Applied Soft Computing 41 (2016) 362–372

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Artificial bee colony algorithm with memory

Xianneng Li, Guangfei Yang*

Faculty of Management and Economics, Dalian University of Technology, No. 2 Linggong Road, Ganjingzi, Dalian 116024, China

ARTICLE INFO

Article history:
Received 3 December 2014
Received in revised form
12 November 2015
Accepted 24 December 2015
Available online 22 January 2016

Keywords:
Artificial bee colony
Swarm intelligence
Bee memory
Foraging

ABSTRACT

Artificial bee colony algorithm (ABC) is a new type of swarm intelligence methods which imitates the foraging behavior of honeybees. Due to its simple implementation with very small number of control parameters, many efforts have been done to explore ABC research in both algorithms and applications. In this paper, a new ABC variant named ABC with memory algorithm (ABCM) is described, which imitates a memory mechanism to the artificial bees to memorize their previous successful experiences of foraging behavior. The memory mechanism is applied to guide the further foraging of the artificial bees. Essentially, ABCM is inspired by the biological study of natural honeybees, rather than most of the other ABC variants that integrate existing algorithms into ABC framework. The superiority of ABCM is analyzed on a set of benchmark problems in comparison with ABC, quick ABC and several state-of-the-art algorithms.

1. Introduction

As a relatively new optimization method inspired by swarm intelligence, artificial bee colony algorithm (ABC) [1,2] imitates the foraging behavior of honeybees to perform its search mechanism. Many studies have confirmed that ABC can achieve very competitive performance comparing with the classical evolutionary algorithms [3]. More importantly, it is very simple to implement and only consists of three control parameters to be tuned, i.e., population size (number of food sources) *SN*, maximal number of generations (terminal condition) *maxGeneration*, and exploration parameter *limit*.[1] Accordingly, researchers have successfully applied ABC to numerous applications [4], i.e., numerical function optimization [5–8], scheduling problems [9–11], clustering [12,13] and program generation [14,15], etc.

Many researchers have focused on the improvement of ABC from different perspectives. Inspired by particle swarm optimization (PSO), Zhu and Kwong [16] developed a gbest-guided ABC (GABC) to improve the search efficiency by incorporating the information of global best (gbest) solution into the original search equation of ABC. Gao and Liu [6] are influenced by differential evolution (DE) to propose a new search equation called ABC/best/1 together with a new chaotic initialization method, which improves the exploitation ability of ABC. Banharnsakun et al. [17] developed a modified search equation so that the solution direction is biased towards the best-so-far solution rather than a randomly selected neighbor one. Li et al. [18] and Xiang and An [19] followed similar concepts to combine the information of best-so-far solution to the search equation in different ways to accelerate the evolution efficiency. The newly proposed search equations are combined with the other search equations to parallel create multiple new solutions, where a greedy selection is applied to select the best one. They argued that the search efficiency can be improved significantly, however, which actually tends to perform an unfair comparison since they require larger number of fitness evaluations than ABC within a predefined *maxGeneration*. Das et al. [8] developed an improved ABC by introducing a fitness learning mechanism with a weighted selection scheme and proximity based stimuli to balance the exploitation and exploration of ABC.

It has been recognized that most of the current ABC variants are dedicated to present new hybrid ABC algorithms or combining operators of existing algorithms into ABC, rather than modeling the natural behavior of honeybees in biology especially neuroscience. A very recent work by Karaboga and Gorkemli [20] indicated that by modeling the foraging behavior of artificial bees in a more accurate way, ABC can achieve better performance than standard ABC in terms of local search ability. They introduced a quick ABC algorithm (qABC) to imitate the real onlooker bees' behavior. That is, Euclidean distance is used to help each onlooker bee to choose the fittest food source within a restricted dancing area rather than to deterministically choose the food source. Apart from this work, there is few study on improving ABC from the perspective of real honeybees.

In this paper, we develop a new ABC variant named ABC with memory algorithm (ABCM). ABCM is inspired by the studies of natural honeybees in neuroscience [21–23] which indicate that in addition to the swarm behavior, honeybees consist of memory ability to allow them efficiently trace new nectar by associatively learning from their previous experience. According to this concept, ABCM imitates a memory mechanism to the artificial bees of ABC to memorize their previous successful experiences of foraging behavior. The memory mechanism is applied to guide the further foraging of the artificial bees, which leads to a more efficient search performance than traditional ABC without memory ability. ABCM is developed in a fairly simple way to preserve the advantages of ABC's simplicity as much as possible. Only one new control parameter related to the memory size called *M* is added in ABCM.

In addition to our work, there are two recent studies which are also dedicated to introduce the memory mechanism into ABC [24,25]. Kiran and Babalik [24] added a memory board to save the solutions whose qualities are better than the average fitness value, where they are only used in the neighbor selection of the onlooker bees to increase the exploitative tendency of ABC. Bayraktar [25] integrated the short term tabu list (STTL) of tabu search to memorize the abandoned solutions, which will be prohibited to be repeatedly generated and visited. The memory mechanisms of these two studies are designed in a solution-level to memorize the entire solutions, while ABCM memorizes the successful search experience of each parameter by the neighboring food source and updating coefficient. In other words, ABCM is designed in a

* Corresponding author. Tel.: +86 41184707917.
E-mail addresses: xianneng@dlut.edu.cn (X. Li), gfyang@dlut.edu.cn (G. Yang).

[1] Since *SN* and *maxGeneration* are the general parameters of almost all evolutionary algorithms, it can be viewed that ABC only consists of one control parameter: *limit*.

http://dx.doi.org/10.1016/j.asoc.2015.12.046
1568-4946/

parameter-level, which is conceptually different from the aforementioned studies. It can be considered as a complementary study rather than a competitor.

In what follows we will organize the paper as follows: Section 2 presents a brief introduction of ABC. Section 3 describes ABCM in details. Section 4 testifies our proposal in the numerical experiments. Finally, Section 5 concludes the paper.

2. Artificial bee colony algorithm (ABC)

Artificial bee colony algorithm (ABC) maintains a population of individuals/solutions, which are specifically named food sources. The population consists of *SN* food sources, which are evolved by three groups of artificial bees, including employed bees, onlooker bees and scout bees.

In the group of employed bees, each bee corresponds to a specific food source, which memorizes the position of its food source. The employed bees search the neighboring region of the food sources to seek better ones. Afterwards, the new food sources are updated and shared with onlooker bees. Onlooker bees work in a different way from employed bees. Exploitation is added by onlooker bees by means of roulette wheel selection. That is, each onlooker bee probabilistically selects a better food source according to its quality to process. The selected food source is evolved by the onlooker bee to search for a better position, which works similarly to the employed bees. Occasionally, a new kind of artificial bees named scout bee is sent to explore the search space. When a food source is not improved after a certain number of trials (defined by a control parameter *limit*) by employed bees and onlooker bees, this food source is considered to have a poor position that will be abandoned, where a complete new food source will be randomly generated by a scout bee to replace it. ABC iteratively sends the three groups of artificial bees to search the solution space until meeting a terminal condition, i.e., the maximal number of generations *maxGeneration*.

Originally, the number of employed bees and the number of onlooker bees are equal to the number of food sources, that is, the population size *SN*. The algorithmic description of ABC is presented in the following parts.

2.1. Initialization

ABC consists a population of food sources with size *SN*. For the numerical optimization problem, each food source consists of a *D*-dimensional parameter vector, which encodes the candidate solution, i.e., $X_i = \{x_i^1, x_i^2, \ldots, x_i^D\}$, $i = 1, 2, \ldots, SN$.

Like the other evolutionary algorithms, ABC generates an initial population of food sources randomly. To cover the search space as much as possible, the initial food sources are uniformly placed within the search space constrained by the predefined minimum and maximum parameter bounds i.e., $X_{\min} = \{x_{\min}^1, x_{\min}^2, \ldots, x_{\min}^D\}$ and $X_{\max} = \{x_{\max}^1, x_{\max}^2, \ldots, x_{\max}^D\}$. For parameter *j* in food source *i*, the initial value x_i^j is generated by

$$x_i^j = x_{\min}^j + rand(0, 1) \times (x_{\max}^j - x_{\min}^j), \tag{1}$$

where, $i = 1, 2, \ldots, SN$ and $j = 1, 2, \ldots, D$. $rand(0, 1)$ is a random value ranging in [0, 1].

2.2. Employed bees

As mentioned above, the number of employed bees are equal to the population size, that is, *SN*. Each employed bee maintains a specific food source. For each food source *i*, its employed bee performs a neighboring search to generate a new vector V_i by updating its vector X_i. Let $V_i = X_i$, the neighboring search is performed by modifying one parameter v_i^j of V_i where $j \in \{1, 2, \ldots, D\}$ is a randomly selected index. The solution search equation is described as follows

$$v_i^j = x_i^j + \phi_i^j \times (x_i^j - x_k^j). \tag{2}$$

Here, $k \in \{1, 2, \ldots, SN\}$ is a randomly selected neighboring food source that guides the vector update of V_i, where *k* should be different from *i*. ϕ_i^j is a random value ranging in $[-1, 1]$.

After obtaining V_i, it is evaluated and compared with X_i. If the quality of V_i is better than X_i, V_i will replace X_i in the population. Otherwise, X_i will be remained in the population. In other words, a greedy selection is used between V_i and X_i.

2.3. Probability calculation

After searching the space, the employed bees return to the hive and share the nectar information of their sources with onlooker bees by dancing. Onlooker bees apply roulette wheel selection to select the food sources. That is, each onlooker bee prefers a food source depending on the nectar information distributed by the employed bees. Therefore, the probability of selecting a food source should be calculated. The probability of selecting a food source *i* by an onlooker bee is denoted by p_i, which is calculated by

$$p_i = \frac{fit_i}{\sum_{j=1}^{SN} fit_j}, \tag{3}$$

where fit_i denotes the fitness value of food source *i*, which is a problem-specific value measuring the quality of the candidate solution. The probability p_i is proportional to the fit_i value, where food sources with higher fitness values will be assigned higher probability values.

For the minimization problems of numerical optimization, fit_i is calculated by

$$fit_i = \begin{cases} \dfrac{1}{1+f_i} & \text{if } f_i \geq 0, \\ 1 + |f_i| & \text{otherwise.} \end{cases} \tag{4}$$

where f_i is the objective function value of food source *i*.

2.4. Onlooker bees

Based on the calculated probabilities, each onlooker bee selects a food source to further search its neighboring area. The searching procedure of onlooker bees is the same as that of employed bees. That is, if an onlooker bee selects a food source *i*, a new vector V_i is produced by Eq. (2). X_i and V_i are compared, where the better one survives in the population.

2.5. Scout bees

It is possible that a food source can not be improved even that the employed bees and onlooker bees have visited it many times. In this case, it is considered that this food source is an abandoned food source that performs poorly in the evolution process, which should be eliminated from the population. To maintain necessary population diversity, once an abandoned food source is found, a scout bee is sent to randomly generate a completely new food source to replace the abandoned one.

To perform this procedure, each food source *i* is given a parameter named $trial_i$, which counts the number of continuously failed trials that the employed bees and onlooker bees have performed on it but cannot improve its quality. Once $trial_i$ reaches a predefined threshold *limit*, this food source *i* is considered as an abandoned food source, which will be replaced by a new food source, and its $trial_i$ will be reset to 0. Originally, ABC only allows one scout bee sent in each generation.