2013 IEEE Congress on Evolutionary Computation
June 20-23, Cancún, México

Accelerating Artificial Bee Colony Algorithm by Using An External Archive

Hui Wang[1,2], *Member, IEEE*, Zhijian Wu[1], Xinyu Zhou[1]
[1]State Key Laboratory of Software Engineering
Wuhan University
Wuhan 430072, China
[2]School of Information Engineering
Nanchang Institute of Technology
Nanchang 330099, China
huiwang@whu.edu.cn

Shahryar Rahnamayan, *Senior Member, IEEE*
Department of Electrical, Computer
and Software Engineering
University of Ontario Institute of Technology
2000 Simcoe Street North, Oshawa, ON L1H 7K4, Canada
shahryar.rahnamayan@uoit.ca

***Abstract*—Artificial bee colony (ABC) is a new optimization technique which has shown to be competitive with some well-known evolutionary algorithms. However, ABC is good at exploration but poor at exploitation. Inspired by JADE (adaptive differential evolution with optional external archive), this paper proposes an improved ABC (IABC) algorithm with an external archive, which stores some best solutions during the search process to guide the search of ABC. Experiments are conducted on several benchmark functions. Computational results show that our approach achieves promising performance in terms of solution accuracy and convergence speed.**

***Keywords*—Artificial bee colony (ABC), swarm intelligence, external archive, global optimization.**

I. Introduction

Artificial bee colony (ABC) algorithm was proposed by Karaboga in 2005 [1], which simulates the foraging behavior of honey bees. Like other swarm intelligence algorithms, ABC also starts with some initial solutions (initial population), and tries to improve them toward optimal solution(s). A recent study has shown that ABC performs significantly better, or at least comparable to other swarm intelligence algorithms [2]. Due to ABC's simple concept, easy implementation yet effectiveness, it has been applied to various optimization areas.

The performance of ABC highly depends on its solution search strategy. Under this strategy, a new candidate solution (for both employed and onlooker bees) is generated by moving its parent solution towards another solution selected randomly from the population. In fact, according to the probability theory, 50% of the time the randomly selected solution is a bad one. So, the new candidate solution is not promising to be a solution better than its parent [3]. To overcome this drawback, some improved solution search strategies have been designed [3–6]. In [3], Zhu and Kwong proposed an improved ABC algorithm called *gbest*-guided ABC (GABC) algorithm, which modified the solution search strategy by incorporating the information of the global best (*gbest*) solution to improve the exploitation. Experimental results show that GABC outperforms the original ABC on majority of test functions. Akay and Karaboga [4] modified the original ABC algorithm by employing two new solution search strategies, including frequency and magnitude of the perturbation. The original ABC modified only one dimension for producing a new solution, while the modified ABC introduced a control parameter that determines how many dimensions to be modified. Results show that the original ABC can effectively solve basic functions, while the modified ABC achieves promising solutions on hybrid complex functions. In [5], [6], Gao and Liu proposed two modified ABC algorithms inspired by the mutation scheme of the Differential Evolution (DE).

In this paper, we also propose an improved ABC (IABC) algorithm by employing a new solution search strategy. The new approach is inspired by JADE [7], in which an external archive is utilized to guide the search of ABC. Experiments are conducted on twelve benchmark functions. Simulation results show that IABC outperforms the original ABC and GABC on the majority of test cases.

The rest of the paper is organized as follows. Section 2 briefly introduces the original ABC algorithm. In Section 3, we describe the proposed approach. Experimental results and discussions are presented in Section 4. Finally, the work is concluded and summarized in Section 5.

II. Artificial Bee Colony

The ABC algorithm is a population-based stochastic algorithm that starts with an initial population of randomly generated bees. The bees are categorized into three groups: employed bees, onlooker bees, and scout bees. The employed bees search their food sources and share the information about these food sources to recruit the onlooker bees. The onlooker bees make a decision to choose a food source from those found by the employed bees, and then further search the food around the selected food source. The food source that has more nectar amount (fitness value) has a higher probability to be selected by the onlooker bees than one of less nectar. The scout bees are translated from a few employed bees, which abandon their food sources and randomly search new ones [3].

For a search problem in a D-dimensional space, the position of a food source represents a potential solution. The nectar amount of a food source is the fitness value of the associate solution. Each food source is exploited by only one employed bee. The number of employed bees or the onlooker bees is equal to the number of solutions in the population.

Let $X_i = x_{i,1}, x_{i,2}, \dots, x_{i,D}$ be the ith food source (solution) in the population, where D is the problem dimension size. Each employed bee generates a new food source V_i around the neighborhood of its parent position as follows:

$$v_{i,j} = x_{i,j} + \phi_{i,j}(x_{i,j} - x_{k,j}), \quad (1)$$

where $i = 1, 2, \dots, SN$, SN is the population size, X_k is a randomly selected solution in the current population ($k \neq i$), $j \in \{1, 2, \dots, D\}$ is a random index, and $\phi_{i,j}$ is a uniformly distributed random number in the range $[-1, 1]$. If the new food source V_i is better than its parent X_i, then V_i replaces X_i.

After all employed bees complete their searches according to Eq. 1, they share their information (nectar amounts and positions of food sources) with the onlooker bees. An onlooker bee chooses a food source according to the probability p_i related to its nectar amount (fitness ratio).

$$p_i = \frac{f_i}{\sum_{i=1}^{SN} f_i}, \quad (2)$$

where f_i is the fitness value of the ith solution in the population. As seen, the probability p_i is proportional to the fitness ratio. The better a food source is, the higher chance to be selected.

If a food source cannot be improved further over a predefined number of cycles, the food source is considered to be abandoned. The value of the predefined number of cycles is another control parameter, called $limit$. Assume that the abandoned source is X_i, then the scout bee randomly searches a new food source to be replaced with X_i. This operation is defined as follows:

$$x_{i,j} = x_j^{min} + rand(0,1)(x_j^{max} - x_j^{min}), \quad (3)$$

where $rand(0,1)$ is a uniformly distributed random number in the range $[0, 1]$, and $[x_j^{min}, x_j^{max}]$ is the boundary box-constraint for the jth variable.

III. Proposed Algorithm: ABC with An External Archive

The original ABC updates only one dimension for producing a new solution. Then, the offspring is similar to their corresponding parent solutions. As a result, the ABC shows slow convergence rate. To tackle this problem, some new solution search strategies have been proposed by utilizing the search information of the global best solution [3], [6], [8]. In [3], Zhu and Kwong proposed a *gbest*-guided ABC (GABC) algorithm inspired by particle swarm optimization (PSO). In GABC, the global best solution (*gbest*) is used to guide the search during searching the solution.

There are several DE mutation schemes, such as DE/rand/k, DE/best/k, and DE/current-to-best/k, where k is equal to 1 or 2. Compared to DE/rand/k, DE/best/k and DE/current-to-best/k benefit from their fast convergence by incorporating best solution information during the search process. However, the best solution information may also cause problems such as premature convergence due to quick losing of population diversity. Based the above analysis, Zhang and Sanderson [7] proposed a new mutation strategy, called DE/current-to-$p$best with external archive (JADE).

Algorithm 1: ABC with External Archive (IABC)

```
1  Randomly generate $SN$ solutions (positions of food sources) as an initial
   population $\{X_i | i = 1, 2, \dots, SN\}$;
2  Calculate the function value $f(X_i)$ of each solution in the population, and
   FEs = $SN$;
3  Add the current $X_{best}$ to the archive A.
4  Initialize $trial_i = 0$ for each solution $X_i$;
5  while FEs ≤ MAX_FEs do
       /* Produce new food sources (solutions) for
          employed bees                                    */
6      for $i = 1$ to $SN$ do
7          Generate a new solution $V_i$ according to Eq. 4;
8          Calculate the function value $f(V_i)$, and FEs = FEs + 1;
9          if $f(V_i) < f(X_i)$ then
10             Replace $X_i$ with $V_i$;
11             $trial_i = 0$;
12         end
13         else
14             $trial_i = trial_i + 1$;
15         end
16     end
17     Calculate the probability $p_i$ according to Eq. 2;
       /* Produce new food sources (solutions) for
          onlooker bees                                    */
18     $i = 1$;
19     $t = 1$;
20     while $t \leq SN$ do
21         if $rand(0,1) < p_i$ then
22             Generate a new solution $V_i$ according to Eq. 4;
23             Calculate the function value $f(V_i)$, and FEs = FEs + 1;
24             If$f(V_i) < f(X_i)$
25             Replace $X_i$ with $V_i$;
26             $trial_i = 0$;
27             else
28                 $trial_i = trial_i + 1$;
29             end
30             $t = t + 1$
31         end
32     end
       /* Produce new food sources (solutions) for Scout
          bees                                             */
33     if The maximum value of $trial_i$ is larger than $limit$ then
34         Generate a random solution according to Eq. 3, and replace $X_i$ with
           the random one;
35         FEs = FEs + 1;
36     end
37     Update the best solution $X_{best}$ found so far;
38     if $X_{best}$ is improved then
39         if The archive size is smaller than $m$ then
40             Add $X_{best}$ to the archive A;
41         end
42         else
43             Randomly remove a solution from the archive A;
44             Add $X_{best}$ to the archive A;
45         end
46     end
47 end
```

Inspired by JADE, this paper proposes an improved ABC algorithm with an external archive (IABC). In IABC, an external archive is constructed by storing some best solutions to guide the search of bees. Denote **A** as the set of an external archive. The new solution search strategy is defined as follows:

$$v_{i,j} = \check{x}_{best,j} + \phi_{i,j}(x_{i,j} - x_{k,j}), \quad (4)$$

where $\check{x}_{best,j}$ is the jth element of a solution randomly selected from **A**, X_k is a randomly selected solution in the current population ($k \neq i$), $j \in \{1, 2, \dots, D\}$ is a random index, and $\phi_{i,j}$ is a uniformly distributed random number in the range $[-1, 1]$.

The archive operation is simple to avoid significant computation overhead. Initially, the archive is set to be entry. After population initialization, the current best solution X_{best} is added to the archive, and the archive size is 1. After each

TABLE I. BENCHMARK FUNCTIONS USED IN THE EXPERIMENTS, WHERE D IS THE DIMENSION OF THE FUNCTIONS

Name	Function	Search range	Global optimum
Sphere	$f_1(x) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]$	0
Schwefel 2.22	$f_2(x) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} x_i$	$[-10, 10]$	0
Schwefel 1.2	$f_3(x) = \sum_{i=1}^{D} (\sum_{j=1}^{i} x_j)^2$	$[-100, 100]$	0
Schwefel 2.21	$f_4(x) = \max \{\lvert x_i \rvert, 1 \le i \le D\}$	$[-100, 100]$	0
Rosenbrock	$f_5(x) = \sum_{i=1}^{n} [100(x_{i+1} - x_i^2)^2 + (1 - x_i^2)^2]$	$[-30, 30]$	0
Step	$f_6(x) = \sum_{i=1}^{D} \lfloor x_i + 0.5 \rfloor$	$[-100, 100]$	0
Quartic with noise	$f_7(x) = \sum_{i=1}^{D} i \cdot x_i^4 + random[0, 1)$	$[-1.28, 1.28]$	0
Schwefel 2.26	$f_8(x) = \sum_{i=1}^{D} -x_i \cdot \sin(\sqrt{\lvert x_i \rvert})$	$[-500, 500]$	$-418.98 \cdot D$
Rastrigin	$f_9(x) = \sum_{i=1}^{D} [x_i^2 - 10 \cos 2\pi x_i + 10]$	$[-5.12, 5.12]$	0
Ackley	$f_{10}(x) = -20 \exp(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2}) - \exp(\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi x_i)) + 20 + e$	$[-32, 32]$	0
Griewank	$f_{11}(x) = \frac{1}{4000} \sum_{i=1}^{D} (x_i)^2 - \prod_{i=1}^{D} \cos(\frac{x_i}{\sqrt{i}}) + 1$	$[-600, 600]$	0
Penalized	$f_{12}(x) = \frac{\pi}{D} \{\sum_{i=1}^{D-1} (y_i - 1)^2 [1 + \sin(\pi y_{i+1})] + (y_D - 1)^2) + (10 \sin^2(\pi y_1)\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4),\ y_i = 1 + \frac{x_i + 1}{4}$ $u(x_i, a, k, m) = \begin{cases} u(x_i, a, k, m), & x_i > a \\ 0, & -a \le x_i \le a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$	$[-50, 50]$	0

generation, if the best solution X_{best} found so far is improved, then X_{best} is added to the archive. If the archive size exceeds a predefined threshold m, then a solution is randomly removed from the archive to keep the archive size at m.

The main steps of our approach IABC are described in Algorithm 1, where $trial_i$ records the number of unchanged function values of X_i, $rand(0, 1)$ is a random number in the range $[0, 1]$, FEs is the number of function evaluations, and MAX_FEs is the maximum number of function evaluations. Compared to the original ABC algorithm, IABC only modifies the solution search strategy. So, both ABC and IABC have the same computational time complexity.

IV. EXPERIMENTAL STUDY

A. Test Functions

In order to verify the performance of IABC, twelve well-known benchmark functions are used in the experiments. All these problems should be minimized. Table I presents the brief descriptions of these benchmark functions. Specific mathematical definitions of these functions can be found in [9].

In the experiments, IABC is compared with the original ABC, and GABC [3] for $D = 30$ and $D = 100$. For the above three algorithms, the same parameter settings are used to obtain a fair competition. The population size SN and $limit$ are set to 100 and 100, respectively [2]. For GABC, the parameter C is set to 1.5 [3]. The archive size m is set to 5. The maximum number of function evaluations (MAX_FEs) is set to 1.5E+05 and 5.0E+05 for $D = 30$ and $D = 100$, respectively. Throughout the experiments, each algorithm is run 30 times per function. The mean and standard deviation results are reported.

B. Simulation Results

Table II presents the computational results of ABC, GABC, and IABC for $D = 30$, where "Mean" represents the mean function values and "Std" indicates the standard deviation. The best results among the comparison are shown in bold. As seen, IABC outperforms the original ABC on all test functions except for f_6. On this function, all three algorithms

TABLE II. RESULTS ACHIEVED BY ABC, GABC, AND IABC WHEN $D = 30$.

Fun	ABC		GABC		IABC	
	Mean	Std	Mean	Std	Mean	Std
f_1	1.14E–15	3.58E–16	4.52E–16	2.79E–16	**1.67E–35**	**6.29E–36**
f_2	1.49E–10	2.34E–10	1.43E–15	3.56E–15	**3.09E–19**	**3.84E–19**
f_3	1.05E+04	3.37E+03	**4.26E+03**	**2.17E+03**	5.54E+03	2.71E+03
f_4	4.07E+01	1.72E+01	1.16E+01	6.32E+00	**1.06E+01**	**4.26E+00**
f_5	1.28E+00	1.05E+00	**2.30E–01**	**3.72E–01**	2.36E–01	3.94E–01
f_6	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_7	1.54E–01	2.93E–01	5.63E–02	3.66E–02	**4.23E–02**	**3.02E–02**
f_8	–12490.5	5.87E+01	**–12569.5**	**3.25E–10**	**–12569.5**	**1.31E–10**
f_9	7.11E–15	2.28E–15	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_{10}	1.60E–09	4.32E–09	3.97E–14	2.83E–14	**3.61E–14**	**1.76E–14**
f_{11}	1.04E–13	3.56E–13	1.12E–16	2.53E–16	**0.00E+00**	**0.00E+00**
f_{12}	5.46E–16	3.46E–16	4.03E–16	2.39E–16	**3.02E–17**	**0.00E+00**

TABLE III. RESULTS ACHIEVED BY ABC, GABC, AND IABC WHEN $D = 100$.

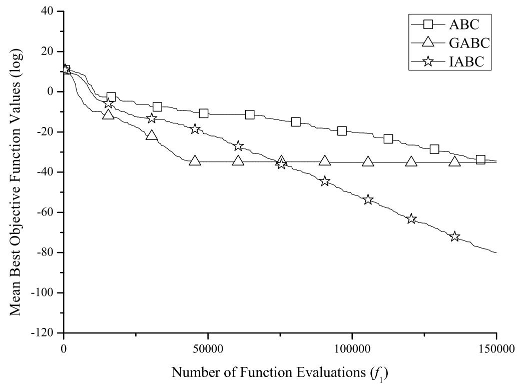
Fun	ABC		GABC		IABC	
	Mean	Std	Mean	Std	Mean	Std
f_1	7.42E–15	5.89E–15	3.37E–15	7.52E–16	**3.23E–33**	**1.45E–34**
f_2	1.09E–09	4.56E–09	6.54E–15	2.86E–15	**4.82E–18**	**3.53E–18**
f_3	1.13E+05	2.62E+04	**9.28E+04**	**2.71E+04**	9.76E+04	2.81E+04
f_4	8.91E+01	4.37E+01	8.37E+01	3.68E+01	**8.29E+01**	**1.28E+01**
f_5	3.46E+00	4.29E+00	2.08E+01	3.46E+00	**2.97E+00**	**2.72E+00**
f_6	1.58E+00	1.68E+00	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_7	1.96E+00	2.57E+00	9.70E–01	7.32E–01	**7.45E–01**	**2.27E–01**
f_8	–40947.5	7.34E+02	**–41898.3**	**5.68E–10**	**–41898.3**	**3.21E–10**
f_9	1.83E–11	2.27E–11	1.95E–14	3.53E–14	**1.42E–14**	**2.63E–14**
f_{10}	3.54E–09	7.28E–10	1.78E–13	5.39E–13	**1.50E–13**	**4.87E–13**
f_{11}	1.12E–14	9.52E–15	1.44E–15	3.42E–15	**7.78E–16**	**5.24E–16**
f_{12}	4.96E–15	3.29E–15	2.99E–15	4.37E–15	**9.05E–18**	**0.00E+00**

TABLE IV. AVERAGE RANKINGS ACHIEVED BY FRIEDMAN TEST FOR THE THREE ABC ALGORITHMS. THE BEST RANKING (WITH THE LOWEST RANKING VALUE) IS SHOWN IN BOLD.

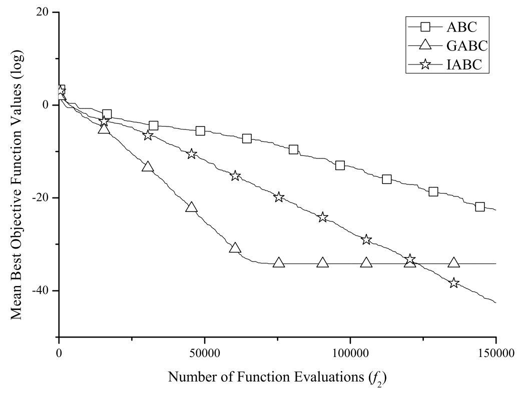
Algorithms	Rankings	
	$D = 30$	$D = 100$
IABC	**1.33**	**1.25**
GABC	1.75	1.92
ABC	2.92	2.83

TABLE V. WILCOXON TEST BETWEEN IABC AND OTHER TWO ABC ALGORITHMS. THE p-VALUES BELOW 0.05 ARE SHOWN IN BOLD.

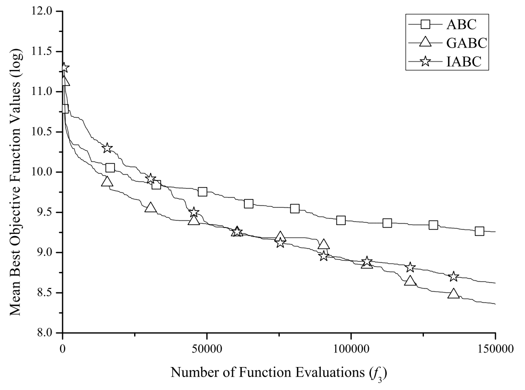
IABC vs.	p-values	
	$D = 30$	$D = 100$
ABC	**3.35E–03**	**1.21E–02**
GABC	3.74E–01	7.45E–02



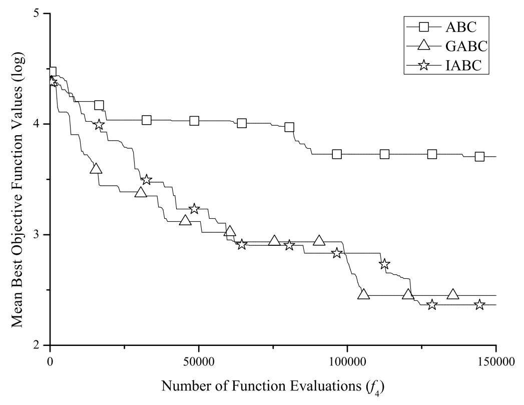
(a) Sphere (f_1)



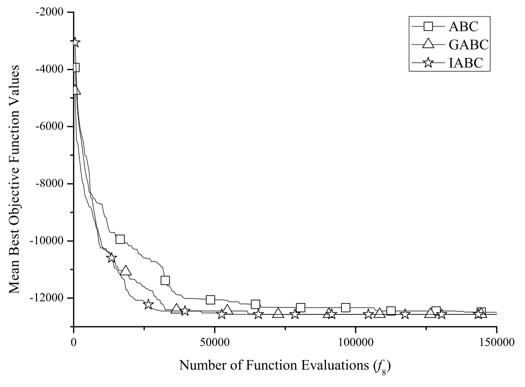
(b) Schwefel 2.22 (f_2)



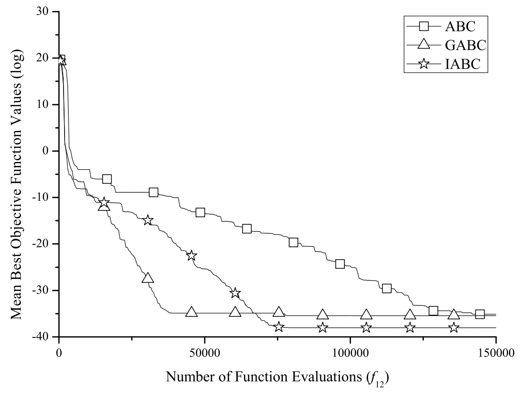
(c) Schwefel 1.2 (f_3)



(d) Schwefel 2.21 (f_4)



(e) Schwefel 2.26 (f_8)



(f) Penalized (f_{12})

Fig. 1. The convergence characteristics of ABC, GABC, and IABC on some sample functions for $D = 30$.

can find the global optimum. Especially for f_1, f_2, f_8-f_{11}, IABC significantly improves the quality of solutions. GABC performs better than IABC on f_3 and f_5. For f_3, it seems that most modified ABC variants also could not achieve reasonable solutions. The possible reason is that the rotated search space hinders the search of ABC which modifies one dimension to produce new solutions. Both GABC and IABC obtain the same results on three functions f_6, f_8, and f_9. For the rest of seven functions, IABC achieves more accurate solutions than GABC.

Fig. 1 presents the convergence performance of ABC, GABC, and IABC on selected functions for $D = 30$. As seen, IABC shows faster convergence speed than ABC. It demonstrates that the new solution search strategy is helpful to accelerate the convergence. GABC converges faster than ABC and IABC at the beginning or middle evolutionary stages, while IABC shows faster convergence speed than ABC and GABC at the last evolutionary stage (exploitation phase).

Table III lists the computational results of ABC, GABC, and IABC for $D = 100$. Like $D = 30$, all three algorithms fail to solve f_3-f_5. As the dimension increases to 100, ABC falls into local minima for f_6, while GABC and IABC still find the global optimum. Though IABC outperforms ABC and GABC on f_9 and f_{11}, its performance is seriously affected by the dimension, because it converges to the global optimum(s) when $D = 30$. For f_1, f_2, f_6, f_8, f_{10}, and f_{12}, IABC obtains similar performance when $D = 30$ and $D = 100$.

In order to compare the performance of multiple algorithms on the twelve problems, Friedman test is conducted [10]. Table IV presents the average ranking of ABC, GABC, and IABC. The best ranking (with the lowest ranking value) is shown in bold. For both $D = 30$ and $D = 100$, the performance of the three algorithms can be sorted by average ranking into the following order: IABC, GABC, and ABC. The best average ranking was obtained by the IABC algorithm, which outperforms the other two algorithms.

Table V shows the p-values of applying Wilcoxon's test between IABC and other two ABC algorithms. It aims to recognize significant differences between the behavior of two algorithms [11], [12]. The p-values below 0.05 (the significant level) are shown in bold. As shown, IABC is significantly better than ABC. Although IABC is not significantly better than GABC, IABC achieves a better ranking than GABC.

V. Conclusion

The original ABC is good at exploration but poor at exploitation. One reason is that the solution search strategy in ABC cannot generate good solutions with a high probability. In order to enhance the exploitation of ABC, an improved ABC (IABC) algorithm is proposed in this paper. The new approach IABC is inspired by JADE, in which an external archive is constructed to guide the search of bees. Every generation, the best solution found so far are added to the archive. New candidate solutions are generated by searching the neighbors of solutions randomly selected from the archive. Experiments are conducted on twelve benchmark functions. Computational results show that IABC outperforms ABC and GABC on the majority of test functions.

Although IABC shows faster convergence than ABC, it only converges faster than GABC at the last evolutionary stage on most functions. It seems that hybridization of GABC and IABC may obtain fast convergence characteristics at all search stages. For the threshold of the archive size m, an empirical value is used in the experiments. The effects of this parameter will be investigated in our future work.

Acknowledgment

This work is supported by the Foundation of State Key Laboratory of Software Engineering (No.SKLSE2012-09-19), the Science and Technology Plan Projects of Jiangxi Provincial Education Department (Nos.GJJ13744, GJJ13745, GJJ13761, and GJJ13762), and the National Natural Science Foundation of China (Nos.61165004 and 61261039).

References

[1] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," *Technical Report-TR06*, Erciyes University, Engineering Faculty, Computer engineering Department, 2005.

[2] D. Karaboga and B. Akay, "A comparative study of artificial bee colony algorithm," *Applied Mathematics and Computation*, vol. 214, pp. 108–132, 2009.

[3] G. Zhu and S. Kwong, "Gbest-guided artificial bee colony algorithm for numerical function optimization," *Applied Mathematics and Computation*, vol. 217, pp. 3166–3173, 2010.

[4] B. Akay and D. Karaboga, "A modified Artificial Bee Colony algorithm for real-parameter optimization," *Information Sciences*, vol. 192, pp. 120–142, 2012.

[5] W. Gao and S. Liu, "Improved artificial bee colony algorithm for global optimization," *Information Processing Letters*, vol. 111, pp. 871–882, 2011.

[6] W. Gao and S. Liu, "A modified artificial bee colony algorithm," *Computers & Operations Research*, vol. 39, pp. 687–697, 2012.

[7] J. Zhang and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[8] W. Gao, S.Liu, and L. Huang, "A global best artificial bee colony algorithm for global optimization," *Journal of Computational and Applied Mathematics*, vol. 236, pp. 2741–2753, 2012.

[9] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, pp. 82–102, 1999.

[10] S. García, D. Molina, M. Lozano, and F. Herrera, "A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: a case study on the CEC'2005 Special Session on Real Parameter Optimization," *Journal of Heuristics*, vol. 15, pp. 617–644, 2009.

[11] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced non-parametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Information Sciences*, vol. 180, no. 20, pp. 2044–2064, 2010.

[12] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.