Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 129271, 10 pages
http://dx.doi.org/10.1155/2015/129271

Research Article

A Novel Artificial Bee Colony Algorithm for Function Optimization

Song Zhang and Sanyang Liu

School of Mathematics and Statistics, Xidian University, Xi'an 710071, China

Correspondence should be addressed to Song Zhang; yxszhang@163.com

Received 6 October 2014; Revised 1 March 2015; Accepted 11 March 2015

Academic Editor: Sishaj P. Simon

It is known that both exploration and exploitation are important in the search equations of ABC algorithms. How to well balance the two abilities in the search process is still a challenging problem in ABC algorithms. In this paper, we propose a novel artificial bee algorithm named as "NABC," by incorporating the information of the global best solution into the solution search equation of the onlookers stage to improve the exploitation. At the same time, we improve the search equation of the employed bees to keep the exploration. The experimental results of NABC tested on a set of 11 numerical benchmark functions show good performance and fast convergence in solving function optimization problems, compared with variant ABC, DE, and PSO algorithms. The application of NABC on solving five standard knapsack problems shows its effectiveness and practicability.

1. Introduction

Intelligent algorithms, such as genetic algorithm (GA) [1], particle swarm optimization (PSO) [2], ant colony optimization (ACO) [3], and Biogeography-Based Optimization (BBO) [4], have shown great success in solving optimization problems which are nonconvex, discontinuous, nondifferentiable, and so on. At the same time, they have many advantages, such as simplicity, ease of implementation, and outstanding performance. Inspired by the intelligent behaviors of honey bee swarms, Karaboga in 2005 [5] proposed the artificial bee colony (ABC) algorithm for function optimizations. A set of experimental results [6–9] show that ABC algorithm is competitive to some other intelligent algorithms. So the ABC algorithm has captured many attentions and has been applied to solve neural networks, flowshop scheduling problems, constrained optimization problems [10–12], and so on.

We know that both exploration and exploitation are necessary for an intelligent algorithm. In order to achieve good performances on problem optimizations in practice, the two abilities should be well balanced, while the two abilities contradict each other and the solution search equations of ABC algorithm are good at exploration but poor at exploitation. So, many researchers mended the search solutions to improve the exploitation and the convergence during the past decades. Inspired by PSO, Zhu and Kwong [13] took advantage of the information of the global best solution to improve the performance of the ABC algorithm named GABC. Basturk and Karaboga [14] proposed a modified ABC algorithm by controlling the frequency of perturbation and introducing the ratio of the variance operator. Gao et al. proposed a novel ABC method called EABC [15] to further improve the performance of ABC. In EABC, in order to balance the exploration and the exploitation, they presented two new search equations to generate candidate solutions in the employed bee phase and the onlookers phase, respectively. Karaboga and Gorkemli presented a new version of ABC algorithm named quick artificial bee colony (qABC) [16], which modeled the behavior of onlooker bees more accurately and improved the performance of ABC in terms of local search ability. And the performance of the qABC depended on the neighborhood radius.

In this paper, we propose two new solution search equations in the stage of the employed bees and the onlookers, respectively, in order to improve the exploitation and keep the exploration. We name the novel ABC algorithm as "NABC." The experiment results tested on 11 numerical function optimizations show that the novel ABC algorithm is superior to the standard ABC algorithm and other improved ABC

algorithms in most cases. And it is better than variant DE and PSO algorithms. At the same time, five knapsack problems are solved by ABC and NABC, which shows that NABC is superior to ABC on solving engineering optimization problems.

The rest of this paper is organized as follows. Section 2 summarizes the standard ABC algorithm. The novel ABC algorithm named NABC is presented and analyzed in Section 3. Section 4 presents and discusses the experimental results of 11 benchmark functions. Section 5 shows the application of NABC on knapsack problems. Finally, the conclusion is drawn in Section 6.

2. Standard ABC Algorithm

Initialization is the first step in the standard ABC algorithm [5]. The position of a food source represents a possible solution to the optimization problem. Each food source is exploited by only one employed bee, so the number of employed bees named SN is equal to the number of food sources. And a random initial population of SN food sources is generated as follows:

$$x_{i,j} = x_{\min,j} + \text{rand}\,(0,1) * \left(x_{\max,j} - x_{\min,j}\right), \quad (1)$$

where $i \in (1, 2, \ldots, \text{SN})$, $j \in (1, 2, \ldots, D)$, and D is the dimension of the numerical function; $x_{\max,j}$ and $x_{\min,j}$ are the lower and upper bounds of $x_{i,j}$.

Second is the search processes of the employed bees, where the onlookers and the scouts are started. The standard ABC algorithm uses the following equation to produce a candidate food source position from the old one:

$$V_{i,j} = x_{i,j} + \phi_{i,j} * \left(x_{i,j} - x_{k,j}\right), \quad (2)$$

where $k \in (1, 2, \ldots, \text{SN})$ and $j \in (1, 2, \ldots, D)$ are random indexes; k is different from i; $\phi_{i,j}$ is a random number in the range $[-1, 1]$. After each candidate source position is produced and evaluated by the employed bees, its performance is compared with that of the old food source position. If the new food source position has an equal or better quality than the old source position, the old one is replaced by the new one. Otherwise, the old one is retained. The greedy selection is used.

Third, the employed bees transfer their food information to the onlookers. The onlookers tend to further search the food around the selected food source. The onlooker chooses a food source depending on the probability value associated with that food source:

$$p_i = \frac{\text{fitness}_i}{\sum_{i=1}^{n} \text{fitness}_i}, \quad (3)$$

where fitness_i is the fitness value of the solution which is obtained by the following equation:

$$\text{fitness}_i = \begin{cases} \dfrac{1}{1 + f_i}, & f_i \geq 0, \\ 1 + \left|f_i\right|, & f_i < 0, \end{cases} \quad (4)$$

where f_i is the objective function value. If the new food source position has an equal or better quality than the old source position, the old one is replaced by the new one. Otherwise, the old one is retained which is the same as the employed bees stage.

At last, if a position cannot be improved further through a number of cycles, which is called Limit for abandonment, that food source is abandoned accordingly. The scout will discover a new food source position randomly using (1) to replace it.

3. A Novel Artificial Bee Colony Algorithm (NABC)

3.1. Modified Search Solutions. The exploration is the ability of investigating the various unknown regions in the solution space. So it is also very important in ABC algorithm. To retain the exploration, we present a modified search equation of the employed bees as follows:

$$V_{i,j} = x_{r,j} + \phi_{i,j} * \left(x_{r,j} - x_{k,j}\right), \quad (5)$$

where the index r, k is randomly chosen from $(1, 2, \ldots, \text{SN})$, and different from the index i. $j \in (1, 2, \ldots, D)$ is a randomly chosen index. $\phi_{i,j}$ is a random number in the range $[-1, 1]$.

Differential evolution (DE) [17] is a simple and efficient evolutionary algorithm for many optimization problems in the real-world applications [18–22]. Inspired by it, many researchers improved variant intelligent algorithms. So inspired by DE and various modified search equations of the ABC algorithms and taking advantage of the information of the global best solution to guide the search of the candidate solutions, a new equation is proposed to serve as the search equation of onlookers, as follows:

$$V_{i,j} = x_{\text{best},j} + \phi_{i,j} * \left(x_{\text{best},j} - x_{k,j}\right), \quad (6)$$

where the index k is a mutually exclusive integer randomly chosen from $(1, 2, \ldots, \text{SN})$, and different from the index i. $x_{\text{best},j}$ is the jth element of the global best solution and $j \in (1, 2, \ldots, D)$ is a randomly chosen index. $\phi_{i,j}$ is a random number in the range $[-1, 1]$. We name the novel ABC algorithm using (5) to replace (2) on the stage of the employed bees and using (6) to replace (2) on the stage of the onlookers as NABC, which can drive the new candidate solution only around the best solution of the previous iteration. Therefore, the modified solution search equation described by (6) can increase the exploitation of ABC algorithm greatly. The best solution in the current population is a very useful source which can be used to improve the convergence performance. Therefore, we modify the solution search equation by applying the global best solution to guide the search of new candidate solutions in order to improve the exploitation. At the same time, the exploration is improved by (5) from the experiments in Section 4.

The new candidate solution is generated around $x_{i,j}$ in GABC [13]. It is different from (6) in which the one is generated around $x_{\text{best},j}$. EABC [15] adopts two different search equations based on different the emphases in the different search stages. NABC uses (5) and (6) to generate the new solution in the stages of employed bees and onlookers, respectively. In the qABC [16], $x_{N_m}^{\text{best}}$ represents the best

Table 1: Benchmark functions used in experiments.

Fun	Name	Search range	Min
$f_1 = \sum_{i=1}^{n} x_i^2$	Sphere	$[-100, 100]$	0
$f_2 = \sum_{i=1}^{n} i x_i^2$	Sumsquare	$[-10, 10]$	0
$f_3 = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	Schwefel2.22	$[-10, 10]$	0
$f_4 = \max\{\lvert x_i \rvert, 1 \le i \le n\}$	Schwefel2.21	$[-100, 100]$	0
$f_5 = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$	Step	$[-100, 100]$	0
$f_6 = \sum_{i=1}^{n} \left[100 \left(x_i^2 - x_{i+1}\right)^2 + (x_i - 1)^2\right]$	Rosenbrock	$[-5, 10]$	0
$f_7 = \sum_{i=1}^{n} \left[x_i^2 - 10 \cos(2\pi x_i) + 10\right]$	Rastrigin	$[-5.12, 5.12]$	0
$f_8 = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	Griewank	$[-600, 600]$	0
$f_9 = 418.98288727243369 * n - \sum_{i=1}^{n} x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	Schwefel2.26	$[-500, 500]$	0
$f_{10} = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n} \sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + e$	Ackley	$[-32, 32]$	0
$f_{11} = \frac{1}{n} \sum_{i=1}^{n} \left(x_i^4 - 16 x_i^2 + 5 x_i\right)$	Himmelblau	$[-5, 5]$	-78.33236

solution among the neighbors of x_m and itself (N_m). But in the NABC, $x_{\text{best},j}$ represents the global best solution. At the same time, in the stages of employed bees and onlookers, the new search equations used in qABC are different from these of the NABC adopted.

3.2. The Novel ABC Algorithm (NABC). The complete computational procedure of the novel ABC algorithm (NABC) is outlined as follows.

Algorithm 1 (NABC). Consider the following steps.

Step 1 (initialization). Preset SN, Limit, and Maxcycle.

Step 2. Randomly generate SN solutions from the search space to create an initial population P.

Step 3. Calculate the function values f of the initial population P.

Step 4. The employed bees stage. Generate a new solution $v_{i,j}$ by (5), if $f(v_{i,j}) < f(x_{i,j})$, $x_{i,j} = v_{i,j}$, $\text{trail}_i = 1$, else $\text{trail}_i = \text{trail}_i + 1$.

Step 5. Calculate the probability values p_i for the solutions $x_{i,j}$ by (3).

Step 6 (the onlookers stage). If rand $< p_i$, generate a new solution $v_{i,j}$ by (6), if $f(v_{i,j}) < f(x_{i,j})$, $x_{i,j} = v_{i,j}$, $\text{trail}_i = 1$, else $\text{trail}_i = \text{trail}_i + 1$.

Step 7 (the scouts stage). If trail > Limit, replace $x_{i,j}$ with a new produced solution by (1).

Step 8. If cycle number > Maxcycle, stop and memorize the best solution achieved so far; otherwise, go to Step 2.

4. Numerical Functions and Experimental Results

4.1. Numerical Functions. It is applied to minimize a set of 11 scalable benchmark functions in Table 1 to test the efficiency of the novel ABC algorithm. The smaller the final result, the better it is.

4.2. Parameter Settings. A set of experiments tested on 11 numerical benchmark functions are performed in this section to compare the performance of the novel ABC algorithm with that of other improved ABC algorithms. Functions f_1–f_4 are unimodal. Function f_5 is a discontinuous step function. The Rosenbrock function of f_6 which is unimodal when $D = 2$ and 3 may have multiple minima in the high dimension case [18]. Functions f_7–f_{11} are multimodal. To compare with other variance algorithms, the dimension D of every function is 30, 60, or 100, respectively, the same as [15]. The dimension D of f_{11} is 150, 300, or 500. The maximum number of generations is 1500, 3000, or 5000, respectively. The population size is 100, namely, SN = 50, and Limit is 200, which is the same as [15]. Each of the experiments is repeated 30 times independently.

4.3. Experimental Results. Figure 1 graphically presents the convergence curves of f_1–f_{10} functions, respectively. We can see from Figure 1 that, in the later stage of evolution, the standard ABC algorithm enters a long period of stagnation.

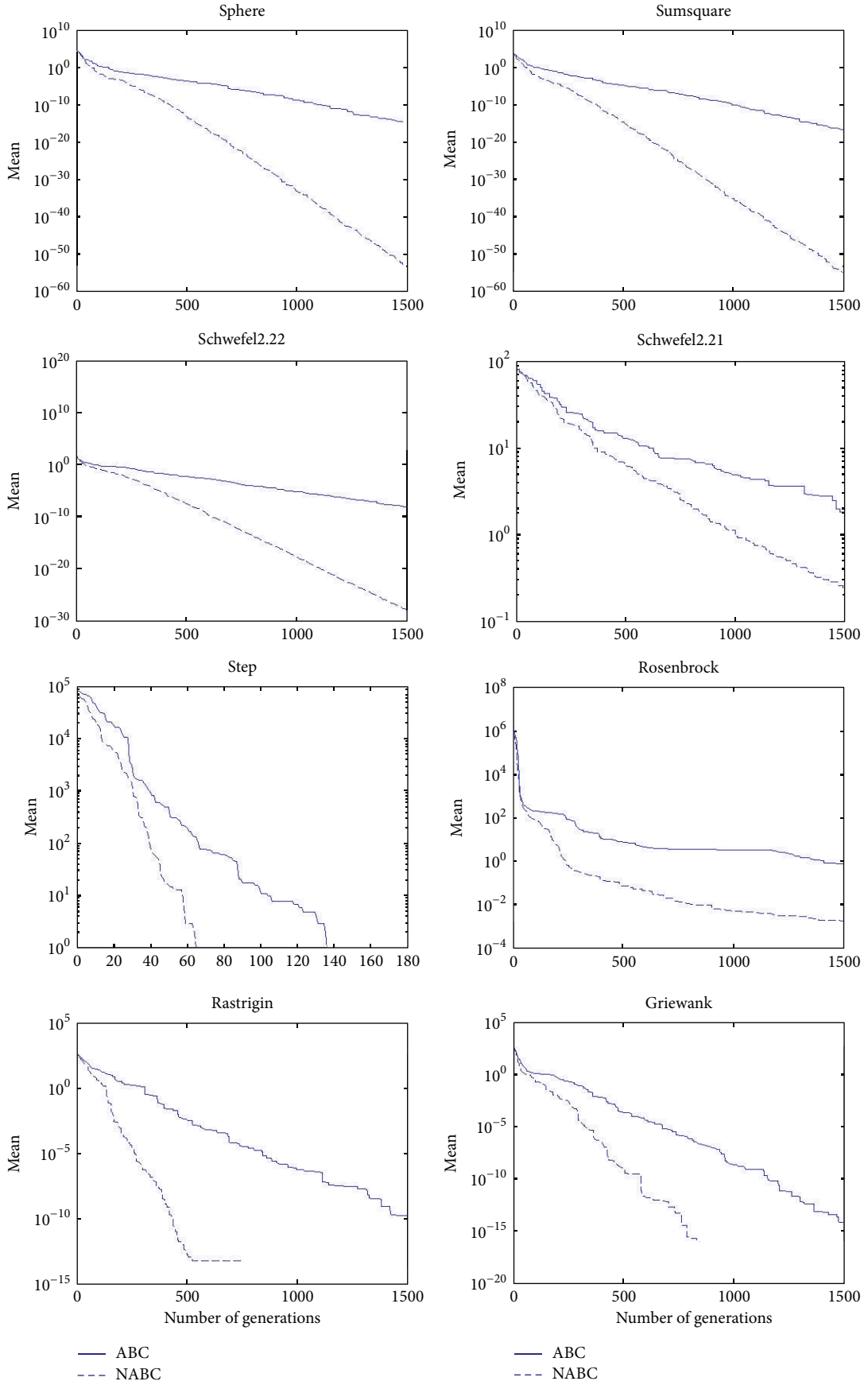

Figure 1: Continued.

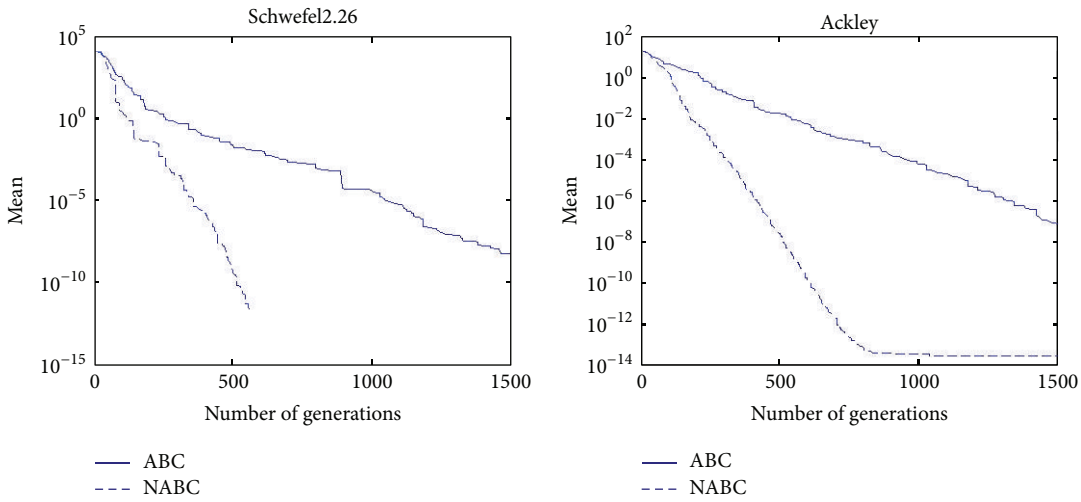

Figure 1: Convergence curves of ABC and NABC algorithms for f_1–f_{10} functions ($D = 30$).

However, NABC still keeps fast convergence and best performance.

The results of NABC compared with the standard ABC algorithm are shown in Table 2 in terms of the best, worst, median, mean, and standard deviation of the solutions. These results show that the convergence rate of NABC is better than the standard ABC algorithm for most test functions. The rate of convergence obviously increases on f_1–f_3, f_5, f_7–f_{11} functions from Figure 1 and Table 2. NABC can find the optimal solutions on functions f_5, f_7, f_8. The convergence rate of NABC is the same order of magnitude as the standard ABC algorithm on f_6. So the superiority of standard ABC algorithm is not very obvious. From the results in Table 2, NABC can obtain better and closer-to-optimal solutions than the standard ABC algorithm for most of the functions. In a word, NABC increases the exploitation greatly. NABC also retains good exploration. Here, "+" indicates that NABC is statistically significantly better than its corresponding competitor algorithm; "≈" stands for that the result of the corresponding algorithm is statistically similar with that of NABC.

NABC is further compared with GABC [13] and EABC [15] in Tables 3 and 4. The results of GABC and EABC are gained from [15] directly. It is clear that NABC works better in almost all the cases and achieves better performance than GABC. The rate of convergence of NABC increases obviously on all functions in Tables 3 and 4. The convergence rate of NABC is a little worse than that of EABC from Table 3. In particular, the convergence rate of Schwefel2.21 and Rosenbrock is better than EABC in Table 4.

At the same time, NABC is compared with four variants of DE (DE [17], jDE [18], JADE [19], and SaDE [20]) and five variants of PSO (FIPS [21], HPSO-TVAC [22], CLPSO [23], FPSO [24], OLPSO-G [25]). The results of DEs and PSOs are gained from [15] directly. The maximum number of function evaluations (FEs) is shown in Tables 5 and 6. Here, FEs is Maxgen∗SN. It can be seen that NABC is better on almost all the test functions except Ackley in JADE and OLPSO-G.

In Table 7, we can find the execution time of ABC is similar with that of NABC. So, NABC is largely faster than ABC and does not add the execution time.

5. Application

5.1. Knapsack Problems. We know that Knapsack problem [26] is an engineering optimization problem. It can be defined an unconstrained optimization problem and described as follows:

$$\min f(x) = -\sum_{i=1}^{n} p_i x_i + \alpha * \max\left(0, \sum_{i=1}^{n} w_i x_i - c\right), \quad (7)$$

where w_i is each item's weight and p_i is each item's profit, respectively. The knapsack has a limited weight capability of c. The objective function of this problem is to pack the knapsack so that the items in it have the maximal total profit. The decision variable x_i is the value one if the item i is packed; otherwise, it is the value zero. And we assume that all profits and weights are positive, and all weights are smaller than the capacity of c. α is a penalty factor which is set to $10e - 20$ in this paper. Tables 8 and 9 are parameters of five standard knapsack problems. The experiments tested on five benchmark 0-1 knapsack problems are performed to compare the performance of NABC with that of standard ABC. The maximum number of generations is set to 2000. The population size is 100, namely, SN = 50, and Limit is 10. Each of the experiments is repeated 30 times independently.

5.2. Data Analysis. In Table 10, we can see that NABC can easily solve the optimum of Knapsack problems. And it can find the optimum in most cases except f_5. ABC has the worst performance, because we can see that it can not find the optimums for f_2 and f_5. In short, NABC has a better performance than ABC on solving the five knapsack problems.

Table 2: Best, worst, median, mean, and standard deviation values obtained by ABC and NABC through 30 independent runs on 11 functions.

Function	Dim		Best	Worst	Median	Mean	SD	Significant
f_1	30	ABC	$5.04e-16$	$7.25e-16$	$5.04e-16$	$6.06e-16$	$9.34e-17$	+
		NABC	$1.13e-55$	$9.66e-54$	$6.66e-55$	$1.10e-54$	$1.71e-54$	
	60	ABC	$1.40e-15$	$1.87e-15$	$1.52e-15$	$1.62e-15$	$2.02e-16$	+
		NABC	$1.03e-53$	$4.66e-52$	$9.06e-53$	$1.48e-52$	$1.37e-52$	
	100	ABC	$2.20e-15$	$9.32e-15$	$5.21e-15$	$3.36e-15$	$2.01e-15$	+
		NABC	$1.61e-52$	$1.21e-51$	$3.69e-51$	$1.45e-51$	$8.91e-52$	
f_2	30	ABC	$4.30e-16$	$6.34e-16$	$4.50e-16$	$5.03e-16$	$7.11e-17$	+
		NABC	$1.64e-57$	$7.82e-56$	$1.16e-56$	$2.70e-56$	$2.48e-56$	
	60	ABC	$1.43e-15$	$2.00e-15$	$1.62e-15$	$1.67e-15$	$1.84e-16$	+
		NABC	$4.43e-55$	$7.82e-53$	$8.01e-54$	$2.08e-53$	$2.33e-53$	
	100	ABC	$2.47e-15$	$3.66e-15$	$3.43e-05$	$3.19e-15$	$4.21e-16$	+
		NABC	$3.51e-53$	$5.75e-52$	$2.66e-52$	$3.01e-52$	$1.56e-52$	
f_3	30	ABC	$2.38e-11$	$1.34e-10$	$6.42e-11$	$7.05e-11$	$5.19e-11$	+
		NABC	$5.06e-30$	$4.11e-29$	$1.73e-29$	$2.02e-29$	$1.10e-29$	
	60	ABC	$9.02e-11$	$8.23e-10$	$5.12e-10$	$4.08e-10$	$1.57e-10$	+
		NABC	$7.12e-29$	$5.72e-28$	$2.57e-28$	$2.92e-28$	$1.54e-28$	
	100	ABC	$2.05e-09$	$5.36e-08$	$4.84e-09$	$3.67e-09$	$4.80e-09$	+
		NABC	$2.85e-28$	$2.81e-27$	$7.35e-28$	$1.04e-27$	$6.88e-28$	
f_4	30	ABC	$8.68e+00$	$1.20e+01$	$1.84e+01$	$1.10e+01$	$1.29e-00$	+
		NABC	$1.77e-01$	$3.18e-01$	$2.68e-01$	$2.67e-01$	$3.55e-02$	
	60	ABC	$4.53e+01$	$5.08e+01$	$5.02e+01$	$4.86e+01$	$1.90e-00$	+
		NABC	$1.90e+00$	$3.16e+00$	$2.74e+00$	$2.71e+00$	$3.14e-01$	
	100	ABC	$6.92e+01$	$7.43e+01$	$7.26e+01$	$7.25e+01$	$1.86e-00$	+
		NABC	$8.32e+00$	$1.10e+01$	$9.79e+00$	$9.92e+00$	$7.04e-01$	
f_5	30	ABC	0	0	0	0	0	
		NABC	0	0	0	0	0	≈
	60	ABC	0	0	0	0	0	
		NABC	0	0	0	0	0	≈
	100	ABC	0	0	0	0	0	
		NABC	0	0	0	0	0	≈
f_6	30	ABC	$8.12e-03$	$1.68e-01$	$1.33e-02$	$4.86e-02$	$4.84e-02$	≈
		NABC	$2.21e-03$	$2.31e-01$	$3.32e-02$	$5.99e-02$	$5.64e-02$	
	60	ABC	$1.08e-02$	$5.88e-01$	$1.56e-01$	$1.41e-01$	$1.65e-01$	
		NABC	$2.88e-04$	$3.88e-01$	$4.73e-02$	$1.02e-01$	$9.95e-02$	≈
	100	ABC	$3.57e-02$	$9.94e-01$	$1.45e-01$	$2.97e-01$	$3.64e-01$	
		NABC	$2.92e-04$	$1.03e+00$	$2.16e-01$	$3.50e-01$	$3.28e-01$	≈
f_7	30	ABC	$5.36e-15$	$1.46e-01$	$4.56e-09$	$9.58e-04$	$1.05e-03$	+
		NABC	0	0	0	0	0	
	60	ABC	$5.87e-12$	$1.99e-00$	$1.25e-09$	$2.68e-02$	$6.32e-02$	+
		NABC	0	0	0	0	0	
	100	ABC	$8.65e-13$	$1.99e-00$	$1.15e-06$	$4.97e-02$	$2.16e-01$	+
		NABC	0	0	0	0	0	
f_8	30	ABC	$8.01e-15$	$1.30e-12$	$8.14e-13$	$2.37e-13$	$4.18e-13$	+
		NABC	0	0	0	0	0	
	60	ABC	$8.32e-14$	$1.34e-11$	$7.20e-13$	$3.45e-12$	$2.89e-12$	+
		NABC	0	0	0	0	0	
	100	ABC	$5.36e-14$	$1.58e-09$	$4.27e-10$	$1.80e-10$	$2.03e-10$	+
		NABC	0	0	0	0	0	

Table 2: Continued.

Function	Dim		Best	Worst	Median	Mean	SD	Significant
f_9	30	ABC	1.54e − 06	2.37e + 02	3.76e − 01	8.86e + 01	8.62e + 01	+
		NABC	−3.64e − 12	−1.82e − 12	−1.82e − 12	−2.00e − 12	5.55e − 13	
	60	ABC	3.55e + 02	7.69e + 02	7.69e + 02	5.40e + 02	1.41e + 02	+
		NABC	2.91e − 11	3.64e − 11	3.64e − 11	3.54e − 11	2.52e − 12	
	100	ABC	7.81e + 02	1.55e + 03	1.51e + 03	1.29e + 03	2.23e + 02	+
		NABC	1.02e − 10	1.16e − 10	1.09e − 10	1.10e − 10	4.58e − 12	
f_{10}	30	ABC	5.36e − 10	1.23e − 08	3.67e − 09	1.45e − 09	2.37e − 09	+
		NABC	2.22e − 14	2.93e − 14	2.93e − 14	2.66e − 14	3.32e − 15	
	60	ABC	8.14e − 09	7.54e − 08	5.38e − 08	4.60e − 08	2.04e − 08	+
		NABC	5.06e − 14	6.84e − 14	6.48e − 14	6.44e − 14	5.26e − 15	
	100	ABC	5.35e − 08	4.28e − 07	1.04e − 07	2.83e − 07	1.57e − 07	+
		NABC	1.04e − 13	1.22e − 13	1.15e − 13	1.16e − 13	4.75e − 15	
f_{11}	150	ABC	−77.4221	−76.9602	−77.0460	−77.1658	2.11e − 01	+
		NABC	−78.3323	−78.3323	−78.3323	−78.3323	1.45e − 14	
	300	ABC	−77.0391	−76.8430	−77.0391	−76.9258	6.76e − 02	+
		NABC	−78.3323	−78.3323	−78.3323	−78.3323	1.45e − 14	
	500	ABC	−76.8562	−76.5849	−76.8562	−76.7220	9.52e − 02	+
		NABC	−78.3323	−78.3323	−78.3323	−78.3323	1.45e − 14	

Table 3: Comparison among GABC, EABC, and NABC on optimizing 8 benchmark functions.

Function	Max.FEs	Dim	GABC		EABC		NABC	
			Mean	SD	Mean	SD	Mean	SD
Sphere	150,000	30	1.37e − 25	2.70e − 25	4.42e − 67	2.71e − 67	1.10e − 54	1.71e − 54
	300,000	60	4.86e − 23	5.14e − 23	2.30e − 64	1.03e − 64	1.48e − 52	1.37e − 52
	500,000	100	9.05e − 22	4.28e − 22	6.37e − 63	2.55e − 63	1.45e − 51	8.91e − 52
Schwefel2.22	150,000	30	5.56e − 15	9.79e − 15	5.51e − 35	7.02e − 35	2.02e − 29	1.10e − 29
	300,000	60	5.61e − 14	9.93e − 15	5.08e − 33	1.95e − 34	2.92e − 28	1.54e − 28
	500,000	100	2.29e − 13	3.97e − 13	4.68e − 32	1.42e − 32	1.04e − 27	6.88e − 28
Schwefel2.21	150,000	30	3.07e − 00	4.72e − 01	6.46e − 01	1.03e − 01	2.67e − 01	3.55e − 02
	300,000	60	3.85e + 01	3.42e − 00	2.49e + 01	2.04e − 00	2.71e + 00	3.14e − 01
	500,000	100	7.04e + 01	1.20e − 00	6.19e + 01	1.24e − 00	9.92e + 00	7.04e − 01
Rosenbrock	150,000	30	1.30e − 00	1.64e − 00	8.67e − 02	7.48e − 02	5.99e − 02	5.64e − 02
	300,000	60	1.66e + 01	3.28e + 01	2.03e − 01	1.32e − 01	1.02e − 01	9.95e − 02
	500,000	100	2.39e + 01	3.34e + 01	5.03e − 01	8.82e − 01	3.50e − 01	3.28e − 01
Griewank	100,000	30	1.40e − 08	1.16e − 08	0	0	0	0
	150,000	60	1.58e − 06	1.00e − 06	0	0	0	0
	250,000	100	2.44e − 06	2.50e − 06	0	0	0	0
Rastrigin	100,000	30	7.08e − 03	1.19e − 03	0	0	0	0
	150,000	60	3.47e − 00	8.78e − 01	0	0	0	0
	250,000	100	9.78e − 00	2.91e − 00	0	0	0	0
Schwefel2.26	50,000	30	6.38e − 00	2.38e + 01	0	0	1.71e − 09	3.01e − 10
	100,000	60	3.30e + 01	4.94e + 01	3.92e − 11	3.56e − 12	1.24e − 09	5.02e − 09
	200,000	100	6.05e + 01	8.05e + 01	1.12e − 10	3.56e − 12	1.22e − 10	2.97e − 11
Ackley	50,000	30	6.45e − 03	2.68e − 03	3.39e − 10	6.95e − 11	2.04e − 08	4.72e − 09
	100,000	60	2.96e − 03	2.14e − 03	1.81e − 09	2.70e − 10	4.45e − 08	1.50e − 08
	150,000	100	6.01e − 02	9.60e − 02	4.53e − 08	1.23e − 08	6.01e − 07	3.23e − 07

Table 4: Comparison among ABC, EABC, and NABC on optimizing 4 benchmark functions with $D = 30, 60, 100$.

Function	Dim	ABC		EABC		NABC	
		Mean	SD	Mean	SD	Mean	SD
Schwefel2.21	30	1.10e + 01	1.29e − 00	6.46e − 01	1.03e − 01	2.67e − 01	3.55e − 02
	60	4.86e + 01	1.90e − 00	2.49e + 01	2.04e − 00	2.71e + 00	3.14e − 01
	100	7.25e + 01	1.86e − 00	6.19e + 01	1.24e − 00	9.92e + 00	7.04e − 01
Rosenbrock	30	4.86e − 02	4.84e − 02	8.67e − 02	7.48e − 02	5.99e − 02	5.64e − 02
	60	1.41e − 01	1.65e − 01	2.03e − 01	1.32e − 01	1.02e − 01	9.95e − 02
	100	2.97e − 01	3.64e − 01	5.03e − 01	8.82e − 01	3.50e − 01	3.28e − 01
Schwefel2.26	30	8.86e + 01	8.62e + 01	−1.23e − 13	1.09e − 13	−2.00e − 12	5.55e − 13
	60	5.40e + 02	1.41e + 02	2.91e − 11	0	3.54e − 11	2.52e − 12
	100	1.29e + 03	2.23e + 02	9.45e − 11	0	1.10e − 10	4.58e − 12
Ackley	30	1.45e − 09	2.37e − 09	1.36e − 14	1.74e − 15	2.66e − 14	3.32e − 15
	60	4.60e − 08	2.04e − 08	4.49e − 14	2.84e − 15	6.44e − 14	5.26e − 15
	100	2.83e − 07	1.57e − 07	9.53e − 14	2.84e − 15	1.16e − 13	4.75e − 15

Table 5: Comparison among NABC, DE, jDE, JADE, and SaDE on optimizing 8 benchmark functions with $D = 30$.

Function	Max.FEs		DE	jDE	JADE	SaDE	NABC
Sphere	150,000	Mean	9.8e − 14	1.46e − 28	1.32e − 54	3.28e − 20	1.10e − 54
		SD	8.4e − 14	1.78e − 28	9.22e − 54	3.62e − 20	1.71e − 54
Schwefel2.22	200,000	Mean	1.6e − 09	9.02e − 24	3.18e − 25	3.51e − 25	7.20e − 40
		SD	1.1e − 09	6.01e − 24	2.05e − 25	2.74e − 25	3.33e − 40
Step	10,000	Mean	4.7e + 03	6.13e + 02	5.62e + 00	5.07e + 01	0
		SD	1.1e + 03	1.72e + 02	1.87e + 00	1.34e + 01	0
Rosenbrock	300,000	Mean	2.1e + 00	1.3e + 01	3.2e − 01	2.1e + 01	4.22e − 02
		SD	1.5e + 00	1.4e + 01	1.1e + 00	7.8e + 00	4.93e − 02
Griewank	50,000	Mean	2.0e − 01	7.29e − 06	1.57e − 08	2.52e − 09	1.09e − 08
		SD	1.1e − 01	1.05e − 05	1.09e − 07	1.24e − 08	2.86e − 08
Rastrigin	100,000	Mean	1.8e + 02	3.32e − 04	1.33e − 01	2.43e + 00	0
		SD	1.3e + 01	6.39e − 04	9.74e − 02	1.60e + 00	0
Ackley	50,000	Mean	1.1e − 01	2.37e − 04	3.35e − 09	3.81e − 06	2.04e − 08
		SD	3.9e − 02	7.10e − 05	2.84e − 09	8.26e − 07	4.72e − 09
Schwefel2.26	100,000	Mean	5.9e + 03	1.70e − 10	2.62e − 04	1.13e − 08	−1.94e − 12
		SD	1.1e + 03	1.71e − 10	3.59e − 04	1.08e − 08	4.61e − 13

Table 6: Comparison among NABC, FIPS, HPSO-TVAC, CLPSO, FPSO, and OLPSO-G on optimizing 8 benchmark functions with 200,000 FEs.

Function		FIPS	HPSO-TVAC	CLPSO	FPSO	OLPSO-G	NABC
Sphere	Mean	2.42e − 13	2.83e − 33	1.58e − 12	2.40e − 16	4.12e − 54	5.45e − 75
	SD	1.73e − 13	3.19e − 33	7.70e − 13	2.00e − 31	6.34e − 54	3.55e − 75
Schwefel2.22	Mean	2.76e − 08	9.03e − 20	2.51e − 08	1.58e − 11	9.85e − 30	7.20e − 40
	SD	9.04e − 09	9.58e − 20	5.84e − 09	1.03e − 22	1.01e − 29	3.33e − 40
Step	Mean	0	0	0	0	0	0
	SD	0	0	0	0	0	0
Rosenbrock	Mean	2.51e + 01	2.39e + 01	1.13e + 01	2.81e + 01	2.15e + 01	6.33e − 02
	SD	5.10e − 01	2.65e + 01	9.85e − 00	2.31e + 02	2.99e + 01	6.34e − 02
Griewank	Mean	9.01e − 12	9.75e − 03	9.02e − 09	1.47e − 03	4.83e − 03	0
	SD	1.84e − 11	8.33e − 03	8.57e − 09	1.28e − 05	8.63e − 03	0
Rastrigin	Mean	6.51e + 01	9.43e − 00	9.09e − 05	7.38e + 01	1.07e − 00	0
	SD	1.33e + 01	3.48e − 00	1.25e − 04	3.70e + 02	9.92e − 01	0
Ackley	Mean	2.33e − 07	7.29e − 14	3.66e − 07	2.17e − 09	7.98e − 15	2.78e − 14
	SD	7.19e − 08	3.00e − 14	7.57e − 08	1.71e − 18	2.03e − 15	2.59e − 15
Schwefel2.26	Mean	9.93e + 02	1.59e + 03	3.82e − 04	1.34e + 03	3.84e + 02	−2.43e − 12
	SD	5.09e + 02	3.26e + 02	1.28e − 05	2.77e + 02	2.17e + 02	8.72e − 13

Table 7: Execution time of ABC and NABC algorithm with $D = 30, 60, 100$ (second).

Dim	Algorithm	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}
30	ABC	19.850	25.535	33.363	18.768	3.969	32.338	40.942	78.680	94.332	47.626
	NABC	17.661	25.697	36.352	20.056	2.768	30.612	18.229	31.331	77.497	42.660
60	ABC	43.236	72.503	104.475	47.454	12.304	81.891	185.687	259.944	322.573	205.303
	NABC	44.900	73.449	108.912	47.991	8.202	80.281	95.126	129.497	265.051	186.870
100	ABC	89.402	163.012	255.115	92.929	37.074	180.789	371.414	574.304	841.482	410.198
	NABC	88.507	168.964	255.134	91.418	21.081	175.828	334.624	308.107	687.463	365.939

Table 8: The five benchmark knapsack problems.

f	Dim	Parameter (w, p, c)
f_1	10	w = (95, 4, 60, 32, 23, 72, 80, 62, 65, 46); p = (55, 10, 47, 5, 4, 50, 8, 61, 85, 87); c = 269
f_2	20	w = (92, 4, 43, 83, 84, 68, 92, 82, 6, 44, 32, 18, 56, 83, 25, 96, 70, 48, 14, 58); p = (44, 46, 90, 72, 91, 40, 75, 35, 8, 54, 78, 40, 77, 15, 61, 17, 75, 29, 75, 63); c = 878
f_3	4	w = (6, 5, 9, 7); p = (9, 11, 13, 15); c = 20
f_4	4	w = (2, 4, 6, 7); p = (6, 10, 12, 13); c = 11
f_5	15	w = (56.358531, 80.874050, 47.987304, 89.596240, 74.660482, 85.894345, 51.353496, 1.498459, 36.445204, 16.589862, 44.569231, 0.466933, 37.788018, 57.118442, 60.716575); p = (0.125126, 19.330424, 58.500931, 35.029145, 82.284005, 17.410810, 71.050142, 30.399487, 9.140294, 14.731285, 98.852504, 11.908322, 0.891140, 53.166295, 60.176397); c = 375;

Table 9: The optimal solutions of the five benchmark knapsack problems.

f	Optimal solution	Optimal value	Value of constraint
f_1	(0, 1, 1, 1, 0, 0, 0, 1, 1, 1)	295	0
f_2	(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 1)	1024	−7
f_3	(1, 1, 0, 1)	35	−2
f_4	(0, 1, 0, 1)	23	0
f_5	(0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1)	481.0694	−20.0392

Table 10: Best, worst, median, mean, and standard deviation values obtained by ABC and NABC.

f	Algorithm	Best	Worst	Median	Mean	SD
f_1	ABC	295	295	295	295	0
	NDABC	295	295	295	295	0
f_2	ABC	1024	1013	1024	1021	4.52
	NDABC	1024	1024	1024	1024	0
f_3	ABC	35	35	35	35	0
	NDABC	35	35	35	35	0
f_4	ABC	23	23	23	23	0
	NDABC	23	23	23	23	0
f_5	ABC	481.069	437.935	475.478	460.729	18.47
	NDABC	481.069	435.786	481.069	470.771	8.79

6. Conclusion

In this paper, we develop a novel artificial bee colony algorithm named NABC. We add the global best solution into the search equation to drive the new candidate solution only around the global best solution in order to improve the exploitation. And the search equation of the employed bees is improved to keep the exploration of algorithm. The experimental results tested on 11 benchmark functions show that the convergence of NABC is much faster than that of other algorithms, and the computing is more effective. At the same time, we apply NABC on solving five standard Knapsack problems and get good optimums. So it is fitted to solve many engineering practical problems.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgments

The authors are grateful to the editor and the anonymous reviewers for their valuable comments and suggestions.

References

[1] K. S. Tang, K. F. Man, S. Kwong, and Q. He, "Genetic algorithms and their applications," *IEEE Signal Processing Magazine*, vol. 13, no. 6, pp. 22–37, 1996.

[2] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, Australia, December 1995.

[3] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53–66, 1997.

[4] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.

[5] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Tech. Rep. TR06, Erciyes University, Kayseri, Turkey, 2005.

[6] B. Basturk and D. Karaboga, "An artificial bee colony (ABC) algorithm for numeric function optimization," in *Proceedings of the IEEE Swarm Intelligence Symposium*, May 2006.

[7] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[8] D. Karaboga and B. Basturk, "On the performance of artificial bee colony (ABC) algorithm," *Applied Soft Computing Journal*, vol. 8, no. 1, pp. 687–697, 2008.

[9] D. Karaboga and B. Akay, "A comparative study of artificial Bee colony algorithm," *Applied Mathematics and Computation*, vol. 214, no. 1, pp. 108–132, 2009.

[10] D. Karaboga, B. Akay, and C. Ozturk, "Artificial bee colony (ABC) optimization algorithm for training feed-forward neural networks," in *Modeling Decisions for Artificial Intelligence*, vol. 4617 of *Lecture Notes in Computer Science*, pp. 318–329, Springer, Berlin, Germany, 2007.

[11] Y.-F. Liu and S.-Y. Liu, "A hybrid discrete artificial bee colony algorithm for permutation flowshop scheduling problem," *Applied Soft Computing Journal*, vol. 13, no. 3, pp. 1459–1463, 2013.

[12] D. Karaboga and B. Basturk, "Artificial bee colony (ABC) optimization algorithm for solving constrained optimization problems," in *Foundations of Fuzzy Logic and Soft Computing*, vol. 4529 of *Lecture Notes in Computer Science*, pp. 789–798, Springer, Berlin, Germany, 2007.

[13] G. P. Zhu and S. Kwong, "Gbest-guided artificial bee colony algorithm for numerical function optimization," *Applied Mathematics and Computation*, vol. 217, no. 7, pp. 3166–3173, 2010.

[14] B. Basturk and D. Karaboga, "A modified artificial bee colony algorithm for real-parameter optimization," *Information Sciences*, vol. 192, pp. 120–142, 2012.

[15] W.-F. Gao, S.-Y. Liu, and L.-L. Huang, "Enhancing artificial bee colony algorithm using more information-based search equations," *Information Sciences*, vol. 270, pp. 112–133, 2014.

[16] D. Karaboga and B. Gorkemli, "A quick artificial bee colony (qABC) algorithm and its performance on optimization problems," *Applied Soft Computing*, vol. 23, pp. 227–238, 2014.

[17] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 23, pp. 689–694, 2010.

[18] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[19] 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.

[20] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2008.

[21] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 204–210, 2004.

[22] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 240–255, 2004.

[23] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[24] M. A. M. de Oca, T. Stützle, M. Birattari, and M. Dorigo, "Frankenstein's PSO: a composite particle swarm optimization algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 1120–1132, 2009.

[25] Z.-H. Zhan, J. Zhang, Y. Li, and Y.-H. Shi, "Orthogonal learning particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 6, pp. 832–847, 2011.

[26] D. Zou, L. Gao, S. Li, and J. Wu, "Solving 0-1 knapsack problem by a novel global harmony search algorithm," *Applied Soft Computing Journal*, vol. 11, no. 2, pp. 1556–1564, 2011.