mathematics

MDPI

Article

A Developed Artificial Bee Colony Algorithm Based on Cloud Model

Ye Jin [1], Yuehong Sun [1,2,*] and Hongjiao Ma [1]

[1] School of Mathematical Sciences, Nanjing Normal University, Nanjing 210023, China; 160902016@stu.njnu.edu.cn (Y.J.); xuminxi525@163.com (H.M.)
[2] Jiangsu Key Laboratory for NSLSCS, Nanjing Normal University, Nanjing 210023, China
* Correspondence: 05234@njnu.edu.cn; Tel.: +86-13451825524

Received: 11 March 2018; Accepted: 10 April 2018; Published: 18 April 2018

Abstract: The Artificial Bee Colony (ABC) algorithm is a bionic intelligent optimization method. The cloud model is a kind of uncertainty conversion model between a qualitative concept $\widetilde{T}$ that is presented by nature language and its quantitative expression, which integrates probability theory and the fuzzy mathematics. A developed ABC algorithm based on cloud model is proposed to enhance accuracy of the basic ABC algorithm and avoid getting trapped into local optima by introducing a new select mechanism, replacing the onlooker bees' search formula and changing the scout bees' updating formula. Experiments on CEC15 show that the new algorithm has a faster convergence speed and higher accuracy than the basic ABC and some cloud model based ABC variants.

Keywords: artificial bee colony algorithm (ABC); cloud model; normal cloud model; Y conditional cloud generator; global optimum

1. Introduction

In recent years, the development of metaheuristics [1–3] has advanced. Many scholars have made a lot of contributions in this area. The Artificial Bee Colony algorithm (ABC) [4] is a novel swarm intelligence algorithm among the sets of metaheuristics. The ABC algorithm is an optimization algorithm which mimics the foraging behavior of the honey bee. It provides a population-based search procedure in which individuals called food positions are modified by the artificial bees with the increasing of iterations. The bee's aim is to discover the places of food sources with high nectar amount.

From 2007 to 2009, Karaboga et al. [5–7] presented a comparative study on optimizing a large set of numerical test functions. They compared the performance of the ABC algorithm with the genetic algorithm (GA), particle swarm optimization (PSO), differential evolution (DE) and evolution strategy (ES). The simulation results show that ABC can be efficiently employed to solve engineering problems with high dimensions. It can produce very good and effective results at a low computational cost by using only three control parameters (population size, maximum number of fitness evaluations, limit). Akay et al. [8] studied the parameter tuning of the ABC algorithm and investigated the effect of control parameters. Afterwards, two modified versions of the ABC algorithm were proposed successively by Akay et al. [9] and Zhang et al. [10] for efficiently solving real-parameter numerical optimization problems. Aderhold et al. [11] studied the influence of the population size about the optimization behavior of ABC and also proposed two variants of ABC which used the new position update of the artificial bees.

However, ABC was unsuccessful on some complex unimodal and multimodal functions [7]. So, some modified artificial colony algorithms were put forward in order to improve the performance of the basic ABC. Zhu et al. [12] proposed an improved ABC algorithm called gbest-guided ABC (GABC) by incorporating the information of global best solution into the solution search equation to guide the search of candidate solutions. The experimental results tested on six benchmark functions showed that GABC outperformed basic ABC. Wu et al. [13] described an improved ABC algorithm to enhance the global search ability of basic ABC. Guo et al. [14] presented a novel search strategy and the improved algorithm is called global ABC which has great advantages of convergence property and solution quality. In 2013, Yu et al. [15] proposed a modified artificial bee colony algorithm in which global best is introduced to modify the update equation of employed and onlooker bees. Simulation results on the problem of peak-to-average power ratio reduction in orthogonal frequency division multiplexing signal and multi-level image segmentation showed that the new algorithm had better performance than the basic ABC algorithm with the same computational complexity. Rajasekhar et al. [16] proposed a simple and effective variant of the ABC algorithm based on the improved self-adaptive mechanism of Rechenbergs 1/5th success rule to enhance the exploitation capabilities of the basic ABC. Yaghoobi [17] proposed an improved artificial bee colony algorithm for global numerical optimization in 2017 from three aspects: initialising the population based on chaos theory; utilising multiple searches in employee and onlooker bee phases; controlling the frequency of perturbation by a modification rate.

Multi-objective evolutionary algorithms (MOEAs) gained wide attention to solve various optimization problems in fields of science and engineering. In 2011, Zou et al. [18] presented a novel algorithm based on ABC to deal with multi-objective optimization problems. The concept of Pareto dominance was used to determine the flight direction of a bee, and the nondominated solution vectors which had been found in an external archive were maintained in the proposed algorithm. The proposed approach was highly competitive and was a viable alternative to solve multi-objective optimization problems.

The performances of Pareto-dominance based MOEAs degrade if the number of objective functions is greater than three. Amarjeet et al. [19] proposed a Fuzzy-Pareto dominance driven Artificial Bee Colony (FP-ABC) to solve the many-objective software module clustering problems (MaSMCPs) effectively and efficiently in 2017. The contribution of the article was as follows: the selection process of candidate solution was improved by fuzzy-Pareto dominance; two external archives had been integrated into the ABC algorithm to balance the convergence and diversity. A comparative study validated the supremacy of the proposed approach compared to the existing many-objective optimization algorithms.

A decomposition-based ABC algorithm [20] was also proposed to handle many-objective optimization problems (MaOPs). In the proposed algorithm, an MaOP was converted into a number of subproblems which were simultaneously optimized by a modified ABC algorithm. With the help of a set of weight vectors, a good diversity among solutions is maintained by the decomposition-based algorithm. The ABC algorithm is highly effective when solving a scalar optimization problem with a fast convergence speed. Therefore, the new algorithm can balance the convergence and diversity well. Moreover, subproblems in the proposed algorithm were handled unequally, and computational resources were dynamically allocated through specially designed onlooker bees and scout bees, which indeed contributed to performance improvements of the algorithm. The proposed algorithm could approximate a set of well-converged and properly distributed nondominated solutions for MaOPs with the high quality of solutions and the rapid running speed.

The basic ABC algorithms is often combined with other algorithms and techniques. In 2016, an additional update equation [21] for all ABC-based optimization algorithms was developed to speed up the convergence utilizing Bollinger bands [22] which is a technical analysis tool to predict maximum or minimum future stock prices. Wang et al. [23] proposed a hybridization method based on krill herd [24] and ABC (KHABC) in 2017. A neighbor food source for onlooker bees in ABC was obtained from the global optimal solutions, which was found by the KHABC algorithm. During the information exchange process, the globally best solutions were shared by the krill and bees. The effectiveness of the proposed methodology was tested for the continuous and discrete optimization.

In this paper, another technique called cloud model [25] will be embedded into the ABC algorithm. Cloud model is an uncertainty conversion model between a qualitative concept and its quantitative expression. In 1999, an uncertainty reasoning mechanism of the cloud model was presented and the cloud model theory was be expanded after that. In addition, Li successfully applied cloud model to inverted pendulum [26]. Some scholars combined cloud model with ABC because cloud model has the characteristics of stable tendentiousness and randomness [27–29].

We propose a new algorithm which inherits the excellent exploration ability of the basic ABC algorithm and the stability and randomness of the cloud model by modifying the selection mechanism of onlookers, the search formula of onlookers and scout bees' update formula. The innovation points of the new algorithm are:

1. The population becomes more diverse in the whole search process by using a different selection mechanism of onlookers, in which the worse individual will have a larger selection probability than in basic ABC;
2. Local search ability of the algorithm can be improved by applying the normal cloud generator as the search formula of onlookers to control the search of onlookers in a suitable range;
3. Historical optimal solutions can be used by Y-conditional cloud generator as scout bees's update formula to ensure that the algorithm not only jumps out of local optimum but also avoids a blind random search.

The remainder of the paper is structured as follows. Section 2 provides the description of the basic ABC algorithm, followed by the details and framework of the developed ABC algorithm based on cloud model, as shown in Section 3. Subsequently, Section 4 gives the experiment results on CEC15 of the comparison among the proposed DCABC, the basic ABC, and other ABC variants based on cloud model. Then in Section 5, the current work is summarized, and the acknowledgements are given in the end.

2. The Basic ABC Algorithm

There are three kinds of bees, namely, employed bees, onlooker bees and scout bees in the ABC algorithm. The total population number is N_s; the number of employed bees is N_e and onlookers is N_u (General define $N_e = N_u = \frac{N_s}{2}$). In the initialization phase, food sources in the population are randomly generated and assigned to employed bees as

$$X_i^j = X_{min}^j + rand(0,1)(X_{max}^j - X_{min}^j), \tag{1}$$

where $j \in \{1, 2, \ldots, D\}$, X_{max}, X_{min} are the upper and lower bounds of the solution vectors, D is the dimension of the decision variable.

Each employed bee X_i generates a new food source V_i in the neighborhood of its present position:

$$V_i^j = X_i^j + \phi_i^j(X_i^j - X_k^j), \tag{2}$$

where $j \in \{1, 2, \ldots, D\}$, $k \in \{1, 2, \ldots, N_e\}$, k must be different from i, k and j are random generating indexes, ϕ_i^j is a random number between $[-1, 1]$. At the same time, we should guarantee V_i in the field of definition domain. V_i will be compared to X_i and the employed bee exploits the better food source by greedy selection mechanism in terms of fitness value fit_i in Equation (3):

$$fit_i = \begin{cases} \dfrac{1}{1+f_i} & f_i \geq 0 \\ 1 + abs(f_i) & f_i < 0 \end{cases} \tag{3}$$

where f_i is the objective value of solution X_i or V_i. Equation (3) is used to calculate fitness values for a minimization problem, while for maximization problems the objective function can be directly used as a fitness function.

An onlooker bee evaluates the fitness value of all employed bees, and uses the roulette wheel method to select a food source X_i updated the same as employed bees according to its probability value P calculated by the following expression:

$$P = \frac{0.9 fit(X_i)}{\max\limits_{m=1}^{N_e} fit(X_m)} + 0.1, \tag{4}$$

If a food source X_i cannot be improved beyond a predetermined number (*limit*) of trial counters, it will be abandoned and the corresponding employed bee will become a scout bee randomly produced by Equation (1). The algorithm will be terminated after repeating a predefined maximum number of cycles, denoted as *Max_Cycles*. The flow chart of ABC algorithm is shown in Algorithm 1.

Algorithm 1: The basic ABC algorithm.

Initialization phase
Initialize the food sources using Equation (1).
Evaluate the fitness value of the food sources using Equation (3), set the current generation $t = 0$.
While $t \leq Max\_Cycles$ **do**
Employed bees phase
Send employed bees to produce new solutions via Equation (2).
Apply greedy selection to evaluate the new solutions.
Calculate the probability using Equation (4).
Onlooker bees phase
Send onlooker bees to produce new solutions via Equation (2).
Apply greedy selection to evaluate the new solutions.
Scout bee phase
Send one scout bee produced by Equation (1) into the search area for discovering a new food source.
Memorize the best solution found so far.
$t = t + 1$.
end while
Return the best solution.

3. A Developed Artificial Bee Colony Algorithm Based on Cloud Model (DCABC)

The ABC algorithm is a relatively new and mature swarm intelligence optimization algorithm. Compared to GA and PSO, ABC has a higher robustness [6]. More and more scholars want to improve the performance of the ABC algorithm. Zhang [27] put forward an algorithm named PABC with the new select scheme based on cloud model. For the individual with a better fitness characteristic,

the value of probability was likely relatively high, and vice versa. Lin et al. [29] proposed an improved ABC algorithm based on cloud model (cmABC) to solve the problem that the basic ABC algorithm suffered from slow convergence and easy stagnation in local optima by calculating food source through the normal cloud particle operator and reducing the radius of the local search space. In cmABC, the author also introduced a new selection strategy that made the inferior individual have more chances to be selected for maintaining diversity. In addition, the best solution found over time was used to explore a new position in the algorithm. A number of experiments on composition functions showed that the proposed algorithm had been improved in terms of convergence speed and solution quality. In this section, we propose a developed ABC algorithm named DCABC which is based on cloud model with a new choice mechanism of onlookers and new search strategies of onlooker bees and scouts.

3.1. Cloud Model

Professor Li presented an uncertainty conversion model between a qualitative concept $\widetilde{T}$ [30] presented by nature language and its quantitative expression which is called cloud model on the basis of traditional fuzzy set theory and probability statistics. He developed and improved a complete set of cloud theory [31] which consists of cloud model, virtual cloud, cloud operations, cloud transform, uncertain reasoning and so on.

Suppose U is a quantitative domain of discourse that are represented by precise values (one-dimensional, two-dimensional or multi-dimensional), and $\widetilde{T}$ is a qualitative concept in U. X is an arbitrary element in U and a random implementation of qualitative concept $\widetilde{T}$. The degree of certainty of X to $\widetilde{T}$ expressed as $\mu(X)$ is an random number that has stable tendency. The distribution of X on the domain of discourse U is called cloud model, or simply 'cloud' for short. Each pare $(X, \mu(X))$ is called a cloud droplet, and cloud model can be formulated as follows:

$$\forall X \in U \longrightarrow \mu(X) \in [0,1] \tag{5}$$

The normal cloud is scattered point cloud model based on normal distribution or half-normal distribution. Normal cloud model uses a set of independent parameters to work together in order to express digital characteristics of a qualitative concept and reflect the uncertainty of the concept. Based on the normal distribution function and membership function, this group of parameters are represented by three digital characteristics including expectation Ex, entropy En, and hyper entropy He.

Expectation Ex is a point which can best represent the qualitative concept in domain of discourse space. It can be considered as the center of gravity of all cloud drop, which can best represent the coordinates of the qualitative concept on number field. Entropy En stands for the measurable granularity of the qualitative concept. Entropy also reflects the uncertainty and fuzzy degree of the qualitative concept. Fuzzy degree means value range that can be accepted by the qualitative concept in domain of discourse. Hyper entropy He is the measure of entropy's uncertainty, namely entropy of En. It reflects randomness of samples which represent qualitative concept values, and reveals the relevance of fuzziness and randomness. Hyper entropy can also reflect the aggregation extent of cloud droplets.

Given three digital characteristics Ex, En and He, forward cloud generator in Equations (6)–(8) can produce N cloud droplets of the normal cloud model (Algorithm 2), which are two-dimensional points (x_i, μ_i) $(i \in \{1, 2, \ldots, N\})$.

$$En_i^{'} = N(En, He^2), \tag{6}$$

$$x_i = N(Ex, (En_i^{'})^2), \tag{7}$$

$$\mu_i = \exp\{-\frac{(x_i - Ex)^2}{2(En_i^{'})^2}\}, \tag{8}$$

Algorithm 2: Forward cloud generator algorithm.

Input: Ex, En, He and N.

Output: quantitative value x_i of ith cloud droplet and its degree of certainty μ_i.

Forward cloud generator

Generate a normal random number $En_i^{'}$ with expectation En and hyper entropy He by Equation (6);

Generate a normal random number x_i with expectation Ex and hyper entropy $En_i^{'}$ by Equation (7).

Drop (x_i, μ_i)

Calculate μ_i by Equation (8);

A cloud droplet(x_i, μ_i) is get.

Repeat

Repeat the above step until N cloud droplets have come into being. (Figure 1)

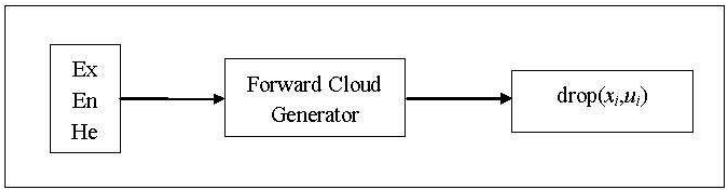

Figure 1. Forward Cloud Generator.

Given three digital characteristics (Ex, En, He) and a specific degree of certainty μ, cloud generator refers to Y-conditional cloud generator based on uncertainty reasoning of cloud model. In other words, every cloud droplet (x_i,μ) has the same degree of certainty which belongs to concept $\widetilde{T}$. The formula of Y-conditional cloud generator (Algorithm 3) is:

$$x_i = Ex \pm \sqrt{-2ln(\mu)} * En_i^{'}, \tag{9}$$

Algorithm 3: Y-conditional cloud generator algorithm.

Input: Ex, En, He, N and μ.

Output: Quantitative values x_i of ith cloud droplet and its degree of certainty μ.

Y-conditional cloud generator

Get a normal random number $En_i^{'}$ with expectation En and hyper entropy He by Equation (6);

Calculate x_i with Ex, $En_i^{'}$ and μ by Equation (9).

Drop (x_i, μ)

A Y-conditional cloud droplet (x_i, μ) is get.

Repeat

Repeat the above step until get N cloud droplets. (Figure 2)

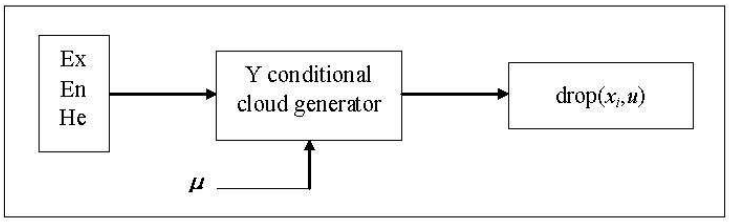

Figure 2. Y-Conditional cloud generator.

3.2. New Choice Mechanism for Onlookers

3.2.1. New Choice Mechanism

In the basic ABC algorithm, onlooker bees choose the good-quality nectars by employing the roulette wheel selection scheme. That is to say, the bigger the nectar's fitness value, the higher the probability it will be chosen by onlookers. The selection mechanism contains three parts: calculating the selection probability of each solution in population according to its fitness value; selecting the candidate solution using the roulette wheel selection method; starting the local search of onlooker bees around the candidate solution. However, the selection scheme is so greedy that it is easy to lead to the rapid decrease of population diversity and fall into local optimum. We hope to obtain a reasonable selection scheme.

Zhang et al. [27] improved the selection strategy based on cloud model with three digital characteristics Ex, En and He in Equation (10):

$$\begin{cases} Ex = \max\limits_{i=1}^{Ne} fit_i \\ En = \frac{Ex - fit_i}{12} \\ He = \frac{En}{3} \end{cases} \tag{10}$$

The possibility of the current individual which is the best can be regarded as the choice probabilities and can be produced by the positive cloud generator. Thinking differently, it will be found that the worst individual also contains useful information after several loop iterations. So, we ensure that the worst individual has larger selection probability. Equation (10) pays more attention to the inferior individuals. Detailed positive cloud generator operations can be described as follows:

$$\begin{cases} Ex = \min\limits_{i=1}^{Ne} fit_i \\ En = \frac{fit_i - Ex}{12} \\ He = \frac{En}{3} \end{cases} \tag{11}$$

The selective probability of the corresponding individual is adjusted as follows:

$$P = \exp\{-\frac{(x - Ex)^2}{2(En^{'})^2}\}, \tag{12}$$

where, $En^{'} = N(En, He^2)$, $x = N(Ex, (En^{'})^2)$, N is a normal random number generator.

We find that the individuals closer to Ex(inferior individuals) will get the higher possibility, namely, selection probability.

3.2.2. Efficiency Analysis

In our proposed algorithm DCABC, Equation (4) is used as the probability selection formula for onlookers when a random number *rand* between 0 and 1 are less than or equal to 0.5; otherwise the selection formula is set by the new choice mechanism in Equation (12). The goal of processing selection probability in two cases is to avoid the algorithm plunging into local optimum.

To test the effectiveness of the current selection mechanism, the modified and the basic ABC run independently on CEC15 [32] with dimensions(D) 10, 30 and 50, respectively. We set the initial population size $N_s = 40$. The number of employed bees equals to the number of onlookers, which is $N_e = N_u = \frac{N_s}{2}$. The value of '*limit*' equals to $N_e * D$ [33]. Every experiment is repeated 30 times. The maximum number of function evaluations ($MaxFES$) is set as $D * 10{,}000$ for all functions [34]. The simulation results is recorded in Table 1. It can be easily observed that the ABC with new choice mechanism is superior to the basic ABC on most functions. This implies that the new choice mechanism improves the performance of the basic ABC.

Table 1. Experimental Results between ABC with the new choice mechanism (NCMABC) and the basic ABC.

Functions	Criteria	ABC (10D)	NCMABC (10D)	ABC (30D)	NCMABC (30D)	ABC (50D)	NCMABC (50D)
f_1	Mean	1.36e+06	**1.11e+06**	3.73e+06	**3.48e+06**	1.20e+07	**1.05e+07**
	Std	1.12e+06	6.92e+05	1.46e+06	1.42e+06	3.69e+06	3.49e+06
	Rank	2	1	2	1	2	1
f_2	Mean	8.57e+02	**7.23e+02**	7.26e+02	**5.84e+02**	1.43e+03	**1.31e+03**
	Std	7.52e+02	8.35e+02	6.06e+02	6.77e+02	1.20e+03	1.07e+03
	Rank	2	1	2	1	2	1
f_3	Mean	2.02e+01	**1.95e+01**	**2.01e+01**	2.02e+01	**2.02e+01**	**2.02e+01**
	Std	3.94e-02	3.12e+00	4.10e-02	4.14e-02	4.52e-02	4.13e-02
	Rank	2	1	1	2	2	1
f_4	Mean	1.26e+01	**1.18e+01**	9.77e+01	**9.56e+01**	2.34e+02	**2.32e+02**
	Std	4.18e+01	3.67e+00	1.80e+01	1.78e+01	2.93e+01	2.89e+01
	Rank	2	1	2	1	2	1
f_5	Mean	4.31e+02	**3.80e+02**	2.42e+03	**2.37e+03**	4.24e+03	**4.19e+03**
	Std	1.49e+02	1.51e+02	2.86e+02	2.71e+02	4.77e+02	3.75e+02
	Rank	2	1	2	1	2	1
f_6	Mean	5.03e+03	**4.47e+03**	1.38e+06	**1.24e+06**	2.20e+06	**1.95e+06**
	Std	4.41e+03	3.10e+03	7.15e+02	6.35e+02	8.30e+02	7.53e+05
	Rank	2	1	2	1	2	1
f_7	Mean	8.72e+01	**7.90e-01**	9.39e+00	**9.18e+00**	1.85e+01	**1.59e+01**
	Std	2.58e+01	2.87e-01	1.25e+00	1.13e+00	8.52e+00	2.02e+00
	Rank	2	1	2	1	2	1
f_8	Mean	**1.32e+04**	1.81e+04	4.11e+05	**3.90e+05**	2.89e+06	**2.15e+06**
	Std	2.14e+04	3.02e+04	3.06e+05	1.96e+05	9.87e+05	8.05e+05
	Rank	1	2	2	1	2	1
f_9	Mean	9.48e+01	**9.32e+01**	1.21e+02	**1.19e+02**	1.62e+02	**1.33e+02**
	Std	2.20e+01	2.35e+01	4.44e+01	4.41e+01	1.13e+02	7.43e+01
	Rank	2	1	2	1	2	1
f_{10}	Mean	**4.37e+03**	4.60e+03	6.88e+05	**5.07e+05**	**8.07e+05**	9.57e+05
	Std	4.24e+03	7.02e+01	3.57e+05	2.90e+05	4.68e+05	4.39e+05
	Rank	1	2	2	1	1	2
f_{11}	Mean	3.01e+02	**2.82e+02**	3.22e+02	**3.21e+02**	**3.58e+02**	3.62e+02
	Std	4.70e-01	4.01e-13	7.47e+00	7.40e+00	1.69e+02	1.65e+02
	Rank	2	1	2	1	1	2
f_{12}	Mean	**1.04e+02**	**1.04e+02**	**1.07e+02**	**1.07e+02**	**1.10e+02**	**1.10e+02**
	Std	7.72e-01	8.03e-01	8.07e-01	6.08e-01	7.41e-01	7.99e-01
	Rank	1	1	1	1	1	1
f_{13}	Mean	**3.13e+01**	3.14e+01	**1.02e+02**	1.04e+02	**1.89e+02**	**1.89e+02**
	Std	2.04e+00	2.21e+00	4.35e+00	3.58e+00	4.91e+00	4.25e+02
	Rank	1	2	1	2	1	1
f_{14}	Mean	1.86e+03	**1.81e+03**	**3.06e+04**	**3.06e+04**	5.00e+04	**4.95e+04**
	Std	1.37e+03	1.31e+03	4.51e+03	5.57e+03	1.77e+03	1.42e+01
	Rank	2	1	1	1	2	1
f_{15}	Mean	**1.00e+02**	**1.00e+02**	**1.00e+02**	**1.00e+02**	1.02e+02	**1.01e+02**
	Std	2.27e-11	5.54e-12	6.56e-02	6.99e-03	1.23e+00	1.14e+00
	Rank	1	1	1	1	2	1
Mean rank		1.67	1.2	1.67	1.13	1.73	1.13
Overall rank		2	1	2	1	2	1

3.3. The New Search Strategy of Onlooker Bees

Lin et al. [29] proposed an improved ABC algorithm based on cloud model (cmABC). By calculating a candidate food source through the normal cloud operator and reducing the radius of the local search, the cmABC algorithm was proved to enhance the convergence speed, exploitation capability and solution quality on the experiments of composition functions. In cmABC, three digital characteristics of cloud model (Ex, En, He) are given as:

$$\begin{cases} Ex = X_i^j \\ En = ex \\ He = \frac{En}{10} \end{cases} \tag{13}$$

where X_i is the current food sources position, $j \in \{1, 2, \ldots, D\}$, ex is variable. The forward cloud generator can produce a normal random number V_i^j, which will correspond to the new food sources position of jth dimension. Detailed operations were described as:

$$\begin{cases} En^{'} = N(En, He^2) \\ V_i^j = N(Ex, En^{'2}) \end{cases} \tag{14}$$

The greater the value of entropy En, the wider the distribution of cloud droplets and vice versa. When the search iteration reached a certain number of times, the population was closer and closer to the optimal solution. A nonlinear decrease strategy to self-adaptive adjust the value of ex was used in cmABC for the sake of improving the precision of solution and controlling the bees' search range:

$$ex = -(E_{max} - E_{min})(t/T_{max})^2 + E_{max} \tag{15}$$

where $t \in \{1, 2, 3, \ldots, T_{max}\}$ was the current number of iterations, T_{max} was the maximum number of cycles. The values of parameters E_{max} and E_{min} were set to 5 and 10^{-4}, respectively. In order not to specify too many parameters, in this paper, three digital characteristics of cloud model (Ex, En, He) are given as

$$\begin{cases} Ex = X_i^j \\ En = \frac{2}{3}|X_i^j - X_k^j| \\ He = \frac{En}{10} \end{cases} \tag{16}$$

where $j \in \{1, 2, \ldots, D\}$, $k \in \{1, 2, \ldots, N_e\}$, k must be different from i, k, j are random generating indexes. This amendment is based on the stable tendency and randomness of normal cloud model. The entropy En is selected by '3σ' principle of normal cloud model, which can control the onlooker bees to search in a suitable range.

3.4. Search Strategy of Scouts Combined with Y Conditional Cloud Generator

Employed and onlooker bees look for a better food source around their neighborhoods in each cycle of the search. If the fitness value of a food source is not improved by a predetermined number of trials that is equal to the value of '*Limit*', then that food source is abandoned by its employed bee and the employed bee associated with that food source becomes a scout bee. In the basic ABC, the scout randomly finds a new food source to replace the abandoned one by Equation (2), which makes the convergence rate of the basic ABC slow for not taking full advantage of the historical optimal solution information. In this section we make the scout bee search a candidate position around the historical optimal value fit_{best} (corresponding to *Globalmin*) by Y-conditional cloud operator. Search strategy of scouts combined with Y-conditional cloud generator is described in Algorithm 4. The purpose of setting $\mu \in (0, 0.5)$ in Step 3 is to guarantee population diversity. Cloud droplets which have smaller membership degrees are farther from center Ex, that is to say the new food source is farther from historical optimum *Globalmin*. However, the historical optimum information is used to generate a scout, therefore aimless searching of scout bees in the basic ABC algorithm can be avoided to a certain degree.

Algorithm 4: Search strategy of scouts combined with Y-conditional cloud generator.

Step 1	Set expectation Ex as *GlobalParams*, which is the position parameters of *Globalmin*.
Step 2	Entropy $En = (X_{max} - X_{min})/N_e$.
Step 3	Hyper entropy $He = En/c2$, where $c2 = 10$.
Step 4	Randomly generate membership degrees $\mu^j \in (0, 0.5)$, where $j \in \{1, 2, \ldots, D\}$.
Step 5	Obtain the new food resource X_i according to Equation (9).

3.5. DCABC Algorithm

Pseudo code of DCABC algorithm proposed for solving unconstrained optimization problems is given in Algorithm 5. $MaxFES$ represents the maximum number of function evaluations. FES represents the number of function evaluations.

Algorithm 5: Pseudo code of DCABC algorithm.

Initialization phase
Using Equation (1) initialize the population of solutions X_i^j, i = 1, 2,. . . ,N_e, j=1, 2,. . . , D.
Evaluate the fitness of the population by Equation (3), set the current $FES = Ne$.
While $FES \leq MaxFES$ **do**
Employed bees phase
Send employed bees to produce new solutions via Equation (2).
Apply greedy selection to evaluate the new solutions.
If *rand* less than or equal to 0.5, Calculate the selective probability using Equation (4);
Otherwise calculate the probability using Equations (11) and (12).
Onlooker bees phase
Send onlooker bees to produce new solutions via Equations (14) and (16).
Apply greedy selection to evaluate the new solutions.
Scout bee phase
Send one scout bee generated by Algorithm 4 into the search area for discovering a new food source.
Memorize the best solution found so far.
end while
Return the best solution.

4. Experimental Study of DCABC

4.1. Evaluation Functions

Comparing the proposed DCABC with the basic ABC and the other ABC variants, such as GABC [12], cmABC [29] and PABC [27], the experimental results of benchmark functions with 10, 30 and 50 decision variables in CEC15 [32] are given under the same machine with an Intel 3.20 GHz CPU, 8GB memory, and the operating system is Windows 7 with MATLAB 9.0 (R2016a). All functions in CEC15 have different optimal values $f(x^*)$.

4.2. Parameters Settings

For all compared algorithms including DCABC, the size of initial population is 40, an equal split of employed bees and onlookers. $'limit'$ equals to $N_e * D$ [33]; The dimension is set as 10, 30 and 50 in turn. In Equation (2) of GABC [12], $C = 1.5$. In cmABC [29], $Emax = 5$, $Emin = 10^{-4}$. The $MaxFES$ is $D * 10{,}000$, which is used as the terminal criterion of five algorithms. Every experiment is repeated 30 times each starting from a random population with different random seeds, the mean results (Mean) and the standard deviation (Std) of each algorithm are recorded with the format of $f(x) - f(x^*)$ in Tables 2–4. The best results are highlighted in boldface. Rank records the performance-rank of five algorithms for dealing with each benchmark function according to their mean results. The overall rank for each algorithm is defined according to their mean rank values over 15 benchmark problems. The number of (Best/2ndBest/Worst) is counted for each algorithm.

4.3. Experiments Analysis

DCABC algorithm is better than four other compared algorithms on dimension 10. It can be seen from Table 2 that DCABC has the best performance on 10 of 15 test problems. DCABC is only worse

than ABC, PABC on one and two functions (f_9, f_3 and f_9). It is worth noting that GABC and cmABC surpass DCABC only on functions f_3, f_5, and f_{14}. GABC and cmABC generate the best results only on functions f_4 and f_9, respectively.

From Table 3, DCABC ranks NO.1 on 11 of 15 functions with dimension 30. Actually, DCABC is superior to ABC and GABC on all the functions. In contrast, DCABC is inferior to cmABC and PABC on functions f_3 and f_9, and cmABC shows the best performance on functions f_3, f_5, f_9, and f_{14}.

Table 2. Experimental Results about ABC and other ABC variants (10D).

Functions	Criteria	ABC	GABC	cmABC	PABC	DCABC
f_1	Mean	1.37e+06	5.88e+05	1.39e+06	1.43e+06	**8.06e+02**
	Std	1.12e+06	4.73e+05	8.96e+05	9.72e+05	2.31e+03
	Rank	3	2	4	5	1
f_2	Mean	8.57e+02	2.32e+03	4.23e+02	8.51e+02	**2.88e+02**
	Std	7.52e+02	2.45e+03	3.52e+02	5.30e+02	1.20e+03
	Rank	4	5	2	3	1
f_3	Mean	2.01e+01	1.88e+01	**1.87e+01**	1.98e+01	2.00e+01
	Std	3.9e-02	5.07e+00	4.77e+00	1.81e+00	7.88e-03
	Rank	5	2	1	3	4
f_4	Mean	1.26e+01	**5.04e+00**	9.30e+00	1.23e+01	7.89e+00
	Std	4.18e+00	1.42e+00	3.28e+00	3.94e+00	2.99e+00
	Rank	5	1	3	4	2
f_5	Mean	4.31e+02	2.31e+02	**1.82e+02**	3.66e+02	2.61e+02
	Std	1.49e+02	1.16e+02	9.83e+01	1.24e+02	1.51e+02
	Rank	5	2	1	4	3
f_6	Mean	5.03e+03	3.16e+03	3.20e+03	4.13e+03	**1.68e+02**
	Std	4.40e+03	2.36e+03	2.96e+03	4.95e+03	1.97e+02
	Rank	5	2	3	4	1
f_7	Mean	8.72e-01	4.02e-01	5.07e-01	8.65e-01	**3.99e-01**
	Std	2.58e-01	2.37e-01	3.29e-01	2.68e-01	4.24e-01
	Rank	4	2	3	5	1
f_8	Mean	1.32e+04	5.04e+03	7.38e+03	1.26e+04	**4.25e+02**
	Std	2.14e+04	4.33e+03	6.80e+03	2.43e+04	1.20e+03
	Rank	5	2	3	4	1
f_9	Mean	9.48e+01	1.00e+02	**6.48e+01**	9.59e+01	1.00e+02
	Std	2.20e+01	5.66e-02	4.79e+01	1.76e+01	4.12e-02
	Rank	2	4	1	3	4
f_{10}	Mean	4.37e+03	1.84e+03	3.28e+03	4.69e+03	**4.41e+02**
	Std	4.24e+03	9.70e+02	5.22e+03	4.73e+03	1.88e+02
	Rank	4	2	3	5	1
f_{11}	Mean	3.00e+2	2.45e+02	2.49e+02	2.90e+02	**1.92e+02**
	Std	4.69e-1	1.14e+02	1.07e+02	4.81e+01	1.44e+02
	Rank	5	2	3	4	1
f_{12}	Mean	1.04e+02	**1.03e+02**	1.04e+02	1.04e+02	**1.03e+02**
	Std	7.72e-01	4.95e-01	6.33e-01	6.29e-01	8.95e-01
	Rank	2	1	2	2	1
f_{13}	Mean	3.13e+01	2.89e+01	2.95e+01	3.11e+01	**2.77e+01**
	Std	2.04e+00	2.70e+00	2.30e+00	2.01e+00	2.74e+00
	Rank	5	2	3	4	1
f_{14}	Mean	1.86e+03	4.76e+02	**4.47e+02**	1.41e+03	1.14e+03
	Std	1.38e+03	8.49e+02	9.03e+02	1.32e+03	1.39e+03
	Rank	5	2	1	4	3
f_{15}	Mean	**1.00e+02**	**1.00e+02**	**1.00e+02**	**1.00e+02**	**1.00e+02**
	Std	2.28e-11	7.75e-06	1.25e-09	1.59e-11	5.42e-04
	Rank	1	1	1	1	1
Mean rank		4.00	2.13	2.26	3.67	1.73
Overall rank		4	2	3	4	1
Best/2nd Best/Worst		1/1/8	3/10/1	5/2/0	1/1/3	10/1/0

Table 3. Experimental Results about ABC and other ABC variants (30D).

Functions	Criteria	ABC	GABC	cmABC	PABC	DCABC
f_1	Mean	3.73e+06	3.11e+06	1.39e+06	1.43e+06	**8.06e+02**
	Std	1.46e+06	2.05e+06	8.96e+05	9.72e+05	2.31e+03
	Rank	5	4	2	3	1
f_2	Mean	7.26e+02	2.85e+03	4.23e+02	8.51e+02	**2.88e+02**
	Std	6.06e+02	3.77e+03	3.52e+02	5.30e+02	1.20e+03
	Rank	3	5	2	4	1
f_3	Mean	2.01e+01	2.02e+01	**1.87e+01**	1.98e+01	2.00e+01
	Std	4.10e-02	8.32e-02	4.77e+00	1.81e+00	7.88e-03
	Rank	4	5	1	2	3
f_4	Mean	9.77e+01	5.50e+01	9.31e+00	1.23e+01	**7.89e+00**
	Std	1.80e+01	9.75e+00	3.28e+00	3.94e+00	2.99e+00
	Rank	5	4	2	3	1
f_5	Mean	2.42e+03	1.92e+03	**1.82e+02**	3.66e+02	2.61e+02
	Std	2.86e+02	2.96e+02	9.83e+01	1.24e+02	1.51e+02
	Rank	5	4	1	3	2
f_6	Mean	1.38e+06	1.45e+06	3.20e+03	4.13e+03	**1.68e+02**
	Std	7.15e+05	7.51e+05	2.96e+03	4.95e+03	1.97e+02
	Rank	4	5	2	3	1
f_7	Mean	9.39e+00	7.04e+00	5.07e-01	8.65e-01	**3.99e-01**
	Std	1.25e+00	1.81e+00	3.29e-01	2.68e-01	4.24e-01
	Rank	5	4	2	3	1
f_8	Mean	4.11e+05	3.31e+05	7.38e+03	1.26e+04	**4.25e+02**
	Std	3.06e+05	1.91e+05	6.80e+03	2.43e+04	1.20e+03
	Rank	5	4	2	3	1
f_9	Mean	1.21e+02	1.05e+02	**6.48e+01**	9.59e+01	1.00e+02
	Std	4.44e+01	4.89e-01	4.79e+01	1.76e+01	4.12e-02
	Rank	5	4	1	2	3
f_{10}	Mean	6.88e+05	6.84e+05	3.28e+03	4.69e+03	**4.41e+02**
	Std	3.57e+05	5.28e+05	5.22e+03	4.73e+03	1.88e+02
	Rank	5	4	2	3	1
f_{11}	Mean	3.22e+02	3.49e+02	2.49e+02	2.90e+02	**1.92e+02**
	Std	7.47e+00	1.11e+02	1.07e+02	4.81e+01	1.44e+02
	Rank	4	5	2	3	1
f_{12}	Mean	1.07e+02	1.07e+02	1.04e+02	1.04e+02	**1.03e+02**
	Std	8.07e-01	5.78e-01	6.33e-01	6.29e-01	8.95e-01
	Rank	3	3	2	2	1
f_{13}	Mean	1.03e+02	9.91e+01	2.95e+01	3.11e+01	**2.77e+01**
	Std	4.35e+00	2.65e+00	2.30e+00	2.01e+00	2.74e+00
	Rank	5	4	2	3	1
f_{14}	Mean	3.06e+04	3.14e+04	**4.47e+02**	1.41e+03	1.14e+03
	Std	4.51e+02	6.51e+02	9.03e+02	1.32e+03	1.39e+03
	Rank	4	5	1	3	2
f_{15}	Mean	**1.00e+02**	**1.00e+02**	**1.00e+02**	**1.00e+02**	**1.00e+02**
	Std	6.56e+02	7.75e-06	1.25e+00	1.59-11	5.42e-04
	Rank	1	1	1	1	1
Mean rank		4.20	4.07	1.67	2.73	1.40
Overall rank		5	4	2	3	1
Best/2nd Best/Worst		1/0/8	1/0/5	5/10/0	1/3/0	11/2/0

In Table 4, DCABC outperforms all compared algorithms with dimension 50 on functions f_1, f_3, f_4, f_5, f_6, f_8, f_9, f_{10}, f_{12}, and f_{15}. DCABC cannot beat ABC, GABC, cmABC and PABC on f_7 and f_{11}. PABC shows the best performance on f_2 and f_{14}. cmABC is superior to all other algorithms on f_3 and f_7. GABC is competitive on function f_{13}. ABC has the best results on function f_{11}, f_{12} and f_{14}. It is worth noting that the overall performance of DCABC is the best.

Table 4. Experimental Results about ABC and other ABC variants (50D).

Functions	Criteria	ABC	GABC	cmABC	PABC	DCABC
f_1	Mean	1.20e+07	1.05e+07	1.00e+07	1.28e+07	**3.92e+02**
	Std	3.69e+06	4.49e+00	2.63e+06	4.65e+06	1.36e+03
	Rank	4	3	2	5	1
f_2	Mean	1.43e+03	7.88e+03	1.52e+03	**1.37e+03**	3.74e+03
	Std	1.20e+03	7.18e+03	1.04e+03	1.32e+03	7.84e+03
	Rank	2	5	3	1	4
f_3	Mean	2.02e+01	2.02e+01	**2.00e+01**	2.01e+01	**2.00e+01**
	Std	4.12e-02	6.70e-02	6.64e-03	3.41e-02	1.56e-02
	Rank	3	3	1	2	1
f_4	Mean	2.34e+02	2.30e+02	2.71e+02	2.19e+02	**1.53e+02**
	Std	2.93e+01	2.90e+01	3.35e+01	3.04+01	2.42e+01
	Rank	4	3	5	2	1
f_5	Mean	4.24e+03	4.76e+03	4.14e+03	4.10e+03	**3.93e+03**
	Std	4.77e+02	4.03e+02	4.09e+02	3.34e+02	4.31e+02
	Rank	5	2	4	3	1
f_6	Mean	2.20e+06	2.45e+06	2.11e+06	2.30e+06	**2.25e+03**
	Std	7.53e+05	1.26e+06	6.63e+05	8.94e+05	1.10e+03
	Rank	3	5	2	4	1
f_7	Mean	1.85e+01	1.91e+01	**1.57e+01**	1.72e+01	3.04e+01
	Std	8.52e+00	9.91e+00	1.48e+00	6.53e+00	1.56e+01
	Rank	3	4	1	2	5
f_8	Mean	2.15e+06	2.97e+06	2.35e+06	3.25e+06	**3.48e+03**
	Std	8.87e+05	1.65e+06	6.22e+05	1.02e+06	1.03e+04
	Rank	2	4	3	5	1
f_9	Mean	1.62e+02	1.08e+02	1.08e+02	1.52e+02	**1.07e+02**
	Std	1.13e+02	5.21e-01	8.44e-01	9.20e+01	5.74e-01
	Rank	4	2	2	3	1
f_{10}	Mean	8.07e+05	1.10e+06	9.27e+05	9.86e+05	**3.59e+03**
	Std	4.68e+05	6.62e+05	3.07e+05	4.57e+05	7.45e+02
	Rank	2	5	3	4	1
f_{11}	Mean	**3.58e+02**	6.68e+02	3.99e+02	4.27e+02	8.16e+02
	Std	1.69e+02	4.05e+02	2.72e+02	2.89e+02	4.05e+02
	Rank	1	4	2	3	5
f_{12}	Mean	**1.10e+02**	1.19e+02	**1.10e+02**	**1.10e+02**	**1.10e+02**
	Std	7.41e-01	4.97e-01	8.35e-01	6.39e-01	9.15e-01
	Rank	1	2	1	1	1
f_{13}	Mean	1.89e+02	**1.85e+02**	1.93e+02	1.88e+02	1.87e+02
	Std	4.91e+00	4.43e+00	5.34e+00	6.20e+00	5.88e+00
	Rank	4	1	5	3	2
f_{14}	Mean	**4.99e+04**	5.51e+04	5.02e+04	**4.99e+04**	5.41e+04
	Std	1.77e+03	6.10+03	2.45e+03	1.76e+03	4.57e+03
	Rank	1	4	2	1	3
f_{15}	Mean	1.02e+02	**1.00e+02**	**1.00e+02**	**1.00e+02**	**1.00e+02**
	Std	1.23e+00	2.00e-07	3.03e-01	1.59-11	3.89e-04
	Rank	2	1	1	1	1
Mean rank		2.73	3.20	2.47	2.67	1.93
Overall rank		4	5	2	3	1
Best/2nd Best/Worst		3/4/1	2/3/3	4/5/2	4/3/2	10/1/2

Unimodal function f_1, hybrid function f_8 and composition functions f_{10} are chosen to exhibit the convergence precision of all compared algorithms. Figures 3–8 are the convergence graphs of five algorithms. The horizontal axis is the number of function evaluations (FES), and the vertical axis is the function values over one independent run. In all the figures, DCABC is represented by the black line with circles, and it has larger descend gradient and gets the minimal error values among the five algorithms. The convergence speed of DCABC is also obviously superior to the other four algorithms.

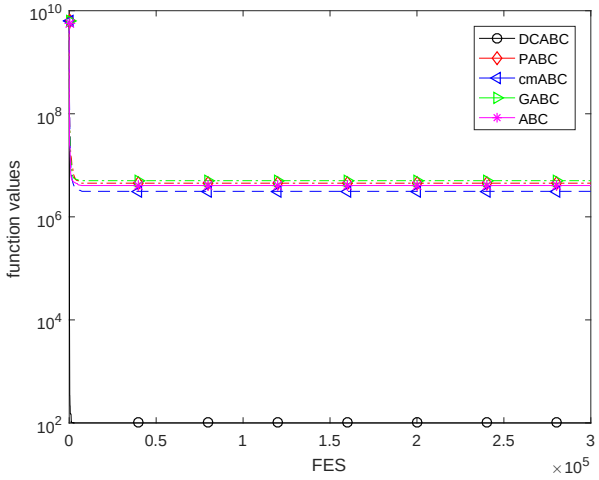

Figure 3. Convergence curves of five algorithms for f_1 with D = 30. (The optimal value of f_1 is 100).

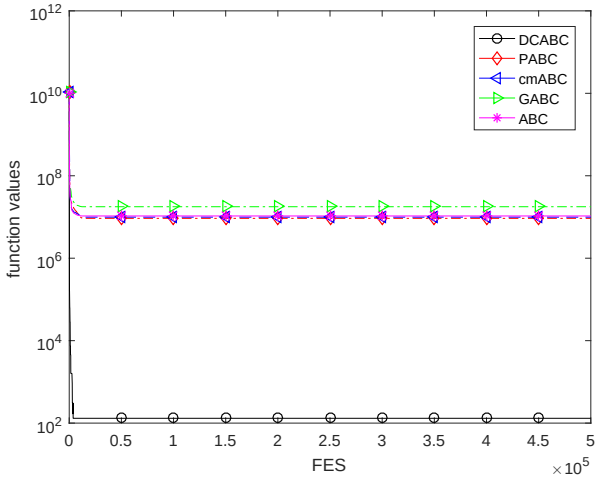

Figure 4. Convergence curves of five algorithms for f_1 with D = 50. (The optimal value of f_1 is 100).

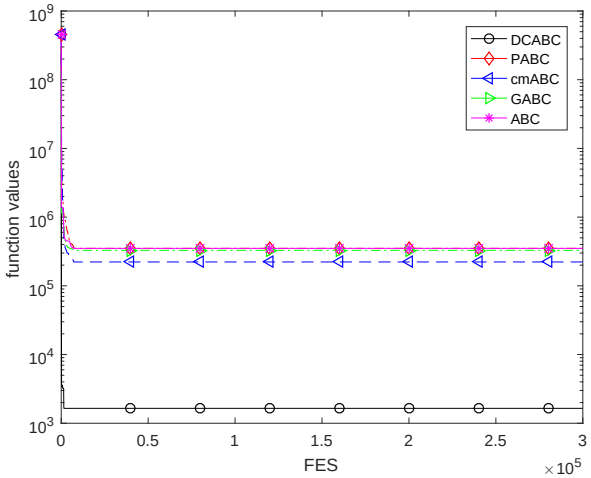

Figure 5. Convergence curves of five algorithms for f_8 with D = 30. (The optimal value of f_8 is 800).

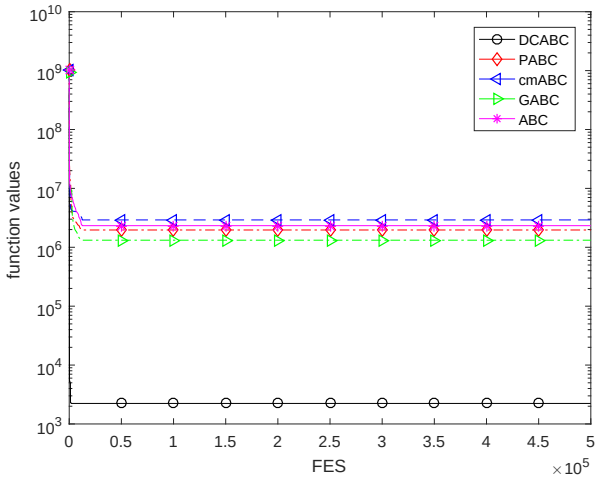

Figure 6. Convergence curves of five algorithms for f_8 with D = 50. (The optimal value of f_8 is 800).

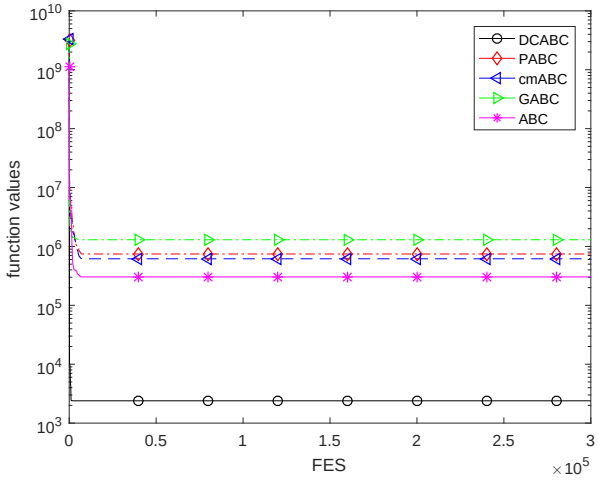

Figure 7. Convergence curves of five algorithms for f_{10} with D = 30. (The optimal value of f_{10} is 1000).

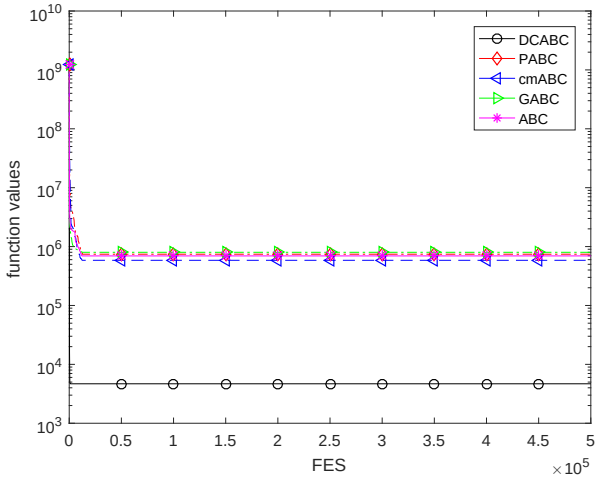

Figure 8. Convergence curves of five algorithms for f_{10} with D = 50. (The optimal value of f_{10} is 1000).

On the whole, compared to the other three modified ABC algorithms, DCABC can show the best performance on most of the functions, that is to say this new algorithm is more stable and the solutions obtained by it have higher precision than other algorithms.

5. Conclusions

In the present study, a developed artificial bee colony algorithm based on cloud model, namely DCABC, is proposed for the continuous optimization. By using a new selection mechanism, the worse individual in DCABC has a larger probability to be selected than in basic ABC. DCABC also improves the local search ability by applying the normal cloud generator as onlookers bees' formula to control the search of onlookers in a suitable range. Moreover, historical optimal solutions are used by Y conditional cloud generator when updating the scout bee to ensure the algorithm jump out of the local optimal. The effectiveness of the proposed method is tested on CEC15. The results clearly show the superiority of DCABC over ABC, GABC, cmABC and PABC.

However, there are quite a few issues that merit further investigation such as the diversity of DCABC. In addition, we hope to show the performance of DCABC by Null Hypothesis Significance Testing (NHST) [35,36] in our future work. We only test the new algorithm on classical benchmark functions and have not used it to solve practical problems, such as fault diagnosis [37], path plan [38], Knapsack [39–41], multi-objective optimization [42], gesture segmentation [43], unit commitment problem [44], and so on. There is an increasing interest in prompting the performance of DCABC, which will be our future research direction.

Acknowledgments: This research is partly supported by Humanity and Social Science Youth foundation of Ministry of Education of China (Grant No. 12YJCZH179), the Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Grant No. 16KJA110001), the National Natural Science Foundation of China (Grant No. 11371197), the Foundation of Jiangsu Key Laboratory for NSLSCS (Grant No. 201601).

Author Contributions: These authors contributed equally to this paper.

Conflicts of Interest: No conflict of interest exists in the submission of this article, and it is approved by all authors for publication.

References

1. Sörensen, K.; Sevaux, M.; Glover, F. A History of Metaheuristics. In *Handbook of Heuristics*; Springer: Berlin/Heidelberg, Germany, 2018.
2. Sörensen, K. Metaheuristics-the metaphor exposed. *Int. Trans. Oper. Res.* **2015**, *22*, 3–18.
3. Črepinšek, M.; Liu, S.; Mernik, M. Exploration and exploitation in evolutionary algorithms: A survey. *ACM Comput. Surv.* **2013**, *45*, 1–33.
4. Karaboga, D. *An Idea Based on Honey bee Swarm for Numerical Optimization*; Technical Report-tr06; Engineering Faculty, Computer Engineering Department, Erciyes University: Kayseri, Turkey, 2005.
5. Karaboga, D.; Basturk, B. Artificial Bee Colony (ABC) optimization algorithm for solving constrained optimization problems. *Found. Fuzzy Log. Soft Comput.* **2007**, *4529*, 789–798.
6. Karaboga, D.; Basturk, B. On the performance of artificial bee colony (ABC) algorithm. *Appl. Soft Comput.* **2008**, *8*, 687–697.
7. Karaboga, D.; Akay, B. A comparative study of artificial bee colony algorithm. *Appl. Math. Comput.* **2009**, *214*, 108–132.
8. Akay, B.; Karaboga, D. Parameter tuning for the artificial bee colony algorithm. *Comput. Collect. Intell.* **2009**, *5796*, 608–619.
9. Akay, B.; Karaboga, D. A modified artificial bee colony algorithm for real-parameter optimization. *Swarm Intell. Appl.* **2012**, *192*, 120–142.
10. Zhang, D.; Guan, X.; Tang, Y.; Tang, Y. Modified artificial bee colony algorithms for numerical optimization. In Proceedings of the 2011 3rd International Workshop on Intelligent Systems and Applications (ISA), Wuhan, China, 28–29 May 2011; pp. 1–4.
11. Aderhold, A.; Diwold, K.; Scheidler, A.; Middendorf, M. Artificial bee colony optimization: A new selection scheme and its performance. *Nat. Inspired Coop. Strateg. Optim. (NICSO 2010)* **2010**, *284*, 283–294.

12. Zhu, G.; Kwong, S. Gbest-guided artificial bee colony algorithm for numerical function optimization. *Appl. Math. Comput.* **2010**, *217*, 3166–3173.
13. Wu, X.; Hao, D.; Xu, C. An improved method of artificial bee colony algorithm. *Appl. Mech. Mater.* **2012**, *101–102*, 315–319.
14. Guo, P.; Cheng, W.; Liang, J. Global artificial bee colony search algorithm for numerical function optimization. In Proceedings of the 2011 Seventh International Conference on Natural Computation (ICNC), Shanghai, China, 26–28 July 2011; Volume 3, pp. 1280–1283.
15. Yu, X.; Zhu, Z. A modified artificial bee colony algorithm with its applications in signal processing. *Int. J. Comput. Appl. Technol.* **2013**, *47*, 297–303.
16. Rajasekhar, A.; Pant, M. An improved self-adaptive artificial bee colony algorithm for global optimisation. *Int. J. Swarm Intell.* **2014**, *1*, 115–132.
17. Yaghoobi, T.; Esmaeili, E. An improved artificial bee colony algorithm for global numerical optimisation. *Int. J. Bio-Inspired Comput.* **2017**, *9*, 251–258.
18. Zou, W.; Zhu, Y.; Chen, H.; Zhang, B. Solving multiobjective optimization problems using artificial bee colony algorithm. *Discret. Dyn. Nat. Soc.* **2011**, *2*, 1–37.
19. Amarjeet; Chhabra, J.K. FP-ABC: Fuzzy Pareto-Dominance Driven Artificial Bee Colony Algorithm for Many-Objective Software Module Clustering. *Comput. Lang. Syst. Struct.* **2018**, *51*, 1–21.
20. Xiang, Y.; Zhou, Y.; Tang, L.; Chen, Z. A Decomposition-Based Many-Objective Artificial Bee Colony Algorithm. *IEEE Trans. Cybern.* **2017**, *99*, 1–14.
21. Koçer, B. Bollinger bands approach on boosting ABC algorithm and its variants. *Appl. Soft Comput.* **2016**, 49, 292–312.
22. Bollinger Bands—Trademark Details. 2011. Available online: Justia.com (accessed on 1 April 2018) .
23. Wang, H.; Yi, J. An improved optimization method based on krill herd and artificial bee colony with information exchange. *Memet. Comput.* **2017**, *2*, 1–22.
24. Gandomi, A. H.; Alavi, A. H. Krill herd: A new bio-inspired optimization algorithm. *Commun. Nonlinear Sci. Numer. Simul.* **2012**, *17*, 4831–4845.
25. Li, D.; Liu, C.; Du, Y.; Han, X. Artificial Intelligence with Uncertainty. *J. Softw.* **2004**, *15*, 1583–1594.
26. Chen, H.; Li, D.; Shen, D.; Zhang, F. A clouds model applied to controlling inverted pendulum. *J. Comput. Res. Dev.* **1999**, *36*, 1180–1187.
27. Zhang, C.; Pang, Y. Sequential blind signal extraction adopting an artificial bee colony Algorithm algorithm. *J. Inf. Comput. Sci.* **2012**, *9*, 5551–5559.
28. He, D.; Jia, R. Cloud model-based Artificial Bee Colony algorithm's application in the logistics location problem. In Proceedings of the International Conference on Information Management, Innovation Management and Industrial Engineering, Sanya, China, 20–21 October 2012.
29. Lin, X.; Ye, D. Artificial Bee Colony algorithm based on cloud mutation. *J. Comput. Appl.* **2012**, *32*, 2538–2541.
30. Li, D.; Meng, H.; Shi, X. Membership clouds and membership clouds generators. *Comput. Res. Dev.* **1995**, *42*, 32–41.
31. Di, K.; Li, D.; Li, D. Cloud theory and its applications in spatial data mining and knowledge discovery. *J. Image Graph.* **1999**, *4*, 930–935.
32. Chen, Q.; Liu, B.; Zhang, Q.; Liang, J.; Suganthan, P.N.; Qu, B. *Problem Definition and Evaluation Criteria for CEC 2015 Special Session and Competition on Bound Constrained Single-Objective Computationally Expensive Numerical Optimization*; Technical Report; Computational Intelligence Laboratory, Zhengzhou University: Zhengzhou, China; Nanyang Technological University: Singapore, 2014.
33. Veček, N.; Liu, S.; Črepinšek, M.; Mernik, M. On the Importance of the Artificial Bee Colony Control Parameter Limit. *Inf. Technol. Control* **2017**, *46*, 566–604.
34. Mernik, M.; Liu, S.; Karaboga, D. On clarifying misconceptions when comparing variants of the Artificial Bee Colony Algorithm by offering a new implementation. *Inf. Sci.* **2015**, *291*, 115–127.
35. Derrac, J.; García, S.; Molina, D.; Herrera, F. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.* **2011**, *1*, 3–18.
36. Veček, N.; Mernik, M.; Črepinšek, M. A chess rating system for evolutionary algorithms: A new method for the comparison and ranking of evolutionary algorithms. *Inf. Sci.* **2014**, *277*, 656–679.

37. Yi, J.; Wang, J.; Wang, G. Improved probabilistic neural networks with self-adaptive strategies for transformer fault diagnosis problem. *Adv. Mech. Eng.* **2016**, *8*, 1–13.
38. Wang, G.; Chu, H.; Mirjalili, S. Three-dimensional path planning for UCAV using an improved bat algorithm. *Aerosp. Sci. Technol.* **2016**, *49*, 231–238.
39. Feng, Y.; Wang, G.; Deb, S.; Lu, M.; Zhao, X. Solving 0-1 knapsack problem by a novel binary monarch butterfly optimization. *Neural Comput. Appl.* **2017**, *28*, 1619–1634.
40. Feng, Y.; Wang, G.; Li, W.; Li, N. Multi-strategy monarch butterfly optimization algorithm for discounted 0-1 knapsack problem. *Neural Comput. Appl.* **2017**, doi:10.1007/s00521-017-2903-1.
41. Feng, Y.; Wang, G.; Dong, J.; Wang, L. Opposition-based learning monarch butterfly optimization with Gaussian perturbation for large-scale 0-1 knapsack problem. *Comput. Electr. Eng.* **2017**, doi:10.1016/j.compeleceng.2017.12.014.
42. Rizk-Allah, R.M.; El-Sehiemy, R.A.; Deb, S.; Wang, G. A novel fruit fly framework for multi-objective shape design of tubular linear synchronous motor. *J. Supercomput.* **2017**, *73*, 1235–1256.
43. Liu, K.; Gong, D.; Meng, F.; Chen, H.; Wang, G. Gesture segmentation based on a two-phase estimation of distribution algorithm. *Inf. Sci.* **2017**, *394–395*, 88–105.
44. Srikanth, K.; Panwar, L.K.; Panigrahi, B.K.; Herrera-Viedma, E.; Sangaiah, A.K.; Wang, G. Meta-heuristic framework: Quantum inspired binary grey wolf optimizer for unit commitment problem. *Comput. Electr. Eng.* **2017**, doi:10.1016/j.compeleceng.2017.07.023.