Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2013, Article ID 696491, 21 pages
http://dx.doi.org/10.1155/2013/696491

Research Article

A Novel Hybrid Bat Algorithm with Harmony Search for Global Numerical Optimization

Gaige Wang[1,2] and Lihong Guo[1]

[1] *Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China*
[2] *Graduate School of Chinese Academy of Sciences, Beijing 100039, China*

Correspondence should be addressed to Lihong Guo; guolh@ciomp.ac.cn

Received 27 June 2012; Revised 27 November 2012; Accepted 15 December 2012

Academic Editor: Marco H. Terra

A novel robust hybrid metaheuristic optimization approach, which can be considered as an improvement of the recently developed bat algorithm, is proposed to solve global numerical optimization problems. The improvement includes the addition of pitch adjustment operation in HS serving as a mutation operator during the process of the bat updating with the aim of speeding up convergence, thus making the approach more feasible for a wider range of real-world applications. The detailed implementation procedure for this improved metaheuristic method is also described. Fourteen standard benchmark functions are applied to verify the effects of these improvements, and it is demonstrated that, in most situations, the performance of this hybrid metaheuristic method (HS/BA) is superior to, or at least highly competitive with, the standard BA and other population-based optimization methods, such as ACO, BA, BBO, DE, ES, GA, HS, PSO, and SGA. The effect of the HS/BA parameters is also analyzed.

1. Introduction

The process of optimization is searching a vector in a function that produces an optimal solution. All of feasible values are available solutions, and the extreme value is optimal solution. In general, optimization algorithms are applied to solve these optimization problems. A simple classification way for optimization algorithms is considering the nature of the algorithms, and optimization algorithms can be divided into two main categories: deterministic algorithms and stochastic algorithms. Deterministic algorithms using gradients such as hill climbing have a rigorous move and will generate the same set of solutions if the iterations commence with the same initial starting point. On the other hand, stochastic algorithms without using gradients often generate different solutions even with the same initial value. However, generally speaking, the final values, though slightly different, will converge to the same optimal solutions within a given accuracy [1]. Generally, stochastic algorithms have two types: heuristic and metaheuristic. Recently, nature-inspired metaheuristic algorithms perform powerfully and efficiently in solving modern nonlinear numerical global optimization problems. To some extent, all metaheuristic algorithms strive for making balance between randomization (global search) and local search [2].

Inspired by nature, these strong metaheuristic algorithms are applied to solve NP-hard problems, such as UCAV path planning [11–14], test-sheet composition [15], WSN deployment [16] and water, and geotechnical and transport engineering [17]. Optimization algorithms cover all searching for extreme value problems. These kinds of metaheuristic algorithms carry out on a population of solutions and always find best solutions. During the 1950s and 1960s, computer scientists studied the possibility of conceptualizing evolution as an optimization tool, and this generated a subset of gradient free approaches named genetic algorithms (GA) [18]. Since then, many other nature-inspired metaheuristic algorithms have emerged, such as differential evolution (DE) [10, 19, 20], particle swarm optimization (PSO) [21–23], biogeography-based optimization (BBO) [24, 25], harmony search (HS) [26, 27], and more recently, the bat algorithm (BA) [28, 29] that is inspired by echolocation behavior of bats in nature.

Firstly presented by Yang in 2010, the bat-inspired algorithm or bat algorithm (BA) [30, 31] is a metaheuristic search algorithm, inspired by the echolocation behavior of

bats with varying pulse rates of emission and loudness. The primary purpose for a bat's echolocation is to act as a signal system to sense distance and hunt for food/prey. A comprehensive review about swarm intelligence involving BA is performed by Parpinelli and Lopes [32]. Furthermore, Tsai et al. proposed an evolving bat algorithm (EBA) to improve the performance of BA with better efficiency [33].

Firstly proposed by Geem et al. in 2001, harmony search (HS) [26] is a new metaheuristic approach for minimizing possibly nondifferentiable and nonlinear functions in continuous space. HS is inspired by the behavior of a musician's improvisation process, where every musician attempts to improve its tune so as to create optimal harmony in a real-world musical performance processes. HS algorithm originates in the similarity between engineering optimization and music improvisation, and the engineers search for a global optimal solution as determined by an objective function, just like the musicians strive for finding aesthetic harmony as determined by aesthetician. In music improvisation, each musician chooses any pitch within the feasible range, together producing one harmony vector. If all the pitches produce a good solution, that experience is reserved in each variable's memory, and the possibility of producing a better solution is also increased next time. Furthermore, this new approach requires few control variables, which makes HS easy to implement, more robust, and very appropriate for parallel computation.

BA is a powerful algorithm in exploitation (i.e., local search), but at times it may trap into some local optima, so that it cannot perform global search well. For bat algorithm, the search depends completely on random walks, so a fast convergence cannot be guaranteed. Firstly presented here, in order to increase the diversity of the population for BA so as to avoid trapping into local optima, a main improvement of adding pitch adjustment operation in HS serving as a mutation operator is made to the BA with the aim of speeding up convergence, thus making the approach more feasible for a wider range of practical applications while preserving the attractive characteristics of the basic BA. That is to say, we combine two approaches to propose a new hybrid metaheuristic algorithm according to the principle of HS and BA, and then this improved BA method is used to search the optimal objective function value. The proposed approach is evaluated on fourteen standard benchmark functions that have ever been applied to verify optimization algorithms on continuous optimization problems. Experimental results show that the HS/BA performs more efficiently and accurately than basic BA, ACO, BBO, DE, ES, GA, HS, PSO, and SGA.

The structure of this paper is organized as follows: Section 2 describes global numerical optimization problem, the HS algorithm, and basic BA in brief. Our proposed approach HS/BA is presented in detail in Section 3. Subsequently, our method is evaluated through fourteen benchmark functions in Section 4. In addition, the HS/BA is also compared with BA, ACO, BBO, DE, ES, GA, HS, PSO and SGA. Finally, Section 5 consists of the conclusion and proposals for future work.

2. Preliminary

To begin with, in this section we will provide a brief background on the optimization problem, harmony search (HS), and bat algorithm (BA).

2.1. Optimization Problem. In computer science, mathematics, management science, and control theory, optimization (also called mathematical optimization or mathematical programming) means the selection of an optimal solution from some set of feasible alternatives. In general, an optimization problem includes minimizing or maximizing a function by systematically selecting input values from a given feasible set and calculating the value of the function. More generally, optimization consists of finding the optimal values of some objective function within a given domain, including a number of different types of domains and different types of objective functions [34].

A global optimization problem can be described as follows.

> Given: a function $f: D \rightarrow R$ from some set D to the real numbers.
>
> Sought: a parameter x_0 in D such that $f(x_0) \leq f(x)$ for all x in D ("minimization") or such that $f(x_0) \geq f(x)$ for all x in D ("maximization").

Such a formulation is named a numerical optimization problem. Many theoretical and practical problems may be modeled in this general framework. In general, D is some subset of the Euclidean space R^n, often specified by a group of constraints, equalities, or inequalities that the components of D have to satisfy. The domain D of f is named the search space, while the elements of D are named feasible solutions or candidate solutions. In general, the function f is called an objective function, cost function (minimization), or utility function (maximization). An optimal solution is an available solution that is the extreme of (minimum or maximum) the objective function.

Conventionally, the standard formulation of an optimization problem is stated in terms of minimization. In general, unless both the feasible region and the objective function are convex in a minimization problem, there may be more than one local minima. A local minimum x^* is defined as a point for which the following expression

$$f(x^*) \leq f(x) \tag{1}$$

holds. More details about the optimization problem can be found in [35].

The branch of numerical analysis and applied mathematics that investigates deterministic algorithms that can guarantee convergence in limited time to the true optimal solution of a nonconvex problem is called global numerical optimization problems. A variety of algorithms have been proposed to solve nonconvex problems. Among them, heuristics algorithms can evaluate approximate solutions to some optimization problems, as described in introduction [36].

2.2. Harmony Search. Firstly developed by Geem et al. in 2001, harmony search (HS) [26] is a relatively new metaheuristic optimization algorithm, and it is based on natural musical performance processes that occur when a musician searches for an optimal state of harmony. The optimization operators of HS algorithm are specified as the harmony memory (HM), which keeps the solution vectors which are all within the search space, as shown in (2); the harmony memory size HMS, which represents the number of solution vectors kept in the HM; the harmony memory consideration rate (HMCR); the pitch adjustment rate (PAR); the pitch adjustment bandwidth (bw)

$$\mathrm{HM} = \left[\begin{array}{cccc|c} x_1^1 & x_2^1 & \ldots & x_D^1 & \mathrm{fitness}\left(x^1\right) \\ x_1^2 & x_2^2 & \ldots & x_D^2 & \mathrm{fitness}\left(x^2\right) \\ \vdots & \vdots & \ldots & \vdots & \vdots \\ x_1^{\mathrm{HMS}} & x_2^{\mathrm{HMS}} & \ldots & x_D^{\mathrm{HMS}} & \mathrm{fitness}\left(x^{\mathrm{HMS}}\right) \end{array}\right]. \quad (2)$$

In more detail, we can explain the HS algorithm with the help of discussing the improvisation process by a music player. When a player is improvising, he or she has three possible options: (1) play any well-known piece of music (a series of pitches in harmony) exactly from his or her memory as the harmony memory consideration rate (HMCR); (2) play something similar to a known piece in player's memory (thus adjusting the pitch slightly); or (3) play totally new or random pitch from feasible ranges. If these three options are formalized for optimization, we have three corresponding components: employment of harmony memory, pitch adjusting, and randomization. Similarly, when each decision variable selects one value in the HS algorithm, it can make use of one of the above three rules in the whole HS procedure. If a new harmony vector is better than the worst harmony vector in the HM, the new harmony vector takes the place of the worst harmony vector in the HM. This procedure is repeated until a stopping criterion is satisfied.

The employment of harmony memory is significant, as it is analogous to select the optimal fit individuals in the GA (genetic algorithm). This will make sure that the best harmonies will be kept on to the new harmony memory. For the purpose of using this memory more effectively, we should properly set the value of the parameter HMCR $\in [0, 1]$. If this rate is very approaching to 1 (too high), almost all the harmonies are utilized in the harmony memory, then other harmonies are not explored well, resulting in potentially wrong solutions. If this rate is very close to 0 (extremely low), only few best harmonies are chosen, and it may have a slow convergence rate. Therefore, generally, HMCR = 0.7 ~ 0.95.

To adjust the pitch slightly in the second component, an appropriate approach is to be applied to adjust the frequency efficiently. In theory, we can adjust the pitch linearly or nonlinearly, but in fact, linear adjustment is utilized. If x_{old} is the current solution (or pitch), then the new solution (pitch) x_{new} is generated by

$$x_{\mathrm{new}} = x_{\mathrm{old}} + \mathrm{bw}\,(2\varepsilon - 1)\,, \quad (3)$$

where ε is a random real number drawn from a uniform distribution $[0, 1]$. Here bw is the bandwidth, controlling the local range of pitch adjustment. Actually, we can see that the pitch adjustment (3) is a random walk.

Pitch adjustment is similar to the mutation operator in GA. Also, we must appropriately set the parameter *PAR* to control the degree of the adjustment. If PAR nears 1 (too high), then the solution is always changing and the algorithm may not converge at all. If it is very close to 0 (too low), then there is very little change and the algorithm may premature. Therefore, we use PAR = 0.1 ~ 0.5 in most simulations as usual.

For the purpose of increasing the diversity of the solutions, the randomization is needed in the third component. Although adjusting pitch has a similar role, it is confined to certain local pitch adjustment and thus corresponds to a local search. The usage of randomization can make the system move further to explore multifarious regions with high solution diversity in order to search for the global optimal solution. In real-world engineering applications, HS has been applied to solve many optimization problems including function optimization, water distribution network, groundwater modeling, energy-saving dispatch, structural design, and vehicle routing.

The mainframe of the basic HS algorithm can be described as shown in Algorithm 1, where D is the number of decision variables and rand is a random real number in interval $(0, 1)$ drawn from uniform distribution. From Algorithm 1, it is clear that there are only two control parameters in HS, which are HMCR and PAR.

2.3. Bat Algorithm. The bat algorithm is a novel metaheuristic swarm intelligence optimization method developed for the global numerical optimization, in which the search algorithm is inspired by social behavior of bats and the phenomenon of echolocation to sense distance.

In [28], for simplicity, bat algorithm is based on idealizing some of the echolocation characteristics of bats, which are the following approximate or idealized rules:

(1) all bats apply echolocation to sense distance, and they always "know" the surroundings in some magical way;

(2) bats fly randomly with velocity v_i and a fixed frequency $f_{\min}$ at position x_i, varying wavelength λ, and loudness A_0 to hunt for prey. They can spontaneously accommodate the wavelength (or frequency) of their emitted pulses and adjust the rate of pulse emission $r \in [0, 1]$, depending on the proximity of their target;

(3) although the loudness can change in different ways, it is supposed that the loudness varies from a minimum constant (positive) $A_{\min}$ to a large A_0.

Based on these approximations and idealization, the basic steps of the bat algorithm (BA) can be described as shown in Algorithm 2.

In BA, each bat is defined by its position x_i^t, velocity v_i^t, frequency f_i, loudness A_i^t, and the emission pulse rate r_i^t

Begin
Step 1: Set the parameters and initialize the HM.
Step 2: Evaluate the fitness for each individual in HM
Step 3: while the halting criteria is not satisfied **do**
for $d = 1{:}D$ **do**
if rand < HMCR **then** // memory consideration
$x_{\text{new}}(d) = x_a(d)$ where $a \in (1, 2, \ldots, \text{HMS})$
if rand < PAR then // pitch adjustment
$x_{\text{new}}(d) = x_{\text{old}}(d) + \text{bw} \times (2 \times \text{rand} - 1)$
endif
else // random selection
$x_{\text{new}}(d) = x_{\min,d} + \text{rand} \times (x_{\max,d} - x_{\min-d})$
endif
endfor d
Update the HM as $x_w = x_{\text{new}}$, if $f(x_{\text{new}}) < f(x_w)$ (minimization objective)
Update the best harmony vector found so far
Step 4: end while
Step 5: Post-processing the results and visualization.
End.

Algorithm 1: Harmony search algorithm.

Begin
Step 1: Initialization. Set the generation counter $t = 1$; Initialize the population of NP bats P randomly and each bat corresponding to a potential solution to the given problem; define loudness A_i, pulse frequency Q_i and the initial velocities $v_i (i = 1, 2, \ldots, \text{NP})$; set pulse rate r_i.
Step 2: While the termination criteria is not satisfied **or** $t <$ MaxGeneration **do**
Generate new solutions by adjusting frequency, and updating velocities and locations/solutions [(4)–(6)]
if (rand > r_i) **then**
Select a solution among the best solutions;
Generate a local solution around the selected best solution
end if
Generate a new solution by flying randomly
if (rand < A_i & $f(x_i) < f(x_*)$) **then**
Accept the new solutions
Increase r_i and reduce A_i
end if
Rank the bats and find the current best x_*
$t = t + 1$;
Step 3: end while
Step 4: Post-processing the results and visualization.
End.

Algorithm 2: Bat algorithm.

in a D-dimensional search space. The new solutions x_i^t and velocities v_i^t at time step t are given by

$$f_i = f_{\min} + \left(f_{\max} - f_{\min}\right)\beta, \tag{4}$$

$$v_i^t = v_i^{t-1} + \left(x_i^t - x_*\right) f_i, \tag{5}$$

$$x_i^t = x_i^{t-1} + v_i^t, \tag{6}$$

where $\beta \in [0, 1]$ is a random vector drawn from a uniform distribution. Here x_* is the current global best location (solution) which is located after comparing all the solutions among all the n bats. Generally speaking, depending on the domain size of the problem of interest, the frequency f is assigned to $f_{\min} = 0$ and $f_{\max} = 100$ in practical implementation. Initially, each bat is randomly given a frequency which is drawn uniformly from $[f_{\min}, f_{\max}]$.

For the local search part, once a solution is selected among the current best solutions, a new solution for each bat is generated locally using random walk

$$x_{\text{new}} = x_{\text{old}} + \varepsilon A^t, \tag{7}$$

where $\varepsilon \in [-1, 1]$ is a scaling factor which is a random number, while $A_t = \langle A_i^t \rangle$ is the average loudness of all the bats at time step t.

The update of the velocities and positions of bats is similar to the procedure in the standard PSO [21], as f_i controls the pace and range of the movement of the swarming particles in essence. To some degree, BA can be considered as a balanced combination of the standard PSO and the intensive local search controlled by the loudness and pulse rate.

Furthermore, the loudness A_i and the rate r_i of pulse emission update accordingly as the iterations proceed as shown in (8)

$$A_i^{t+1} = \alpha A_i^t,$$
$$r_i^{t+1} = r_i^0 \left[1 - \exp\left(-\gamma t\right)\right], \tag{8}$$

where α and γ are constants. In essence, α is similar to the cooling factor of a cooling schedule in the simulated annealing (SA) [37]. For simplicity, we set $\alpha = \gamma = 0.9$ in this work.

3. Our Approach: HS/BA

Based on the introduction of HS and BA in previous section, we will explain how we combine the two approaches to form the proposed bat algorithm with harmony search (HS/BA) in this section, which modifies the solutions with poor fitness in order to add diversity of the population to improve the search efficiency.

In general, the standard BA algorithm is adept at exploiting the search space, but at times it may trap into some local optima, so that it cannot perform global search well. For BA, the search depends completely on random walks, so a fast convergence cannot be guaranteed. Firstly presented here, in order to increase the diversity of the population for BA so as to avoid trapping into local optima, a main improvement of adding pitch adjustment operation in HS serving as a mutation operator is made to the BA with the aim of speeding up convergence, thus making the approach more feasible for a wider range of practical applications while preserving the attractive characteristics of the basic BA. In this paper, a hybrid metaheuristic algorithm by inducing the pitch adjustment operation in HS as a mutation operator into bat algorithm, so-called harmony search/bat algorithm (HS/BA), is used to optimize the benchmark functions. The difference between HS/BA and BA is that the mutation operator is used to improve the original BA generating a new solution for each bat. In this way, this method can explore the new search space by the mutation of the HS algorithm and exploit the population information with BA, and therefore can avoid trapping into local optima in BA. In the following, we will show the algorithm HS/BA which is an improvement of HS and BA.

The critical operator of HS/BA is the hybrid harmony search mutation operator, which composes the improvisation of harmony in HS with the BA. The core idea of the proposed hybrid mutation operator is based on two considerations. Firstly, poor solutions can take in many new used features from good solutions. Secondly, the mutation operator can improve the exploration of the new search space. In this way, the strong exploration abilities of the original HS and the exploitation abilities of the BA can be fully developed.

For bat algorithm, as the search relies entirely on random walks, a fast convergence cannot be guaranteed. Described here for the first time, a main improvement of adding mutation operator is made to the BA, including three minor improvements, which are made with the aim of speeding up convergence, thus making the method more practical for a wider range of applications, but without losing the attractive features of the original method.

The first improvement is that we use fixed frequency f and loudness A instead of various frequency f_i and A_i^t. Similar to BA, in HS/BA, each bat is defined by its position x_i^t, velocity v_i^t, the emission pulse rate r_i^t and the fixed frequency f, and loudness A in a d-dimensional search space. The new solutions x_i^t and velocities v_i^t at time step t are given by

$$v_i^t = v_i^{t-1} + \left(x_i^t - x_*\right) f,$$
$$x_i^t = x_i^{t-1} + v_i^t, \tag{9}$$

where x_* is the current global best location (solution) which is located after comparing all the solutions among all the n bats. In our experiments, we make $f = 0.5$. Through a series of simulation experiments on benchmarks in Section 4.2, it was found that setting the parameter of pulse rate r to 0.6 and the loudness A to 0.95 produced the best results.

The second improvement is to add mutation operator in an attempt to increase diversity of the population to improve the search efficiency and speed up the convergence to optima. For the local search part, once a solution is selected among the current best solutions, a new solution for each bat is generated locally using random walk by (7) when ξ is larger than pulse rate r, that is, $\xi > r$, where $\xi \in [0, 1]$ is a random real number drawn from a uniform distribution; while when $\xi \leq r$, we use pitch adjustment operation in HS serving as a mutation operator updating the new solution to increase diversity of the population to improve the search efficiency, as shown in (3). Through testing benchmarks in Section 4.2, it was found that setting the parameter of harmony memory consideration rate HMCR to 0.95 and pitch adjustment rate PAR to 0.1 produced the best results.

The last improvement is the addition of elitism scheme into the HS/BA. As with other population-based optimization algorithms, we typically incorporate some sort of elitism in order to retain the best solutions in the population. This prevents the best solutions from being corrupted by pitch adjustment operator. Note that we use an elitism approach to save the property of the bats that has the best solution in the HS/BA process, so even if pitch adjustment operation ruins its corresponding bat, we have saved it and can revert back to it if needed.

Based on the above-mentioned analyses, the mainframe of the harmony search/bat algorithm (HS/BA) is presented in Algorithm 3.

Begin
Step 1: Initialization. Set the generation counter $t = 1$; initialize the population of NP bats P randomly and each bat corresponding to a potential solution to the given problem; define loudness A; set frequency Q, the initial velocities v, and pulse rate r; set the harmony memory consideration rate HMCR, the pitch adjustment rate PAR and bandwidth bw; set maximum of elite individuals retained KEEP.
Step 2: Evaluate the quality f for each bat in P determined by the objective function $f(x)$.
Step 3: While the termination criteria is not satisfied or $t <$ MaxGeneration **do**
Sort the population of bats P from best to worst by order of quality f for each bat.
Store the KEEP best bats as KEEPBAT.
for $i = 1$:NP (all bats) **do**
$v_i^t = v_i^{t-1} + (v_i^t - x_*) \times Q$
$x_i^t = x_i^{t-1} + v_i^t$
if (rand $> r$) **then**
$x_u^t = x_* + \alpha\varepsilon^t$
end if
for $j = 1$:D (all elements) **do** //Mutate
if (rand < HMCR) **then**
$r_1 = \lceil \text{NP} * \text{rand} \rceil$
$x_v(j) = x_{r_1}(j)$ where $r_1 \in (1, 2, \ldots, \text{HMS})$
if (rand < PAR) **then**
$x_v(j) = x_v(j) + \text{bw} \times (2 \times \text{rand} - 1)$
endif
else
$x_v(j) = x_{\min,\, j} + \text{rand} \times (x_{\max,j} - x_{\min - j})$
endif
endfor j
Evaluate the fitness for the offsprings x_u^t, x_i^t, x_v^t
Select the offspring x_k^t with the best fitness among the offsprings x_u^t, x_i^t, x_v^t.
if (rand < A) **then**
$x_{r_1}^t = x_k^t$;
end if
Replace the KEEP worst bats with the KEEP best bats KEEPBAT stored.
end for i
$t = t + 1$;
Step 4: end while
Step 5: Post-processing the results and visualization;
End.

Algorithm 3: The hybrid meta-heuristic algorithm of HS/BA.

4. Simulation Experiments

In this section, we test the performance of the proposed meta-heuristic HS/BA to global numerical optimization through a series of experiments conducted on benchmark functions.

To allow a fair comparison of running times, all the experiments were conducted on a PC with a Pentium IV processor running at 2.0 GHz, 512 MB of RAM and a hard drive of 160 Gbytes. Our implementation was compiled using MATLAB R2012a (7.14) running under Windows XP3. No commercial BA or HS tools were used in the following experiments.

4.1. General Performance of HS/BA. In order to explore the benefits of HS/BA, in this subsection we compared its performance on global numeric optimization problem with nine other population-based optimization methods, which are ACO, BA, BBO, DE, ES, GA, HS, PSO, and SGA. ACO (ant colony optimization) [38, 39] is a swarm intelligence algorithm for solving optimization problems which is based on the pheromone deposition of ants. BBO (biogeography-based optimization) [24] is a new excellent powerful and efficient evolution algorithm, developed for the global optimization inspired by the immigration and emigration of species between islands (or habitats) in search of more compatible islands. DE (differential evolution) [19] is a simple but excellent optimization method that uses the difference between two solutions to probabilistically adapt a third solution. An ES (evolutionary strategy) [40] is an algorithm that generally distributes equal importance to mutation and recombination and that allows two or more parents to reproduce an offspring. A GA (genetic algorithm) [18] is a search heuristic that mimics the process of natural evolution. PSO (particle swarm optimization) [21] is also a swarm

intelligence algorithm which is based on the swarm behavior of fish and bird schooling in nature. A stud genetic algorithm (SGA) [41] is a GA that uses the best individual at each generation for crossover.

In all experiments, we will use the same parameters for HS, BA and HS/BA that are loudness $A = 0.95$, pulse rate $r = 0.6$, scaling factor $\varepsilon = 0.1$, the harmony memory consideration rate HMCR = 0.95, and the pitch adjustment rate PAR = 0.1. For ACO, BBO, DE, ES, GA, PSO, and SGA, we set the parameters as follows. For ACO, initial pheromone value $\tau_0 = 1E - 6$, pheromone update constant $Q = 20$, exploration constant $q_0 = 1$, global pheromone decay rate $\rho_g = 0.9$, local pheromone decay rate $\rho_l = 0.5$, pheromone sensitivity $s = 1$, and visibility sensitivity $\beta = 5$; for BBO, habitat modification probability = 1, immigration probability bounds per gene = [0, 1], step size for numerical integration of probabilities = 1, maximum immigration and migration rates for each island = 1, and mutation probability = 0.005; for DE, a weighting factor $F = 0.5$ and a crossover constant CR = 0.5; for the ES, the number of offspring $\lambda = 10$ produced in each generation and standard deviation $\sigma = 1$ for changing solutions. For the GA, we used roulette wheel selection and single-point crossover with a crossover probability of 1 and a mutation probability of 0.01. For PSO, we set an inertial constant = 0.3, a cognitive constant = 1, and a social constant for swarm interaction = 1. For the SGA, we used single-point crossover with a crossover probability of 1 and a mutation probability of 0.01.

Well-defined problem sets are favorable for evaluating the performance of optimization methods proposed in this paper. Based on mathematical functions, benchmark functions can be applied as objective functions to perform such tests. The properties of these benchmark functions can be easily achieved from their definitions. Fourteen different benchmark functions are applied to verify our proposed metaheuristic algorithm HS/BA. Each of the functions in this study has 20 independent variables (i.e., $D = 20$).

The benchmark functions described in Table 1 are standard testing functions. The properties of the benchmark functions are given in Table 2. The modality property means the number of the best solutions in the search space. Unimodal benchmark functions only have an optimum, which is the global optimum. Multimodal benchmark functions have at least two optima in their search space, indicating that they have more than one local optima except the global optimum. More details of all the benchmark functions can be found in [6].

We set population size $NP = 50$, an elitism parameter $Keep = 2$, and maximum generation $Maxgen = 50$ for each algorithm. We ran 100 Monte Carlo simulations of each algorithm on each benchmark function to get representative performances. Tables 3 and 4 illustrate the results of the simulations. Table 3 shows the average minima found by each algorithm, averaged over 100 Monte Carlo runs. Table 4 shows the absolute best minima found by each algorithm over 100 Monte Carlo runs. In other words, Table 3 shows the average performance of each algorithm, while Table 4 shows the best performance of each algorithm. The best value achieved for each test problem is shown in bold. Note that the normalizations in the tables are based on different scales, so values cannot be compared between the two tables.

From Table 3, we see that, on average, HS/BA is the most effective at finding objective function minimum on ten of the fourteen benchmarks (F02, F03, F06–F11, and F13-F14). ACO is the second most effective, performing the best on the benchmarks F04 and F05. BBO and SGA are the third most effective, performing the best on the benchmarks F12 and F01, respectively, when multiple runs are made. Table 4 shows that HS/BA performs the best on thirteen of the fourteen benchmarks (F02–F14), while BBO performs the second best at finding objective function minimum on the only benchmark (F01) when multiple runs are made. In addition, statistical analysis [42] on these values obtained by the ten methods on 14 benchmark functions based on the Friedman's test [43] reveals that the differences in the obtained average and the best function minima are statistically significant ($P = 1.66 * 10^{-17}$ and $P = 7.25 * 10^{-17}$, resp.) at the confidence level of 5%.

Furthermore, the computational requirements of the ten optimization methods were similar. We collected the average computational time of the optimization methods as applied to the 14 benchmarks discussed in this section. The results are shown in Table 3. BA was the quickest optimization method. HS/BA was the third fastest of the ten algorithms. However, it should be noted that in the vast majority of real-world applications, it is the fitness function evaluation that is by far the most expensive part of a population-based optimization algorithm.

Furthermore, convergence graphs of ACO, BA, BBO, DE, ES, GA, HS, HS/BA, PSO, and SGA are shown in Figures 1–14 which represent the process of optimization. The values shown in Figures 1–14 are the mean objective function optimum achieved from 100 Monte Carlo simulations, which are the true objective function values, not normalized.

Figure 1 shows the results obtained for the ten methods when the F01 Ackley function is applied. From Figure 1, clearly, we can draw the conclusion that BBO is significantly superior to all the other algorithms including HS/BA during the process of optimization, while HS/BA performs the second best on this multimodal benchmark function. Here, all the other algorithms show the almost same starting point, and HS/BA has a faster convergence rate than other algorithms, while HS/BA is outperformed by BBO after 13 generations. Although slower, SGA eventually finds the global minimum close to HS/BA. Obviously, BBO, HS/BA, and SGA outperform ACO, BA, DE, ES, GA, HS, and PSO during the whole process of searching the global minimum.

Figure 2 illustrates the optimization results for F02 Fletcher-Powell function. In this multimodal benchmark problem, it is obvious that HS/BA outperforms all other methods during the whole progress of optimization (except the generation 20–33, BBO, and HS/BA performs approximately equally).

Figure 3 shows the optimization results for F03 Griewank function. From Table 3 and Figure 3, we can see that HS/BA performs the best on this multimodal benchmark problem. Looking at Figure 3 carefully, HS/BA shows a faster convergence rate initially than ACO; however, it is outperformed

Table 1: Benchmark functions.

No.	Name	Definition	Source
F01	Ackley	$f(\vec{x}) = 20 + e - 20 \cdot e^{-0.2 \cdot \sqrt{(1/n)\sum_{i=1}^{n} x_i^2}} - e^{(1/n)\sum_{i=1}^{n} \cos(2\pi x_i)}$	[3]
F02	Fletcher-Powell	$f(\vec{x}) = \sum_{i=1}^{n} (A_i - B_i)^2, \quad A_i = \sum_{j=1}^{n} \left(a_{ij} \sin \alpha_j + b_{ij} \cos \alpha_j\right)$ $B_i = \sum_{j=1}^{n} \left(a_{ij} \sin x_j + b_{ij} \cos x_j\right)$	[4]
F03	Griewank	$f(\vec{x}) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	[5]
F04	Penalty #1	$f(\vec{x}) = \frac{\pi}{30} \left\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 \cdot [1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4), \quad y_i = 1 + 0.25(x_i + 1)$	[6]
F05	Penalty #2	$f(\vec{x}) = 0.1 \left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n-1} (x_i - 1)^2 \cdot [1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	[6]
F06	Quartic *with noise*	$f(\vec{x}) = \sum_{i=1}^{n} (i \cdot x_i^4 + U(0, 1))$	[7]
F07	Rastrigin	$f(\vec{x}) = 10 \cdot n + \sum_{i=1}^{n} (x_i^2 - 10 \cdot \cos(2\pi x_i))$	[8]
F08	Rosenbrock	$f(\vec{x}) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$	[7]
F09	Schwefel 2.26	$f(\vec{x}) = 418.9829 \times D - \sum_{i=1}^{D} x_i \sin\left(\lvert x_i \rvert^{1/2}\right)$	[9]
F10	Schwefel 1.2	$f(\vec{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$	[10]
F11	Schwefel 2.22	$f(\vec{x}) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	[10]
F12	Schwefel 2.21	$f(\vec{x}) = \max_i \{\lvert x_i \rvert, 1 \le i \le n\}$	[6]

Table 1: Continued.

No.	Name	Definition	Source
F13	Sphere	$f(\vec{x}) = \sum_{i=1}^{n} x_i^2$	[7]
F14	Step	$f(\vec{x}) = 6 \cdot n + \sum_{i=1}^{n} \lfloor x_i \rfloor$	[7]

* In benchmark function F02, the matrix elements $\mathbf{a}_{n\times n}, \mathbf{b}_{n\times n} \in (-100, 100), \boldsymbol{\alpha}_{n\times 1} \in (-\pi, \pi)$ are drawn from uniform distribution [4].
* In benchmark functions F04 and F05, the definition of the function $u(x_i, a, k, m)$ is given by $u(x_i, a, k, m) = k(x_i - a)^m, x_i > a; 0, -a \le x_i \le a; k(-x_i - a)^m, x_i < -a$.

Table 2: Properties of benchmark functions; lb denotes lower bound, ub denotes upper bound, and opt denotes optimum point.

No.	Function	lb	ub	opt	Continuity	Modality
F01	Ackley	−32.768	32.768	0	Continuous	Multimodal
F02	Fletcher-Powell	$-\pi$	π	0	Continuous	Multimodal
F03	Griewank	−600	600	0	Continuous	Multimodal
F04	Penalty #1	−50	50	0	Continuous	Multimodal
F05	Penalty #2	−50	50	0	Continuous	Multimodal
F06	Quartic *with noise*	−1.28	1.28	1	Continuous	Multimodal
F07	Rastrigin	−5.12	5.12	0	Continuous	Multimodal
F08	Rosenbrock	−2.048	2.048	0	Continuous	Unimodal
F09	Schwefel 2.26	−512	512	0	Continuous	Multimodal
F10	Schwefel 1.2	−100	100	0	Continuous	Unimodal
F11	Schwefel 2.22	−10	10	0	Continuous	Unimodal
F12	Schwefel 2.21	−100	100	0	Continuous	Unimodal
F13	Sphere	−5.12	5.12	0	Continuous	Unimodal
F14	Step	−5.12	5.12	0	Discontinuous	Unimodal

Table 3: Mean normalized optimization results in fourteen benchmark functions. The values shown are the minimum objective function values found by each algorithm, averaged over 100 Monte Carlo simulations.

	ACO	BA	BBO	D*E*	*ES*	GA	HS	HSBA	PSO	SGA
F01	2.31	3.33	1.15	2.02	3.38	2.72	3.47	1.09	2.66	**1.00**
F02	24.58	25.82	1.58	8.94	24.35	5.45	15.69	**1.00**	13.96	1.33
F03	3.16	60.72	1.93	5.44	23.85	3.22	77.22	**1.00**	25.77	1.42
F04	**1.00**	3.0*E*38	4.0*E*32	5.6*E*33	2.7*E*38	3.1*E*32	1.4*E*39	2.3*E*32	4.1*E*36	9.6*E*31
F05	**1.00**	1.1*E*8	299.42	1.5*E*6	4.6*E*8	5.4*E*3	4.1*E*8	215.51	5.5*E*7	111.10
F06	489.01	6.8*E*3	35.32	308.29	1.8*E*4	274.83	1.5*E*4	**1.00**	2.5*E*3	10.09
F07	8.09	11.55	1.28	6.56	11.87	6.17	10.22	**1.00**	8.44	1.80
F08	42.25	29.01	2.29	7.59	59.99	9.05	47.85	**1.00**	12.04	2.15
F09	3.17	20.26	1.99	13.58	13.33	1.81	19.92	**1.00**	17.61	1.15
F10	1.75	3.73	1.38	2.95	4.93	1.25	4.22	**1.00**	2.48	1.48
F11	1.05	19.70	1.83	7.14	23.12	11.13	19.45	**1.00**	13.22	2.46
F12	1.86	4.03	**1.00**	2.99	3.91	1.92	3.74	1.38	2.38	1.12
F13	98.30	150.84	3.80	19.03	226.52	47.74	182.32	**1.00**	72.91	4.02
F14	7.73	120.48	3.93	13.31	102.56	11.53	146.55	**1.00**	63.44	3.28
Tim*E*	2.74	**1.00**	1.32	1.64	1.67	1.79	2.33	1.43	2.03	1.76

* The values are normalized so that the minimum in each row is 1.00. These are not the absolute minima found by each algorithm, but the average minima found by each algorithm.

by ACO after 5 generations. For other algorithms, although slower, BBO, GA, and SGA eventually find the global minimum close to HS/BA, while BA, DE, ES, HS, and PSO fail to search the global minimum within the limited iterations.

Figure 4 shows the results for F04 Penalty #1 function. From Figure 4, it is obvious that HS/BA outperforms all other methods during the whole progress of optimization in this multimodal benchmark function. Although slower, SGA and BBO perform the second and third best at finding the global minimum.

Figure 5 shows the performance achieved for F05 Penalty #2 function. For this multimodal function, similar to F04 Penalty #1 function shown in Figure 4, HS/BA is significantly superior to all the other algorithms during the process of optimization. Here, PSO shows a faster convergence rate initially than HS/BA; however, it is outperformed by HS/BA after 3 generations.

Figure 6 shows the results achieved for the ten methods when using the F06 Quartic (*with noise*) function. From Table 3 and Figure 6, we can conclude that HS/BA performs the best in this multimodal function. Looking carefully at Figure 6, PSO and HS/BA show the same initial fast convergence towards the known minimum, as the procedure proceeds, HS/BA gets closer and closer to the minimum, while PSO comes into being premature and traps into the local minimum. BA, ES, GA, HS, and PSO do not manage

Table 4: Best normalized optimization results in fourteen benchmark functions. The values shown are the minimum objective function values found by each algorithm.

	ACO	BA	BBO	DE	ES	GA	HS	HSBA	PSO	SGA
F01	1.85	2.31	**1.00**	1.43	2.25	2.03	2.30	1.05	1.91	1.04
F02	10.42	14.48	1.09	4.22	10.39	4.03	9.91	**1.00**	8.28	1.33
F03	2.73	46.22	1.87	4.47	21.38	8.02	43.24	**1.00**	16.75	1.76
F04	4.4*E*6	5.0*E*6	1.2*E*3	1.9*E*4	2.2*E*6	3.0*E*4	4.2*E*6	**1.00**	3.755	3.22
F05	3.0*E*4	3.2*E*4	51.01	386.33	1.6*E*4	884.46	2.6*E*4	**1.00**	4.113	4.57
F06	58.51	992.55	6.50	24.69	808.48	59.24	746.91	**1.00**	189.71	2.02
F07	5.71	8.53	1.25	5.13	7.94	5.23	7.47	**1.00**	6.00	1.68
F08	26.42	25.03	1.48	3.70	33.52	6.16	21.50	**1.00**	7.75	1.45
F09	2.43	8.66	1.28	4.90	5.93	2.09	7.28	**1.00**	7.40	1.42
F10	1.89	4.94	1.18	2.66	3.00	2.08	2.76	**1.00**	2.01	1.74
F11	7.74	12.61	1.21	3.36	12.14	6.01	9.60	**1.00**	7.81	1.62
F12	1.33	2.33	1.44	1.69	2.08	1.74	2.11	**1.00**	1.70	1.21
F13	28.04	56.57	2.17	5.54	60.57	19.71	52.86	**1.00**	20.98	2.28
F14	4.28	54.02	1.97	4.85	33.61	10.60	49.11	**1.00**	19.72	1.85

* The values are normalized so that the minimum in each row is 1.00. These are the absolute best minima found by each algorithm.

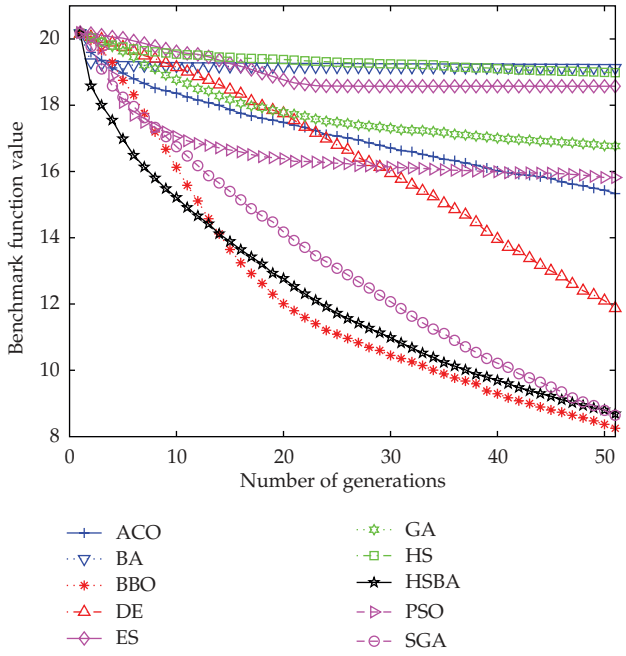

Figure 1: Comparison of the performance of the different methods for the F01 Ackley function.

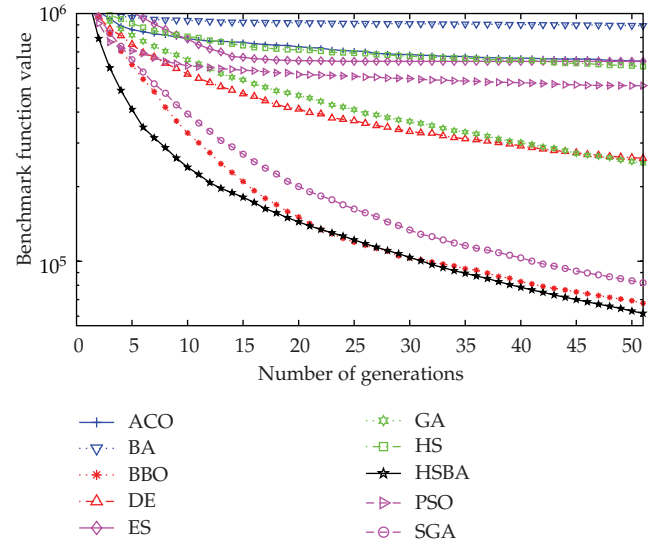

Figure 2: Comparison of the performance of the different methods for the F02 Fletcher-Powell function.

to succeed in this benchmark function within maximum number of generations. At last, BBO, DE, and SGA converge to the value very close to HS/BA.

Figure 7 shows the optimization results for the F07 Rastrigin function. Very clearly, HS/BA has the fastest convergence rate at finding the global minimum and significantly outperforms all other approaches. Looking carefully at Figure 7, approximately, we can divide all the algorithms into three groups: one group including BBO, HS/BA, and SGA performing the best; the other group including ACO, DE, GA, and PSO performing the second best; another group including BA, ES, and HS performing the worst.

Figure 8 shows the results for F08 Rosenbrock function. From Table 3 and Figure 8, we can see that HS/BA is superior to the other algorithms during the optimization process in this relatively simple unimodal benchmark function. Also, we see that, similar to the F06 Quartic (*with noise*) function shown in Figure 6, PSO shows a faster convergence rate initially than HS/BA; however, it is outperformed by HS/BA after 4 generations and is premature to the local minimum.

Figure 9 shows the equivalent results for the F09 Schwefel 2.26 function. From Figure 9, very clearly, though HS/BA is outperformed by BBO between the generations 10–30, it has the stable convergence rate at finding the global minimum

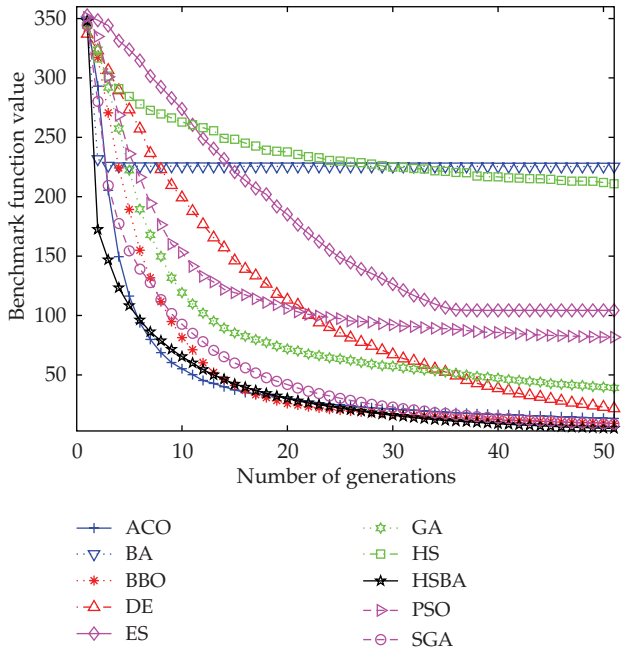

Figure 3: Comparison of the performance of the different methods for the F03 Griewank function.

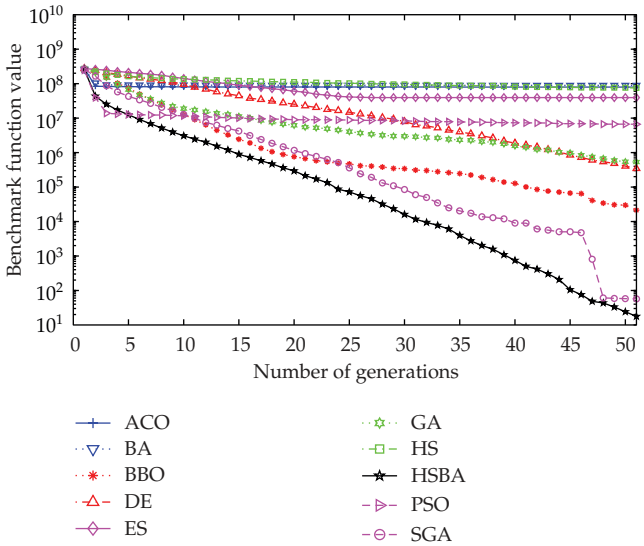

Figure 4: Comparison of the performance of the different methods for the F04 Penalty #1 function.

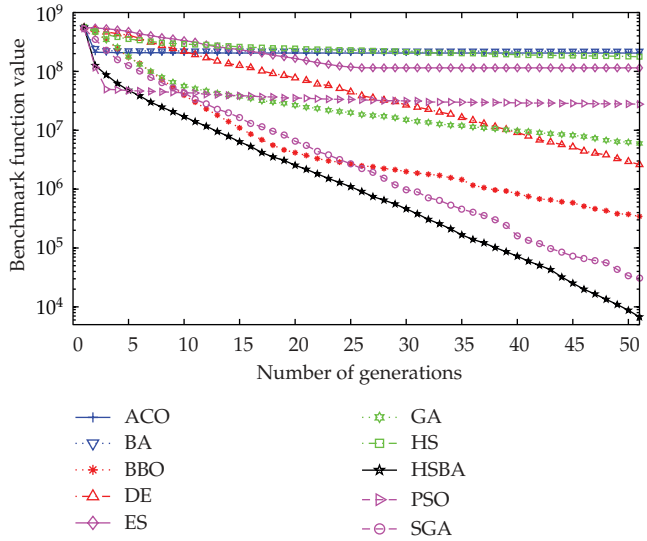

Figure 5: Comparison of the performance of the different methods for the F05 Penalty #2 function.

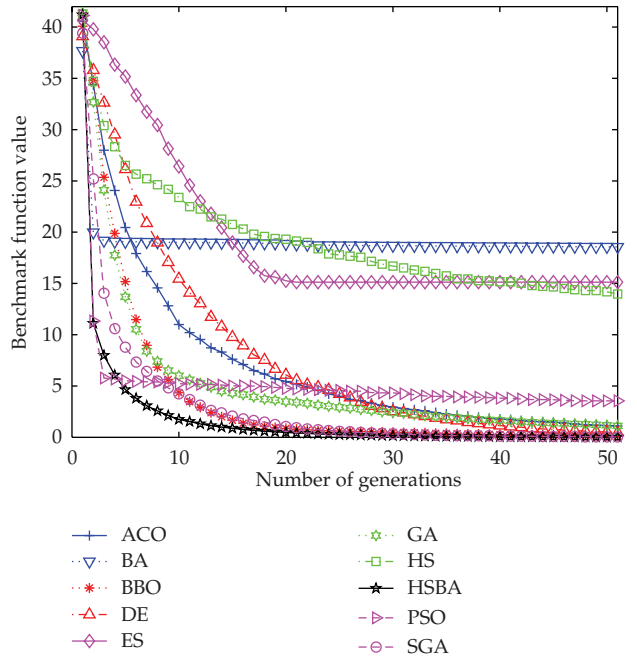

Figure 6: Comparison of the performance of the different methods for the F06 Quartic (*with noise*) function.

and significantly outperforms all other approaches in this multimodal benchmark function.

Figure 10 shows the results for F10 Schwefel 1.2 function. From Figure 10, we can see that HS/BA performs far better than other algorithms during the optimization process in this relative simple unimodal benchmark function. PSO shows a faster convergence rate initially than HS/BA; however, it is outperformed by HS/BA after 3 generations. Looking carefully at Figure 10, akin to F07 Rastrigin function shown in Figure 7, all the algorithms except BA can be approximately divided into three groups: one group including BBO and HS/BA performing the best; the other group including ACO, GA, PSO, and SGA performing the second best; another group including DE, ES, and HS performing the worst.

Figure 11 shows the results for F11 Schwefel 2.22 function. Very clearly, BBO shows a faster convergence rate initially than HS/BA; however, it is outperformed by HS/BA after 40 generations. At last, HS/BA reaches the optimal solution significantly superior to other algorithms. BBO is only inferior

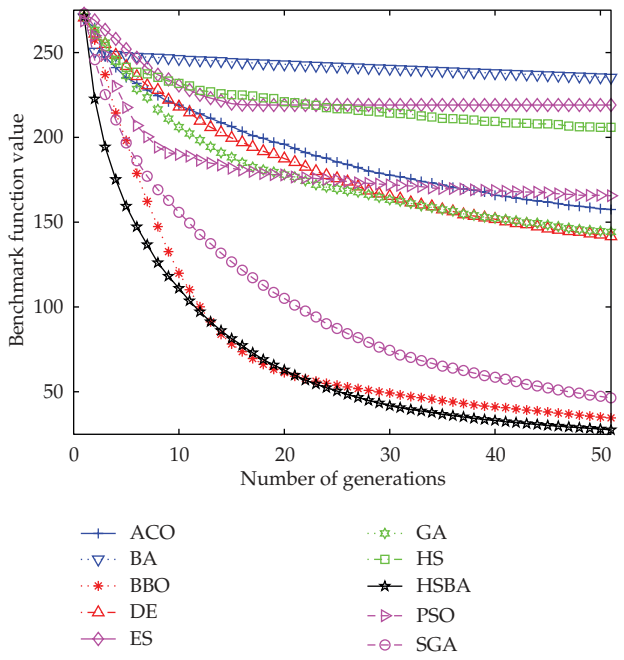

Figure 7: Comparison of the performance of the different methods for the F07 Rastrigin function.

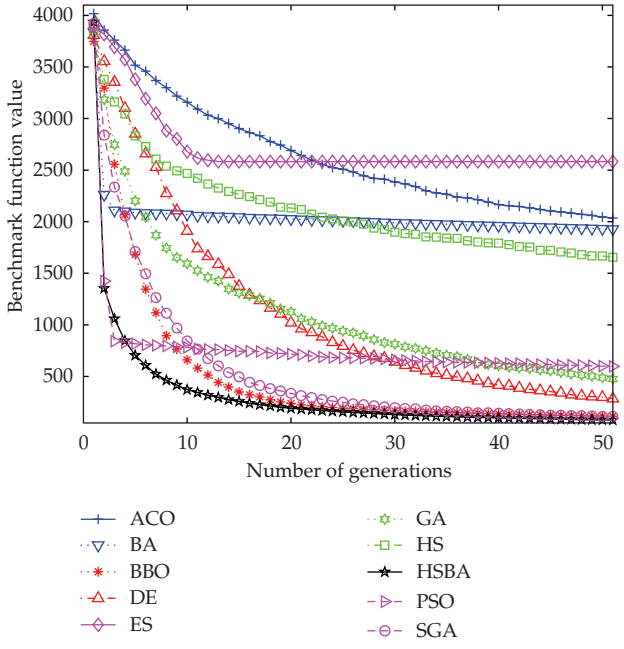

Figure 8: Comparison of the performance of the different methods for the F08 Rosenbrock function.

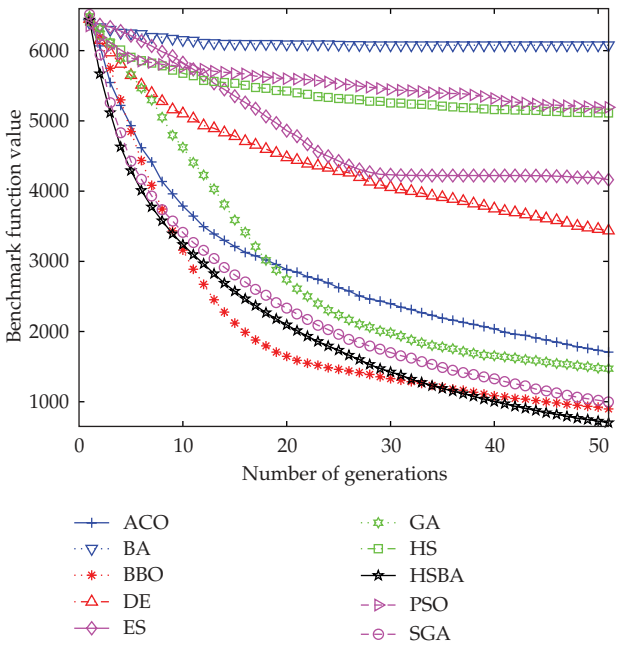

Figure 9: Comparison of the performance of the different methods for the F09 Schwefel 2.26 function.

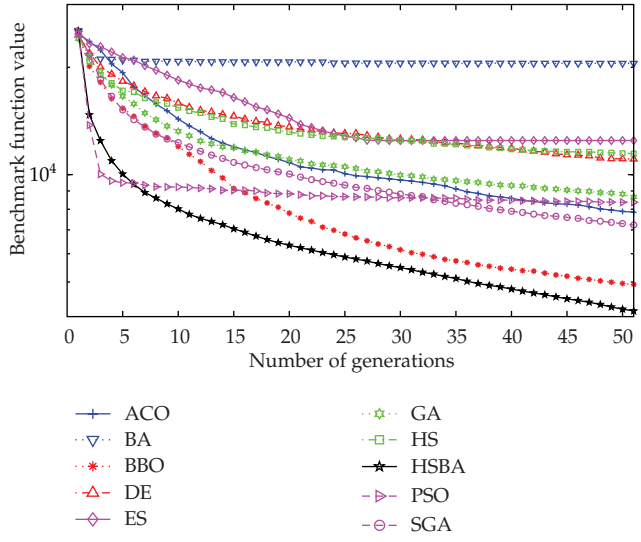

Figure 10: Comparison of the performance of the different methods for the F10 Schwefel 1.2 function.

to HS/BA and performs the second best in this unimodal function.

Figure 12 shows the results for F12 Schwefel 2.21 function. Very clearly, HS/BA has the fastest convergence rate at finding the global minimum and significantly outperforms all other approaches.

Figure 13 shows the results for F13 Sphere function. From Table 3 and Figure 13, HS/BA has the fastest convergence rate at finding the global minimum and outperforms all other approaches. Looking carefully at Figure 13, for BBO and SGA, we can see that BBO has a faster convergence rate than SGA, but SGA does finally converge to the value of BBO that is approaching to HS/BA.

Figure 14 shows the results for F14 Step function. Apparently, HS/BA shows the fastest convergence rate at finding the global minimum and significantly outperforms all other approaches. Looking carefully at Figure 14, for BBO and SGA,

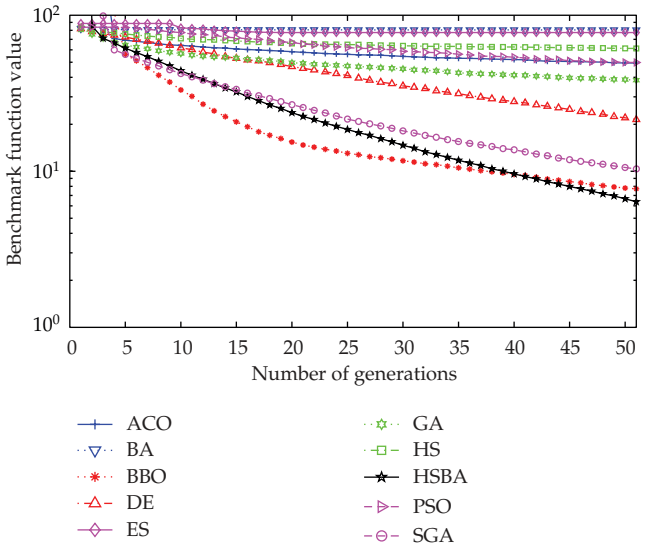

Figure 11: Comparison of the performance of the different methods for the F11 Schwefel 2.22 function.

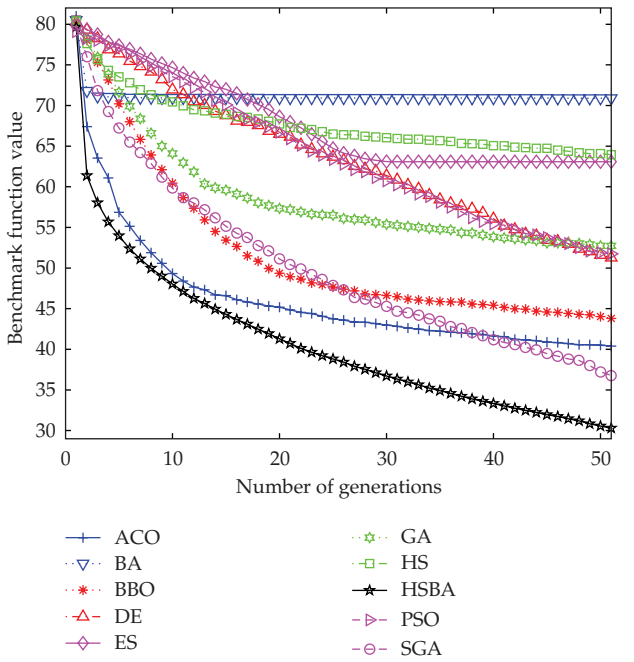

Figure 12: Comparison of the performance of the different methods for the F12 Schwefel 2.21 function.

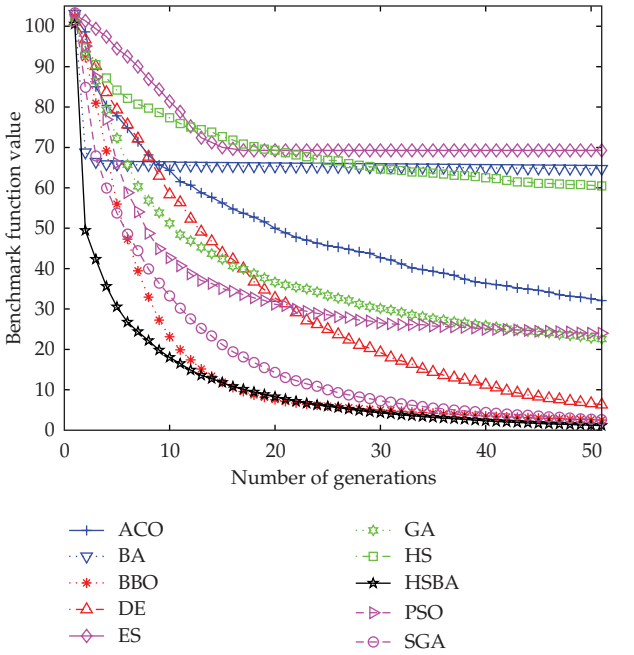

Figure 13: Comparison of the performance of the different methods for the F13 Sphere function.

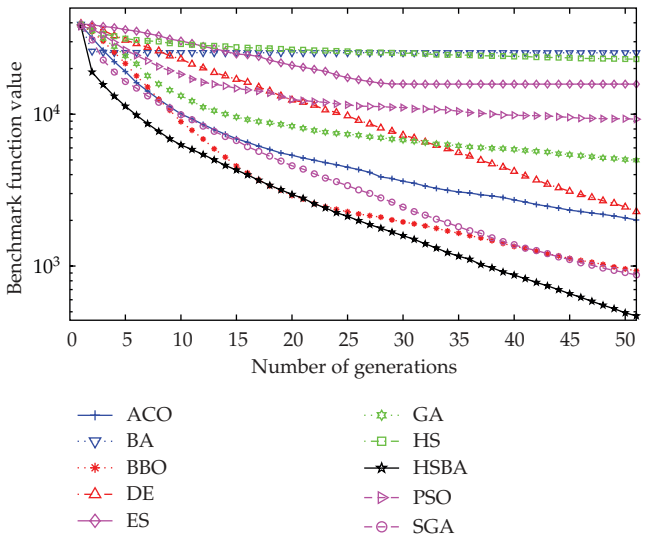

Figure 14: Comparison of the performance of the different methods for the F14 Step function.

we can see that BBO has a faster convergence rate than SGA, but SGA does finally converge to the value of BBO.

From the above-analyses about Figures 1–14, we can come to the conclusion that our proposed hybrid metaheuristic algorithm HS/BA significantly outperforms the other nine algorithms. In general, ACO, BBO, and SGA are only inferior to the HS/BA, and ACO, BBO, and SGA perform better than HS/BA in the benchmarks F04-F05, the benchmark F12, and the benchmark F01, respectively. Further, benchmarks F04, F05, F06, F08, and F10 illustrate that PSO has a faster convergence rate initially, while later it converges slower and slower to the true objective function value. At last, we must point out that, in [24], Simon compared BBO with seven state-of-the-art EAs over 14 benchmark functions and a real-world sensor selection problem. The results demonstrated the good performance of BBO. It is also indirectly demonstrated that our proposed hybrid metaheuristic method HS/BA is a more powerful and efficient optimization algorithm than other population-based optimization algorithms.

Table 5: Best normalized optimization results in fourteen benchmark functions with different A. The numbers shown are the best results found after 100 Monte Carlo simulations HS/BA algorithm.

	A										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
F01	1.66	1.33	1.55	1.40	**1.00**	1.33	1.21	1.22	1.16	1.13	1.14
F02	1.25*E*4	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F03	3.41	11.15	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F04	2.73*E*4	**1.00**	1.25*E*4	1.04	1.04	1.04	1.04	1.04	1.04	**1.00**	**1.00**
F05	5.03*E*5	5.30	1.33	1.68*E*4	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F06	**1.00**	2.56	5.47	4.73	17.81	9.90	2.89	6.74	9.90	2.60	5.57
F07	10.14	14.92	4.35	1.10	1.10	15.37	1.01	**1.00**	**1.00**	**1.00**	**1.00**
F08	38.49	1.08	14.06	1.08	1.08	1.08	99.01	1.01	**1.00**	**1.00**	**1.00**
F09	285.18	404.58	1.38	4.74	1.19	1.19	1.19	367.91	1.01	**1.00**	**1.00**
F10	665.22	4.68	1.69*E*3	15.27	1.18	1.18	1.18	1.18	1.21	**1.00**	**1.00**
F11	20.03	**1.00**	9.96	11.54	39.22	9.96	9.96	9.96	9.96	38.97	8.49
F12	10.32	**1.00**	13.22	3.23	14.69	2.79	2.79	2.79	2.79	2.79	19.31
F13	1.46	4.15	2.84	4.47	1.15	1.56	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F14	527.52	13.57	3.95	1.01	1.15	12.91	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	1	4	2	2	3	3	5	6	7	**10**	**10**

4.2. Influence of Control Parameter. In [28], Dr. Yang concluded that if the parameters in BA can be adjusted properly, it can outperform GA, HS (harmony search) [26], and PSO. The choice of the control parameters is of vital importance for different problems. To investigate the influence of the loudness A, pulse rate r, harmony memory consideration rate HMCR, and pitch adjustment rate PAR on the performance of HS/BA, we carry out this experiment in the benchmark problem with different parameters. All other parameter settings are kept unchanged (unless noted otherwise in the following paragraph). The results are recorded in Tables 5–12 after 100 Monte Carlo runs. Among them, Tables 5, 7, 9, and 11 show the best minima found by HS/BA algorithm over 100 Monte Carlo runs. Tables 6, 8, 10, and 12 show the average minima found by HS/BA algorithm, averaged over 100 Monte Carlo runs. In other words, Tables 5, 7, 9, 11 and Tables 6, 8, 10, 12 show the best and average performance of HS/BA algorithm, respectively. In each table, the last row is the total number of functions on which HS/BA performs the best with some parameters.

4.2.1. Loudness: A. Tables 5 and 6 recorded the results performed in the benchmark problem with the loudness $A = 0, 0.1, 0.2, \ldots, 0.9, 1.0$ and fixed pulse rate $r = 0.6$, harmony memory consideration rate HMCR = 0.95, and pitch adjustment rate PAR = 0.1. From Tables 5 and 6, obviously, it can be seen that (i) for the three benchmark functions F02, F03, and F04, HS/BA performs slightly differently, that is to say, these three benchmark functions are insensitive to the parameter A. (ii) For benchmark functions F01, F06, F11, and F12, HS/BA performs better on smaller $A(<0.5)$. (iii) However, for functions F05, F07–10, F13, and F14, HS/BA performs better on bigger $A(>0.5)$. In sum, HS/BA performs the best when A is equal or very close to 1.0. So, we set $A = 0.95$, which is very close to 0.9 and 1.0 in other experiments.

4.2.2. Pulse Rate: r. Tables 7 and 8 recorded the results performed on the benchmark problem with the pulse rate $r = 0, 0.1, 0.2, \ldots, 0.9, 1.0$ and fixed loudness$A = 0.95$, harmony memory consideration rate HMCR = 0.95, and pitch adjustment rate PAR = 0.1. From Tables 7 and 8, we can evidently conclude that HS/BA performs significantly better on bigger $r(>0.5)$ than on smaller $r(<0.5)$, and HS/BA performs the best when r is equal or very close to 0.6. So, we set $r = 0.6$ in other experiments.

4.2.3. Harmony Memory Consideration Rate: HMCR. Tables 9 and 10 recorded the results performed in the benchmark problem with the harmony memory consideration rate HMCR $= 0, 0.1, 0.2, \ldots, 0.9, 1.0$, fixed loudness $A = 0.95$, pulse rate $r = 0.6$, and pitch adjustment rate PAR = 0.1. From Tables 9 and 10, we can recognize that the function values evaluated by HS/BA are better/smaller on bigger HMCR (>0.5) than on smaller r (<0.5), and most benchmarks reach minimum when HMCR is equal or very close to 1. So, we set HMCR = 0.95 in other experiments.

4.2.4. Pitch Adjustment Rate: PAR. Tables 11 and 12 recorded the results performed on the benchmark problems with the pitch adjustment rate PAR $= 0, 0.1, 0.2, \ldots, 0.9, 1.0$, fixed loudness $A = 0.95$, pulse rate $r = 0.6$, and harmony memory consideration rate HMCR = 0.95. From Tables 11 and 12, we can recognize that the function values for HS/BA vary little with the increasing PAR, and HS/BA reaches optimum/minimum in most benchmarks when PAR is equal or very close to 0.1. So, we set PAR = 0.1 in other experiments.

4.3. Discussion. In the HS/BA, the bats fly in the sky using echolocation to find food/prey (i.e., best solutions). Four other parameters are the loudness (A), the rate of pulse emission (r), harmony memory consideration rate (HMCR),

Table 6: Mean normalized optimization results in fourteen benchmark functions with different A. The numbers shown are the minimum objective function values found by HS/BA algorithm, averaged over 100 Monte Carlo simulations.

	A										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
F01	1.01	1.01	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F02	1.52	**1.00**	1.46	1.17	1.08	1.37	1.88	1.65	1.67	1.79	1.48
F03	6.69	10.41	1.06	1.01	1.03	1.03	1.01	1.02	1.05	**1.00**	**1.00**
F04	201.31	1.26	4.10	4.21	3.33	2.72	**1.00**	5.19	2.91	**1.00**	3.11
F05	591.22	16.82	4.25	5.87	1.01	4.14	24.04	**1.00**	3.68	7.33	9.00
F06	**1.00**	4.90E4	637.99	258.71	4.95E4	2.18	2.17	2.17	2.18	2.17	2.17
F07	8.57	2.21E6	6.43	272.34	110.48	2.11E4	1.02	1.01	**1.00**	**1.00**	**1.00**
F08	49.80	1.14E4	1.72E4	161.60	224.59	91.19	1.74E4	1.03	1.11	**1.00**	**1.00**
F09	82.37	1.73E6	1.06	3.14	74.45	103.10	42.26	7.94E3	**1.00**	1.37	1.29
F10	90.31	1.45E3	2.45E5	2.32E3	23.34	22.34	31.38	12.89	2.34E3	1.40	**1.00**
F11	4.98	**1.00**	3.15E4	2.33	14.41	520.23	443.65	616.26	249.91	4.78E4	2.14
F12	3.69	2.10E4	4.57E4	**1.00**	2.12E4	266.35	233.13	198.80	276.14	112.01	2.14E4
F13	1.90	8.25	4.48E6	2.14E6	**1.00**	6.15	254.51	222.75	189.96	263.86	107.01
F14	66.91	1.95E5	1.17E4	1.24E3	21.04	1.86E3	29.40	23.92	**1.00**	18.35	24.75
	1	2	1	2	2	1	2	2	4	5	5

Table 7: Best normalized optimization results in fourteen benchmark functions with different r. The numbers shown are the best results found after 100 Monte Carlo simulations of HS/BA algorithm.

	r										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
F01	1.84	1.84	1.84	4.84	1.59	1.71	**1.00**	1.34	1.09	1.16	1.01
F02	9.44E3	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F03	3.73	5.46	1.86	1.86	1.86	1.86	**1.00**	1.72	1.21	1.61	1.01
F04	11.42	1.11	24.29	1.23	1.26	**1.00**	1.29	1.29	1.29	1.29	1.29
F05	1.77E5	2.30	1.15	1.09E4	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F06	62.44	29.96	48.58	15.60	26.34	7.80	1.82	**1.00**	35.20	2.39	1.55
F07	8.31	4.48	2.58	1.29	1.29	3.49	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F08	9.61	1.18	4.20	1.37	1.37	1.37	6.95	1.01	1.01	**1.00**	**1.00**
F09	373.94	444.22	1.68	3.35	1.68	1.68	**1.00**	291.98	1.01	1.01	1.01
F10	386.93	3.16	13.97	4.63	1.58	1.58	**1.00**	1.58	309.31	1.58	1.01
F11	42.55	**1.00**	18.29	21.35	20.18	11.11	19.04	21.35	15.81	18.76	11.55
F12	114.40	**1.00**	141.84	44.48	125.47	44.48	44.48	44.48	44.48	44.48	69.43
F13	6.39	6.37	6.01	2.96	3.02	**1.00**	1.52	1.39	**1.00**	2.57	2.49
F14	120.52	4.04	2.33	**1.00**	1.16	3.40	**1.00**	**1.00**	1.16	1.16	1.16
	0	3	1	2	2	4	**8**	5	4	4	4

and pitch adjustment rate (PAR). The appropriate update for the pitch adjustment rate (PAR) and the rate of pulse emission (r) balances the exploration and exploitation behavior of each bat, respectively, as the loudness usually decreases once a bat has found its prey/solution, while the rate of pulse emission increases in order to raise the attack accuracy.

For all of the standard benchmark functions that have been considered, the HS/BA has been demonstrated to perform better than or be equal to the standard BA and other acclaimed state-of-the-art population-based algorithms with the HS/BA performing significantly better in some functions. The HS/BA performs excellently and efficiently because of its ability to simultaneously carry out a local search, still searching globally at the same time. It succeeds in doing this due to the local search via harmony search algorithm and global search via bat algorithm concurrently. A similar behavior may be performed in the PSO by using multiswarm from a particle population initially [44]. However, HS/BA's advantages include performing simply and easily, and only having four parameters to regulate. The work carried out here demonstrates the HS/BA to be robust over all kinds of benchmark functions.

Benchmark evaluation is a good way for verifying the performance of the metaheuristic algorithms, but it also has limitations. First, we did not make any special effort to tune the optimization algorithms in this section. Different tuning

Table 8: Mean normalized optimization results in fourteen benchmark functions with different r. The numbers shown are the minimum objective function values found by HS/BA algorithm, averaged over 100 Monte Carlo simulations.

	r										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
F01	1.84	1.99	1.86	1.94	1.59	**1.00**	1.42	1.34	1.09	1.16	1.71
F02	3.31	5.38	4.57	3.08	3.82	1.14	**1.00**	2.89	2.79	4.02	3.11
F03	3.73	5.46	5.33	2.29	3.36	2.41	**1.00**	1.72	1.21	1.61	1.36
F04	11.42	1.11	24.29	1.49*E*4	1.13*E*3	3.94*E*3	2.78*E*4	19.66	175.22	1.14	**1.00**
F05	16.09	128.62	32.69	24.46	64.40	12.77	29.59	76.17	4.56	**1.00**	4.46
F06	62.44	26.96	48.58	15.60	26.34	7.80	1.82	**1.00**	35.20	2.39	1.55
F07	3.30	1.78	1.34	1.03	1.23	1.39	1.55	**1.00**	1.19	1.36	3.49
F08	4.88	3.93	3.94	1.94	2.19	3.83	3.53	4.33	**1.00**	2.27	3.14
F09	1.39	1.66	1.14	**1.00**	1.21	1.15	1.15	1.09	1.08	1.04	1.09
F10	7.21	9.94	8.75	3.60	8.46	6.11	4.83	4.14	5.76	**1.00**	2.71
F11	3.82	4.23	3.73	2.05	1.81	**1.00**	1.71	2.20	1.42	1.68	1.89
F12	1.64	2.17	2.04	1.47	1.80	1.86	**1.00**	1.21	1.13	1.34	1.56
F13	6.39	6.37	6.01	2.96	3.02	1.90	1.52	1.39	**1.00**	2.57	2.49
F14	5.74	15.43	7.42	6.82	6.44	3.17	**1.00**	4.85	2.21	1.74	2.09
	0	0	0	1	0	2	**4**	2	2	2	1

Table 9: Best normalized optimization results in fourteen benchmark functions with different HMCR. The numbers shown are the best results found after 100 Monte Carlo simulations of HS/BA algorithm.

	HMCR										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
F01	1.64	1.64	1.64	1.64	1.64	1.64	1.64	1.51	1.28	**1.00**	1.63
F02	1.87*E*4	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F03	19.82	26.49	3.71	3.71	3.71	3.71	3.71	3.71	3.71	2.06	**1.00**
F04	1.26*E*5	**1.00**	1.87*E*4	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F05	6.07*E*5	5.34	**1.00**	1.87*E*4	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F06	108.92	246.31	365.04	345.40	338.57	234.65	143.49	45.28	23.30	**1.00**	25.75
F07	9.22	14.28	5.34	**1.00**	**1.00**	9.30	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F08	18.10	1.08	7.72	1.08	1.08	1.08	35.94	**1.00**	**1.00**	**1.00**	**1.00**
F09	280.04	391.61	1.27	6.81	1.27	1.27	1.27	227.95	**1.00**	**1.00**	**1.00**
F10	551.65	8.76	1.76*E*3	11.70	1.64	1.64	1.64	1.64	274.48	**1.00**	**1.00**
F11	14.67	**1.00**	6.18	6.18	13.99	6.18	6.18	6.18	4.40	2.69	6.16
F12	7.68	**1.00**	11.10	2.73	10.21	2.73	2.73	2.73	2.73	2.73	4.73
F13	7.72	26.75	15.90	17.76	6.12	10.40	6.12	5.95	2.55	**1.00**	2.25
F14	537.16	14.28	5.34	**1.00**	**1.00**	7.13	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
	0	4	2	4	5	3	5	6	7	**11**	**9**

parameter values in the optimization algorithms might result in significant differences in their performance. Second, real-world optimization problems may not have much of a relationship to benchmark functions. Third, benchmark tests might result in different conclusions if the grading criteria or problem setup change. In our work, we examined the mean and best results obtained with a certain population size and after a certain number of generations. However, we might arrive at different conclusions if (for example) we change the generation limit, or look at how many generations it takes to reach a certain function value, or if we change the population size. In spite of these caveats, the benchmark results shown here are promising for HS/BA and indicate that this novel method might be able to find a niche among the plethora of population-based optimization algorithms.

We note that CPU time is a bottleneck to the implementation of many population-based optimization algorithms. If an algorithm cannot converge fast, it will be impractical, since it would take too long to find an optimal or suboptimal solution. HS/BA does not seem to require an unreasonable amount of computational effort; of the ten optimization algorithms compared in this paper, HS/BA was the third fastest. Nevertheless, finding mechanisms to accelerate HS/BA could be an important area for further research.

Table 10: Mean normalized optimization results in fourteen benchmark functions with different HMCR. The numbers shown are the best results found after 100 Monte Carlo simulations of HS/BA algorithm.

	HMCR										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
F01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	1.01	**1.00**	1.01
F02	747.56	7.87	5.25	3.43	5.09	7.10	3.07	1.87	1.60	**1.00**	1.01
F03	9.13	3.12*E*4	1.09	1.07	1.05	1.06	1.05	1.02	1.01	1.01	**1.00**
F04	2.10*E*6	1.00*E*4	3.95*E*4	9.48*E*3	5.97*E*3	2.95*E*3	1.51*E*3	1.50*E*3	46.94	**1.00**	2.37
F05	3.24*E*6	3.27*E*4	2.09*E*4	4.14*E*4	7.13*E*3	1.54*E*3	8.93*E*3	1.53*E*3	629.48	**1.00**	203.73
F06	**1.00**	5.94*E*4	422.04	632.15	6.00*E*4	1.92	1.91	1.91	1.91	1.91	1.91
F07	10.63	2.07*E*6	9.00	216.72	324.55	3.07*E*4	1.04	1.03	1.02	1.01	**1.00**
F08	59.50	9.88*E*3	3.04*E*4	141.80	217.01	324.68	3.07*E*4	1.07	1.07	1.03	**1.00**
F09	226.87	4.95*E*6	3.08	8.91	114.22	173.60	259.41	2.45*E*4	1.45	**1.00**	1.71
F10	257.37	2.83*E*4	9.35*E*5	1.37*E*4	97.09	66.56	99.25	149.40	1.39*E*4	**1.00**	2.73
F11	6.07	**1.00**	1.83*E*4	2.02	16.61	390.40	262.99	403.00	603.61	5.72*E*4	1.84
F12	3.00	2.79*E*4	3.60*E*4	**1.00**	2.82*E*4	270.95	194.14	130.77	200.40	300.16	2.84*E*4
F13	2.32	9.66	5.61*E*6	1.89*E*6	**1.00**	8.15	267.78	191.86	129.23	198.05	296.65
F14	184.66	3.84*E*5	1.17*E*4	1.85*E*3	**1.00**	5.71*E*3	25.11	55.39	39.60	26.57	41.49
	1	1	0	1	2	0	0	0	0	**6**	**3**

Table 11: Best normalized optimization results in fourteen benchmark functions with different PAR. The numbers shown are the best results found after 100 Monte Carlo simulations of HS/BA algorithm.

	PAR										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
F01	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F02	3.91*E*3	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F03	**1.00**	4.25	1.51	2.20	2.20	2.12	2.20	1.99	2.20	1.28	1.80
F04	**1.00**	2.55	65.54	2.55	2.55	2.55	2.55	2.55	2.55	2.55	2.55
F05	2.52	**1.00**	1.71	1.69*E*3	1.71	1.71	1.71	1.71	1.71	1.71	1.71
F06	**1.00**	59.87	34.18	22.85	46.52	23.44	43.54	44.84	36.88	18.99	8.12
F07	8.07	**1.00**	1.48	2.55	2.55	13.06	2.55	2.55	2.55	2.55	2.55
F08	5.43	**1.00**	1.92	**1.00**	**1.00**	**1.00**	11.45	**1.00**	**1.00**	**1.00**	**1.00**
F09	76.72	2.52	1.17	**1.00**	1.71	1.71	1.71	155.56	1.71	1.71	1.71
F10	929.10	1.48	**1.00**	4.91	2.55	2.55	2.55	2.22	2.35*E*3	2.55	2.55
F11	3.18*E*3	**1.00**	5.82*E*3	3.99*E*3	3.39*E*3	5.82*E*3	4.91*E*3	5.82*E*3	5.82*E*3	9.58*E*3	5.82*E*3
F12	305.06	**1.00**	537.28	97.34	187.58	97.34	97.34	97.34	97.34	97.34	398.27
F13	1.92	4.22	5.87	10.07	12.98	4.66	1.48	4.01	3.17	**1.00**	4.09
F14	88.12	**1.00**	1.48	2.55	2.55	4.91	2.55	2.55	2.55	2.55	2.55
	4	**8**	3	4	3	3	2	3	3	4	3

5. Conclusion and Future Work

This paper proposed a hybrid metaheuristic HS/BA method for optimization problem. We improved the BA by combining original harmony search (HS) algorithm and evaluating the HS/BA on multimodal numerical optimization problems. A novel type of BA model has been presented, and an improvement is applied to mutate between bats using harmony search algorithm during the process of bats updating. Using the original configuration of the bat algorithm, we generate the new harmonies based on the newly generated bat each iteration after bat's position has been updated. The new harmony vector substitutes the newly generated bat only if it has better fitness. This selection scheme is rather greedy, which often overtakes original HS and BA. The HS/BA attempts to take merits of the BA and the HS in order to avoid all bats getting trapped in inferior local optimal regions. The HS/BA enables the bats to have more diverse exemplars to learn from, as the bats are updated each iteration and also form new harmonies to search in a larger search space. This new method can speed up the global convergence rate without losing the strong robustness of the basic BA. From the analysis of the experimental results, we observe that the proposed HS/BA makes good use of the information in past solutions more effectively to generate better quality solutions frequently, when compared to the other population-based

Table 12: Mean normalized optimization results in fourteen benchmark functions with different PAR. The numbers shown are the best results found after 100 Monte Carlo simulations of HS/BA algorithm.

	PAR										
	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
F01	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
F02	4.07	**1.00**	**1.00**	4.52	2.77	2.57	3.41	5.36	3.69	4.89	1.03
F03	**1.00**	1.27*E*4	1.34	1.35	1.35	1.35	1.34	1.35	1.34	1.34	1.34
F04	1.15	81.27	9.39*E*3	**1.00**	1.01	1.01	129	**1.00**	**1.00**	1.01	1.01
F05	345.73	50.61	86.19	9.07*E*3	25.01	1.08	1.01	1.91	3.33	**1.00**	1.18
F06	**1.00**	5.42*E*6	4.27*E*4	4.74*E*4	5.48*E*6	577.88	577.88	577.88	577.88	577.87	577.87
F07	4.86	1.53	**1.00**	95.43	105.96	1.22*E*4	1.34	1.36	1.32	1.33	1.35
F08	12.69	76.00	8.76*E*3	17.03	69.13	76.72	8.85*E*3	1.10	1.05	1.04	**1.00**
F09	63.45	230.47	1.82	1.67	12.78	48.60	53.49	6.09*E*3	1.11	1.30	**1.00**
F10	133.55	12.51	1.26	1.97*E*3	11.48	4.64	16.45	18.93	1.99*E*3	**1.00**	1.88
F11	63.02	**1.00**	6.39*E*3	79.44	59.24	3.96*E*3	1.42*E*3	5.81*E*3	6.45*E*3	7.45*E*5	79.81
F12	3.90	8.81*E*3	8.90*E*3	**1.00**	8.90*E*3	44.40	47.78	17.25	70.11	77.84	8.99*E*3
F13	**1.00**	21.66	2.07*E*3	6.69	5.80	4.30	271.67	282.46	105.39	429.26	476.62
F14	46.14	**1.00**	36.10	55.93	1.22	6.40*E*3	42.15	32.89	34.81	12.72	51.29
	4	**4**	3	3	1	1	1	2	2	3	3

optimization algorithms such as ACO, BA, BBO, DE, ES, GA, HS, PSO, and SGA. Based on the results of the ten approaches on the test problems, we can conclude that the HS/BA significantly improves the performances of the HS and the BA on most multimodal and unimodal problems.

In this work, 14 benchmark functions are used to evaluate the performance of our approach; we will test our approach on more problems, such as the high-dimensional ($D \geq 20$) CEC 2010 test suit [45] and the real-world problems. Moreover, we will compare HS/BA with other metaheuristic method, such as DLHS [46, 47], SGHS [48], adaptive DE [49], CLPSO [50], and DMS-PSO [51]. In addition, we only consider the unconstrained function optimization in this work. Our future work consists on adding the diversity rules into HS/BA for constrained optimization problems, such as constrained real-parameter optimization CEC 2010 test suit [52].

In the field of optimization, there are many issues worthy of further study, and efficient optimization method should be developed depending on the analysis of specific engineering problem. Our future work will focus on the two issues: on the one hand, we would apply our proposed approach HS/BA to solve practical engineering optimization problems, and, obviously, HS/BA can become a fascinating method for real-world engineering optimization problems; on the other hand, we would develop new meta-hybrid approach to solve optimization problem.

Acknowledgments

This work was supported by State Key Laboratory of Laser Interaction with Material Research Fund under Grant no. SKLLIM0902-01 and Key Research Technology of Electric-discharge Non-chain Pulsed DF Laser under Grant no. LXJJ-11-Q80.

References

[1] G. Wang, L. Guo, A. H. Gandomi et al., "Lévy-flight krill herd algorithm," *Mathematical Problems in Engineering*, vol. 2013, Article ID 682073, 14 pages, 2013.

[2] A. H. Gandomi, X. S. Yang, S. Talatahari, and A. H. Alavi, *Metaheuristic Applications in Structures and Infrastructures*, Elsevier, Waltham, Mass, USA, 2013.

[3] D. H. Ackley, "An empirical study of bit vector function optimization," in *Genetic Algorithms and Simulated Annealing*, L. Davis, Ed., pp. 170–204, Pitman, London, UK, 1987.

[4] R. Fletcher and M. J. D. Powell, "A rapidly convergent descent method for minimization," *The Computer Journal*, vol. 6, no. 2, pp. 163–168, 1963.

[5] A. O. Griewank, "Generalized descent for global optimization," *Journal of Optimization Theory and Applications*, vol. 34, no. 1, pp. 11–39, 1981.

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

[7] K. A. De Jong, *Analysis of the Behavior of a Class of Genetic Adaptive Systems*, University of Michigan, 1975.

[8] L. A. Rastrigin, *Extremal Control Systems*, Nauka, Moscow, Russian, 1974, Theoretical Foundations of Engineering Cybernetics Series.

[9] S.-K. S. Fan and J.-M. Chang, "Dynamic multi-swarm particle swarm optimizer using parallel PC cluster systems for global optimization of large-scale multimodal functions," *Engineering Optimization*, vol. 42, no. 5, pp. 431–451, 2010.

[10] W. Gong, Z. Cai, and C. X. Ling, "DE/BBO: a hybrid differential evolution with biogeography-based optimization for global numerical optimization," *Soft Computing*, vol. 15, no. 4, pp. 645–665, 2010.

[11] G. Wang, L. Guo, H. Duan, L. Liu, and H. Wang, "A modified firefly algorithm for UCAV path planning," *International Journal of Hybrid Information Technology*, vol. 5, no. 3, pp. 123–144, 2012.

[12] G. Wang, L. Guo, H. Duan, L. Liu, and H. Wang, "A hybrid meta-heuristic DE/CS algorithm for UCAV path planning," *Journal of Information and Computational Science*, vol. 9, no. 16, pp. 1–8, 2012.

[13] G. Wang, L. Guo, H. Duan, L. Liu, H. Wang, and M. Shao, "Path planning for uninhabited combat aerial vehicle using hybrid meta-heuristic DE/BBO algorithm," *Advanced Science, Engineering and Medicine*, vol. 4, no. 6, pp. 550–564, 2012.

[14] G. Wang, L. Guo, H. Duan, H. Wang, L. Liu, and M. Shao, "A hybrid meta-heuristic DE/CS algorithm for UCAV three-dimension path planning," *The Scientific World Journal*, vol. 2012, Article ID 583973, 11 pages, 2012.

[15] H. Duan, W. Zhao, G. Wang, and X. Feng, "Test-sheet composition using analytic hierarchy process and hybrid metaheuristic algorithm TS/BBO," *Mathematical Problems in Engineering*, vol. 2012, Article ID 712752, 22 pages, 2012.

[16] G. Wang, L. Guo, H. Duan, L. Liu, and H. Wang, "Dynamic deployment of wireless sensor networks by biogeography based optimization algorithm," *Journal of Sensor and Actuator Networks*, vol. 1, no. 2, pp. 86–96, 2012.

[17] X. S. Yang, A. H. Gandomi, S. Talatahari, and A. H. Alavi, *Metaheuristics in Water, Geotechnical and Transport Engineering*, Elsevier, Waltham, Mass, USA, 2013.

[18] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, New York, NY, USA, 1998.

[19] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[20] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.

[21] 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.

[22] A. H. Gandomi, G. J. Yun, X.-S. Yang, and S. Talatahari, "Chaos-enhanced accelerated particle swarm optimization," *Communications in Nonlinear Science and Numerical Simulation*, vol. 18, no. 2, pp. 327–340, 2013.

[23] S. Talatahari, X. S. Yang, R. Sheikholeslami, and A. H. Gandomi, "Optimal parameter estimation for muskingum model using hybrid CSS and PSO method," *Journal of Computational and Theoretical Nanoscience*. In press.

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

[25] G. Wang, L. Guo, H. Duan, L. Liu, H. Wang, and M. Shao, "Hybridizing harmony search with biogeography based optimization for global numerical optimization," *Journal of Computational and Theoretical Nanoscience*. In press.

[26] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[27] G. Wang, L. Guo, H. Duan, H. Wang, L. Liu, and J. Li, "Incorporating mutation scheme into krill herd algorithm for global numerical optimization," *Neural Computing and Applications*, 2012.

[28] X. S. Yang, *Nature-Inspired Metaheuristic Algorithms*, Luniver Press, Frome, UK, 2nd edition, 2010.

[29] A. H. Gandomi, X.-S. Yang, A. H. Alavi, and S. Talatahari, "Bat algorithm for constrained optimization tasks," *Neural Computing & Applications*. In press.

[30] X. S. Yang and A. H. Gandomi, "Bat algorithm: a novel approach for global engineering optimization," *Engineering Computations*, vol. 29, no. 5, pp. 464–483, 2012.

[31] G. Wang, L. Guo, H. Duan, L. Liu, and H. Wang, "A Bat algorithm with mutation for UCAV path planning," *The Scientific World Journal*, vol. 2012, Article ID 418946, 15 pages, 2012.

[32] R. Parpinelli and H. Lopes, "New inspirations in swarm intelligence: a survey," *International Journal of Bio-Inspired Computation*, vol. 3, no. 1, pp. 1–16, 2011.

[33] P. W. Tsai, J. S. Pan, B. Y. Liao, M. J. Tsai, and V. Istanda, "Bat algorithm inspired algorithm for solving numerical optimization problems," *Applied Mechanics and Materials*, vol. 148, pp. 134–137, 2012.

[34] Wikipedia, "Mathematical optimization," http://en.wikipedia.org/wiki/Numerical_optimization.

[35] W. Y. Zhang, S. Xu, and S. J. Li, "Necessary conditions for weak sharp minima in cone-constrained optimization problems," *Abstract and Applied Analysis*, vol. 2011, Article ID 909520, 11 pages, 2012.

[36] Wikipedia, "Global optimization," http://en.wikipedia.org/wiki/Global_optimization.

[37] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[38] M. Dorigo and T. Stutzle, *Ant Colony Optimization*, The MIT Press, Cambridge, Mass, USA, 2004.

[39] M. Dorigo, *Optimization, Learning and Natural Algorithms*, Politecnico di Milano, Italy, 1992.

[40] H. G. Beyer, *The Theory of Evolution Strategies*, Springer, New York, NY, USA, 2001.

[41] W. Khatib and P. Fleming, "The stud GA: a mini revolution?" in *Parallel Problem Solving From Nature*, pp. 683–691, 1998.

[42] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.

[43] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *The Annals of Mathematical Statistics*, vol. 11, no. 1, pp. 86–92, 1940.

[44] R. Brits, A. P. Engelbrecht, and F. van den Bergh, "Locating multiple optima using particle swarm optimization," *Applied Mathematics and Computation*, vol. 189, no. 2, pp. 1859–1883, 2007.

[45] K. Tang, X. Li, P. N. Suganthan, Z. Yang, and T. Weise, "Benchmark functions for the CEC'2010 special session and competition on large scale global optimization," Nature Inspired Computation and Applications Laboratory, USTC, Hefei, China, 2010.

[46] Q.-K. Pan, P. N. Suganthan, J. J. Liang, and M. F. Tasgetiren, "A local-best harmony search algorithm with dynamic subpopulations," *Engineering Optimization*, vol. 42, no. 2, pp. 101–117, 2010.

[47] Q.-K. Pan, P. N. Suganthan, J. J. Liang, and M. F. Tasgetiren, "A local-best harmony search algorithm with dynamic sub-harmony memories for lot-streaming flow shop scheduling problem," *Expert Systems with Applications*, vol. 38, no. 4, pp. 3252–3259, 2011.

[48] Q.-K. Pan, P. N. Suganthan, M. F. Tasgetiren, and J. J. Liang, "A self-adaptive global best harmony search algorithm for continuous optimization problems," *Applied Mathematics and Computation*, vol. 216, no. 3, pp. 830–848, 2010.

[49] S. M. Islam, S. Das, S. Ghosh, S. Roy, and P. N. Suganthan, "An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization," *IEEE Transactions on Systems, Man, and Cybernetics B*, vol. 42, no. 2, pp. 482–500, 2012.

[50] 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.

[51] S. Z. Zhao, P. N. Suganthan, Q.-K. Pan, and M. Fatih Tasgetiren, "Dynamic multi-swarm particle swarm optimizer with harmony search," *Expert Systems with Applications*, vol. 38, no. 4, pp. 3735–3742, 2011.

[52] R. Mallipeddi and P. Suganthan, *Problem Definitions and Evaluation Criteria for the CEC 2010 Competition on Constrained Real-Parameter Optimization*, Nanyang Technological University, Singapore, 2010.