Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 739437, 14 pages
http://dx.doi.org/10.1155/2015/739437

Research Article

Artificial Bee Colony Algorithm Combined with Grenade Explosion Method and Cauchy Operator for Global Optimization

Jian-Guo Zheng,[1] Chao-Qun Zhang,[1,2] and Yong-Quan Zhou[2]

[1]*Glorious Sun School of Business and Management, Donghua University, Shanghai 200051, China*
[2]*College of Information Science and Engineering, Guangxi University for Nationalities, Nanning 530006, China*

Correspondence should be addressed to Chao-Qun Zhang; chaozi_0771@163.com

Received 11 June 2015; Accepted 15 July 2015

Academic Editor: Xiaoyu Song

Artificial bee colony (ABC) algorithm is a popular swarm intelligence technique inspired by the intelligent foraging behavior of honey bees. However, ABC is good at exploration but poor at exploitation and its convergence speed is also an issue in some cases. To improve the performance of ABC, a novel ABC combined with grenade explosion method (GEM) and Cauchy operator, namely, ABCGC, is proposed. GEM is embedded in the onlooker bees' phase to enhance the exploitation ability and accelerate convergence of ABCGC; meanwhile, Cauchy operator is introduced into the scout bees' phase to help ABCGC escape from local optimum and further enhance its exploration ability. Two sets of well-known benchmark functions are used to validate the better performance of ABCGC. The experiments confirm that ABCGC is significantly superior to ABC and other competitors; particularly it converges to the global optimum faster in most cases. These results suggest that ABCGC usually achieves a good balance between exploitation and exploration and can effectively serve as an alternative for global optimization.

1. Introduction

Algorithms dealing with solving optimization problems can be classified into different groups according to their characteristics such as population-based, stochastic, deterministic, and iterative. An algorithm working with a group of solutions and trying to improve them is called population-based [1]. Two important classes of population-based optimization algorithms are evolutionary algorithms and swarm intelligence-based algorithms [2]. Swarm intelligence is an innovative artificial intelligence technique based on collective behavior of self-organized systems [3]. One of the most recent population-based methods is artificial bee colony (ABC) algorithm [4] which simulates the intelligent behavior of a honey bee swarm for seeking a high quality food source in nature. Karaboga [4] first proposed ABC in 2005 for numerical optimization, and then afterwards Karaboga and his collaborators [5–9] applied it to train feed-forward neural network and solve integer programming problems, data clustering, constrained optimization problems, and real-parameter optimization problems. Their results [2, 4–6, 8–11] demonstrate that ABC is simple in concept, few in parameters, easy for implementation, and more effective than some other population-based algorithms such as genetic algorithm (GA), particle swarm optimization (PSO), ant colony optimization (ACO), and differential evolution (DE) algorithm. Therefore, ABC has aroused much interest and has been successfully used in different fields [1–20].

Exploration and exploitation are extremely important mechanisms in a robust search process. Exploration is the ability to search the space to find promising new solutions, while exploitation is the ability to find the optimum in the neighborhood of a good solution [9]. Unfortunately, ABC is good at exploration but poor at exploitation and its convergence speed is also an issue in some cases [12–18], so a lot of its variants [1, 3, 7–9, 12–20] have been proposed in recent years to further improve the performance of ABC. For example, [13] introduced a gbest-guided ABC (GABC) by incorporating the information of global best (gbest) solution into the solution search equation to improve the exploitation;

[14] proposed a novel ABC with Powell's method (PABC); that is, the search equation in the onlooker bees' phase is modified and Powell's method is further used as a local search tool to improve the search ability; [15] designed a modified ABC combined with the best solution, chaotic systems, and opposition-based learning method (ABCbest) in which the initial population and scout bees are generated by combining chaotic systems with opposition-based learning method, and each bee searches only around the best solution of the previous iteration to improve the exploitation; [16] presented a new ABC based on modified search equation and orthogonal learning (CABC) where a modified search equation is applied to generate a candidate solution to improve the search ability of ABC, and the orthogonal experimental design is used to form an orthogonal learning strategy for variant ABCs to discover more useful information from the search experiences; [17] developed a bare bone ABC (BABC) which uses a Gaussian search equation to produce a new candidate individual in the onlooker bees' phase and integrates a parameter adaptation strategy and a fitness-based neighborhood mechanism into the employed bees' phase for better performance; moreover, the proposed framework was applied to GABC [13], ABCbest [15], and CABC [16], and the corresponding advanced ABCs were termed as BGABC, BABCbest, and BCABC, respectively. The experiments suggest that ABC variants perform competitively and effectively. Note that most of the papers which are based on improvement of ABC try to improve the local search capability (i.e., exploitation ability) [21]. However, there is no specific algorithm to substantially achieve the best solution for all optimization problems. Some algorithms only give a better solution for some particular problems than others. Hence it is very necessary to search for a well-improved or new optimization method.

To improve the performance of ABC in terms of poor balance between exploitation and exploration and convergence speed, we notice that grenade explosion method (GEM) [22–25] which mimics the mechanism of a grenade explosion usually converges to the global minimum faster than some other methods such as GA and ABC. Thereby, two modified versions of ABC inspired by GEM, namely, GABC1 and GABC2, were first proposed to enhance the basic ABC's exploitation ability in [18]. GEM is embedded in the employed bees' phase of GABC1, whereas it is embedded in the onlooker bees' phase of GABC2. The experiments show that GABC1 has similar or better performance than GABC2 in most cases, but GABC2 performs more robustly and effectively than GABC1; they significantly outperform the competitors. In addition, a mutation operator based on Cauchy random numbers [26], namely, Cauchy operator, is appropriate for global search because of its higher probability of making longer jumps. Thus a hybrid algorithm ABCGC which combines ABC with GEM and Cauchy operator is proposed for global optimization. GEM is embedded in the onlooker bees' phase, which greatly enhances the exploitation ability and accelerates convergence of ABCGC; meanwhile, Cauchy operator is introduced into the scout bees' phase to help ABCGC jump out of local optimum and further enhance its exploration ability. To evaluate the performance of the proposed algorithm, a set of 22 well-known benchmark functions is tested among ABCGC, ABC, and 8 ABC variants. Furthermore, a set of 6 classical functions is used to compare ABCGC with ABC and other 5 state-of-the-art algorithms. Besides, the effects of each modification on the performance of ABCGC are analyzed on the same 6 functions.

The rest of this paper is organized as follows. The basic ABC is introduced in Section 2. In Section 3, the proposed algorithm is described in detail. The performance of ABCGC, ABC, and several other compared algorithms is compared and discussed in Section 4. Finally, conclusions and future work are presented in Section 5.

2. Artificial Bee Colony Algorithm

A honey bee colony manages to discover the highest quality food sources in nature. Thereby, ABC [4] takes concepts from honey bee intelligent foraging behavior to discover good solutions in an optimization problem. In ABC, the colony of artificial bees contains three groups: employed, onlooker, and scout bees. The first half of the colony consists of employed bees and the second half includes onlooker bees. Employed bees search food sources and share the information about these food sources to recruit onlooker bees. Onlooker bees select the food sources found by all employed bees according to the probability proportional to the quality of food sources and further exploit the selected food sources. Scout bees are translated from a few employed bees, which abandon their food sources through a predetermined number of cycles called *limit* and search new ones. The position of a food source represents a possible solution to the optimization problem, and the profitability of a food source corresponds to the quality (fitness) of the associated solution. Each food source is exploited by only one employed or onlooker bee. In other words, the number of food sources is equal to the number of employed bees.

At the beginning of an optimization, an initial population containing SN solutions is generated randomly. SN is the number of food sources. Each solution (food source) X_i ($i = 1, 2, \ldots, SN$) is a D-dimensional vector and let $X_i = \{x_{i1}, x_{i2}, \ldots, x_{iD}\}$ represent the ith food source in the population, where D denotes the number of optimization parameters (dimension). And then, the population of the positions is subject to repeated cycles, $cycle = 1, 2, \ldots$, and maximum cycle number (MCN), of the search processes of the employed, onlooker, and scout bees.

In ABC, the fitness function is defined as

$$\text{fit}(X_i) = \begin{cases} \dfrac{1}{1 + f(X_i)} & f(X_i) \geq 0 \\ 1 + \text{abs}(f(X_i)) & f(X_i) < 0, \end{cases} \tag{1}$$

where $f(X_i)$ is the objective function value of solution X_i and $\text{fit}(X_i)$ is the fitness value of X_i.

The probability of a food source being selected by an onlooker bee can be presented by

$$p(X_i) = \frac{\text{fit}(X_i)}{\sum_{n=1}^{SN} \text{fit}(X_n)}. \tag{2}$$

Step 1. Preset the values of control parameters: D, SN, $limit$, MCN
Step 2. Initialize the population of food sources using (4)
Step 3. Evaluate the population using (1)
Step 4. $cycle = 1$
Step 5. repeat
Step 6. Produce new food sources for the employed bees and evaluate them,
then apply the greedy selection process {employed bees' phase}:
for $i = 1$ to SN
Produce a new food source V_i from X_i (based on X_k, $i \neq k$) using (3)
Calculate the fitness of the food source V_i using (1)
Apply the greedy selection between the new food source and the old one
end for
Step 7. Calculate the probability values for food sources using (2)
Step 8. Produce new food sources for the onlooker bees from the food source X_i selected depending on $p(X_i)$
and evaluate them, then apply the greedy selection process {onlooker bees' phase}:
for $i = 1$ to SN
if $random < p(X_i)$
Produce a new food source V_i from X_i (based on X_k, $i \neq k$) using (3)
Calculate the fitness of the food source V_i using (1)
Apply the greedy selection between the new food source and the old one
end if
end for
Step 9. Determine the abandoned food source for the scout bee, if exists,
and replace it with a new randomly produced one using (4) {scout bees' phase}
Step 10. Memorize the best food source achieved so far
Step 11. $cycle = cycle + 1$
Step 12. until $cycle = MCN$

Algorithm 1: Main steps and some related pseudocodes of ABC.

After an employed bee discovers or an onlooker bee selects the food source X_i, they exploit a neighboring food source V_i. V_i is determined by changing only one parameter of X_i, namely, $v_{ij} \neq x_{ij}$, while the rest of V_i keep the same value as X_i. v_{ij} is generated by

$$v_{ij} = x_{ij} + \phi_{ij}\left(x_{ij} - x_{kj}\right), \tag{3}$$

where $k \in \{1, 2, \ldots, SN\}$ is a randomly chosen index and k must be different from i, $j \in \{1, 2, \ldots, D\}$ is a randomly chosen dimension, and ϕ_{ij} is a random number in the range from -1 to 1. Note that after an employed or onlooker bee determines a new candidate food source in the neighborhood of its currently associated food source using (3), a greedy selection mechanism [8–11] is applied between the new food source and the old one; that is to say, if the new food source is better than the old one, it is substituted for the old one. Otherwise, the old one is retained in the memory.

If the abandoned food source is X_i, a scout bee produces a new food source according to

$$x_{ij} = x_{\min j} + \text{rand}\,(0, 1)\left(x_{\max j} - x_{\min j}\right), \tag{4}$$

where $x_{\min j}$ and $x_{\max j}$ are the lower and upper bounds of the variable x_{ij}, respectively.

The main steps and some related pseudocodes of ABC are outlined in Algorithm 1.

3. The Proposed Algorithm ABCGC

In this section, ABCGC which is a modified version of the basic ABC based on GEM and Cauchy operator to improve its performance is proposed. The proposed algorithm follows the general procedure of ABC. The essential difference between ABCGC and ABC is different exploitation and exploration strategies adopted by their onlooker bees and scout bees, respectively. That is to say, the main steps of ABCGC remain the same as ABC except for Steps 8 and 9. Figure 1 visualizes the framework for ABCGC.

3.1. New Exploitation Strategy Adopted by Onlooker Bees. In ABC, after an onlooker bee selects the food source X_i, it further exploits a neighboring food source V_i. V_i is determined by changing only one parameter of X_i; namely, $v_{ij} \neq x_{ij}$. v_{ij} is generated by (3). In (3), v_{ij} is modified from x_{ij} based on a comparison with the randomly selected position from its neighboring solution x_{kj}. As can be seen from (3), the difference between x_{ij} and x_{kj} is a difference of position in the randomly chosen dimension j. j is a crucial parameter since it directly influences the position of a new food source. However, the randomly chosen dimension j may not always guide ABC toward more high fitted positions of food sources and lead to slow convergence or even make the search easily trapped in local optimum. Then which dimension among

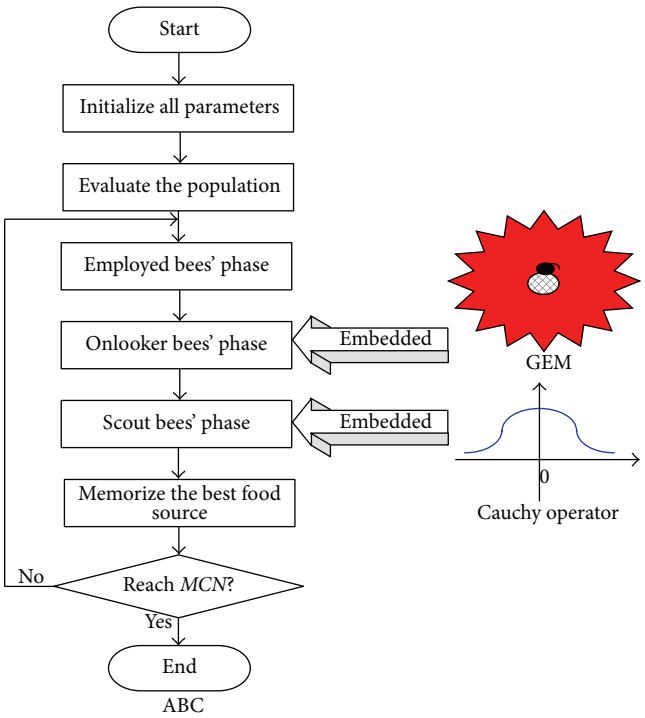

Figure 1: Framework for ABCGC.

all the dimensions is the best choice for an onlooker bee to update the new candidate food source?

GEM first presented by Ahrari et al. [22] in 2009 is inspired by the mechanism of a grenade explosion, where objects are hit by pieces of shrapnel. Damage caused by each piece of shrapnel hitting an object is calculated. A high value for damage per piece in an area indicates there are valuable objects in that area. To intensify the damage, the next grenade is thrown where the greatest damage occurs. This process will result in finding the best place for throwing the grenade. Ahrari et al. [22–25] used a set of classical benchmark functions and some randomly generated multimodal functions to test the performance of GEM; the results show that this simple and robust method can often spot high fitted regions quite fast and converge to the exact location of the global minimum.

Therefore, GEM is introduced into the onlooker bees' phase of ABC to select the optimal search dimension instead of a random chosen one for each onlooker bee in hope that they collectively move towards the optimal position. Here, the overall damage caused by the hit is considered as the "fitness" of a solution. Note that the number of pieces of shrapnel per grenade should be large enough so that far regions can be explored for new high fitted regions and the algorithm would not be trapped in local optimum. To eliminate the need for setting the parameters of GEM, there is only one grenade and let the grenade throw D pieces of shrapnel in each cycle.

In each cycle of ABCGC, D pieces of shrapnel are thrown in all the dimensions (i.e., each dimension is exploited by only one shrapnel) to gather information around the current position of the grenade (old food source); meanwhile, each onlooker bee computes each candidate food source along which each shrapnel is thrown and evaluates corresponding damage-per-shrapnel value (fitness) and then makes a decision on a new candidate food source with the greatest damage (the highest fitness), which means the selected optimal search dimension is biased towards the global or near-global optimal position more quickly. Consequently, in ABCGC, a new candidate solution based on the optimal search dimension for an onlooker bee is produced by

$$v_{iOSD} = x_{iOSD} + \phi_{iOSD}\left(x_{iOSD} - x_{kOSD}\right)$$
$$\text{s.t.} \quad \text{fit}\left(V_{iOSD}\right) = \max\left\{\text{fit}\left(V_{it}\right) \mid t = 1, 2, \ldots, D\right\}, \tag{5}$$

where $k \in \{1, 2, \ldots, SN\}$ is a randomly chosen index and $k \neq i$; $OSD \in \{1, 2, \ldots, D\}$ represents the optimal search dimension; ϕ_{iOSD} is a random number in the range from -1 to 1; V_{it} denotes a new candidate food source V_i generated by just changing the value of old food source X_i in dimension t, namely, $v_{it} \neq x_{it}$, while the rest of V_{it} keep the same value as X_{it}; V_{iOSD} has a similar meaning as V_{it} and also indicates that V_i obtains the maximum fitness in dimension OSD instead of other dimensions.

Similarly, after an onlooker bee determines a new candidate food source in the neighborhood of its currently associated food source using (5) and (1), a greedy selection mechanism is applied between the new food source and the old one.

From the above explanation, the pseudocodes of Step 8 of ABCGC are presented in Algorithm 2 (the main differences from ABC are highlighted in bold).

3.2. New Exploration Strategy Adopted by Scout Bees. Although a search can easily fall into local optimum, Cauchy operator ensures that the search is executed in the global region and does not trap in local optimum prematurely [30]. For example, [26] combined evolutionary programming (EP) with Cauchy operator, and the new EP significantly outperforms the classical EP with Gaussian mutation; [31] applied Bayesian techniques to enhance the PSO's searching ability in the exploitation of past particle positions and used Cauchy operator for exploring the better solution; [32] introduced Chaos and Cauchy operator into the improved biogeography-based optimization algorithm to search for the optimal solution of the core backbone network. Their results confirm that Cauchy operator is appropriate for global search due to its higher probability of making longer jumps. Figure 2 shows the probability density functions of standard Cauchy and Gaussian distributions. On the interval $[-3, 3]$, Gaussian function has a large probability, but its probability is almost 0 on the intervals $(-\infty, -3)$ and $(3, \infty)$. Cauchy distribution has a similar shape as Gaussian distribution, but it has more probability in its long tail area, so the possibility of Cauchy distribution to generate a random number away from the origin is higher than Gaussian distribution. In other words, Cauchy distribution-based random numbers explore a relatively wider search space than Gaussian distribution-based random numbers. Thereby, Cauchy operator is introduced into the scout bees' phase of ABC to generate a wider

Step 8. **Produce and evaluate new food sources for each onlooker bee in all dimensions of each associated food source according to its probability and determine the optimal search dimension** (*OSD*) **and the best new candidate food source using (5) and (1)**, then apply the greedy selection process {onlooker bees' phase}:

```
for i = 1 to SN
  if random < p(X_i)
    for t = 1 to D
      Produce a new food source V_it from X_i (based on X_k, i ≠ k) using (3)
      Calculate the fitness of a food source V_it using (1)
      if t = 1
        V_iOSD = V_i1
        fit(V_iOSD) = fit(V_i1)
        OSD = 1
      else
        if fit(V_it) > fit(V_iOSD)
          V_iOSD = V_it
          fit(V_iOSD) = fit(V_it)
          OSD = t
        end if
      end if
    end for
    Apply the greedy selection between the new food source and the old one
  end if
end for
```

Algorithm 2: Pseudocodes of Step 8 of ABCGC.

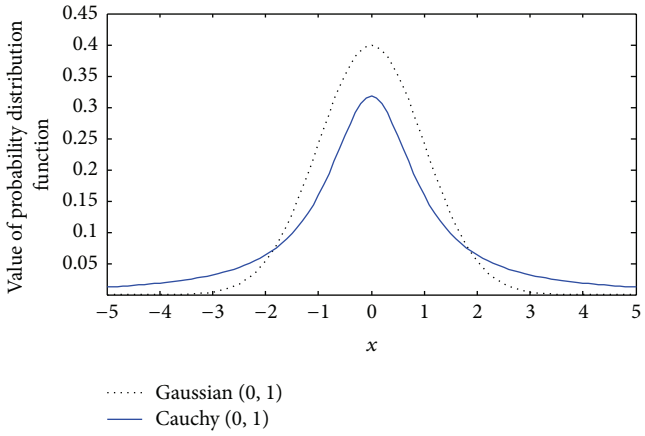

Figure 2: Probability density functions of standard Cauchy and Gaussian distributions.

solution instead of a random produced one for each scout bee. This will help ABC escape from local optimum and further enhance the ability of global exploration.

In ABCGC, if the abandoned food source is X_i, a scout bee produces a new food source according to

$$x_{ij} = x_{ij}\text{Cauchy}\,(0,1)\,, \tag{6}$$

where Cauchy$(0,1)$ is the standard Cauchy distribution, which denotes a random value from a Cauchy distribution centered at 0 with a scale parameter equal to 1. Concretely, Cauchy$(0,1)$ is defined as

$$\text{Cauchy}\,(0,1) = \frac{1}{\pi\left(1 + x_{ij}^2\right)}. \tag{7}$$

Based on above considerations, Step 9 of ABCGC is listed in Algorithm 3 (the main differences from ABC are highlighted in bold).

4. Experiments and Discussion

It is a common practice to compare different algorithms using different benchmark problems in the field of optimization [33]. In order to verify the performance of ABCGC, a set of 22 well-known benchmark functions is tested among the proposed algorithm, ABC, and 8 improved ABC algorithms. Besides, a set of 6 classical functions is used to compare ABCGC with ABC and other 5 state-of-the-art algorithms; moreover, the effects of each modification on the performance of ABCGC are analyzed on the same 6 functions. Table 1 shows the list of various algorithms used in the paper.

ABCGC is developed in MATLAB version 7.0.4.365(R14) and the MATLAB codes of ABC are downloaded from Karaboga's website (http://mf.erciyes.edu.tr/abc/software.htm). In order to make fair comparisons, the parameter settings for different methods are chosen to be the same in all our experiments. In ABC, ABCG, ABCC, and ABCGC, both the employed bees and the onlooker bees are 50% of the colony,

Step 9. Determine the abandoned food source for the scout bee, if exists, and replace it with **a new produced one based on Cauchy operator using (6) and (7)** {scout bees' phase}

Algorithm 3: Step 9 of ABCGC.

Table 1: List of various algorithms used in the paper.

Nomenclature	Algorithm	Author and reference
ABC	The basic artificial bee colony algorithm	Karaboga [4]
BABC	Bare bone artificial bee colony algorithm	Gao et al. [17]
GABC	Gbest-guided artificial bee colony algorithm	Zhu and Kwong [13]
BGABC	Bare bone GABC	Gao et al. [17]
PABC	Artificial bee colony algorithm with Powell's method	Gao et al. [14]
ABCbest	Artificial bee colony algorithm combined with the best solution, chaotic systems, and opposition-based learning method	Gao et al. [15]
BABCbest	Bare bone ABCbest	Gao et al. [17]
CABC	Artificial bee colony algorithm based on modified search equation and orthogonal learning	Gao et al. [16]
BCABC	Bare bone CABC	Gao et al. [17]
BSO-RPTVW	Bee swarm optimization with repulsion factor, penalizing fitness, and time-varying weights	Akbari et al. [27]
IMDE (1st process)	Intersect mutation differential evolution algorithm for 1st process	Zhou et al. [28]
IMDE (2nd process)	Intersect mutation differential evolution algorithm for 2nd process	Zhou et al. [28]
DFO	Drosophila food-search optimization algorithm where modified quadratic approximation is used	Das and Singh [29]
DFO (QA)	Drosophila food-search optimization algorithm where quadratic approximation is used	Das and Singh [29]
ABCG (i.e., GABC2)	Artificial bee colony algorithm combined with GEM	Zhang et al. [18]
ABCC	Artificial bee colony algorithm with Cauchy operator	Algorithm proposed in this paper
ABCGC	Artificial bee colony algorithm combined with GEM and Cauchy operator	Algorithm proposed in this paper

respectively, and the number of scout bees is selected as one. All experiments are repeated 50 times and run on a portable computer with an Intel Core i5-3317U 1.70 GHz CPU and 4 GB RAM under Windows 7. The mean and standard deviation of the function values found by different algorithms under the same conditions have been recorded. Since all the functions are minimization problems, a smaller value represents the better one. Results in boldface indicate the best values obtained by different algorithms for each function.

4.1. Comparison among ABC Algorithms. ABCGC, ABC, and 8 ABC variants are used to minimize a set of 22 well-known benchmark functions given in [14, 16, 17]. Tables 2 and 3 show the details of these functions. Different functions have different characteristics. A function is unimodal if it has only one optimum, while a multimodal function has more than one local optimum [2]. Multimodal functions are used to test the ability of algorithms getting rid of local minima. A function is separable if it can be rewritten as a sum of functions of just one variable [10], while a nonseparable function cannot be rewritten in this form due to the complex interrelation among variables. Therefore, a nonseparable function is more difficult than the separable one and difficulty increases if the function is multimodal. In Tables 2 and 3, Cha denotes the characteristics of a function; U, M, S, and N in the Cha column represent that a function is unimodal, multimodal, separable, and nonseparable, respectively. f_1–f_{10} are unimodal. More concretely, f_1–f_6 and f_8 are continuous functions; f_7 is a discontinuous function; f_9 is a noisy function and f_{10} is unimodal for $D = 2$ and $D = 3$ but may have multiple minima in high dimension cases. f_{11}–f_{22} are multimodal and the number of their local minima increases exponentially with the problem dimension. ABCGC follows the same parameter settings; that is, SN, $limit$, the maximum number of function evaluations, and the dimension of f_1–f_{20} and f_{21}-f_{22} are fixed at 100, 200, 200000, 60, and 200, respectively. The results of the compared algorithms are all derived directly from [14, 17]. Table 4 shows the comparison results.

As seen from Table 4, all the algorithms easily find the minimum of f_7 which is a region rather than a point. Both

Table 2: Details of benchmark functions used to compare ABCGC with other ABC algorithms.

Function	Range	Formulation	Minimum	Cha
Sphere	$[-100, 100]$	$f_1(X) = \sum_{i=1}^{D} x_i^2$	0	US
Elliptic	$[-100, 100]$	$f_2(X) = \sum_{i=1}^{D} \left(10^6\right)^{(i-1)/(D-1)} x_i^2$	0	US
SumSquare	$[-10, 10]$	$f_3(X) = \sum_{i=1}^{D} i x_i^2$	0	US
SumPower	$[-1, 1]$	$f_4(X) = \sum_{i=1}^{D} \lvert x_i \rvert^{i+1}$	0	US
Schwefel 2.22	$[-10, 10]$	$f_5(X) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	0	UN
Schwefel 2.21	$[-100, 100]$	$f_6(X) = \max\{\lvert x_i \rvert, 1 \le i \le D\}$	0	UN
Step	$[-100, 100]$	$f_7(X) = \sum_{i=1}^{D} \left(\lfloor x_i + 0.5 \rfloor\right)^2$	0	US
Exponential	$[-1.28, 1.28]$	$f_8(X) = \exp\left(0.5 \sum_{i=1}^{D} x_i^2\right) - 1$	0	UN
Quartic	$[-1.28, 1.28]$	$f_9(X) = \sum_{i=1}^{D} i x_i^4 + \text{random}[0, 1)$	0	US
Rosenbrock	$[-5, 10]$	$f_{10}(X) = \sum_{i=1}^{D-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2\right]$	0	UN
Rastrigin	$[-5.12, 5.12]$	$f_{11}(X) = \sum_{i=1}^{D} \left[x_i^2 - 10\cos\left(2\pi x_i\right) + 10\right]$	0	MS
NCRastrigin	$[-5.12, 5.12]$	$f_{12}(X) = \sum_{i=1}^{D} \left[y_i^2 - 10\cos\left(2\pi y_i\right) + 10\right]$, $y_i = \begin{cases} x_i & \lvert x_i \rvert < 0.5 \\ 0.5\,\text{round}\left(2x_i\right) & \lvert x_i \rvert \ge 0.5 \end{cases}$	0	MS
Griewank	$[-600, 600]$	$f_{13}(X) = \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	0	MN
Schwefel 2.26	$[-500, 500]$	$f_{14}(X) = 418.98288727243369D - \sum_{i=1}^{D} x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	0	MS
Ackley	$[-32, 32]$	$f_{15}(X) = 20 + e - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D} \cos\left(2\pi x_i\right)\right)$	0	MN

PABC and ABCGC reach the global optimum of f_8. BABC, BGABC, PABC, ABCbest, BABCbest, CABC, BCABC, and ABCGC obtain the optimum of f_{11} and f_{12}. BABC, BGABC, PABC, BABCbest, BCABC, and ABCGC find the minimum of f_{13}. BABC, BGABC, BABCbest, BCABC, and ABCGC reach the global optimum of f_{20}. On f_{14}, BABC, BGABC, BABCbest, and BCABC have the same mean value and outperform the other algorithms, and ABCGC ranks the second. On f_{16}, BABC, BGABC, BABCbest, CABC, and BCABC have the same mean value and perform better than the rest of the algorithms, and ABCGC ranks the fifth. On f_{17}, BABC, BGABC, PABC, BABCbest, CABC, and BCABC have the same mean value and outperform the compared algorithms, and ABCGC ranks the fourth. BCABC performs best on f_{18}, and ABCGC ranks the fifth. On f_{19}, BABCbest, CABC, and BCABC have the same mean value and outperform the other algorithms, and ABCGC ranks the seventh. On f_{21}, BABC, BGABC, PABC, ABCbest, BABCbest, CABC, BCABC, and ABCGC have the same mean value and perform better than ABC and GABC, but ABCGC has the smallest standard deviation. However, only ABCGC can obtain the optimum of f_1, f_2, f_3, and f_4. Note that the minimum value of f_5 found by ABCGC is 0 and its mean value is $8.23e - 264$, which greatly surpass the values obtained by the rest of the methods; moreover, ABCGC significantly outperforms the competitors on f_6, f_9, f_{10}, f_{15}, and f_{22}. To sum up, ABCGC has similar or better performance than ABC, BABC, GABC, BGABC, PABC, ABCbest, BABCbest, CABC, and BCABC on 22, 17, 19, 17, 19, 19, 18, 18, and 17 out of 22 functions, respectively. These results clearly indicate that the proposed algorithm is greatly superior to the nine compared algorithms on almost all the functions.

Table 3: Details of benchmark functions used to compare ABCGC with other ABC algorithms.

Function	Range	Formulation	Minimum	Cha
Penalized 1	$[-50, 50]$	$f_{16}(X) = \frac{\pi}{D}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i-1)^2\left[1+10\sin^2(\pi y_{i+1})\right] + (y_D-1)^2\right\} + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ $y_i = 1 + 0.25(x_i + 1)$ $u(x_i, a, k, m) = \begin{cases} k(x_i-a)^m & x_i > a \\ 0 & -a \le x_i \le a \\ k(-x_i-a)^m & x_i < -a \end{cases}$	0	MN
Penalized 2	$[-50, 50]$	$f_{17}(X) = 0.1\left\{\sin^2(\pi x_1) + \sum_{i=1}^{D-1}(x_i-1)^2\left[1+\sin^2(3\pi x_{i+1})\right] + (x_D-1)^2\left[1+\sin^2(2\pi x_D)\right]\right\} + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	0	MN
Alpine	$[-10, 10]$	$f_{18}(X) = \sum_{i=1}^{D} x_i\lvert\sin(x_i) + 0.1x_i\rvert$	0	MS
Levy	$[-10, 10]$	$f_{19}(X) = \sum_{i=1}^{D-1}(x_i-1)^2\left[1+\sin^2(3\pi x_{i+1})\right] + \sin^2(3\pi x_1) + \lvert x_D-1\rvert\left[1+\sin^2(3\pi x_D)\right]$	0	MN
Weierstrass	$[-0.5, 0.5]$	$f_{20}(X) = \sum_{i=1}^{D}\left\{\sum_{k=0}^{k_{\max}}\left[a^k\cos\left(2\pi b^k(x_i+0.5)\right)\right]\right\} - D\sum_{k=0}^{k_{\max}}\left[a^k\cos\left(2\pi b^k 0.5\right)\right]$ $a = 0.5, b = 3, k_{\max} = 20$	0	MN
Himmelblau	$[-5, 5]$	$f_{21}(X) = \frac{1}{D}\sum_{i=1}^{D}\left(x_i^4 - 16x_i^2 + 5x_i\right)$	-78.33236 for $D = 200$	MS
Michalewicz	$[0, \pi]$	$f_{22}(X) = -\sum_{i=1}^{D}\sin(x_i)\sin^{20}\left(\frac{ix_i^2}{\pi}\right)$	-198.5568 for $D = 200$	MS

Table 4: Mean values obtained by ten algorithms on f_1–f_{20} with $D = 60$ and f_{21}-f_{22} with $D = 200$.

Function	ABC	BABC	GABC	BGABC	PABC	ABCbest	BABCbest	CABC	BCABC	ABCGC
f_1	$1.01e-12$	$2.45e-33$	$2.02e-21$	$2.32e-35$	$4.49e-43$	$1.27e-29$	$2.17e-37$	$2.63e-34$	$2.14e-41$	**0**
f_2	$1.54e-08$	$7.70e-30$	$1.01e-17$	$7.12e-32$	$1.20e-41$	$5.68e-26$	$1.31e-33$	$3.43e-30$	$5.19e-38$	**0**
f_3	$3.81e-13$	$2.91e-34$	$5.92e-22$	$5.02e-36$	$8.85e-45$	$4.39e-30$	$2.46e-38$	$1.64e-34$	$1.35e-41$	**0**
f_4	$1.69e-17$	$1.01e-75$	$7.02e-29$	$1.87e-83$	$4.79e-49$	$1.36e-58$	$2.69e-105$	$2.65e-47$	$6.82e-91$	**0**
f_5	$3.63e-07$	$7.88e-18$	$1.44e-11$	$3.30e-19$	$1.21e-16$	$5.26e-16$	$1.33e-20$	$5.19e-18$	$4.30e-22$	**$8.23e-264$**
f_6	$5.30e+01$	$4.46e+01$	$4.64e+01$	$3.64e+01$	$3.55e+01$	$3.28e+01$	$2.94e+01$	$3.99e+01$	$3.40e+01$	**$1.34e-04$**
f_7	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_8	$1.99e-15$	$3.55e-16$	$3.38e-08$	$4.44e-16$	**0**	$4.66e-16$	$4.88e-16$	$7.10e-16$	$4.44e-16$	**0**
f_9	$6.90e-01$	$1.32e-01$	$3.63e-01$	$1.54e-01$	$1.66e-01$	$1.63e-01$	$1.15e-01$	$1.99e-01$	$1.11e-01$	**$6.62e-07$**
f_{10}	$4.38e-01$	$8.57e-02$	$1.09e+01$	$4.14e-01$	$2.82e-01$	$2.38e+01$	$1.03e-00$	$3.27e-01$	$2.05e-01$	**$4.60e-03$**
f_{11}	$7.85e-01$	**0**	$1.17e-13$	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_{12}	$1.39e-00$	**0**	$2.01e-11$	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_{13}	$6.73e-11$	**0**	$2.47e-02$	**0**	**0**	$1.79e-15$	**0**	$1.49e-08$	**0**	**0**
f_{14}	$4.89e-00$	**$3.63e-11$**	$4.51e-10$	**$3.63e-11$**	$5.38e-11$	$2.01e-10$	**$3.63e-11$**	$1.85e-10$	**$3.63e-11$**	$4.48e-11$
f_{15}	$3.93e-06$	$7.31e-14$	$6.39e-11$	$7.46e-14$	$5.37e-14$	$9.66e-14$	$6.60e-14$	$9.31e-14$	$6.11e-14$	**$8.88e-16$**
f_{16}	$1.47e-14$	**$7.85e-33$**	$1.33e-23$	**$7.85e-33$**	$7.89e-33$	$8.50e-32$	**$7.85e-33$**	**$7.85e-33$**	**$7.85e-33$**	$2.96e-15$
f_{17}	$5.96e-13$	**$1.34e-32$**	$3.72e-22$	**$1.34e-32$**	**$1.34e-32$**	$1.68e-30$	**$1.34e-32$**	**$1.34e-32$**	**$1.34e-32$**	$2.99e-15$
f_{18}	$6.03e-04$	$1.22e-16$	$1.52e-05$	$6.32e-16$	$3.26e-12$	$3.61e-14$	$3.71e-15$	$4.51e-18$	**$1.35e-23$**	$1.49e-15$
f_{19}	$7.72e-10$	$1.34e-31$	$6.79e-16$	$1.34e-31$	$1.64e-31$	$9.46e-30$	**$1.34e-32$**	**$1.34e-32$**	**$1.34e-32$**	$1.83e-15$
f_{20}	$5.96e-04$	**0**	$1.33e-10$	**0**	$2.74e-14$	$2.55e-14$	**0**	$8.52e-15$	**0**	**0**
f_{21}	−77.9536	**−78.3323**	−78.3322	**−78.3323**	**−78.3323**	**−78.3323**	**−78.3323**	**−78.3323**	**−78.3323**	**−78.3323**
f_{22}	−174.3284	−186.9572	−176.5457	−185.11092	−176.7006	−181.1403	−188.8318	−187.3753	−191.0282	**−199.4474**

Table 5: Details of benchmark functions used to compare ABCGC with other methods.

Function	Range	Formulation	Minimum	Cha
Sphere	$[-100, 100]$	$f_1(X) = \sum_{i=1}^{D} x_i^2$	0	US
Rosenbrock	$[-30, 30]$	$f_2(X) = \sum_{i=1}^{D-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2\right]$	0	UN
Rastrigin	$[-5.12, 5.12]$	$f_3(X) = \sum_{i=1}^{D} \left[x_i^2 - 10\cos\left(2\pi x_i\right) + 10\right]$	0	MS
Ackley	$[-30, 30]$	$f_4(X) = 20 + e - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos\left(2\pi x_i\right)\right)$	0	MN
Griewank	$[-600, 600]$	$f_5(X) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	0	MN
Schaffer	$[-100, 100]$	$f_6(X) = 0.5 + \frac{\sin^2\left(\sqrt{\sum_{i=1}^{D} x_i^2}\right) - 0.5}{\left(1 + 0.001\sum_{i=1}^{D} x_i^2\right)^2}$	0	MN

The above results are reasonable. In ABCGC, onlooker bees always select the optimal search dimension instead of a random chosen one in each cycle, which is very useful for local search and fine tuning; on the other hand, ABCGC has global search ability which prevents the search from premature convergence due to the exploration of a wider solution space carried out by scout bees and neighbor solution production mechanism performed by employed and onlooker bees. Generally, there is a good balance between exploitation and exploration; hence ABCGC has better performance in terms of global optimization.

4.2. Comparison with Other State-of-the-Art Algorithms. To further evaluate the performance of the proposed algorithm, a set of 6 classical benchmark functions from [29] is used to compare ABCGC with ABC and 5 state-of-the-art

Table 6: Mean values obtained by ABCGC, ABC, and other 5 state-of-the-art algorithms.

Function	D	ABC	BSO-RP TVW	IMDE		DFO	DFO (QA)	ABCGC
				1st process	2nd process			
Sphere	10	$4.9e-28$	$1.7e-118$	**0**	**0**	**0**	**0**	**0**
	20	$3.1e-25$	$7.2e-114$	$2.1e-267$	**0**	**0**	$2.58e-182$	**0**
	30	$6.4e-24$	$3.2e-111$	$7.2e-238$	$6.2e-296$	**0**	$5.74e-252$	**0**
Rosenbrock	10	$9.3e-03$	$1.9e-09$	**0**	**0**	$4.90e-03$	$8.18e-08$	$1.40e-03$
	20	$1.3e-02$	$9.0e-09$	**0**	**0**	$7.04e-03$	$4.99e-05$	$1.91e-03$
	30	$3.4e-02$	$6.0e-08$	**0**	**0**	$3.11e-03$	$1.28e-03$	$2.90e-03$
Rastrigin	10	$4.4e-24$	$8.6e-93$	**0**	**0**	**0**	**0**	**0**
	20	$2.0e-22$	$6.9e-91$	**0**	**0**	**0**	$8.6e-19$	**0**
	30	$8.6e-22$	$2.6e-86$	**0**	**0**	**0**	$2.3e-18$	**0**
Ackley	10	$3.2e+00$	**$8.0e-53$**	$5.9e-16$	$5.9e-16$	$1.88e-15$	$2.22e-15$	$8.88e-16$
	20	$1.8e+00$	**$6.8e-53$**	$4.1e-15$	$4.1e-15$	$5.77e-15$	$7.54e-15$	$8.88e-16$
	30	$9.8e-01$	**$5.7e-54$**	$4.1e-15$	$4.1e-15$	$7.91e-15$	$1.55e-14$	$8.88e-16$
Griewank	10	$8.5e-05$	$1.8e-20$	**0**	**0**	**0**	$1.4e-19$	**0**
	20	$6.4e-04$	$9.7e-20$	**0**	**0**	**0**	$2.68e-19$	**0**
	30	$1.4e-03$	$2.4e-19$	**0**	**0**	**0**	$8.13e-19$	**0**
Schaffer	2	$9.4e-08$	$3.5e-56$	$2.9e-03$	$5.0e-03$	**0**	**0**	**0**

algorithms: BSO-RPTVW, IMDE (1st process), IMDE (2nd process), DFO, and DFO (QA). The details of these functions are given in Table 5. Based on their characteristics, the functions may be divided into three groups: functions with no local minima, many local minima, and a few local minima. Sphere and Rosenbrock functions are high-dimensional unimodal functions. Rastrigin, Ackley, and Griewank functions are high-dimensional multimodal functions with many local minima and highly nonlinear in nature; moreover, the number of local minima of these functions increases exponentially with increase of dimension. Schaffer function is a low-dimensional function with a smaller number of local minima. Therefore, these functions are widely used to test the performance of global optimization algorithms [2, 13, 18, 27–29]. As mentioned in [29], SN is set to 100 and MCN is 5000, 7500, and 10000 for f_1–f_5 with 10, 20, and 30 dimensions, respectively, but for Schaffer function with $D = 2$, MCN is equal to 2000 in this experiment. The results of ABC, BSO-RPTVW, IMDE (1st process), IMDE (2nd process), DFO, and DFO (QA) are quoted from [29]. Table 6 shows the comparison results.

From Table 6, DFO and ABCGC obtain the global optimum of Sphere function and perform better than the rest of the methods. On Rosenbrock function, IMDE (1st process) and IMDE (2nd process) reach the minimum and outperform all other algorithms, and ABCGC ranks the fourth. IMDE (1st process), IMDE (2nd process), DFO, and ABCGC can find the global optimum of Rastrigin and Griewank functions, while the others cannot. On Ackley function, BSO-RPTVW performs best, and ABCGC is better than the rest of the methods except for the function with $D = 10$, but ABCGC ranks the third on the function with $D = 10$. ABC, BSO-RPTVW, IMDE (1st process), and IMDE (2nd process) are not able to reach the minimum of Schaffer function, while DFO, DFO (QA), and ABCGC do it. These results clearly show that ABCGC has similar or better performance than ABC, BSO-RPTVW, IMDE (1st process), IMDE (2nd process), DFO, and DFO (QA) in 16, 10, 12, 12, 16, and 13 out of 16 cases. In other words, ABCGC performs significantly better than the compared algorithms in most cases. These also demonstrate that ABCGC is available for local and global optimization due to a good balance between the local search process carried out by employed and onlooker bees and the global search process managed by scout bees.

4.3. Effects of Each Modification on the Performance of ABCGC. In order to further study the effects of each modification, the basic ABC combined with GEM is named as ABCG (i.e., GABC2), and the basic ABC with Cauchy operator is named as ABCC. A set of 6 functions given in Table 5 is compared with the convergence performance of ABC, ABCG, ABCC, and ABCGC to see how much each new search strategy makes contribution to improving the performance of ABCGC. In the literature [1, 2, 4–6, 8–18, 21–31, 33], SN, $limit$, and D are often set to 10, $SN \times D$, and 10, 50, 100, respectively. Consequently, the parameter settings for the four algorithms are as follows: $SN = 10$, $limit = SN \times D$, $D = 10, 50, 100$, and $MCN \in \{1, 2, \ldots, 300\}$. The convergence characteristics of the algorithms are shown in Figures 3, 4, and 5.

It can be observed from Figures 3, 4, and 5 that ABCC has similar performance to ABC on Sphere, Rosenbrock, Rastrigin, Ackley, and Griewank functions with $D = 10, 50, 100$, and it has a slightly bigger fluctuation than ABC due to wider solutions generated by ABCC, but ABCC performs slightly better than ABC. In addition, the performances of ABCG and ABCGC are very close on the same 5 functions with the three dimensions, and they exhibit much better convergence

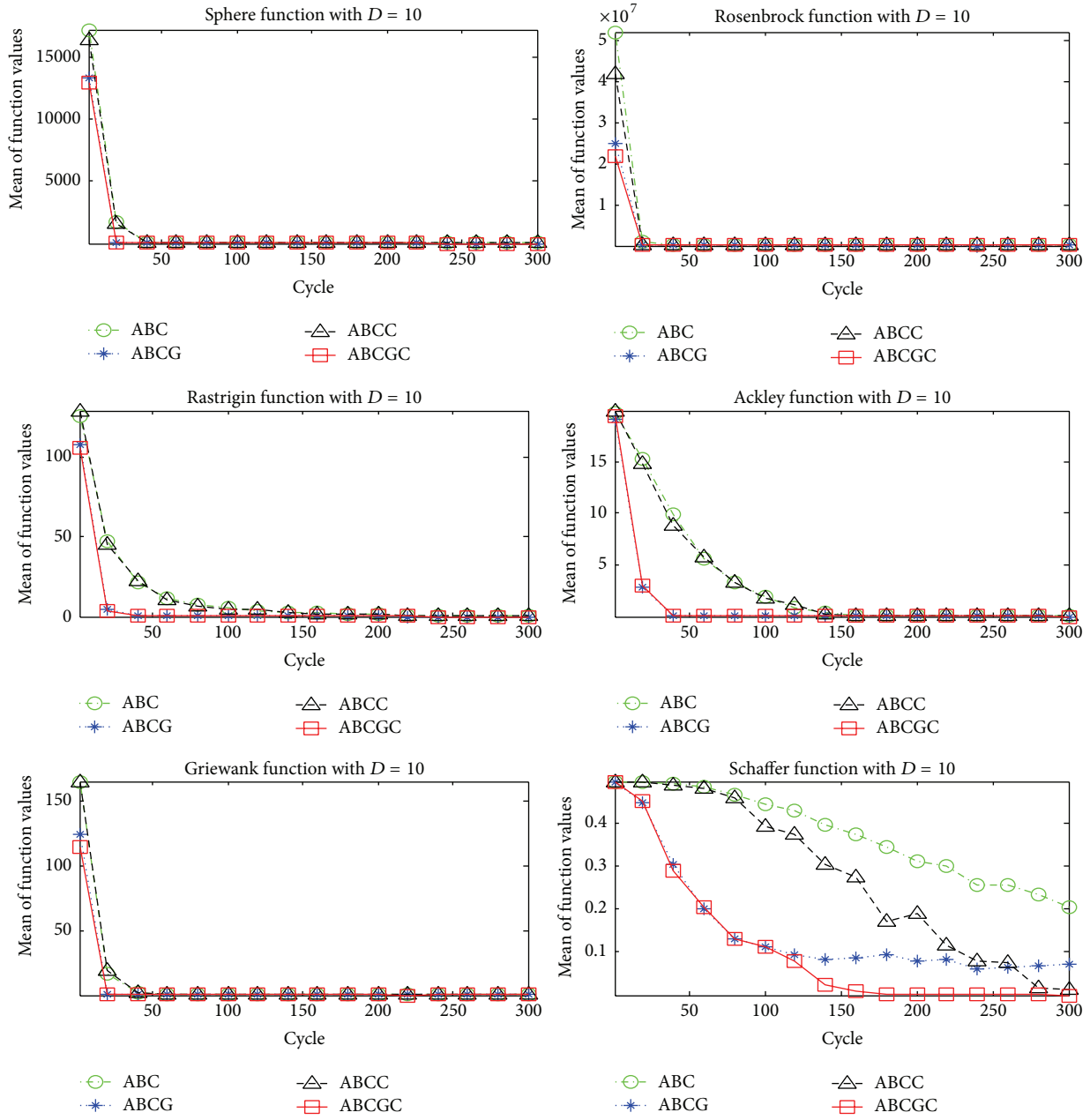

Figure 3: Convergence curves of ABC, ABCG, ABCC, and ABCGC on the 6 functions with $D = 10$.

than ABC and ABCC, but ABCGC slightly outperforms ABCG. More excitingly, both ABCG and ABCGC can obtain the global optimum of Rastrigin function after 260 cycles. Schaffer function is one of the most difficult standard test functions and its global minimum is very close to local minima [2]. More interestingly, on Schaffer function with $D = 10$, ABCC shows a bigger fluctuation than other algorithms and performs much better than ABC; moreover, ABCC performs worse than ABCG before 270 cycles but surpasses ABCG after 270 cycles. Note that ABCGC significantly outperforms the rest of the algorithms on Schaffer function with $D = 10$. For example, the mean value of ABC, ABCG, and ABCC is about 1000, 100, and 300 times that of ABCGC after 300 cycles, respectively. On Schaffer function with $D = 50$ and $D = 100$, the performance of ABC, ABCG, ABCC, and ABCGC decreases as D increases; ABCGC has similar or better performance than ABCG and ABCC also has similar or better performance than ABC. However, ABCG and ABCGC greatly outperform ABC and ABCC. Taken as a whole, ABCGC converges the fastest and performs more robustly and effectively than ABC, ABCG, and ABCC on all the 6 functions. These results demonstrate that ABC is good at exploration but poor at exploitation, which result in slow convergence. On the other hand, the new exploitation strategy adopted by onlooker bees in ABCG and ABCGC biases the solution towards the global or near-global optimum more quickly, and the new exploration strategy adopted by scout bees in ABCC and ABCGC further helps to maintain large

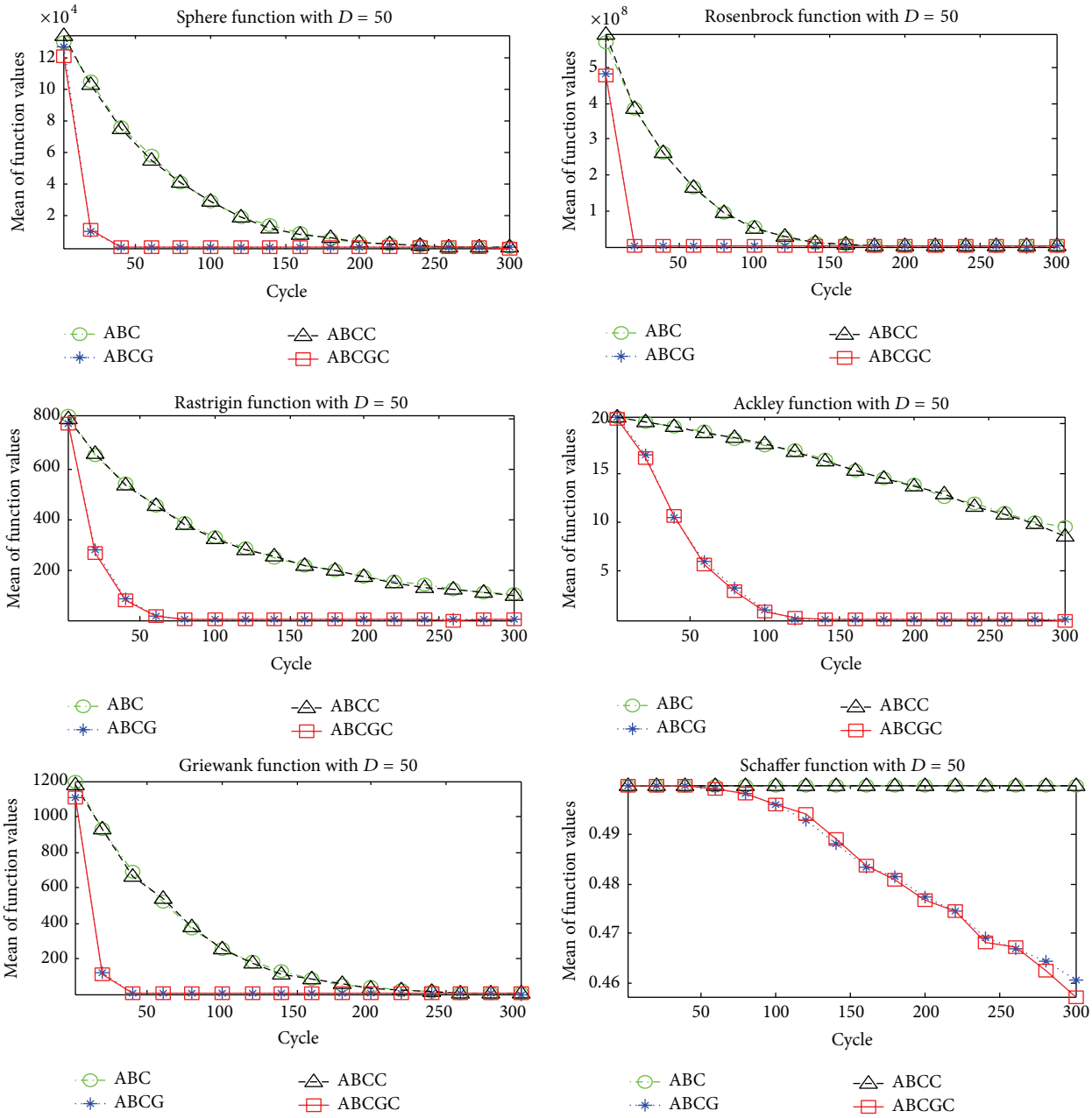

Figure 4: Convergence curves of ABC, ABCG, ABCC, and ABCGC on the 6 functions with $D = 50$.

population diversity and avoids premature convergence; that is to say, the new exploitation and exploration strategies work well together to improve the performance of ABCGC rather than contradict each other.

5. Conclusions

To address the problems of poor balance between exploitation and exploration and slow convergence of the basic ABC, a new hybrid ABC for global optimization is proposed, namely, ABCGC. In the proposed algorithm, the new exploitation strategy inspired by GEM makes onlooker bees always select the optimal search dimension instead of a random chosen one in each cycle, which greatly enhances the exploitation ability and accelerates convergence of ABCGC; meanwhile, the new exploitation strategy with Cauchy operator generates a wider solution instead of a randomly produced one for each scout bee to help ABCGC escape from local optimum and further enhance its exploration ability. Comprehensive experiments are conducted on two sets of well-known benchmark functions to verify the performance of ABCGC. The results show that ABCGC works best and achieves overall better performance than the compared algorithms in terms of solution quality and convergence characteristics; particularly it finds the global optimum faster in most cases. These confirm that ABCGC usually achieves a good balance

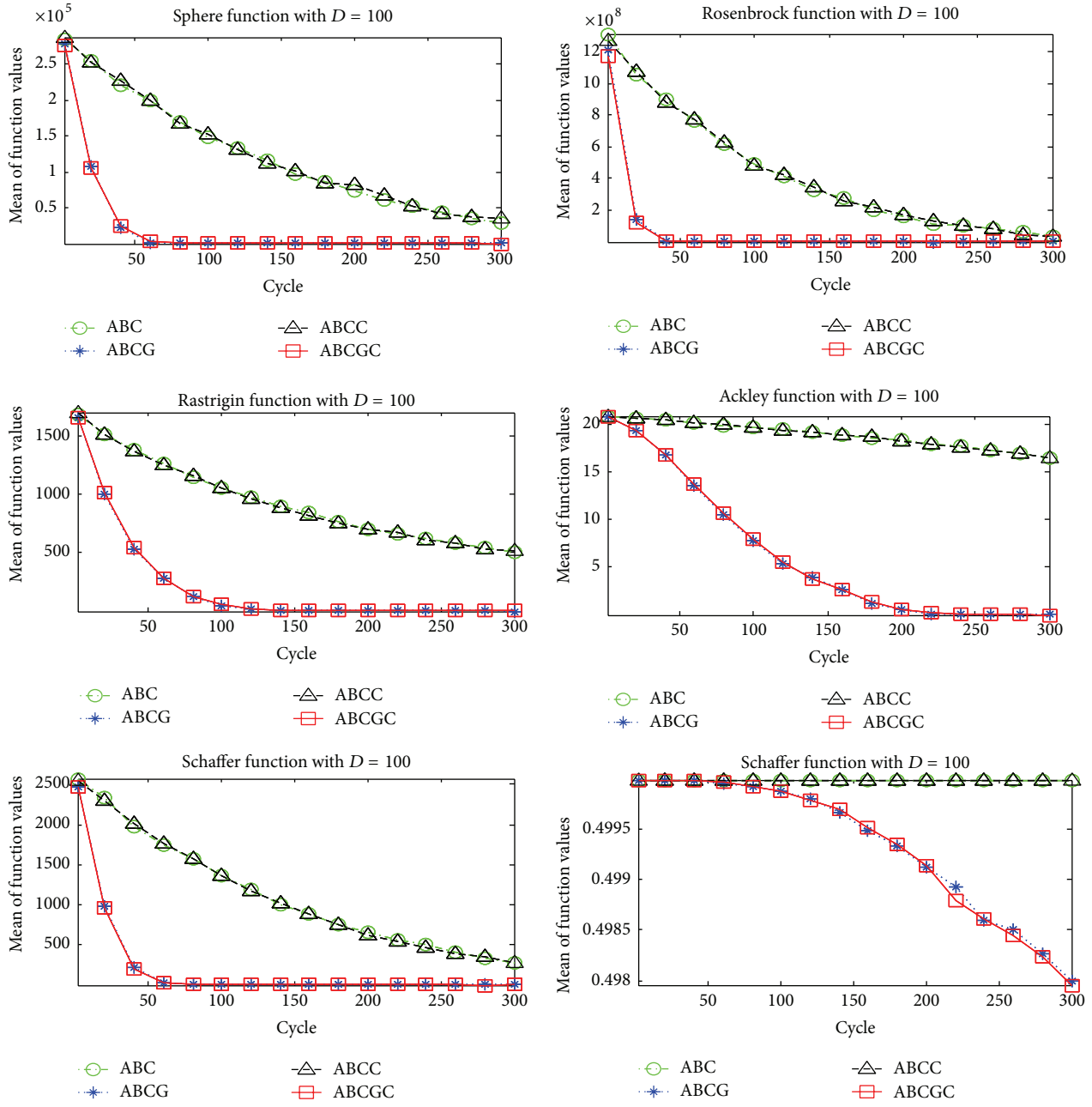

Figure 5: Convergence curves of ABC, ABCG, ABCC, and ABCGC on the 6 functions with $D = 100$.

between exploitation and exploration. Thus it can effectively serve as an alternative for global optimization.

In future work, there is a need for an adaptive scheme instead of a fixed one, which will dynamically adjust the number of pieces of shrapnel per grenade in each cycle of ABCGC. In addition, it is desired to further apply the proposed variants to real-world applications.

Conflict of Interests

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

Acknowledgments

This work is supported by National Science Foundation of China under Grant nos. 61463007 and 21466008, by National Science Foundation of Shanghai under Grant no. 15ZR1401600, and by Key Project of Guangxi University for Nationalities under Grant no. 2012MDZD035. The authors would like to thank the editors and the anonymous reviewers for their kind assistance, constructive comments, and recommendations, which have significantly improved the presentation of this paper. They would like to express their appreciation to Karaboga for sharing his basic ABC codes.

References

[1] M. H. Kashan, N. Nahavandi, and A. H. Kashan, "DisABC: a new artificial bee colony algorithm for binary optimization," *Applied Soft Computing*, vol. 12, no. 1, pp. 342–352, 2012.

[2] 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.

[3] M. F. Tasgetiren, Q.-K. Pan, P. N. Suganthan, and A. Oner, "A discrete artificial bee colony algorithm for the no-idle permutation flowshop scheduling problem with the total tardiness criterion," *Applied Mathematical Modelling*, vol. 37, no. 10-11, pp. 6758–6779, 2013.

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

[5] 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, pp. 318–329, Springer, Berlin, Germany, 2007.

[6] B. Akay and D. Karaboga, "Solving integer programming problems by using artificial bee colony algorithm," in *AI*IA 2009: Emergent Perspectives in Artificial Intelligence*, vol. 5883 of *Lecture Notes in Computer Science*, pp. 355–364, Springer, Berlin, Germany, 2009.

[7] C. Ozturk, E. Hancer, and D. Karaboga, "Dynamic clustering with improved binary artificial bee colony algorithm," *Applied Soft Computing Journal*, vol. 28, pp. 69–80, 2015.

[8] D. Karaboga and B. Akay, "A modified artificial bee colony (ABC) algorithm for constrained optimization problems," *Applied Soft Computing*, vol. 11, no. 3, pp. 3021–3031, 2011.

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

[10] 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.

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

[12] S. Biswas, S. Das, S. Debchoudhury, and S. Kundu, "Co-evolving bee colonies by forager migration: a multi-swarm based artificial bee colony algorithm for global search space," *Applied Mathematics and Computation*, vol. 232, pp. 216–234, 2014.

[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] W.-F. Gao, S.-Y. Liu, and L.-L. Huang, "A novel artificial bee colony algorithm with Powell's method," *Applied Soft Computing Journal*, vol. 13, no. 9, pp. 3763–3775, 2013.

[15] W.-F. Gao, S.-Y. Liu, and L.-L. Huang, "A global best artificial bee colony algorithm for global optimization," *Journal of Computational and Applied Mathematics*, vol. 236, no. 11, pp. 2741–2753, 2012.

[16] W.-F. Gao, S.-Y. Liu, and L.-L. Huang, "A novel artificial bee colony algorithm based on modified search equation and orthogonal learning," *IEEE Transactions on Cybernetics*, vol. 43, no. 3, pp. 1011–1024, 2013.

[17] W. Gao, F. T. Chan, L. Huang, and S. Liu, "Bare bones artificial bee colony algorithm with parameter adaptation and fitness-based neighborhood," *Information Sciences*, vol. 316, pp. 180–200, 2015.

[18] C.-Q. Zhang, J.-G. Zheng, and Y.-Q. Zhou, "Two modified artificial bee colony algorithms inspired by grenade explosion method," *Neurocomputing*, vol. 151, no. 3, pp. 1198–1207, 2015.

[19] J. M. Chaves-González, M. A. Pérez-Toledano, and A. Navasa, "Software requirement optimization using a multiobjective swarm intelligence evolutionary algorithm," *Knowledge-Based Systems*, vol. 83, pp. 105–115, 2015.

[20] J. Liang and C. Lee, "A modification artificial bee colony algorithm for optimization problems," *Mathematical Problems in Engineering*, vol. 2015, Article ID 581391, 14 pages, 2015.

[21] M. S. Kıran and O. Fındık, "A directed artificial bee colony algorithm," *Applied Soft Computing*, vol. 26, pp. 454–462, 2015.

[22] A. Ahrari, M. Shariat-Panahi, and A. A. Atai, "GEM: a novel evolutionary optimization method with improved neighborhood search," *Applied Mathematics and Computation*, vol. 210, no. 2, pp. 376–386, 2009.

[23] A. Ahrari and A. A. Atai, "Grenade Explosion Method—a novel tool for optimization of multimodal functions," *Applied Soft Computing Journal*, vol. 10, no. 4, pp. 1132–1140, 2010.

[24] A. Ahrari and R. Ahrari, "On the utility of randomly generated functions for performance evaluation of evolutionary algorithms," *Optimization Letters*, vol. 4, no. 4, pp. 531–541, 2010.

[25] A. Ahrari, M. R. Saadatmand, M. Shariat-Panahi, and A. A. Atai, "On the limitations of classical benchmark functions for evaluating robustness of evolutionary algorithms," *Applied Mathematics and Computation*, vol. 215, no. 9, pp. 3222–3229, 2010.

[26] X. Yao, Y. Liu, and G.-M. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.

[27] R. Akbari, A. Mohammadi, and K. Ziarati, "A novel bee swarm optimization algorithm for numerical function optimization," *Communications in Nonlinear Science and Numerical Simulation*, vol. 15, no. 10, pp. 3142–3155, 2010.

[28] Y. Zhou, X. Li, and L. Gao, "A differential evolution algorithm with intersect mutation operator," *Applied Soft Computing Journal*, vol. 13, no. 1, pp. 390–401, 2013.

[29] K. N. Das and T. K. Singh, "Drosophila food-search optimization," *Applied Mathematics and Computation*, vol. 231, pp. 566–580, 2014.

[30] Y.-Y. Li, L.-C. Jiao, P.-D. Li, and B. Wu, "A hybrid memetic algorithm for global optimization," *Neurocomputing*, vol. 134, pp. 132–139, 2014.

[31] L. Zhang, Y. Tang, C. Hua, and X. Guan, "A new particle swarm optimization algorithm with adaptive inertia weight based on Bayesian techniques," *Applied Soft Computing Journal*, vol. 28, pp. 138–149, 2015.

[32] F.-F. Dong, D.-C. Liu, J. Wu et al., "Constructing core backbone network based on survivability of power grid," *International Journal of Electrical Power & Energy Systems*, vol. 67, pp. 161–167, 2015.

[33] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching-learning-based optimization: an optimization method for continuous non-linear large scale problems," *Information Sciences*, vol. 183, pp. 1–15, 2012.