Lévy-Flight Krill Herd Algorithm

Article *in* Mathematical Problems in Engineering · February 2013

DOI: 10.1155/2013/682073

CITATIONS
58

READS
179

7 authors, including:

Gai-Ge Wang
Ocean University of China
119 PUBLICATIONS **3,155** CITATIONS
SEE PROFILE

Amir H Gandomi
Stevens Institute of Technology
224 PUBLICATIONS **8,833** CITATIONS
SEE PROFILE

Amir H. Alavi
University of Missouri
165 PUBLICATIONS **6,109** CITATIONS
SEE PROFILE

Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2013, Article ID 682073, 14 pages
http://dx.doi.org/10.1155/2013/682073

Research Article

Lévy-Flight Krill Herd Algorithm

Gaige Wang,[1,2] Lihong Guo,[1] Amir Hossein Gandomi,[3] Lihua Cao,[1] Amir Hossein Alavi,[4] Hong Duan,[5] and Jiang Li[1]

[1] *Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun, Jilin 130033, China*
[2] *University of Chinese Academy of Sciences, Beijing 100039, China*
[3] *Department of Civil Engineering, University of Akron, Akron, OH 443253905, USA*
[4] *Department of Civil and Environmental Engineering, Engineering Building, Michigan State University, East Lansing, MI 48824, USA*
[5] *School of Computer Science and Information Technology, Northeast Normal University, Changchun 130117, China*

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

Received 3 November 2012; Accepted 20 December 2012

Academic Editor: Siamak Talatahari

To improve the performance of the krill herd (KH) algorithm, in this paper, a Lévy-flight krill herd (LKH) algorithm is proposed for solving optimization tasks within limited computing time. The improvement includes the addition of a new local Lévy-flight (LLF) operator during the process when updating krill in order to improve its efficiency and reliability coping with global numerical optimization problems. The LLF operator encourages the exploitation and makes the krill individuals search the space carefully at the end of the search. The elitism scheme is also applied to keep the best krill during the process when updating the krill. Fourteen standard benchmark functions are used to verify the effects of these improvements and it is illustrated that, in most cases, the performance of this novel metaheuristic LKH method is superior to, or at least highly competitive with, the standard KH and other population-based optimization methods. Especially, this new method can accelerate the global convergence speed to the true global optimum while preserving the main feature of the basic KH.

1. Introduction

In current competitory world, human beings make an attempt at extracting the maximum output or profit from a restricted amount of usable resources. In the case of engineering optimization, such as design optimization of tall steel buildings [1], optimum design of gravity retaining walls [2], water, geotechnical and transport engineering [3], and structural optimization and design [4, 5], engineers would attempt to design structures that satisfy all design requirements with the minimum possible cost. Most real-world engineering optimization problems could be converted into general global optimization problems. Therefore, the study of global optimization is of vital importance for the engineering optimization. In this issue, many biological intelligent techniques [6] as optimization tools have been developed and applied to solve engineering optimization problems for engineers. A general classification way for these techniques is considering the nature of the techniques, and optimization techniques can be classified as two main groups: classical methods and modern intelligent algorithms. Classical methods such as hill climbing have a rigorous move and will generate the same set of solutions if the iterations start with the same initial starting point. On the other hand, modern intelligent algorithms often generate different solutions even with the same initial value. However, in general, the final solutions, though slightly different, will converge to the same optimal values within a given accuracy. The emergence of metaheuristic optimization algorithm as a blessing from the artificial intelligence and mathematical theorem has opened up a new facet to carry out the optimization of a function. Recently, nature-inspired metaheuristic algorithms perform powerfully and efficiently in solving modern nonlinear numerical global optimization problems. To some extent, all metaheuristic algorithms make an attempt at relieving the conflict between diversification/exploration/randomization (global search) and intensification/exploitation (local search) [7, 8].

Inspired by nature, these strong metaheuristic algorithms have been proposed to solve NP-hard tasks, such as UCAV path planning [9, 10], test-sheet composition [11], and parameter estimation [12]. These kinds of metaheuristic methods perform on a population of solutions and always find optimal or suboptimal solutions. During the 1960s and 1970s, computer scientists studied the possibility of formulating evolution as an optimization method and eventually this generated a subset of gradient free methods, namely, genetic algorithms (GAs) [13, 14]. In the last two decades, a huge number of techniques were developed on function optimization, such as bat algorithm (BA) [15, 16], differential evolution (DE) [17, 18], genetic programming (GP) [19], harmony search (HS) [20, 21], particle swarm optimization (PSO) [22–24], cuckoo search (CS) [25, 26], and, more recently, the krill herd (KH) algorithm [27] that is based on imitating the krill herding behavior in nature.

Firstly proposed by Gandomi and Alavi in 2012, inspired by the herding behavior of krill individuals, KH algorithm is a novel swarm intelligence method for optimizing possibly nondifferentiable and nonlinear complex functions in continuous space [27]. In KH, the time-dependent position of the krill individuals involves three main components: (i) movement led by other individuals, (ii) foraging motion, and (iii) random physical diffusion. One of the notable advantages of the KH algorithm is that derivative information is unnecessary, because it uses a random search instead of a gradient search use in classical methods. Moreover, comparing with other population-based metaheuristic methods, this new method needs few control variables, in principle only a separate parameter Δt (time interval) to tune, which makes KH easy to implement, more robust and fits for parallel computation.

KH is an effective and powerful algorithm in exploration, but at times it may trap into some local optima so that it cannot implement global search well. For KH, the search depends completely on random search, so there is no guarantee for a fast convergence. In order to improve KH in optimization problems, a method has been proposed [28], which introduces a more focused mutation strategy into KH to add the diversity of population.

On the other hand, many researchers have centralized on theories and applications of statistical techniques, especially of Lévy distribution. And recently huge advances are acquired in many fields. One of these fields is the applications of Lévy flight in optimization methods. Previously, Lévy flights have been used together with some metaheuristic optimization methods such as firefly algorithm [29], cuckoo search [30], krill herd algorithm [31], and particle swarm optimization [32].

Firstly presented here, an effective Lévy-flight KH (LKH) method is originally proposed in this paper, in order to accelerate convergence speed, thus making the approach more feasible for a wider range of real-world engineering applications while keeping the desirable characteristics of the original KH. In LKH, first of all, a standard KH algorithm is implemented to shrink the search apace and select a good candidate solution set. And then, for more precise modeling of the krill behavior, a local Lévy-flight (LLF) operator is added to the algorithm. This operator is applied to exploit the limited promising area intensively to get better solutions so as to improve its efficiency and reliability for solving global numerical optimization problems. The proposed method is evaluated on fourteen standard benchmark functions that have ever been applied to verify optimization methods in continuous optimization problems. Experimental results show that the LKH performs more efficiently and effectively than basic KH, ABC, ACO, BA, CS, DE, ES, GA, HS, PBIL, and PSO.

The structure of this paper is organized as follows. Section 2 gives a description of basic KH algorithm and Lévy flight in brief. Our proposed LKH method is described in detail in Section 3. Subsequently, our method is evaluated through fourteen benchmark functions in Section 4. In addition, the LKH is also compared with ABC, ACO, BA, CS, DE, ES, GA, HS, KH, PBIL, and PSO in that section. Finally, Section 5 involves the conclusion and proposals for future work.

2. Preliminary

At first, in this section, a background on the krill herd algorithm and Lévy flight will be provided in brief.

2.1. Krill Herd Algorithm. Krill herd (KH) [27] is a new metaheuristic optimization method [4] for solving optimization tasks, which is based on the simulation of the herding of the krill swarms in response to particular biological and environmental processes. The time-dependent position of an individual krill in 2D space is decided by three main actions presented as follows:

(i) movement affected by other krill individuals,

(ii) foraging action,

(iii) random diffusion.

KH algorithm adopted the following Lagrangian model in a d-dimensional decision space as in the following (1):

$$\frac{dX_i}{dt} = N_i + F_i + D_i, \tag{1}$$

where N_i, F_i, and D_i are the motion led by other krill individuals, the foraging motion, and the physical diffusion of the ith krill individual, respectively.

In movement affected by other krill individuals, the direction of motion, α_i, is approximately computed by the target effect (target swarm density), local effect (a local swarm density), and a repulsive effect (repulsive swarm density). For a krill individual, this movement can be defined as

$$N_i^{\text{new}} = N^{\max}\alpha_i + \omega_n N_i^{\text{old}}, \tag{2}$$

and $N^{\max}$ is the maximum induced speed, ω_n is the inertia weight of the motion induced in $[0, 1]$, and N_i^{old} is the last motion induced.

The foraging motion is estimated by the two main components. One is the food location and the other one is

Table 1: Benchmark functions.

Number	Name	Definition
F01	Ackley	$f(\vec{x}) = 20 + e - 20 \cdot e^{-0.2 \cdot \sqrt{(1/n)\sum_{i=1}^{n} x_i^2}} - e^{(1/n)\sum_{i=1}^{n} \cos(2\pi x_i)}$
F02	Fletcher-Powell	$f(\vec{x}) = \sum_{i=1}^{n} (A_i - B_i)^2, \ A_i = \sum_{j=1}^{n} \left(a_{ij} \sin \alpha_j + b_{ij} \cos \alpha_j\right)$ $B_i = \sum_{j=1}^{n} \left(a_{ij} \sin x_j + b_{ij} \cos x_j\right)$
F03	Griewank	$f(\vec{x}) = \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$
F04	Penalty #1	$f(\vec{x}) = \frac{\pi}{30} \left\{10\sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 \cdot [1 + 10\sin^2(\pi y_{i+1})] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4), \ y_i = 1 + 0.25(x_i + 1)$
F05	Penalty #2	$f(\vec{x}) = 0.1 \left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n-1} (x_i - 1)^2 \cdot [1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$
F06	Quartic *with noise*	$f(\vec{x}) = \sum_{i=1}^{n} (i \cdot x_i^4 + U(0,1))$
F07	Rastrigin	$f(\vec{x}) = 10 \cdot n + \sum_{i=1}^{n} (x_i^2 - 10 \cdot \cos(2\pi x_i))$
F08	Rosenbrock	$f(\vec{x}) = \sum_{i=1}^{n-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$
F09	Schwefel 2.26	$f(\vec{x}) = 418.9829 \times D - \sum_{i=1}^{D} x_i \sin\left(\lvert x_i \rvert^{1/2}\right)$
F10	Schwefel 1.2	$f(\vec{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$
F11	Schwefel 2.22	$f(\vec{x}) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$
F12	Schwefel 2.21	$f(\vec{x}) = \max_i \{\lvert x_i \rvert, 1 \le i \le n\}$
F13	Sphere	$f(\vec{x}) = \sum_{i=1}^{n} x_i^2$
F14	Step	$f(\vec{x}) = 6 \cdot n + \sum_{i=1}^{n} \lfloor x_i \rfloor$

* In benchmark function F02, the matrix elements $\mathbf{a}_{n\times n}$, $\mathbf{b}_{n\times n} \in (-100, 100)$, and $\boldsymbol{\alpha}_{n\times 1} \in (-\pi, \pi)$ are draw from uniform distribution.

* In benchmark functions F04 and F05, the definition of the function $u(x_i, a, k, m)$ is as follows: $u(x_i, a, k, m) = \{k(x_i - a)^m, \ x_i > a; \ 0, \ -a \le x_i \le a; \ k(-x_i - a)^m, \ x_i < -a\}$.

the prior knowledge about the food location. For the ith krill individual, this motion can be approximately formulated as follows:

$$F_i = V_f \beta_i + \omega_f F_i^{\text{old}}, \tag{3}$$

where

$$\beta_i = \beta_i^{\text{food}} + \beta_i^{\text{best}}, \tag{4}$$

and V_f is the foraging speed, ω_f is the inertia weight of the foraging motion between 0 and 1, F_i^{old} is the last foraging motion.

The random diffusion of the krill individuals can be considered to be a random process in essence. This motion can be described in terms of a maximum diffusion speed and a random directional vector. It can be indicated as follows:

$$D_i = D^{\max}\delta, \tag{5}$$

where $D^{\max}$ is the maximum diffusion speed, and δ is the random directional vector and its arrays are random values in $[-1, 1]$.

Based on the three above-mentioned movements, using different parameters of the motion during the time, the position vector of a krill individual during the interval t to $t + \Delta t$ is expressed by the following equation:

$$X_i(t + \Delta t) = X_i(t) + \Delta t \frac{dX_i}{dt}. \tag{6}$$

It should be noted that Δt is one of the most important parameters and should be fine-tuned in terms of the specific real-world engineering optimization problem. This is because this parameter can be treated as a scale factor of the speed vector. More details about the three main motions and KH algorithm can be found in [27].

2.2. Lévy Flights. Usually, the hunt of food by animals takes place in the form of random or quasi-random. That is to say, all animals feed in a walk path from one location to another at random. However, the direction it selects relies only on a mathematical model [33]. One of the remarkable models is called Lévy flights.

Lévy flights are a class of random walk in which the steps are determined in terms of the step lengths, and the jumps are distributed according to a Lévy distribution. More recently, Lévy flights have subsequently been applied to improve and optimize searching. In the case of CS, the random walking steps of a cuckoo are determined by a Lévy flight [34]:

$$\begin{aligned} X_i^{t+1} &= X_i^t + \beta L(s, \lambda), \\ L(s, \lambda) &= \frac{\lambda \Gamma(\lambda) \sin(\pi\lambda/2)}{\pi} \frac{1}{s^{1+\lambda}}, \quad (s, s_0 > 0). \end{aligned} \tag{7}$$

Here, $\beta > 0$ is the step size scaling factor, which should be related to the scales of the problem of interest. The random walk via Lévy flight is more efficient in exploring the search space as its step length is much longer in the long run. Some of the new solutions should be generated by Lévy walk around the best solution obtained so far; this will speed up the local search.

3. Our Approach: LKH

In general, the standard KH algorithm is adept at exploring the search space and locating the promising region of global optimal value, but it is not relatively good at exploiting solution. In order to improve the exploitation of KH, a new distribution Lévy flight performing local search, called local Lévy-flight (LLF) operator, is introduced to form a novel Lévy-flight krill herd (LKH) algorithm. In LKH, to begin with, standard KH algorithm with high convergence speed is used to shrink the search region to a more promising area. And then, LLF operator with good exploitation ability is applied to exploit the limited area intensively to get better solutions. In this way, the strong exploration abilities of the original KH and the exploitation abilities of the LLF operator can be fully extracted. The difference between LKH and KH is that the LLF operator is used to perform local search and fine-tune the original KH generating a new solution for each krill instead of random walks originally used in KH. As a matter of fact, according to the figuration of LKH, the original KH in LKH focuses on the exploration/diversification at the beginning of the search to evade trapping into local optima in a multimodal landscape; while later LLF operator encourages the exploitation/intensification and makes the krill individuals search the space carefully at the end of the search. Therefore, our proposed LKH method can fully exploit the merits of different search techniques and overcome the lack of the exploitation of the KH and solve the conflict between exploration and exploitation effectively. The detailed explanation of our method is described as follows.

To start with, standard KH algorithm utilizes three main actions to search the promising area in the solution space and use these actions to guide the generation of the candidate solutions for the next generation. It has been demonstrated that [27] KH performs well in both convergence speed and final accuracy on unimodal problems and many simple multimodal problems. Therefore, in LKH, we employ the merit of the fast convergence of KH to implement global search. In addition, KH is able to shrink the search region towards the promising area within a few generations. However, sometimes KH's performance on complex multimodal problems is unsatisfying; accordingly, another search technique with good exploitation ability is crucial to exploit the limited area carefully to get optimal solutions.

To improve the exploitation ability of the KH algorithm, genetic reproduction mechanisms have been incorporated into the standard KH algorithm. Gandomi and Alavi have proved that the KH II (KH with crossover operator only) performs the best among serials of KH methods [27]. In our present work, we use a more focused local search technique, local Lévy-flight (LLF) operator, in the local search part of the LKH algorithm, which can increase diversity of the population in an attempt to avoid premature convergence and exploit a small region in the later run phase to refine the final solutions. The main step of LLF operator used in the LKH algorithm is presented in Algorithm 1.

Here, $t \in [0, t_{\max}]$ and $t_{\max}$ is the maximum of generations. d is the number of decision variables. NP is the size of the parent population. A is max Lévy-flight step size.

Begin
$a = A/(t\hat{}2)$; % Smaller step for local walk
NoSteps = $\lceil d * \text{exprnd}(2 * t_{\max}) \rceil$;
Determine the next step size dx by performing Lévy flight as shown in Section 2.2;
$r = \lceil \text{NP} * \text{rand} \rceil$
for $j = 1$ to d ***do***
if rand ≤ 0.5 % Random number to determine direction
$V_i(j) = a \times dx(j) + X_{\text{NP}-r+1}(j)$
else
$V_i(j) = a \times dx(j) - X_{\text{NP}-r+1}(j)$
end if
end for j
Evaluate the offspring V_i
if V_i is better than X_i ***then***
$X_i = V_i$
end if
End.

Algorithm 1: Local Lévy-flight (LLF) operator.

$X_i(j)$ is the jth variable of the solution X_i. V_i is the offspring. $\lceil d * \text{exprnd}(2 * t_{\max}) \rceil$ is a random integer number between 1 and $d * \text{exprnd}(2 * t_{\max})$ drawn from exponential distribution. $\text{exprnd}(2 * t_{\max})$ returns an array of random numbers chosen from the exponential distribution with mean parameter $2 * t_{\max}$. Similarly, $\lceil d * \text{rand} \rceil$ is a random integer number between 1 and d drawn from uniform distribution. And rand is a random real number in interval (0, 1) drawn from uniform distribution.

In addition, another important improvement is the addition of elitism strategy into the LKH. Clearly, KH has some fundamental elitism. However, it can be further improved. As with other population-based optimization algorithms, we combine some sort of elitism so as to store the optimal solutions in the population. Here, we use a more centralized elitism on the best solutions, which can stop the best solutions from being ruined by three motions and LLF operator in LKH. In the main cycle of the LKH, to start with, the *KEEP* best solutions are retained in a variable *KEEPKRILL*. Generally speaking, the *KEEP* worst solutions are substituted by the *KEEP* best solutions at the end of the every iteration. There is a guarantee that this elitism strategy can make the whole population not decline to the population with worse fitness than the former. Note that we use an elitism strategy to save the property of the krill that has the best fitness in the LKH process, so even if three motions and LLF operator corrupt their corresponding krill, we have retained it and can recuperate to its preceding good status if needed.

Based on the above analyses, the main steps of Lévy-flight krill herd method can be simply presented in Algorithm 2.

4. Simulation Experiments

In this section, the performance of our proposed method LKH is tested to global numerical optimization through a series of experiments implemented in benchmark functions.

To allow an unprejudiced comparison of CPU time, all the experiments were carried out on a PC with a Pentium IV processor running at 2.0 GHz, 512 MB of RAM, and a hard drive of 160 GB. Our execution was compiled using MATLAB R2012b (8.0) running under Windows XP3. No commercial KH or other optimization tools were used in our simulation experiments.

Well-defined problem sets benefit for testing the performance of optimization algorithms proposed in this paper. Based on numerical functions, benchmark functions can be considered as objective functions to fulfill such tests. In our present study, fourteen different benchmark functions are applied to test our proposed metaheuristic LKH method. The formulation of these benchmark functions are given in Table 1 and the properties of these benchmark functions are presented in Table 2. More details of all the benchmark functions can be found in [35, 36]. We must point out that, in [35], Yao et al. have used 23 benchmarks to test optimization algorithms. However, for the other low-dimensional benchmark functions (such as $d = 2$, 4, and 6), all the methods perform almost identically with each other [37], because these low-dimensional benchmarks are too simple to clarify the performance difference among different methods. Therefore, in our present work, only fourteen high-dimensional complex benchmarks are applied to verify our proposed LKH algorithm.

4.1. General Performance of LKH. In order to explore the merits of LKH, in this section, we compared its performance on global numeric optimization problems with eleven population-based optimization methods, which are ABC, ACO, BA, CS, DE, ES, GA, HS, KH, PBIL, and PSO. ABC (artificial bee colony) [38] is an intelligent optimization algorithm based on the smart behavior of honey bee swarm. ACO (ant colony optimization) [39] is a swarm intelligence algorithm for solving optimization problems which is based on the pheromone deposition of ants. BA (bat algorithm) [16] is a new powerful metaheuristic optimization method inspired by the echolocation behavior of bats with varying

Begin
Step 1: Initialization. Set the generation counter $t = 1$; initialize the population P of NP krill individuals randomly and each krill corresponds to a potential solution to the given problem; set the foraging speed V_f, the maximum diffusion speed $D^{\max}$, and the maximum induced speed $N^{\max}$; set max Lévy flight step size A and elitism parameter
KEEP: how many of the best krill to keep from one generation to the next.
Step 2: Fitness evaluation. Evaluate each krill individual according to its position.
Step 3: While the termination criteria is not satisfied or $t <$ MaxGeneration ***do***
Sort the population/krill from best to worst.
Store the *KEEP* best krill as *KEEPKRILL*.
for $i = 1 :$ NP (all krill) ***do***
Perform the following motion calculation.
Motion induced by the presence of other individuals
Foraging motion
Physical diffusion
Update the krill individual position in the search space by (6).
Fine-tune X_{t+1} by performing LLF operator as shown in Algorithm 1.
Evaluate each krill individual according to its new position X_{t+1}.
end for *i*
Replace the *KEEP* worst krill with the *KEEP* best krill stored in *KEEPKRILL*.
Sort the population/krill from best to worst and find the current best.
$t = t + 1$;
Step 4: end while
Step 5: Post-processing the results and visualization.
End.

Algorithm 2: Lévy-flight krill herd algorithm.

pulse rates of emission and loudness. CS (cuckoo search) [40] is a metaheuristic optimization algorithm inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds. DE (differential evolution) [17] is a simple but excellent optimization method that uses the difference between two solutions to probabilistically adapt a third solution. An ES (evolutionary strategy) [41] 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) [13] is a search heuristic that mimics the process of natural evolution. HS (harmony search) [20] is a new metaheuristic approach inspired by behavior of musician' improvisation process. PBIL (probability-based incremental learning) [42] is a type of genetic algorithm where the genotype of an entire population (probability vector) is evolved rather than individual members. PSO (particle swarm optimization) [22] is also a swarm intelligence algorithm which is based on the swarm behavior of fish and bird schooling in nature. In addition, it should be noted that, in [27], Gandomi and Alavi have proved that, comparing all the algorithms, the KH II (KH with crossover operator) performed the best which confirms the robustness of the KH algorithm. Therefore, in our work, we use KH II as a standard KH algorithm.

In our experiments, we will use the same parameters for KH and LKH that are the foraging speed $V_f = 0.02$, the maximum diffusion speed $D^{\max} = 0.005$, the maximum induced speed $N^{\max} = 0.01$, and max Lévy-flight step size $A = 1.0$ (only for LKH). For ACO, DE, ES, GA, PBIL, and PSO, we set the same parameters as [36, 43]. For ABC, the number of colony size (employed bees and onlooker bees) NP = 50, the number of food sources $FoodNumber$ = NP/2, and maximum search times $limit = 100$ (a food source which could not be improved through "limit" trials is abandoned by its employed bee). For BA, we set loudness $L = 0.95$, pulse rate $r = 0.5$, and scaling factor $\varepsilon = 0.1$; for CS, a discovery rate $p_a = 0.25$. For HS, we set harmony memory accepting rate = 0.75 and pitch adjusting rate = 0.7.

We set population size NP = 50 and maximum generation $Maxgen = 50$ for each method. We ran 100 Monte Carlo simulations of each method 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 method, averaged over 100 Monte Carlo runs. Table 4 shows the absolute best minima found by each method over 100 Monte Carlo runs. That is to say, Table 3 shows the average performance of each method, while Table 4 shows the best performance of each method. The best value achieved for each test problem is marked in bold. Note that the normalizations in the tables are based on different scales, so values cannot be compared between the two tables. Each of the functions in this study has 20 independent variables (i.e., $d = 20$).

From Table 3, we see that, on average, LKH is the most effective at finding objective function minimum on twelve of the fourteen benchmarks (F01–F08, F10, and F12–F14). ABC and GA are the second most effective, performing the best on the benchmarks F11 and F09 when multiple runs are made,

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

Number	Function	lb	ub	opt	Continuity	Modality
F01	Ackley	−32.768	32.768	0	Continuous	Multimodal
F02	Fletcher-Powell	$-\pi$	π	0	Continuous	Multimodal
F03	Griewangk	−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.

	ABC	ACO	BA	CS	DE	ES	GA	HS	KH	LKH	PBIL	PSO
F01	5.26	6.20	7.81	6.84	4.85	7.54	6.70	7.74	1.85	**1.00**	7.84	6.53
F02	2.63	10.01	14.42	6.82	3.80	9.77	4.25	9.42	3.67	**1.00**	9.02	8.27
F03	31.49	10.22	182.64	61.09	17.19	78.85	32.57	155.54	4.59	**1.00**	176.91	64.57
F04	8.5*E*5	3.3*E*7	4.5*E*7	2.6*E*6	1.1*E*5	1.9*E*7	2.7*E*5	2.9*E*7	9.2*E*3	**1.00**	4.3*E*7	2.5*E*6
F05	1.2*E*6	1.7*E*7	2.8*E*7	3.0*E*6	2.9*E*5	1.3*E*7	5.1*E*5	2.2*E*7	3.5*E*4	**1.00**	2.6*E*7	3.1*E*6
F06	3.4*E*5	3.3*E*5	5.5*E*6	7.6*E*5	1.2*E*5	4.1*E*6	3.3*E*5	3.8*E*6	3.8*E*4	**1.00**	4.6*E*6	8.8*E*5
F07	1.22	2.30	3.42	2.64	1.99	3.16	2.04	2.89	1.25	**1.00**	3.18	2.34
F08	15.83	85.99	89.29	25.57	13.40	110.42	23.00	75.23	5.34	**1.00**	90.29	26.64
F09	1.77	1.11	3.93	2.86	2.24	2.73	**1.00**	3.33	2.10	1.93	3.47	3.35
F10	51.56	43.79	123.64	29.15	68.30	72.56	48.30	66.46	33.17	**1.00**	75.49	51.20
F11	**1.00**	2.84	4.57	2.79	1.23	4.32	2.19	3.52	1.50	4.22	3.51	2.58
F12	14.65	9.13	15.73	11.05	11.98	14.52	12.31	14.91	2.46	**1.00**	15.61	12.52
F13	5.6*E*3	1.5*E*4	3.2*E*4	1.2*E*4	3.1*E*3	3.3*E*4	1.1*E*4	3.0*E*4	851.62	**1.00**	3.2*E*4	1.2*E*4
F14	205.78	95.69	1.1*E*3	427.88	103.81	700.63	227.85	1.0*E*3	29.20	**1.00**	1.2*E*3	411.03
Time	2.40	3.22	1.08	2.02	1.95	2.03	2.38	2.77	4.66	4.30	**1.00**	2.37
Total	1	0	0	0	0	0	1	0	0	12	0	0

*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.

respectively. Table 4 shows that LKH performs the best on twelve of the fourteen benchmarks which are F01–F04, F06–F08, and F10–F14. ACO and GA are the second most effective, performing the best on the benchmarks F05 and F09 when multiple runs are made, respectively.

Moreover, the computational times of the twelve optimization methods were alike. We collected the average computational time of the optimization methods as applied to the 14 benchmarks considered in this section. The results are given in Table 3. From Table 3, PBIL was the quickest optimization method, and LKH was the eleventh fastest of the twelve algorithms. This is because that the evaluation of step size by Lévy flight is too time consuming. However, we must point out that in the vast majority of real-world engineering applications, it is the fitness function evaluation that is by far the most expensive part of a population-based optimization algorithm.

In addition, in order to further prove the superiority of the proposed LKH method, convergence plots of ABC, ACO, BA, CS, DE, ES, GA, HS, KH, LKH, PBIL, and PSO are illustrated in Figures 1–14 which mean the process of optimization. The values shown in Figures 1–14 are the average objective function optimum obtained from 100 Monte Carlo simulations, which are the true objective function value, not normalized. Most importantly, note that the best global solutions of the benchmarks (F04, F05, F11, and F14) are illustrated in the form of the semilogarithmic convergence plots. KH is short for KH II in the legends of the figures.

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

	ABC	ACO	BA	CS	DE	ES	GA	HS	KH	LKH	PBIL	PSO
F01	39.13	51.65	64.63	54.80	39.31	66.38	35.21	74.39	6.59	**1.00**	73.39	56.78
F02	13.81	52.37	68.76	37.85	18.34	52.09	5.64	41.98	16.07	**1.00**	36.30	40.23
F03	13.52	6.71	91.05	37.41	14.32	53.67	5.95	121.61	1.78	**1.00**	84.88	46.62
F04	309.00	1.26	3.0*E*8	9.9*E*6	7.7*E*4	4.6*E*8	282.60	5.5*E*8	645.76	**1.00**	1.2*E*9	1.2*E*7
F05	2.2*E*5	**1.00**	1.7*E*8	1.8*E*7	1.9*E*6	2.0*E*8	2.1*E*4	2.0*E*8	7.3*E*4	2.70	3.1*E*8	2.8*E*7
F06	1.2*E*6	4.9*E*6	6.1*E*7	7.6*E*6	3.9*E*6	1.8*E*8	3.1*E*6	1.2*E*8	1.3*E*6	**1.00**	1.8*E*8	2.4*E*7
F07	2.00	4.53	6.30	4.95	3.68	6.28	3.21	4.75	1.98	**1.00**	5.90	4.90
F08	10.40	70.89	54.16	15.03	12.74	109.61	17.46	58.83	5.56	**1.00**	70.98	19.22
F09	4.60	2.18	9.10	6.98	6.28	7.80	**1.00**	10.10	4.83	4.53	10.01	8.14
F10	560.93	236.20	703.61	265.75	929.67	893.67	280.77	553.77	265.62	**1.00**	1.1*E*3	495.02
F11	20.01	32.33	75.00	44.09	29.25	83.22	31.79	88.27	23.70	**1.00**	86.51	54.41
F12	39.39	18.51	45.36	28.77	34.87	48.27	23.82	48.37	4.08	**1.00**	45.44	36.44
F13	1.0*E*4	3.8*E*4	6.7*E*4	2.1*E*4	7.9*E*3	9.0*E*4	1.0*E*4	1.0*E*5	1.5*E*3	**1.00**	7.8*E*4	3.5*E*4
F14	595.00	470.00	6.5*E*3	1.7*E*3	633.50	4.5*E*3	592.00	5.8*E*3	113.00	**1.00**	7.0*E*3	1.8*E*3
Total	0	1	0	0	0	0	1	0	0	12	0	0

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

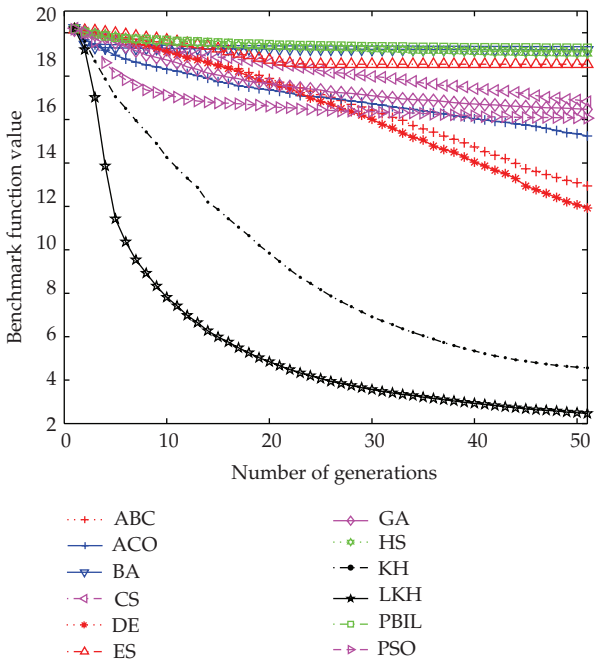

Figure 1: Comparison of the performance of the different methods for the F01 Ackley function.

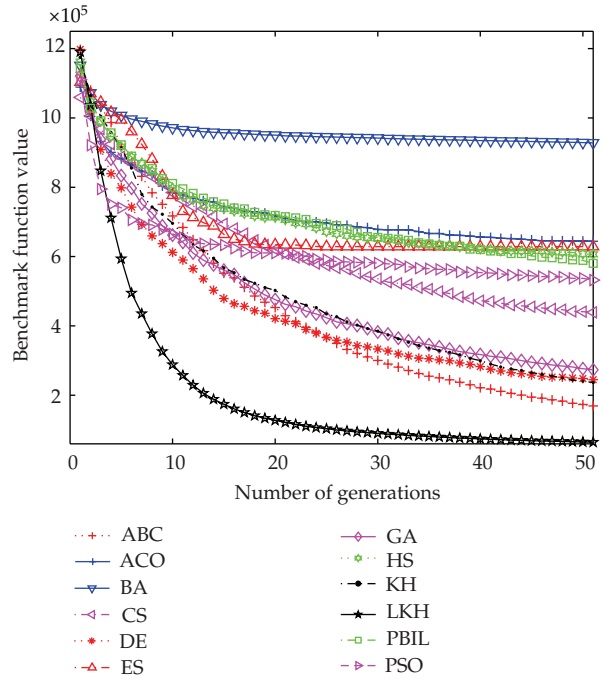

Figure 2: Comparison of the performance of the different methods for the F02 Fletcher-Powell function.

Figure 1 shows the results obtained for the twelve methods when the F01 Ackley function is applied. From Figure 1, clearly, we can draw the conclusion that LKH is significantly superior to all the other algorithms during the process of optimization. For other algorithms, although slower, KH II eventually finds the global minimum close to LKH, while ABC, ACO, BA, CS, DE, ES, GA, HS, PBIL, and PSO fail to search the global minimum within the limited generations. Here, all the algorithms show the almost same starting point; however, LKH outperforms them with fast and stable convergence rate.

Figure 2 illustrates the optimization results for F02 Fletcher-Powell function. In this multimodal benchmark problem, it is clear that LKH outperforms all other methods during the whole progress of optimization. Other algorithms do not manage to succeed in this benchmark function within maximum number of generations. At last, ABC and KH II converge to the value that is significantly inferior to LKH's.

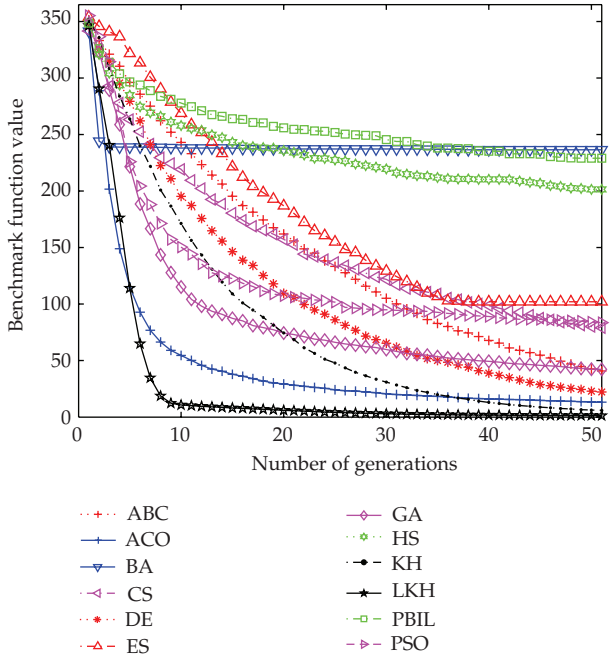

Figure 3: Comparison of the performance of the different methods for the F03 Griewank function.

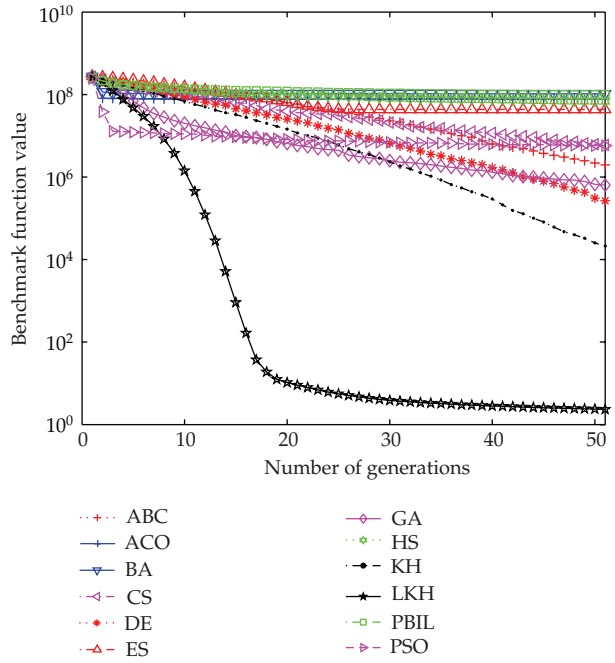

Figure 4: Comparison of the performance of the different methods for the F04 Penalty #1 function.

Figure 3 shows the optimization results for F03 Griewank function. From Figure 3, we can see that the figure shows that there is a little difference between the performance of LKH and KH II. However, from Table 3 and Figure 3, we can conclude that, LKH performs better than KH II in this multimodal function. Through carefully looking at Figure 6, ACO has a fast convergence initially towards the known minimum, as the procedure proceeds LKH gets closer and closer to the minimum, while ACO comes into being premature and traps into the local minimum.

Figure 4 shows the results for F04 Penalty #1 function. From Figure 4, clearly, LKH outperforms all other methods during the whole progress of optimization in this multimodal function. Eventually, KH II performs the second best at finding the global minimum. Although slower later, DE performs the third best at finding the global minimum.

Figure 5 shows the performance achieved for F05 Penalty #2 function. For this multimodal function, similar to the F04 Penalty #2 function as shown in Figure 4, LKH is significantly superior to all the other algorithms during the process of optimization. Here, KH II shows a stable convergence rate in the whole optimization process and eventually it performs the second best at finding the global minimum that is significantly superior to the other algorithms.

Figure 6 shows the results achieved for the twelve methods when using the F06 Quartic (*with noise*) function. For this case, the figure shows that there is a little difference among the performance of DE, GA, KH II, and LKH. From Table 3 and Figure 6, we can conclude that LKH performs the best in this multimodal function. KH II, DE, and GA perform as well and have ranks of 2, 3, and 4, respectively. Through carefully looking at Figure 6, PSO has a fast convergence

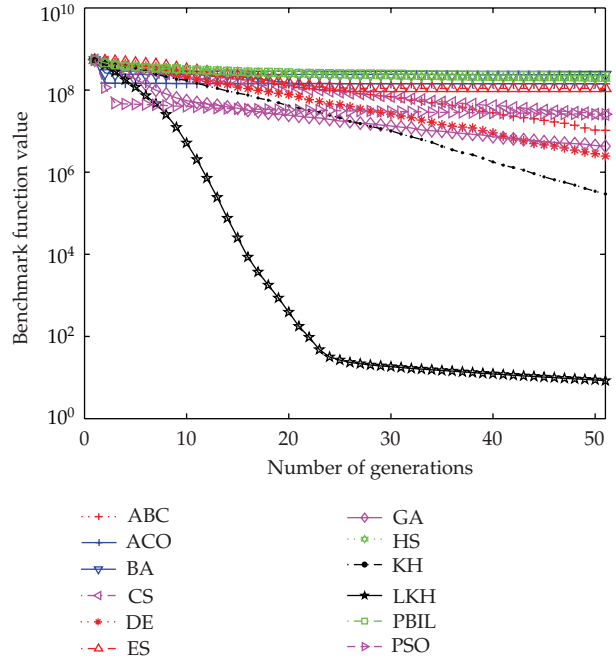

Figure 5: Comparison of the performance of the different methods for the F05 Penalty #2 function.

initially towards the known minimum; as the procedure proceeds, LKH gets closer and closer to the minimum, while PSO comes into being premature and traps into the local minimum.

Figure 7 shows the optimization results for the F07 Rastrigin function. In this multimodal benchmark problem,

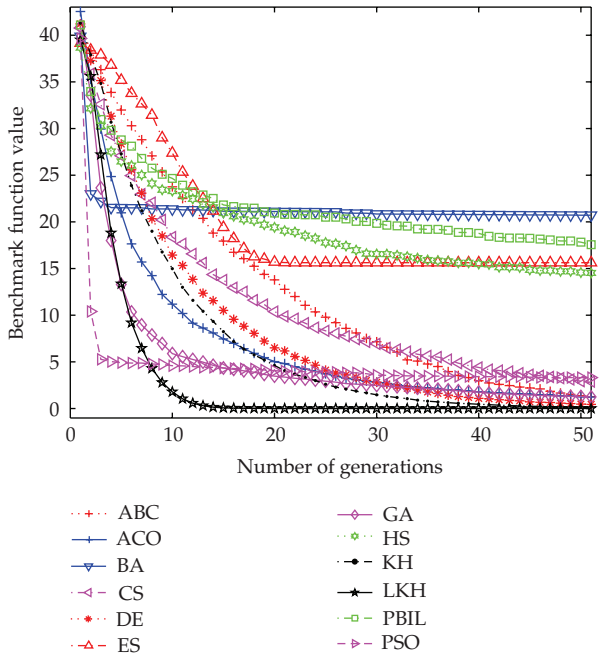

Figure 6: Comparison of the performance of the different methods for the F06 Quartic (*with noise*) function.

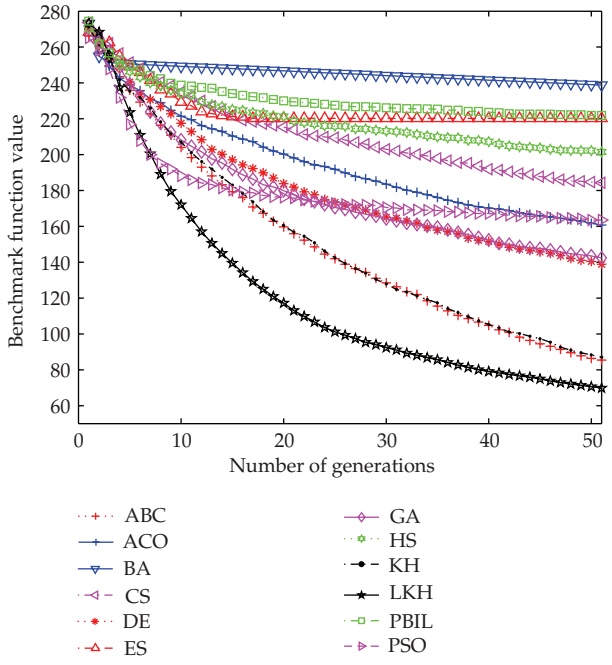

Figure 7: Comparison of the performance of the different methods for the F07 Rastrigin function.

it is obvious that LKH outperforms all other methods during the whole progress of optimization. For other algorithms, the figure shows that there is little difference between the performance of ABC and KH II. From Table 3 and Figure 7, we can conclude that, KH II performs slightly better than ABC in this multimodal function. In addition, other algorithms do not manage to succeed in this benchmark function within the maximum number of generations.

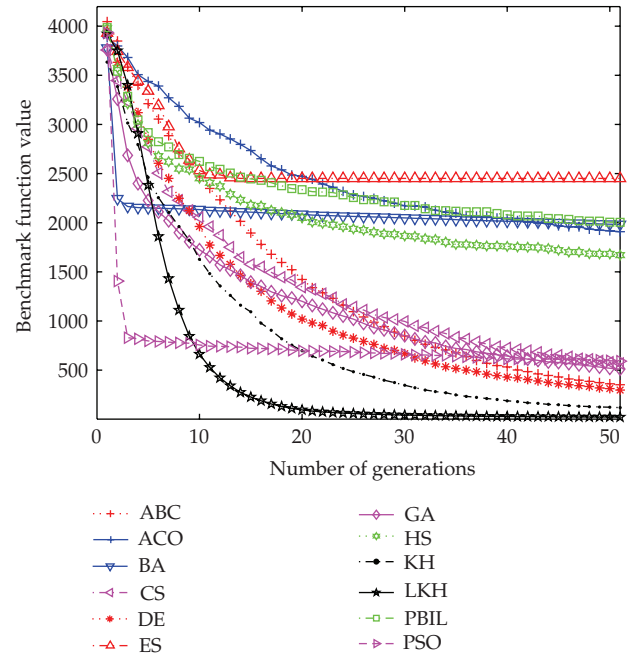

Figure 8: Comparison of the performance of the different methods for the F08 Rosenbrock function.

Figure 8 shows the results for F08 Rosenbrock function. From Figure 8, we can conclude that LKH performs the best in this unimodal function. In addition, KH II, DE, and ACO perform very well and have ranks of 2, 3, and 4, respectively. Through carefully looking at Figure 8, PSO has a fast convergence initially towards the known minimum; however, it is outperformed by LKH after 10 generations. For other algorithms, they do not manage to succeed in this benchmark function within the maximum number of generations.

Figure 9 shows the equivalent results for the F09 Schwefel 2.26 function. From Figure 9, clearly, GA is significantly superior to other algorithms including LKH during the process of optimization, while ACO and ABC perform the second and the third best in this multimodal benchmark function, respectively. Unfortunately, LKH only performs the fourth in this multimodal benchmark function.

Figure 10 shows the results for F10 Schwefel 1.2 function. For this case, LKH, CS, KH II, and ACO perform the best and have ranks of 1, 2, 3, and 4, respectively. Looking carefully at Figure 7, LKH has the fastest and stable convergence rate at finding the global minimum and significantly outperforms all other approaches.

Figure 11 shows the results for F11 Schwefel 2.22 function. From Figure 11, similar to the F09 Schwefel 2.26 function as shown in Figure 9, it is clear that ABC is significantly superior to other algorithms including LKH during the process of optimization. For other algorithms, DE and KH II perform very well and have ranks of 2 and 3, respectively. Unfortunately, LKH only performs the tenth best in this unimodal benchmark function among the twelve methods.

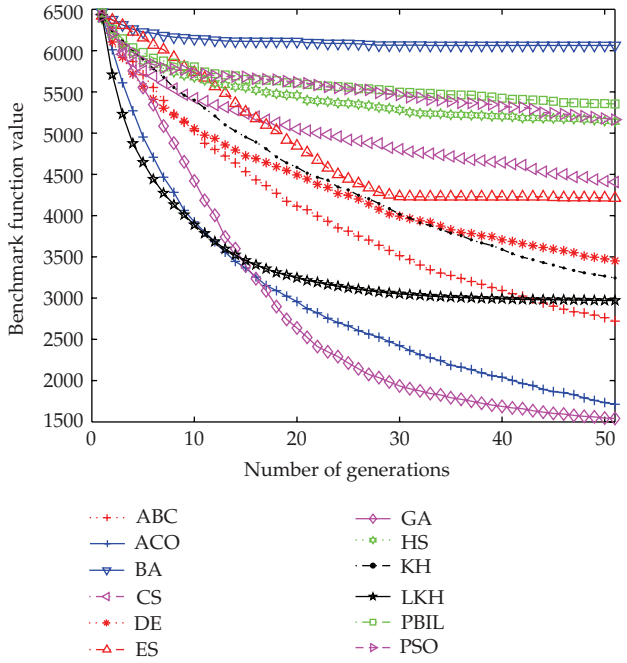

Figure 9: Comparison of the performance of the different methods for the F09 Schwefel 2.26 function.

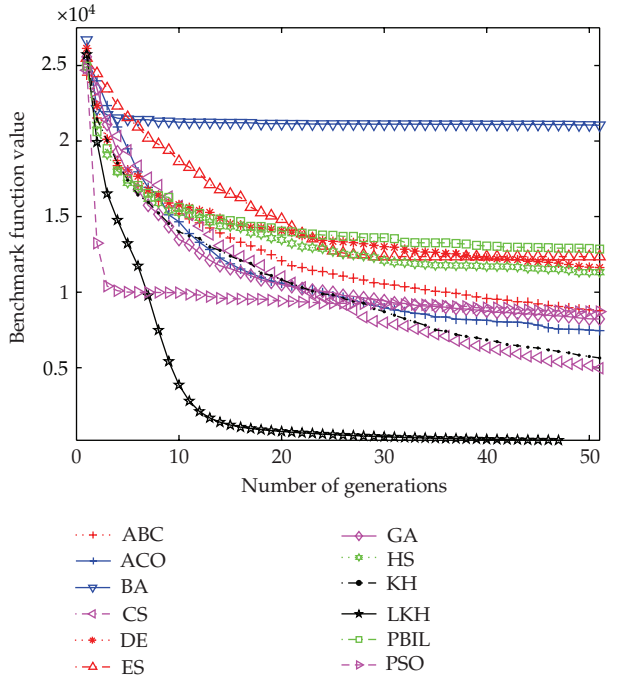

Figure 10: Comparison of the performance of the different methods for the F10 Schwefel 1.2 function.

Figure 12 shows the results for F12 Schwefel 2.21 function. Very clearly, LKH has the fastest convergence rate at finding the global minimum and significantly outperforms all other methods. For other algorithms, KH II and ACO that are only inferior to LKH perform very well and have ranks of 2 and 3, respectively.

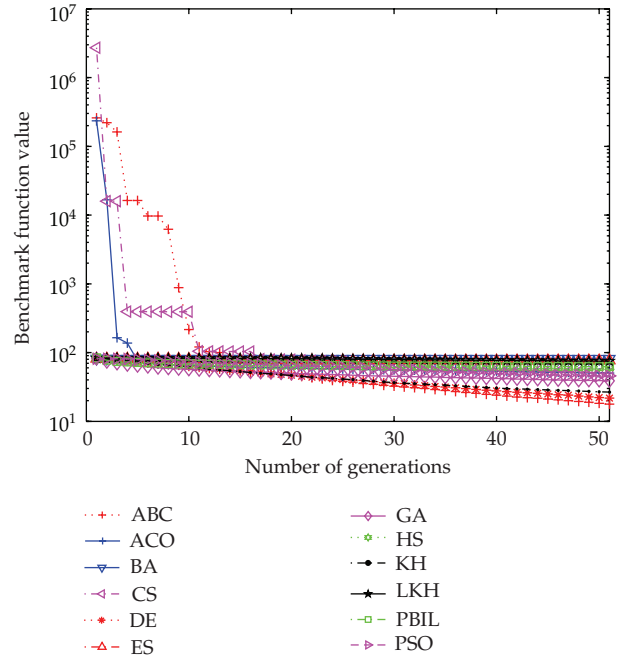

Figure 11: Comparison of the performance of the different methods for the F11 Schwefel 2.22 function.

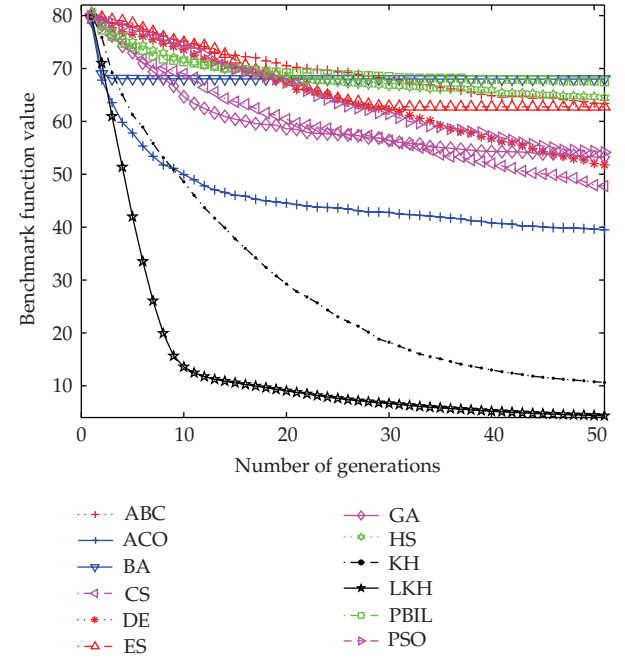

Figure 12: Comparison of the performance of the different methods for the F12 Schwefel 2.21 function.

Figure 13 shows the results for F13 Sphere function. From Figure 13, LKH shows the fastest convergence rate at finding the global minimum and significantly outperforms all other methods. In addition, KH II, DE, and ACO perform very well and have ranks of 2, 3, and 4, respectively.

Figure 14 shows the results for F14 Step function. Clearly, LKH shows the fastest convergence rate at finding the

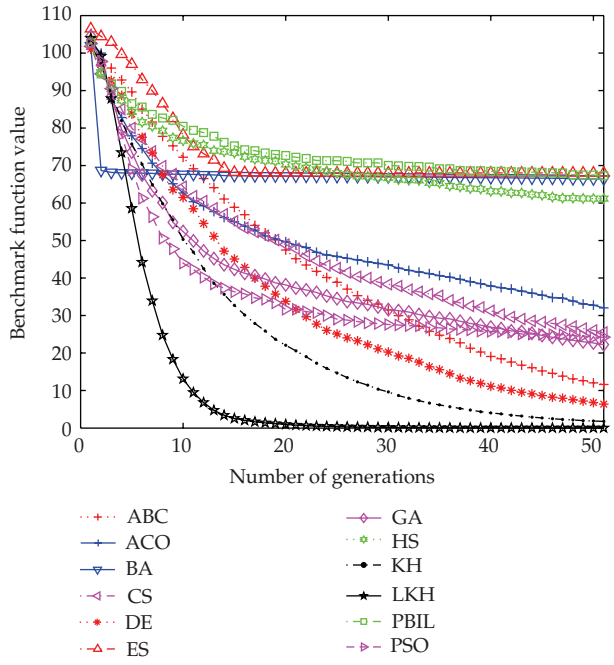

Figure 13: Comparison of the performance of the different methods for the F13 Sphere function.

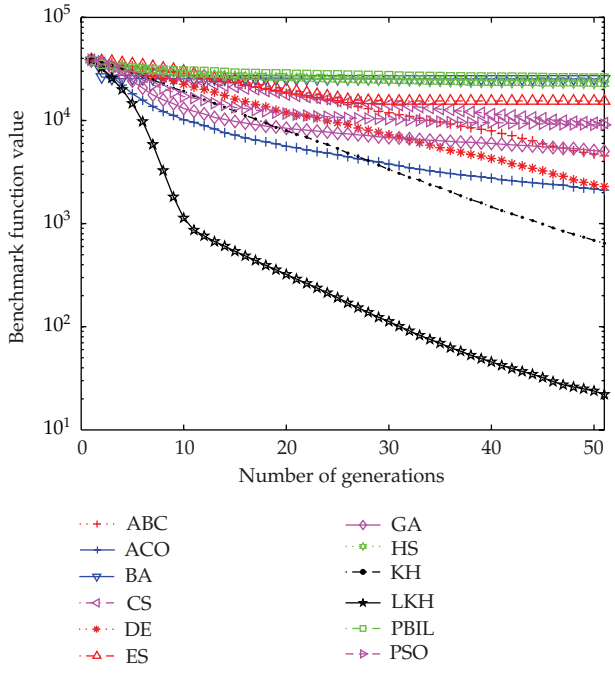

Figure 14: Comparison of the performance of the different methods for the F14 Step function.

global minimum and significantly outperforms all other approaches. Though slow, KH II performs the second best at finding the global minimum that is only inferior to the LKH.

From the above analyses about Figures 1–14, we can come to a conclusion that our proposed hybrid metaheuristic LKH algorithm significantly outperforms the other eleven algorithms. In general, KH II is only inferior to LKH and performs the second best among twelve methods. ABC, ACO, DE, and GA perform the third best only inferior to the LKH and KH II; ABC and GA especially perform better than LKH on benchmark functions F11 and F09, respectively. Furthermore, the illustration of benchmarks F04, F05, F06, F08, and F10 shows that PSO has a faster convergence rate initially, while later, it converges slower and slower to the true objective function value.

4.2. Discussion. For all of the standard benchmark functions considered in this section, the LKH method has been demonstrated to perform better than, or at least highly competitive with, the standard KH and other eleven acclaimed state-of-the-art population-based methods. The advantages of LKH involve performing simply and easily and have few parameters to regulate. The work here proves the LKH to be robust, powerful, and effective over all types of benchmark functions.

Benchmark evaluating is a good way for testing the performance of the metaheuristic methods, but it is also not flawless and has some limitations. First, we did not do much work painstakingly to carefully regulate the optimization methods in this section. In general, different tuning parameter values in the optimization methods might lead to significant differences in their performance. Second, real-world optimization problems may have little of a relationship to benchmark functions. Third, benchmark tests may arrive at fully different conclusions if the grading criteria or problem setup changes. In our present work, we looked into the mean and best values obtained with some population size and after some number of iterations. However, we might reach different conclusions if, for example, we change the population size, or look at how many population size it needs to reach a certain function value, or if we change the iteration. Despite these caveats, the benchmark results represented here are prospective for LKH and show that this novel method might be capable of finding a niche among the plethora of population-based optimization methods.

Note that running time is a bottleneck to the implementation of many population-based optimization algorithms. If an algorithm converges too slowly, it will be impractical and infeasible, since it would take too long to search an optimal or suboptimal solution. LKH seems not to require an unreasonable amount of computational time; of the twelve comparative optimization methods used in this paper, LKH was the eleventh fastest. How to speed up the LKH's convergence is worthy of further study.

In our study, 14 benchmark functions have been applied to evaluate the performance of our LKH method; we will test our proposed method on more optimization problems, such as the high-dimensional ($d \geq 20$) CEC 2010 test suit [44] and the real-world engineering problems. Moreover, we will compare LKH with other optimization algorithms. In addition, we only consider the unconstrained function optimization in this study. Our future work consists of adding the other techniques into LKH for constrained optimization problems, such as constrained real-parameter optimization CEC 2010 test suit [45].

5. Conclusion and Future Work

Due to the limited performance of KH on complex problems, LLF operator has been introduced into the standard KH to develop a novel Lévy-flight krill herd (LKH) algorithm for optimization problems. In LKH, at first, original KH algorithm is applied to shrink the search region to a more promising area. Thereafter, LLF operator is implemented as a critical complement to perform the local search to exploit the limited area intensively to get better solutions. In principle, KH takes full advantage of the three motions in the population and has experimentally demonstrated very good performance on the multimodal problems. In a rugged region of the fitness landscape, KH may fail to proceed to better solutions [27]. Then, LLF operator is adaptively launched to reboost the search. The LKH makes an attempt at taking merits of the KH and Lévy flight in order to avoid all krill getting trapped in inferior local optimal regions. The LKH enables the krill to have more diverse exemplars to learn from as the krill are updated each generation and also form new krill to search in a larger search space. With both techniques combined, LKH can balance exploration and exploitation and effectively solve complex multimodal problems.

Furthermore, this new method can speed up the global convergence rate without losing the strong robustness of the basic KH. From the analysis of the experimental results, we can see that the Lévy-flight KH clearly improves the reliability of the global optimality and they also enhance the quality of the solutions. Based on the results of the twelve methods on the test problems, we can conclude that LKH significantly improves the performances of the KH on most multimodal and unimodal problems. In addition, LKH is simple and implements easily.

In the field of numerical optimization, there are considerable issues that deserve further study, and some more efficient optimization methods 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 LKH method to solve real-world civil engineering optimization problems [46], and, obviously, LKH can be a promising method for these optimization problems. On the other hand, we would develop more new metaheuristic methods to solve optimization problems more efficiently and effectively.

Acknowledgments

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

References

[1] S. Gholizadeh and F. Fattahi, "Design optimization of tall steel buildings by a modified particle swarm algorithm," *The Structural Design of Tall and Special Buildings*. In press.

[2] S. Talatahari, R. Sheikholeslami, M. Shadfaran, and M. Pourbaba, "Optimum design of gravity retaining walls using charged system search algorithm," *Mathematical Problems in Engineering*, vol. 2012, Article ID 301628, 10 pages, 2012.

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

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

[5] S. Gholizadeh and A. Barzegar, "Shape optimization of structures for frequency constraints by sequential harmony search algorithm," *Engineering Optimization*. In press.

[6] S. Chen, Y. Zheng, C. Cattani, and W. Wang, "Modeling of biological intelligence for SCM system optimization," *Computational and Mathematical Methods in Medicine*, vol. 2010, Article ID 769702, 10 pages, 2012.

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

[8] X. S. Yang, *Engineering Optimization: An Introduction with Metaheuristic Applications*, Wiley & Sons, NJ, USA, 2010.

[9] 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.

[10] 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.

[11] 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.

[12] W.-H. Ho and A. L.-F. Chan, "Hybrid Taguchi-differential evolution algorithm for parameter estimation of differential equation models with application to HIV dynamics," *Mathematical Problems in Engineering*, vol. 2011, Article ID 514756, 14 pages, 2011.

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

[14] M. Shahsavar, A. A. Najafi, and S. T. A. Niaki, "Statistical design of genetic algorithms for combinatorial optimization problems," *Mathematical Problems in Engineering*, vol. 2011, Article ID 872415, 17 pages, 2011.

[15] G. Wang and L. Guo, "A novel hybrid bat algorithm with harmony search for global numerical optimization," *Journal of Applied Mathematics*. In press.

[16] 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.

[17] 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.

[18] A. H. Gandomi, X.-S. Yang, S. Talatahari, and S. Deb, "Coupled eagle strategy and differential evolution for unconstrained and constrained global optimization," *Computers & Mathematics with Applications*, vol. 63, no. 1, pp. 191–200, 2012.

[19] A. H. Gandomi and A. H. Alavi, "Multi-stage genetic programming: a new strategy to nonlinear system modeling," *Information Sciences*, vol. 181, no. 23, pp. 5227–5239, 2011.

[20] 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.

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

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

[23] S. Talatahari, M. Kheirollahi, C. Farahmandpour, and A. H. Gandomi, "A multi-stage particle swarm for optimum design of truss structures," *Neural Computing & Applications*. In press.

[24] 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.

[25] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems," *Engineering with Computers*, vol. 29, no. 1, pp. 1–19, 2013.

[26] G. Wang, L. Guo, H. Duan, L. Liu, H. Wang, and W. Jianbo, "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.

[27] A. H. Gandomi and A. H. Alavi, "Krill Herd: a new bio-inspired optimization algorithm," *Communications in Nonlinear Science and Numerical Simulation*, vol. 17, no. 12, pp. 4831–4845, 2012.

[28] 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*. In press.

[29] G. Wang, L. Guo, H. Duan, H. Wang, and L. Liu, "A new improved firefly algorithm for global numerical optimization," *Journal of Computational and Theoretical Nanoscience*. In press.

[30] 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.

[31] G. Wang, L. Guo, A. H. Gandomi et al., "A new improved krill herd algorithm for global numerical optimization," *Neurocomputing*. In press.

[32] S. Yang and J. Lee, "Multi-basin particle swarm intelligence method for optimal calibration of parametric Lévy models," *Expert Systems with Applications*, vol. 39, no. 1, pp. 482–493, 2012.

[33] P. Barthelemy, J. Bertolotti, and D. S. Wiersma, "A Lévy flight for light," *Nature*, vol. 453, no. 7194, pp. 495–498, 2008.

[34] A. Natarajan, S. Subramanian, and K. Premalatha, "A comparative study of cuckoo search and bat algorithm for Bloom filter optimisation in spam filtering," *International Journal of Bio-Inspired Computation*, vol. 4, no. 2, pp. 89–99, 2012.

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

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

[37] X. Li, J. Wang, J. Zhou, and M. Yin, "A perturb biogeography based optimization with mutation for global numerical optimization," *Applied Mathematics and Computation*, vol. 218, no. 2, pp. 598–609, 2011.

[38] 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.

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

[40] X. S. Yang and S. Deb, "Engineering optimisation by cuckoo search," *International Journal of Mathematical Modelling and Numerical Optimisation*, vol. 1, no. 4, pp. 330–343, 2010.

[41] H.-G. Beyer, *The Theory of Evolution Strategies*, Springer, Berlin, Germany, 2001.

[42] B. Shumeet, "Population-Based Incremental Learning: A Method for Integrating Genetic Search Based Function Optimization and Competitive Learning," Tech. Rep. CMU-CS-94-163, Carnegie Mellon University, Pittsburgh, Pa, USA, 1994.

[43] 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.

[44] 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," Inspired Computation and Applications Laboratory, USTC, Hefei, China, 2010.

[45] 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.

[46] P. Lu, S. Chen, and Y. Zheng, "Artificial intelligence in civil engineering," *Mathematical Problems in Engineering*, vol. 2012, Article ID 145974, 22 pages, 2012.