A Quick Artificial Bee Colony -qABC- Algorithm for Optimization Problems

Dervis Karaboga
Department of Computer Engineering
Erciyes University
Kayseri, TURKEY
karaboga@erciyes.edu.tr

Beyza Gorkemli
Department of Computer Engineering
Erciyes University
Kayseri, TURKEY
bgorkemli@erciyes.edu.tr

***Abstract*— Artificial bee colony (ABC) algorithm is one of the most popular swarm intelligence based algorithms. ABC has been applied to solve several problems in various fields and also many researchers have attempted to improve ABC's performance by making some modifications. This works aims to model the behaviour of foragers of artificial bee colony more accurately and improve the performance of ABC algorithm in terms of local search ability. Therefore, a new definition is introduced for the onlookers of ABC. The new ABC is tested on a set of benchmark problems.**

Keywords-optimization; swarm intelligence; artificial bee colony

I. INTRODUCTION

ABC algorithm is one of the most popular swarm intelligence based algorithms. It was introduced by Karaboga in 2005 [1,2]. Since that time, ABC has been applied to solve several problems in various fields. A good survey regarding ABC applications and its modifications has been just published [3]. In order to improve its performance, several researchers attempted to implement ABC in parallel [4, 5, 6, 7, 8]. Also, concepts related to evolutionary optimization algorithms have been integrated with ABC. Inspired by PSO, Zhu and Kwong proposed an improved ABC algorithm called gbest-guided ABC by incorporating the information of global best solution into the solution search equation in [9]. Xu and Lei applied an improved version of ABC named ABC_SA, which is presented to prevent the algorithm from sliding into local optimum by introducing Metropolis acceptance criteria into ABC's searching process, to solve multiple sequence alignment problem [10]. Tuba et al. presented a novel algorithm named GABC which integrates ABC algorithm with self-adaptive guidance adjusted for engineering optimization problems in [11]. Li et al. proposed an improved ABC algorithm in which inertia weight and acceleration coefficients are introduced to improve the search process of ABC algorithm [12]. Banharnsakun et al. proposed an effective scheduling method based on best-so-far ABC for solving the JSSP, in which the solution direction is biased toward the best-so-far solution rather a neighbouring solution as proposed in the original ABC method [13]. Bi and Wang presented an improved ABC called fast mutation ABC in which a mutation strategy based on opposition-based learning is employed instead of the behaviour of scouts [14]. Inspired by DE, Gao and Liu proposed an improved solution search equation for ABC, which is based on that the bee searches only around the best solution of the previous iteration to improve the exploitation process [15].

In order to make ABC more powerful, it was combined with some traditional and evolutionary optimization algorithms, too. This type ABC is called hybridized ABC. A hybrid simplex ABC algorithm which combines Nelder-Mead simplex method with ABC was introduced and used to improve the search efficiency in computation by Kang et al. and used for inverse analysis problems by Kang et al. [16, 17]. A new hybrid algorithm, which is based on the concepts of ABC and greedy randomized adaptive search procedure, for optimally clustering n objects into k clusters is presented in [18]. Xiao and Chen introduced a hybrid ABC with artificial immune network algorithm and used this algorithm to solve multi-mode resource constrained multi-project scheduling problem [19]. Bin and Qian described a differential ABC algorithm for global numerical optimization [20]. Sharma and Pant suggested the incorporation of DE operators in the structure of basic ABC algorithm in [21]. Kang et al. presented a novel hybrid Hooke Jeeves ABC algorithm with intensification search based on the Hooke Jeeves pattern search and the ABC and demonstrated how the standard ABC can be improved by incorporating a hybridization strategy [22].

This paper's aim is not to introduce a hybrid ABC algorithm or to integrate an operator of an existing algorithm into ABC. The aim is to model the behaviour of foragers in ABC more accurately and improve its performance in terms of local search ability. Therefore, a new definition is introduced for onlookers of ABC and the improved ABC is tested on a set of benchmark problems. The rest of the paper is organized as follows; section 2 presents the standard ABC and the new strategy is described in section 3. The simulation results are demonstrated in section 4 and in section 5, the conclusion is given.

II. STANDARD ABC ALGORITHM

In ABC, the colony of artificial bees contains three groups of bees: employed bees associated with specific food sources, onlooker bees watching the dance of employed bees within the hive to choose a food source, and scout bees searching for food sources randomly. Both onlookers and scouts are also called

This work was supported by Research Fund of the Erciyes University. Project Number: FBA-11-3536

unemployed bees. Initially, all food source positions are discovered by scout bees. Thereafter, the nectar of food sources are exploited by employed bees and onlooker bees, and this continual exploitation will ultimately cause them to become exhausted. Then, the employed bee which was exploiting the exhausted food source becomes a scout bee in search of further food sources once again. In other words, the employed bee whose food source has been exhausted becomes a scout bee. In ABC, the position of a food source represents a possible solution to the problem and the nectar amount of a food source corresponds to the quality (fitness) of the associated solution. The number of employed bees is equal to the number of food sources (solutions) since each employed bee is associated with one and only one food source.

The general algorithmic structure of the ABC approach is given as follows:

Initialization Phase

REPEAT

Employed Bees Phase

Onlooker Bees Phase

Scout Bees Phase

Memorize the best solution achieved so far

UNTIL (Cycle = Maximum Cycle Number or a Maximum CPU time)

In the initialization phase, all the population of food sources are initialized with (1).

$$x_{m,i} = l_i + rand(0,1) * (u_i - l_i) \tag{1}$$

where $x_{m,i}$ is the i. dimension of the food source m, l_i and u_i are the lower and upper bounds of the parameter $x_{m,i}$, respectively.

Employed bees search for new food sources (v_m) having more nectar within the neighbourhood of the food source x_m in their memory. They find a neighbour food source and evaluate its fitness. They determine a neighbour food source by using (2).

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

where x_k is a randomly selected food source, i is a randomly chosen parameter index and $\phi_{m,i}$ is a random number within the range [-1,1]. After producing the new food source v_m, its profitability is calculated and a greedy selection is applied between v_m and x_m.

The fitness of the solution $fit(x_m)$ can be calculated using (3).

$$fit(x_m) = \begin{cases} \dfrac{1}{1+f(x_m)} & if \quad f(x_m) \geq 0 \\ 1 + abs(f(x_m)) & if \quad f(x_m) < 0 \end{cases} \tag{3}$$

where $f(x_m)$ is the objective function value of solution x_m.

Employed bees share their food source information with onlooker bees waiting in the hive. Depending on this information, then onlooker bees probabilistically choose their food sources.

The probability value p_m with which x_m is chosen by an onlooker bee can be calculated by (4).

$$p_m = \frac{fit(x_m)}{\sum_{m=1}^{SN} fit(x_m)} \tag{4}$$

After a food source for an onlooker bee is probabilistically selected, a neighbour source v_m is determined by using (2), and its fitness value is computed. As in the employed bees phase, a greedy selection is applied between v_m and x_m. Hence more onlookers are recruited to richer sources and positive feedback behaviour appears.

Employed bees whose solutions cannot be improved through a predetermined number of trials, called "limit" herein, become scouts and their solutions are abandoned. Then, the converted scouts start to search for new solutions, randomly. Hence those sources which are initially poor or have been made poor by exploitation are abandoned and negative feedback behaviour arises to balance the positive feedback.

III. A New Definition Proposed for the Search Behaviour of Onlookers

In real honey bee colonies, while an employed bee exploits the food source that she visited before, an onlooker chooses a food source region depending on dances of employed bees and she tries to find a rich food source after reaching that region and exploit it. Therefore, she examines the food sources in that region where she visits first time and chooses the fittest one. In other words, employed bees and onlookers choose food sources in a region, in a different way. However, in standard ABC algorithm, artificial employed bees and onlookers determine their new sources by using the same formula, (2). It is more logical that the way onlooker searches fittest food source after reaching a food source region should be modeled by a different formula from (2). In the following, a new definition is presented for the behavior of onlookers. The new definition proposed is :

$$v^{best}_{N_m,i} = x^{best}_{N_m,i} + \phi_{m,i}(x^{best}_{N_m,i} - x_{k,i}) \tag{5}$$

where $x_{N_m}^{best}$ is the best solution among the neighbours of x_m and itself (N_m). In order to define the neighbourhood, a similarity measure in terms of structure of solutions can be used. For different representations of a solution, different similarity measures can be defined. Therefore, this proposed formula, (5) can be used for binary optimization problems or combinatorial optimization problems, too. For instance, for a numerical optimization problem, the neighbourhood of x_m can be described depending on the mean Euclidean distance between x_m and the rest of solutions. Mean Euclidean distance for x_m , md_m , is calculated by (6).

$$md_m = \frac{\sum_{j=1}^{SN} d(m, j)}{SN - 1} \tag{6}$$

where $d(m, j)$ is the Euclidean distance between x_m and x_j .

A solution of which Euclidean distance from x_m is less than the mean Euclidean distance, md_m , can be accepted as the neighbour. It means that, an onlooker watches the dances of the employed bees in the hive and selects the region of the food source x_m . After reaching the region of x_m , examining the all food sources in N_m , she choses the best food source $x_{N_m}^{best}$ to improve. If there are S solutions in N_m which includes x_m , too, the best solution in N_m is defined by (7).

$$fit(x_{N_m}^{best}) = \max(fit(x_{N_m}^{1}),\ fit(x_{N_m}^{2}),\ ...,\ fit(x_{N_m}^{S})) \tag{7}$$

A more general definition can be written to determine a neighbour of x_m as following:

$$\begin{aligned} &\textit{if } d(m, j) \le r * md_m \textit{ then } x_j \textit{ is the neighbour of } x_m, \\ &\textit{else not} \end{aligned} \tag{8}$$

where r is called "neighbourhood radius" and $r \ge 0$. In (8), if $r = 0$, the (5) will be equal to the (2) and then the new ABC will turn to be the standard ABC since $x_{N_m}^{best}$ becomes x_m . As the value of r increases, the neighbourhood of x_m enlarges or its neighbourhood gets smaller while the value of r decreases.

IV. SIMULATION RESULTS

Performance of new ABC was tested on 4 well-known benchmark problems. Table 1 shows the test problems, bounds of the search spaces and the global optimum values for the problems. Table 2 shows the parameter settings for standard ABC and new ABC. 30 runs were carried out for each test case with random seeds. The mean values (mean) and standard deviation values (SD) obtained for the test problems are presented in Table 3. As seen from the table, both algorithms performances in terms of mean and SD of final values are similar. The convergence graphics of standard and new ABCs for test problems are shown in Fig. 1-4. For all problems, when the convergence graphics are examined it can be concluded that the new ABC is much quicker than standard ABC. In terms of cycle number, standard ABC requires usually two times more cycles than new ABC to reach the same mean value.

When (5) is used for onlookers to produce new solutions, the local convergence performance of ABC is significantly improved. In other words, ABC with this strategy converges to the optimum point of search space very quickly. Therefore, ABC employing this strategy is called "quick ABC- qABC".

TABLE I. TEST PROBLEMS

Function	Interval	Global Optimum
Sphere	[-100, 100]	$F_{\min} = 0,\ X = (0,0,...)$
Rosenbrock	[-30, 30]	$F_{\min} = 0,\ X = (1,1,...)$
Rastrigin	[-5.12, 5.12]	$F_{\min} = 0,\ X = (0,0,...)$
Griewank	[-600,600]	$F_{\min} = 0,\ X = (0,0,...)$

TABLE II. PARAMETER SETTINGS

Control Parameters	Value
ColonySize	40
MaxCycle	2000
Limit	200
r	1
Dimension	10

TABLE III. PERFORMANCE COMPARISON OF STANDARD ABC AND QABC ALGORITHMS

Function	Standard ABC		qABC	
	Mean	*SD*	*Mean*	*SD*
Sphere	9.596627E-17	1.438378E-17	1.856297E-16	9.822076E-17
Rosenbrock	0.281847	0.608359	0.223045	0.266708
Griewank	1.073216E-16	1.992911E-17	1.110223E-16	0
Rastrigin	0	0	5.921189E-17	3.188658E-16

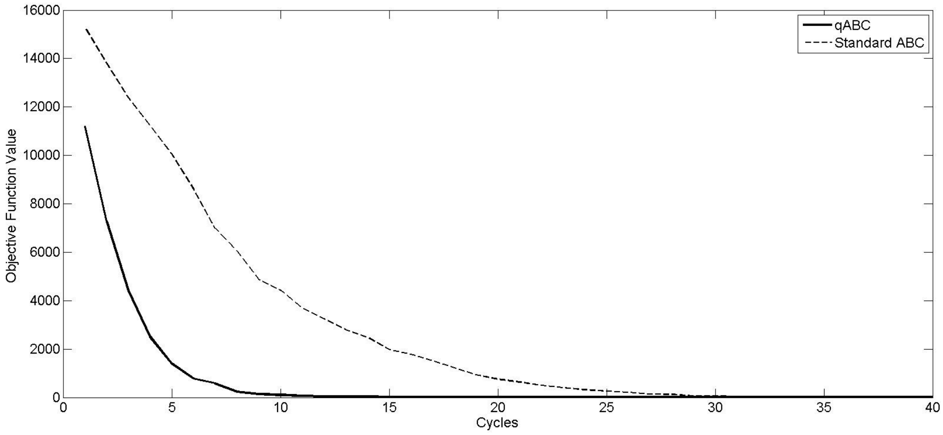

Figure 1. ABC and qABC algorithms convergence performance on Sphere function

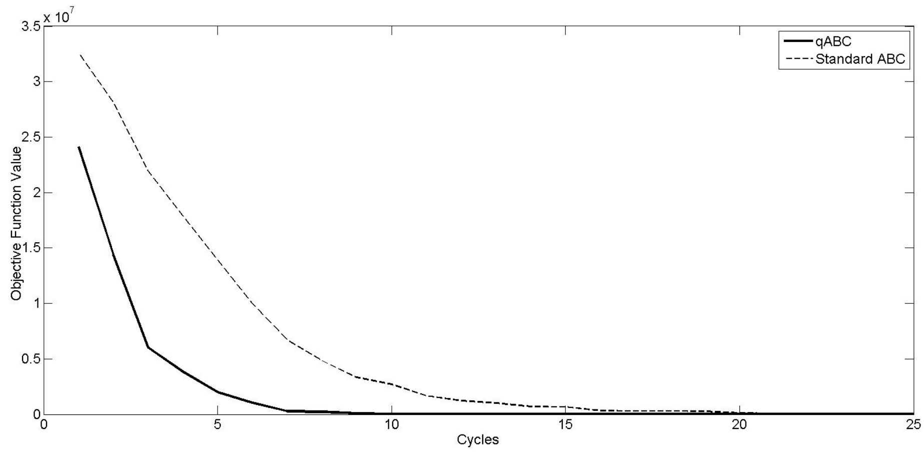

Figure 2. ABC and qABC algorithms convergence performance on Rosenbrock function

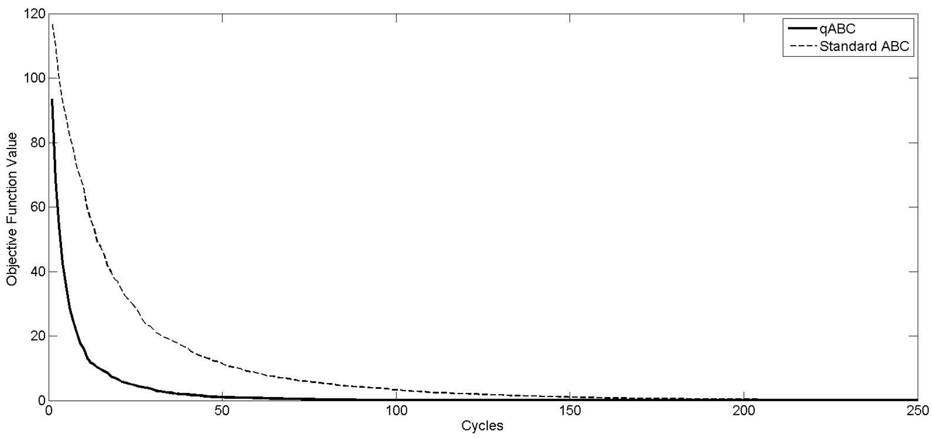

Figure 3. ABC and qABC algorithms convergence performance on Rastrigin function

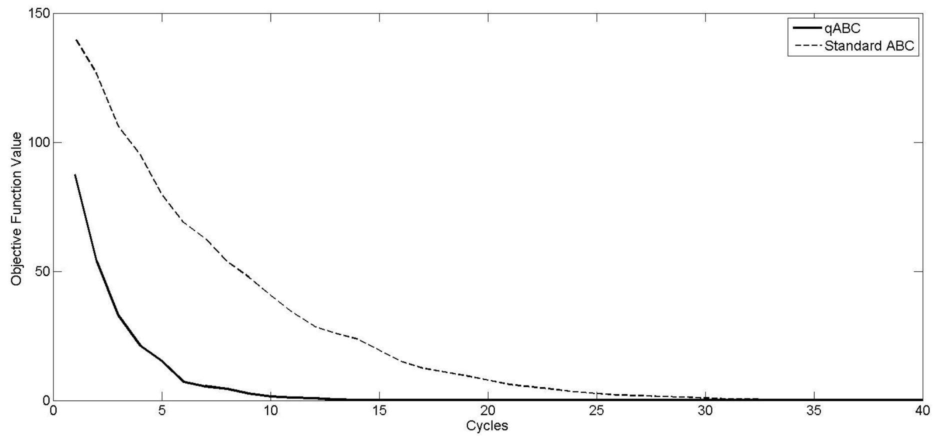

Figure 4. ABC and qABC algorithms convergence performance on Griewank function

V. CONCLUSIONS

In this paper, a new definition for the behaviour of onlooker bees of ABC algorithm was proposed. The new algorithm was called as quick ABC -qABC. qABC and standard ABC were tested on 4 well-known benchmark problems and the results obtained were compared. Simulation results showed that, the new definition significantly improves the convergence performance of ABC and the convergence of qABC is much better than standard ABC. The new definition can be used for all type optimization problems, such as binary and combinatorial optimization problems.

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, "Artificial bee colony algorithm," www.scholarpedia.org/article/Artificial_bee_colony_algorithm, Scholarpedia, vol. 5, no. 3, pp. 6915-6915, 2010.

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

[4] P.W. Tsai, J.S. Pan, B.Y. Liao, and S.C. Chu, "Enhanced artificial bee colony optimization," Int. J. Innov. Comput. Inf. Control., vol. 5, pp. 5081-5092, 2009.

[5] H. Narasimhan, "Parallel artificial bee colony (pabc) algorithm," in NaBIC 2009, world congress on nature biologically inspired computing, 2009, pp. 306–311.

[6] M. Subotic, M. Tuba, and N. Stanarevic, "Parallelization of the artificial bee colony (abc) algorithm," in 11th WSEAS international conference on nural networks and 11th WSEAS international conference on evolutionary computing and 11th WSEAS international conference on Fuzzy systems, World Scientific and Engineering Academy and Society (WSEAS). Stevens Point, Wisconsin, USA, NN'10/EC'10/FS'10, 2010, pp. 191–196.

[7] W. Zou, Y. Zhu, H. Chen, and X. Sui, "A clustering approach using cooperative artificial bee colony algorithm," Discret. Dyn. Nat. Soc., doi:10.1155/2010/459796, 2010.

[8] M. Subotic, M. Tuba, and N. Stanarevic, "Different approaches in parallelization of the artificial bee colony algorithm," Int. J. Math. Model. Method. Appl. Sci., vol. 5, no. 4, pp. 755–762, 2011.

[9] G. Zhu and S. Kwong, "Gbest-guided artificial bee colony algorithm for numerical function optimization," Appl. Math. Comput., doi:10.1016/j.amc.2010.08.049, 2010.

[10] X. Xu and X. Lei, "Multiple sequence alignment based on abc_sa," in Artificial intelligence and computational intelligence, lecture notes in computer science, vol. 6320, 2010, pp. 98–105.

[11] M. Tuba, N. Bacanin, and N. Stanarevic, "Guided artificial bee colony algorithm," in European computing conference (ECC'11), 2011, pp. 398–403.

[12] G. Li, P. Niu, and X. Xiao, "Development and investigation of efficient artificial bee colony algorithm for numerical function optimization," Appl. Soft. Comput., doi:10.1016/j.asoc.2011.08.040, 2011.

[13] A. Banharnsakun, B. Sirinaovakul, and T. Achalakul, "Job shop scheduling with the best-so-far abc," Eng. Appl. Artif. Intell., doi:10.1016/j.engappai.2011.08.003, 2011.

[14] X. Bi and Y. Wang, "An improved artificial bee colony algorithm," in 3rd international conference on computer research and development (ICCRD), vol. 2, 2011, pp. 174–177.

[15] W.F. Gao and S.Y. Liu, "A modified artificial bee colony algorithm," Comput. Oper. Res., vol. 39, no. 3, pp. 687–697, 2012.

[16] F. Kang, J. Li, and Q. Xu, "Hybrid simplex artificial bee colony algorithm and its application in material dynamic parameter back analysis of concrete dams," J. Hydraul. Eng., vol. 40, no. 6, pp. 736–742, 2009.

[17] F. Kang, J. Li, and Q. Xu, "Structural inverse analysis by hybrid simplex artificial bee colony algorithms," Comput. Struct., vol. 87, no. 13–14, pp. 861–870, 2009.

[18] Y. Marinakis, M. Marinaki, and N. Matsatsinis, "A hybrid discrete artificial bee colony—grasp algorithm for clustering," in CIE: 2009 international conference on computers and industrial engineering, vols. 1-3, 2009, pp. 548–553.

[19] R. Xiao and T. Chen, "Enhancing abc optimization with ai-net algorithm for solving project scheduling problem," in 7th international conference on natural computation (ICNC), vol. 3, 2011, pp. 1284–1288.

[20] W. Bin and C.H. Qian, "Differential artificial bee colony algorithm for global numerical optimization," J. Comput., vol. 6, no. 5, pp. 841–848, 2011.

[21] T. K. Sharma and M. Pant, "Differential operators embedded artificial bee colony algorithm," Int. J. Appl. Evol. Comput., vol. 2, no. 3, pp. 1–14, 2011.

[22] F. Kang, J. Li, Z. Ma, and H. Li, "Artificial bee colony algorithm with local search for numerical optimization," J. Softw., vol. 6, no. 3, pp. 490–497, 2011.