by population average nearness

$$F = \frac{2 \sum_{u=1}^{2N-1} \sum_{v=u+1}^{2N} L(u, v)}{2N(2N-1)}. \quad (8)$$

$F \leq 1$ is obtained by $0 \leq L(u, v) \leq 1$; if individuals in bee colony are identical, then the diversity is the worst; F is the maximum value 1.

In the update nectar formula of ABC algorithm, since r_{ij} is randomly generated numerical value in $[-1, 1]$, the relationship between nectar source and the diversity of employed bees and bee colony is ignored. Therefore, let r_{ij} therefore, let adjustment formula of r_{ij} be:

$$r_{ij}(k) = \frac{1}{(a - bF(k-1))}, \quad (9)$$

where $r_{ij}(k)$ is the update coefficient of k th generation bee colony, $F(k-1)$ is diversity measurement of $(k-1)$ th bee colony, and a, b are constants. Update formula of improved nectar is

$$V_{ij}(k) = x_{ij} + r_{ij}(k) \times (x_{ij} - x_{kj}). \quad (10)$$

3.3. Dual Population Search Strategy. Since the update methods of ABC and PSO Algorithm individual, as well as different optimization strategies, the effect of optimization also varies. In order to improve the population diversity of ABC algorithm and accelerate the speed of algorithm searching for optimal solution, inspired by reference [8, 22], taking into consideration the advantages of particle swarm algorithm which has a simple structure, easy to implement, few parameters, and fast algorithm convergence rate in early stage, the advantages of this algorithm and ABC algorithm are combined to propose a dual population search strategy. Main ideas of the strategy are to randomly divide the population into two groups, each group using different optimization strategies to find optimal solution. Better solution is selected as the algorithm optimal solution after comparison. The specific process is as follows.

Step 1. Initialize population behaviors using chaos reverse learning strategy mentioned above.

Step 2. Initialized population is randomly divided into two groups; one group uses improved ABC algorithm mentioned above, nectar update using formula (10); another group uses particle swarm algorithm, individual update using formula (5).

Step 3. Two kinds of populations are searching for optimal solution in accordance with their respective search strategy under algorithm termination condition. And based on the idea of survival of the fittest, respective proceeds in accordance with the respective optimal solutions are compared, and position of the better solution is recorded.

Diversity of the population is ensured by mixing two populations and two populations parallel searching; at the same time, algorithm convergence rate is improved to a large extent; the algorithm has a higher convergence rate in reasonable computational complexity.

4. Convergence Analysis

IMABC algorithm in this paper determines convergence according to methods given in the literature [23–25].

4.1. Convergence Criteria. If the result of the iteration of optimization problem $\{A, f\}$ is x_k , then the next iteration is $x_{k+1} = Q(x_k, \eta)$, of which A is solution space, f is fitness function, and η is the solution which the algorithm has found. The function

$$R_{\delta, Z^+} = \begin{cases} \{x \in A \mid f(x) < \beta + \delta\}, & \beta < |\infty|, \delta > 0; \\ \{x \in A \mid f(x) < -Z^+\}, & \beta = -\infty; \end{cases} \quad (11)$$

is defined as the optimal area; if the algorithm finds a point in R_{δ, Z^+} , then the algorithm can be considered to find the optimal algorithm or approximate optimal solution.

Condition 1. $f(Q(x, \eta)) \leq f(x)$; if $\eta \in A$, then $f(Q(x, \eta)) \leq f(\eta)$. If the algorithm satisfies this condition, it can be stated that fitness is nonincremental.

Condition 2. For any Borel subset B of A , if in the set B the Lebesgue measure $\nu[B] > 0$, then $\prod_{k=0}^{\infty} (1 - u_k[B]) = 0$. If the algorithm satisfies the condition, it can be stated that after bee colony unlimitedly searches optimization, the probability of global optimum that cannot be found is 0.

Theorem 1. Set the function f as measurable; A is a measurable subset of R^n , algorithm Q satisfies Conditions 1 and 2, and $\{x_k\}_{k=0}^{\infty}$ is the solution sequence generated by algorithm Q ; there $\lim_{k \rightarrow \infty} P(x_k \in R_{\delta, Z^+}) = 1$.

4.2. Algorithm Convergence Analysis

Lemma 2. IMABC algorithm meets Condition 1.

Proof. The algorithm uses chaotic reverse learning strategies to initialize population, double-population search is conducted in each iteration, and the optimal value is saved; that is,

$$H(P_{g,t} V_{i,t} x_i) = \begin{cases} P_{g,t} x_i, & f(V_{i,t}) \leq f(x_i), \\ V_{i,t} x_i, & f(V_{i,t}) > f(x_i). \end{cases} \quad (12)$$

Condition 1 is met. $\square$

Definition 3. Assuming optimal solution is g_{best} ; optimal solution set is defined as $G = \{s = (X) \mid f(X) = f(g_{\text{best}}), s \in S\}$.

Theorem 4 (see [13, 23–25]). In the algorithm, for bee colony state sequence $\{s(t); t \geq 0\}$, set G as a closed set in state space S .

Proof. Set $\forall s_i \in G, \forall s_j \notin G$; for any transfer step length $l, l \geq 1$, the probability P_{s_i, s_j}^l of bee colony state transferred from s_i to s_j by l steps can be obtained by bee colony algorithm

$$P_{s_i, s_j}^l = \sum_{s_1 \in S} \cdots \sum_{s_{l-1} \in S} P(T_s(s_i) = s_{r_1}) \times P(T_s(s_{r_1}) = s_{r_2}) \cdots P(T_s(s_{r_{l-1}}) = s_j), \quad (13)$$

where $P(T_s(s_{r_{c-1}}) = s_{r_c})$ is the probability of bee colony state transferred from s_{c-1} to $s_c, 1 \leq c \leq l$, and the probability is determined by transition probability of each bee; that is, $P(T_s(s_{r_{c-1}}) = s_{r_c}) = \prod_{m=1}^{SN} P(T_s(X_{im}) = X_{jm})$. SN is the number of bees. $P(T_s(s_{r_{c-1}}) = s_{r_c})$ exists in each expression in (13), since $s_{r_{c-1}} \in G, s_{r_c} \notin G, f(X_c) > f(X_{c-1}) = f(g_{\text{best}}) = \inf(f(a)), a \in A$; at least there exists $P(T_s(s_{r_{c-1}}) = s_{r_c}) = 0, P_{s_i, s_j}^l = 0$, so set G as a closed set in state space S . $\square$

Theorem 5 (see [23–25]). Bee colony state space S does not have a nonempty closed set M , making $M \cap G = \phi$.

Proof. Assume that there exists a nonempty closed set M and $M \cap G = \phi$. Set $s_i = (g_{\text{best}}, g_{\text{best}}, \dots, g_{\text{best}}) \in G, \forall s_j = (x_{j1}, x_{j2}, \dots, x_{jd}) \in M$; since $f(X_{jc}) > f(g_{\text{best}})$, after a finite number of iterations, the probability of scouter transferred from X_i to X_j is $p_{sc}(T_s(X_i) = X_j) > 0$. Therefore when the step size is large enough, in expansion P_{s_i, s_j}^l there must be a certain product expression greater than 0; that is $P(T_s(s_{r_{c-1}}) = s_{r_{c+i+1}}) > 0$. From (13) in Theorem 4, $P_{s_i, s_j}^l > 0$; M is not a closed set, which is contradicted with the question conditions. Therefore, there is no closed set in the state space S except G . $\square$

Theorem 6 (see [24]). Assume Markov Chain has a nonempty closed set E and there is no other nonempty closed set O ; let $E \cap O = \phi$. Therefore, when $j \in E, \lim_{n \rightarrow \infty} P(X_n = j) = \pi_j$ and when $j \notin E, \lim_{n \rightarrow \infty} P(X_n = j) = 0$ [2].

Theorem 7. When bee colony is unlimitedly iterated and optimized, all state sequences are present in the optimal state set

FIGURE 1: The N-S chart of chaotic phase.

FIGURE 2: The N-S chart of reverse learning phase.

G. Theorem 7 is established which can be drawn from Theorems 4–6.

Lemma 8. *Algorithm satisfies Condition 2.*

Proof. By Theorem 7, after bee colony is unlimitedly optimized, the probability of no global optimum is 0; then there is $\prod_{k=0}^{\infty} (1 - u_k[B]) = 0$. $\square$

Theorem 9. *IMABC algorithm converges to global optimum.*

As the bee colony algorithm satisfies Conditions 1 and 2, the algorithm can be obtained and converged to the global optimum by Theorem 1. The algorithm in this paper uses two optimization algorithms and the optimal solutions obtained from two populations are recorded; the two populations are independent. As long as the global optimal solution probability of one of the optimization algorithms converges to 1, the global optimal solution probability of the entire

FIGURE 3: f_1 function convergence performance comparison. FIGURE 4: f_2 function convergence performance comparison.

algorithm also converges to 1. Therefore, this algorithm is a global convergence algorithm.

5. Simulation Experiment and Results Analysis

In order to verify the validity of above analysis and the improved algorithm performance, comparison experiments were done on this improved algorithm (abbreviated as IMABC), traditional ABC algorithm, and improved integration algorithm (abbreviated as PABC) combined by ABC algorithm proposed by [7] and PSO algorithm. In the simulation experiment, 10 test functions [26–29] were selected; $f_1 \sim f_8$ are high-dimensional functions, where f_1 and f_2 are unimodal functions; $f_3 \sim f_8$ are multimodal functions; and

FIGURE 5: f_3 function convergence performance comparison. FIGURE 7: f_5 function convergence performance comparison. FIGURE 6: f_4 function convergence performance comparison. FIGURE 8: f_6 function convergence performance comparison.

f_9 and f_{10} are two-dimensional functions. Table 1 lists names, dimensions, definitions, ranges, and theory global optimal solutions of these test functions. In the experiments, population sizes of the three algorithms are all 40, K is 300, limit is 40, and the corresponding maximum number of iterations is 5000, where parameter settings of dual population strategy particle swarm algorithm are seen in [27]; that is, $c_1 = c_2 = 2$, $w = 0.4$. For each test function, every algorithm is randomly run 30 times to find the best value, the worst value, average, and standard deviation. The best and worst values reflect the solution quality; average tells the accuracy that algorithm can achieve under a given number of function evaluations, reflecting the algorithm convergence rate; variance reflects

the stability and robustness of the algorithm. Results are shown in Table 2.

As can be seen from the data comparison in Table 2, among most standard test functions, whether it is the solution quality or algorithm convergence accuracy and stability, IMABC algorithm has been greatly more improved than PABC algorithm and ABC algorithm. In functions f_1 and f_9 , although compared to PABC algorithm IMABC algorithm's minimum, the worst value, average, and variance all slightly increase, significant improvement is achieved when compared to the standard ABC algorithm. In functions f_2 , f_5 , f_6 , and f_8 , the improved algorithm is significantly better than the standard ABC algorithms and PABC algorithm in various

TABLE 1: Dimension, search space, and optimal value of test functions.

Function	Mathematical representation	Dimension (D)	Range of search (S)	Theoretical optimum $f_{\min}$
Sphere (f_1)	$f_1(x) = \sum_{i=1}^D x_i^2$	50	$[-100, 100]^D$	0
Rosenbrock (f_2)	$f_2(x) = \sum_{i=1}^D \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	50	$[-30, 30]^D$	0
Griewank (f_3)	$f_3(x) = \frac{1}{4000} \sum_{i=1}^D x_i^2 - \prod_{i=1}^D \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	50	$[-600, 600]^D$	0
Rastrigin (f_4)	$f_4(x) = \left(\sum_{i=1}^D x_i^2 - 10 \cos(2\pi x_i) + 10 \right)$	50	$[-5.12, 5.12]^D$	0
Ackley (f_5)	$f_5(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^D x_i^2}\right) - \exp\left(-0.2 \frac{1}{D} \sum_{i=1}^D \cos(2\pi x_i)\right) + 20 + e$	50	$[-30, 30]^D$	0
Levill (f_6)	$f_6(x) = \sin(3\pi x_1)^2 + (x_1 - 1)^2(1 + \sin(3\pi x_1)^2) + (x_2 - 1)^2(1 + \sin(2\pi x_2)^2)$	50	$[-10, 10]^D$	0
Schweffel's problem (f_7)	$f_7(x) = \sum_{i=1}^D x_i + \prod_{i=1}^D x_i $	50	$[-500, 500]^D$	-837.9658
Alpine (f_8)	$f_8(x) = \sum_{i=1}^D x_i \sin(x_i) + 0.1 x_i $	50	$[-100, 100]^D$	0
Cross (f_9)	$f_{10}(x) = \left \sin(x_1) \sin(x_2) e^{100 - \sqrt{x_1^2 + x_2^2}/\pi} + 1 \right $	2	$[-10, 10]^D$	0
Schaffer's F6 (f_{10})	$f_9(x) = 0.5 + \frac{\sin^2\left(\sqrt{x_1^2 + x_2^2}\right) - 0.5}{(1 + 0.001(x_1^2 + x_2^2))^2}$	2	$[-100, 100]^D$	0

TABLE 2: Results comparison of 10 test functions of 3 algorithms.

Function	Algorithm	Best	Worst	Mean	Std.
f_1	ABC	$8.85038e - 015$	$7.93808e - 005$	$2.65944e - 006$	$1.44905e - 005$
	PABC	$2.50765e - 015$	$4.22993e - 014$	$9.22438e - 015$	$8.65892e - 015$
	IMABC	$3.48561e - 017$	$8.9191e - 017$	$6.2547e - 017$	$1.33411e - 017$
f_2	ABC	5.90141	44.6327	18.4201	10.1225
	PABC	0.185485	7.19489	1.54921	1.72969
	IMABC	0.000204645	0.00921194	0.00381647	0.00285877
f_3	ABC	$4.44089e - 015$	$3.56306e - 009$	$1.32808e - 010$	$6.48804e - 010$
	PABC	$4.996e - 015$	$1.44799e - 010$	$7.14078e - 012$	$2.67773e - 011$
	IMABC	0	0	0	0
f_4	ABC	$1.20019e - 008$	11.6005	2.66884	2.94451
	PABC	$7.42517e - 013$	$1.32277e - 005$	$5.01149e - 007$	$2.40813e - 006$
	IMABC	0	0	0	0
f_5	ABC	$6.59917e - 013$	$1.04211e - 010$	$9.28022e - 012$	$1.86668e - 011$
	PABC	$4.21885e - 013$	$2.06368e - 011$	$4.90198e - 012$	$4.5731e - 012$
	IMABC	$8.88178e - 016$	$2.66454e - 015$	$2.54611e - 015$	$6.48634e - 016$
f_6	ABC	0.000957132	0.259845	0.0545307	0.0597714
	PABC	$1.18297e - 013$	$3.51652e - 009$	$5.77724e - 010$	$9.57809e - 010$
	IMABC	$6.08383e - 021$	$1.4873e - 018$	$3.51197e - 019$	$2.97612e - 019$
f_7	ABC	-837.767	-826.26	-833.611	3.18261
	PABC	-730.356	-730.356	-730.356	$3.46891e - 013$
	IMABC	-837.966	-837.966	-837.966	0
f_8	ABC	2.37327	6.15694	4.09479	1.21718
	PABC	0.0443285	0.419608	0.200788	0.142808
	IMABC	$2.99468e - 016$	$5.48062e - 016$	$4.18311e - 016$	$8.39077e - 017$
f_9	ABC	$4.85217e - 005$	$4.95895e - 005$	$4.87645e - 005$	$3.21009e - 007$
	PABC	$4.84822e - 005$	$4.86088e - 005$	$4.85033e - 005$	$3.7978e - 008$
	IMABC	$4.84822e - 005$	$4.84822e - 005$	$4.84822e - 005$	$4.03232e - 018$
f_{10}	ABC	0.000235347	0.0851231	0.0240707	0.0235249
	PABC	$8.98791e - 011$	$8.1512e - 005$	$6.98519e - 006$	$1.63762e - 005$
	IMABC	0	0	0	0

test results; especially in functions f_3 , f_4 , f_7 , and f_{10} , IMABC algorithm not only has good test results but also can converge to optimal solution, showing good searching performance.

In order to compare algorithm optimization effect more visually, IMABC algorithm, PABC algorithm, and ABC algorithm are compared. Corresponding test function convergence curves are given in Figures 3 to 12. According to the figures, because of the use of a new initialization method and dual population parallel search strategy, as well as the dynamic self-adaptation of nectar location update, the improved Artificial Bee Colony algorithm can jump out of local optimal solution and gradually converge to the global optimal solution when processing multimodal functions, and has a faster convergence rate when in processing unimodal functions and low-dimensional functions. While processing unimodal functions and low-dimensional functions, this improved algorithm has a faster convergence rate. It can be seen from Figures 3, 4, 7, and 8, since standard test function has a high complexity in 50th-dimension, three

algorithms are all unable to converge to the optimal solution, but this improved algorithm, compared to ABC algorithm and PABC algorithm, has a faster convergence rate and significantly superior convergence accuracy; it can be seen from Figures 5, 6, 9, and 12 that, compared to the other two algorithms, IMABC algorithm has higher convergence accuracy and can converge to a global optimal solution faster and stabilize. As can be seen from Figure 10, this proposed algorithm gradually approaches function optimal solution with the increase of iterations, although it cannot converge; the extent of approaching and search accuracy are significantly better than the other two algorithms. As can be seen from Figure 11, IMABC algorithm and the other two algorithms are all approaching function optimal solution with the increase of iterations. When the number of iterations is more than 60 times, this algorithm has little difference with PABC algorithm. But when the number of iterations is less than 60 times, it can be seen that IMABC algorithm has higher convergence rate. Therefore, it can be concluded that

FIGURE 9: f_7 function convergence performance comparison. FIGURE 11: f_9 function convergence performance comparison. FIGURE 10: f_8 function convergence performance comparison.

the overall optimization performance of this proposed IMABC algorithm is superior to the standard ABC algorithm and the PABC improved algorithm proposed in [7].

6. Conclusion

In order to avoid falling into local optimum resulting from premature and improve convergence rate of ABC algorithm, an improved artificial bee colony algorithm based on multipolicy optimization was proposed. In order to improve the global search ability and keep the algorithm diversity, improved algorithm proposed a chaotic reverse learning initialization method on the basis of existing research results; in order to avoid the algorithm falling into a local optimum,

FIGURE 12: f_{10} function convergence performance comparison.

improved algorithm introduced dual population search mechanism into search phase of the algorithm. Advantages of particle swarm algorithm and standard ABC algorithm were merged; meanwhile, algorithm convergence rate was increased. In addition, in order to improve the algorithm population diversity and global search capability, the concept of population similarity degree was introduced into the improved algorithm, and an indicator of population diversity measure was proposed to dynamic self-adaptive adjustment of the nectar location. Experimental results of 10 standard test functions optimization showed that this proposed algorithm improved more greatly than standard ABC algorithm and PABC in optimization efficiency, optimization performance, and robustness.

Furthermore, this improved algorithm also has certain limitations: though optimization performance is improved,

the algorithm complexity is increased to a certain extent. How to ensure algorithm jumping out of local optima and having high convergence rate, at the same time, possessing low algorithm complexity, will be the next step in research.

Conflict of Interests

The author declares that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work was supported by the Scientific Research Program of the Higher Education Institution of Xinjiang (no. XJEDU2010S48).

References

- [1] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Tech. Rep., Erciyes University, Engineering Faculty, Computer Engineering Department, Kayseri, Turkey, 2005.
- [2] D. Karaboga and B. Basturk, "Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems," in *Foundations of Fuzzy Logic and Soft Computing*, vol. 4529 of *Lecture Notes in Computer Science*, pp. 789–798, Springer, Berlin, Germany, 2007.
- [3] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Applied Soft Computing*, vol. 8, no. 1, pp. 687–697, 2008.
- [4] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
- [5] A. H. Anderson, "A comparison of two privacy policy languages: EPAL and XACML," in *Proceedings of the 3rd ACM Workshop on Secure Web Services (SWS '06)*, pp. 53–60, ACM, 2006.
- [6] A. Singh, "An artificial bee colony algorithm for the leaf-constrained minimum spanning tree problem," *Applied Soft Computing Journal*, vol. 9, no. 2, pp. 625–631, 2009.
- [7] G. P. Zhu and S. Kwong, "Gbest-guided artificial bee colony algorithm for numerical function optimization," *Applied Mathematics and Computation*, vol. 217, no. 7, pp. 3166–3173, 2010.
- [8] A. Abraham, R. K. Jathoth, and A. Rajasekhar, "Hybrid differential artificial bee colony algorithm," *Journal of Computational and Theoretical Nanoscience*, vol. 9, no. 2, pp. 1–9, 2012.
- [9] N. Karaboga, "A new design method based on artificial bee colony algorithm for digital IIR filters," *Journal of the Franklin Institute*, vol. 346, no. 4, pp. 328–348, 2009.
- [10] L. Wang, G. Zhou, and Y. Xu, "An artificial bee colony algorithm for solving hybrid flow-shop scheduling problem with unrelated parallel machines," *Control Theory & Applications*, vol. 29, no. 12, pp. 1551–1556, 2012.
- [11] J.-Q. Li, Q.-K. Pan, and K.-Z. Gao, "Pareto-based discrete artificial bee colony algorithm for multi-objective flexible job shop scheduling problems," *International Journal of Advanced Manufacturing Technology*, vol. 55, no. 9–12, pp. 1159–1169, 2011.
- [12] Q. Li, J. Gong, and J.-F. Tang, "Multi-objective particle swarm optimization algorithm for cross-training programming," *Control Theory & Applications*, vol. 30, no. 1, pp. 18–22, 2013.
- [13] Z.-H. Zhan, J. Zhang, Y. Li, and H. S.-H. Chung, "Adaptive particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 6, pp. 1362–1381, 2009.
- [14] C. W. Jiang and B. Etorre, "A hybrid method of chaotic particle swarm optimization and linear interior for reactive power optimisation," *Mathematics and Computers in Simulation*, vol. 68, no. 1, pp. 57–65, 2005.
- [15] B. Liu, L. Wang, Y.-H. Jin, F. Tang, and D.-X. Huang, "Improved particle swarm optimization combined with chaos," *Chaos, Solitons and Fractals*, vol. 25, no. 5, pp. 1261–1271, 2005.
- [16] R. S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 64–79, 2008.
- [17] G. P. Zhu and S. Kwong, "Gbest-guided artificial bee colony algorithm for numerical function optimization," *Applied Mathematics and Computation*, vol. 217, no. 7, pp. 3166–3173, 2010.
- [18] Y. Liu and L. Ma, "Bees algorithm for function optimization," *Control and Decision*, vol. 27, no. 6, pp. 886–889, 2012.
- [19] W. F. Gao and S. Y. Liu, "Improved artificial bee colony algorithm for global optimization," *Information Processing Letters*, vol. 111, no. 17, pp. 871–882, 2011.
- [20] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.
- [21] D.-X. Zhang, Z.-H. Guan, and X.-Z. Liu, "Adaptive particle swarm optimization algorithm with dynamically changing inertia weight," *Control and Decision*, vol. 23, no. 11, pp. 1253–1257, 2008.
- [22] B. Alatas, "Chaotic bee colony algorithms for global numerical optimization," *Expert Systems with Applications*, vol. 37, no. 8, pp. 5682–5687, 2010.
- [23] J. C. Zeng, J. Jie, and Z. H. Cui, *Particle Swarm Algorithm*, Science Press, Beijing, China, 2004.
- [24] J.-P. Luo, X. Li, and M.-R. Chen, "The Markov model of shuffled frog leaping algorithm and its convergence analysis," *Acta Electronica Sinica*, vol. 38, no. 12, pp. 2875–2880, 2010.
- [25] H. Zhang, H. Wang, and Z. Hu, "Analysis of particle swarm optimization algorithm global convergence method," *Computer Engineering and Applications*, vol. 47, no. 34, pp. 61–63, 2011.
- [26] X. Yao, Y. Liu, and G. M. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.
- [27] J. Kennedy and R. Mendes, "Population structure and particles swarm performance," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1671–1676, Honolulu, Hawaii, USA, 2002.
- [28] F. Kang, J. J. Li, and Z. Y. Ma, "Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions," *Information Sciences*, vol. 181, no. 16, pp. 3508–3531, 2011.
- [29] Q.-K. Pan, M. F. Tasgetiren, P. N. Suganthan, and T. J. Chua, "A discrete artificial bee colony algorithm for the lot-streaming flow shop scheduling problem," *Information Sciences*, vol. 181, no. 12, pp. 2455–2468, 2011.