Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 465082, 17 pages
http://dx.doi.org/10.1155/2014/465082

Research Article

Wolf Pack Algorithm for Unconstrained Global Optimization

Hu-Sheng Wu[1,2] and Feng-Ming Zhang[1]

[1] *Materiel Management and Safety Engineering Institute, Air Force Engineering University, Xi'an 710051, China*
[2] *Materiel Engineering Institute, Armed Police Force Engineering University, Xi'an 710086, China*

Correspondence should be addressed to Hu-Sheng Wu; wuhusheng0421@gmail.com

Received 28 June 2013; Revised 13 January 2014; Accepted 27 January 2014; Published 9 March 2014

Academic Editor: Orwa Jaber Housheya

Copyright © 2014 H.-S. Wu and F.-M. Zhang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The wolf pack unites and cooperates closely to hunt for the prey in the Tibetan Plateau, which shows wonderful skills and amazing strategies. Inspired by their prey hunting behaviors and distribution mode, we abstracted three intelligent behaviors, scouting, calling, and besieging, and two intelligent rules, winner-take-all generation rule of lead wolf and stronger-survive renewing rule of wolf pack. Then we proposed a new heuristic swarm intelligent method, named wolf pack algorithm (WPA). Experiments are conducted on a suit of benchmark functions with different characteristics, unimodal/multimodal, separable/nonseparable, and the impact of several distance measurements and parameters on WPA is discussed. What is more, the compared simulation experiments with other five typical intelligent algorithms, genetic algorithm, particle swarm optimization algorithm, artificial fish swarm algorithm, artificial bee colony algorithm, and firefly algorithm, show that WPA has better convergence and robustness, especially for high-dimensional functions.

1. Introduction

Global optimization is a hot topic with applications in many areas, such as science, economy, and engineering. Generally; unconstrained global optimization problems can be formulated as follows:

$$\min \text{ or } \max f(X), \quad X = (x_1, x_2, \ldots, x_n), \tag{1}$$

where $f : R^n \rightarrow R$ is a real-valued objective function, $X \in R^n$, and n is the number of parameters to be optimized.

As many real-world problems are becoming increasingly complex; global optimization, especially using traditional methods, is becoming a challenging task [1]. Because of its great search space, high-dimensional global optimization problems are more difficult [2]. Fortunately, many algorithms inspired by nature have become powerful tools for these problems [3–5]. Since, with long time of biological evolution and natural selection, there are many marvelous swarm intelligence phenomenons in nature, which are wonderful and can give us endless inspiration. The remarkable swarm behavior of animals such as swarming ants, schooling fish, and flocking birds has for long captivated the attention of naturalists and scientists [6]. People have developed many intelligent optimization methods to solve complex global problems in recent decades. In 1995, inspired by social behavior and movement dynamics of birds, Kennedy proposed the particle swarm optimization algorithm (PSO) [7]. In 1996, inspired by social division and foraging behavior of ant colonies, Dorigo proposed the ant colony optimization algorithm (ACO) [8]. In 2002, inspired by foraging behavior of fish schools, Li proposed the artificial fish swarm algorithm (AFSA) [9]. In 2005, motivated by the intelligent foraging behavior of honeybee swarms, Karaboga proposed the artificial bee colony (ABC) algorithm [10]. In 2008, based on the flashing behavior of fireflies, Doctor Yang proposed firefly algorithm (FA) [11]. Researchers even give some conceptions of swarm intelligent algorithms such as rats herds algorithm, mosquito swarms algorithm, and dolphins herds algorithm [12]. Birds, fishes, ants, and bees do not have any human complex intelligence such as logical reasoning and synthetic judgment, but under the same aim, food, they stand out powerful swarm intelligence through constantly adapting environment and mutual cooperation, which give us many new ideas for solving complex problems.

The wolf pack is marvelous. Harsh living environment and constant evolution for centuries have created their

rigorous organization system and subtle hunting behavior. Wolves tactics of Mongolia cavalry in Genghis Khan period, submarine tactics of Nazi Admiral Doenitz in World War II and U.S. military wolves attack system for electronic countermeasures all highlight great charm of their swarm intelligence. [13] proposes a wolf colony algorithm (WCA) to solve the optimization problem. But the accuracy and efficiency of WCA are not good enough and easily fall into local optima, especially for high-dimensional functions. So, in this paper, we reanalyzed collaborative predation behavior and prey distribution mode of wolves and proposed a new swarm intelligence algorithm, called wolf pack algorithm (WPA); Moreover, the efficiency and robustness of the new algorithm were tested by compared experiments.

The remainder of this paper is structured as follows. In Section 2, the predation behaviors and prey distribution of wolves are analyzed. In Section 3, WPA is described. Section 4 describes the experimental setup, followed by experimental results and analysis. Finally, conclusion and future work are presented in Section 5.

2. System Analyzing of Wolf Pack

Wolves are gregarious animals and have clearly social work division. There is a lead wolf; some elite wolves act as scouts and some ferocious wolves in a wolf pack. They cooperate well with each other and take their respective responsibility for the survival and thriving of wolf pack.

Firstly, the lead wolf, as a leader under the law of the jungle, is always the smartest and most ferocious one. It is responsible for commanding the wolves and constantly making decision by evaluating surrounding situation and perceiving information from other wolves. These can avoid the wolves in danger and command the wolves to smoothly capture prey as soon as possible.

Secondly, the lead wolf sends some elite wolves to hunt around and look for prey in the probable scope. Those elite wolves are scouts. They walk around and independently make decision according to the concentration of smell left by prey; and higher concentration means the prey is closer to the wolves. So they always move towards the direction of getting stronger smell.

Thirdly, once a scout wolf finds the trace of prey, it will howl and report that to lead wolf. Then the lead wolf will evaluate this situation and make a decision whether to summon the ferocious wolves to round up the prey or not. If they are summoned, the ferocious wolves will move fast towards the direction of the scout wolf.

Fourthly, after capturing the prey, the prey is not distributed equitably, but in an order from the strong to the weak. That is to say that, the stronger the wolf is, the more the food it will get is. Although this distribution rule will make some weak wolf dead for lack of food, it makes sure that the wolves that have the ability to capture prey get more food so as to keep being strong and can capture more prey successfully in the next time. The rule avoids that the whole pack starves to death and ensures its continuance and proliferating. In what follows, the author made detailed description and realization for the above intelligent behaviors and rules.

3. Wolf Pack Algorithm

3.1. Some Definitions. If the predatory space of the artificial wolves is a $N \times D$ Euclidean space, N is the number of wolves, D is the number of variables. The position of one wolf i is a vector $\mathbf{X}_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$, and x_{id} is the dth variable value of the ith artificial wolf. $Y = f(\mathbf{X})$ represents the concentration of prey's smell perceived by artificial wolves, which is also the objective function value.

The distance between two wolves p and q is described as $L(p, q)$. Several distance measurements can be selected according to specific problems. For example, hamming distance can be used in WPA for 0-1 discrete optimization, while Manhattan distance (MD) and Euclidean distance (ED) can be used in WPA for continuous numerical function optimization. In this paper, we mainly discuss the latter problem, and the selection of distance measurements will be discussed in Section 4.2.1. Moreover, because the problems of maximum value and minimal value can convert to each other, only the maximum value problem is discussed in what follows.

3.2. The Description of Intelligent Behaviors and Rules. The cooperation between lead wolf, scout wolves, and ferocious wolves makes nearly perfect predation, while prey distribution from the strong to the weak makes the wolf pack thrives towards the direction of the prey that it most probably can be able to capture. The whole predation behavior of wolf pack is abstracted three intelligent behaviors, scouting, calling, and besieging behavior, and two intelligent rules, winner-take-all generating rule for the lead wolf and the stronger-survive renewing rule for the wolf pack.

(1) The winner-take-all generating rule for the lead wolf: the artificial wolf with the best objective function value is lead wolf. During each iteration, compare the function value of the lead wolf with the best one of other wolves; if the value of lead wolf is not better, it will be replaced. Then the best wolf becomes lead wolf. Rather than acting the three intelligent behaviors, the lead wolf directly goes into the next iteration until it is replaced by other better wolf.

(2) Scouting behavior: S_num elite wolves except the lead wolf are considered as the scout wolves; they search the solution in predatory space. Y_i is the concentration of prey smell perceived by the scout wolf i. Y_{lead} is the concentration of prey smell perceived by the lead wolf.

If $Y_i > Y_{\text{lead}}$, that means the scout wolf is nearer to the prey and probably captures prey, so the scout wolf i becomes lead wolf and $Y_{\text{lead}} = Y_i$.

If $Y_i < Y_{\text{lead}}$, the scout wolf i, respectively, takes astep towards h different directions; the step length is $step_a$. After taking a step towards the pth direction, the state of the scout wolf i is formulated below:

$$x_{id}^{p} = x_{id} + \sin\left(2\pi \times \frac{p}{h}\right) \times \text{step}_a^d, \quad p = \{1, 2, \ldots, h\}. \quad (2)$$

It should be noted that h is different for each wolf because of their different seeking ways. So h is randomly selected in $[h_{\min}, h_{\max}]$ and it must be an integer. Y_{i0} is the concentration of prey smell perceived by the scout wolf i and Y_{ip} represents

the one after it took a step towards the pth direction. If $\max\{Y_{i1}, Y_{i2}, \ldots, Y_{ih}\} > Y_{i0}$, the wolf i steps forward and its position X_i is updated. Then repeat the above until $Y_i > Y_{\text{lead}}$ or the maximum number of repetitions $T_{\max}$ is reached.

(3) Calling behavior: the lead wolf will howl and summon M_num ferocious wolves to gather around the prey. Here, the position of the lead wolf is considered as the one of the prey so that the ferocious wolves aggregate towards the position of lead wolf. $step_b$ is the step length; g_d^k is the position of artificial lead wolf in the dth variable space at the kth iteration. The position of the ferocious wolf i in the kth iterative calculation is updated according to the following equation:

$$x_{id}^{k+1} = x_{id}^{k} + \text{step}_b^d \cdot \frac{\left(g_d^k - x_{id}^k\right)}{\left|g_d^k - x_{id}^k\right|}. \tag{3}$$

This formula consists of two parts; the former is the current position of wolf i, which represents the foundation for prey hunting; the latter represents the aggregate tendency of other wolves towards the lead wolf, which shows the lead wolf's leadership to the wolf pack.

If $Y_i > Y_{\text{lead}}$, the ferocious wolf i becomes lead wolf and $Y_{\text{lead}} = Y_i$; then the wolf i takes the calling behavior; If $Y_i < Y_{\text{lead}}$, the ferocious wolf i keeps on aggregating towards the lead wolf with a fast speed until $L(i,l) < L_{\text{near}}$; the wolf takes besieging behavior. $L(i,l)$ is the distance between the wolf i and the lead wolf l; L_{near} is the distance determinant coefficient as a judging condition, which determine whether wolf i changes state from aggregating towards the lead wolf to besieging behavior. The different value of L_{near} will affect algorithmic convergence rate. There will be a discussion in Section 4.2.2.

Calling behavior shows information transferring and sharing mechanism in wolf pack and blends the idea of social cognition.

(4) Besieging behavior: after large-steps running towards the lead wolf, the wolves are close to the prey, then all wolves except the lead wolf will take besieging behavior for capturing prey. Now, the position of lead wolf is considered as the position of prey. In particular, G_d^k reprensents the position of prey in the dth variable space at the kth iteration. The position of wolf i is updated according to the following equation:

$$x_{id}^{k+1} = x_{id}^{k} + \lambda \cdot \text{step}_c^d \cdot \left|G_d^k - x_{id}^k\right|. \tag{4}$$

λ is a random number uniformly distributed at the interval $[-1, 1]$; $step_c$ is the step length of wolf i when it takes besieging behavior. Y_{i0} is the concentration of prey smell perceived by the wolf i and Y_{ik} represents the one after it took this behavior. If $Y_{i0} < Y_{ik}$, the position $\mathbf{X}_i$ is updated; otherwise it not changed.

There are $step_a$, $step_b$, and $step_c$ in the three intelligent behaviors, and the three-step length in dth variable space should have the following relationship:

$$\text{step}_a^d = \frac{\text{step}_b^d}{2} = 2 \cdot \text{step}_c^d = S. \tag{5}$$

S is step coefficient and represents the fineness degree of artificial wolf searching for prey in resolution space.

(5) The stronger-survive renewing rule for the wolf pack: the prey is distributed from the strong to the weak, which will result in some weak wolves dead. The algorithm will generate R wolves while deleting R wolves with bad objective function values. Specifically, with the help of the lead wolf's hunting experience, in the dth variable space, position of the ith one of R wolves is defined as follows:

$$x_{id} = g_d \cdot \text{rand}, \quad i = \{1, 2, \ldots, R\}. \tag{6}$$

g_d is the position of artificial lead wolf in the dth variable space, $rand$ is a random number uniformly distributed at the interval $[-0.1, 0.1]$.

When the value of R is larger, it is better for sustaining wolf's diversity and making the algorithm have the ability to open up new resolution space. But if R is too large, the algorithm will nearly be a random search approach. Because the number and scale of prey captured by wolves are different in natural word, which will lead to different number of weak wolf dead. R is an integer and randomly selected at the interval $[n/(2 * \beta), n/\beta]$. β is the population renewing proportional coefficient.

3.3. Algorithm Description. As described in the previous section, WPA has three artificial intelligent behaviors and two intelligent rules. There are scouting behavior, calling behavior, and besieging behavior and winner-take-all rule for generating lead wolf and the stronger-survive renewing rule for wolf pack.

Firstly, the scouting behavior accelerates the possibility that WPA can fully traverse the solution space; Secondly, the winner-take-all rule for generating lead wolf and the calling behavior make the wolves move towards the lead wolf whose position is the nearest to the prey and most likely capturing prey. The winner-take-all rule and calling behavior also make wolves arrive at the neighborhood of the global optimum only after a few iterations elapsed, since the step of wolves in calling behavior is the largest one. Thirdly, with a small step, step_c, besieging behavior makes WPA algorithm have the ability to open up new solution space and carefully search the global optima in good solution area. Fourthly, with the help of stronger-survive renewing rule for the wolf pack, the algorithm can get several new wolves whose positions are near the best wolf, lead wolf, which allows for more latitude of search space to anchor the global optimum while keeping population diversity in each iteration.

All the above make WPA possesses superior performance in accuracy and robustness, which will be seen in Section 4.

Having discussed all the components of WPA, the important computation steps are detailed below.

Step 1 (initialization). Initialize the following parameters, the initial position of artificial wolf i ($\mathbf{X}_i$), the number of the wolves (N), the maximum number of iterations ($k_{\max}$), the step coefficient (S), the distance determinant coefficient (L_{near}), the maximum number of repetitions in scouting behavior ($T_{\max}$), and the population renewing proportional coefficient (β).

Table 1: Benchmark functions in experiments.

No.	Functions	Formulation	Global extremum	D	C	Range
1	Rosenbrock	$f(\vec{x}) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$	$f_{\min}(\vec{x}) = 0$	2	UN	$(-2.048, 2.048)$
2	Colville	$f(\vec{x}) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 + 10.1(x_2 - 1)^2 + (x_4 - 1)^2 + 19.8(x_2 - 1)(x_4 - 1)$	$f_{\min}(\vec{x}) = 0$	4	UN	$(-10, 10)$
3	Sphere	$f(\vec{x}) = \sum_{i=1}^{D} x_i^2$	$f_{\min}(\vec{x}) = 0$	200	US	$(-100, 100)$
4	Sumsquares	$f(\vec{x}) = \sum_{i=1}^{D} i x_i^2$	$f_{\min}(\vec{x}) = 0$	150	US	$(-10, 10)$
5	Booth	$f(\vec{x}) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$	$f_{\min}(\vec{x}) = 0$	2	MS	$(-10, 10)$
6	Bridge	$f(\vec{x}) = \frac{\sin\sqrt{x_1^2 + x_2^2}}{\sqrt{x_1^2 + x_2^2}} + \exp\left(\frac{\cos 2\pi x_1 + \cos 2\pi x_2}{2}\right) - 0.7129$	$f_{\max}(\vec{x}) = 3.0054$	2	MN	$(-1.5, 1.5)$
7	Ackley	$f(\vec{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos 2\pi x_i\right) + 20 + e$	$f_{\min}(\vec{x}) = 0$	50	MN	$(-32, 32)$
8	Griewank	$f(\vec{x}) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$f_{\min}(\vec{x}) = 0$	100	MN	$(-600, 600)$

D: dimension; C: characteristic; U: unimodal; M: multimodal; S: separable; N: nonseparable.

Step 2. The wolf with best function value is considered as lead wolf. In practical computation, $S\_num = M\_num = n - 1$, which means that wolves except for lead wolf act with different behavior as different status. So, here, except for lead wolf, according to formula (2), the rest of the $n - 1$ wolves firstly act as the artificial scout wolves to take scouting behavior until $Y_i > Y_{\text{lead}}$ or the maximum number of repetition $T_{\max}$ is reached and then go to Step 3.

Step 3. Except for the lead wolf, the rest of the $n - 1$ wolves secondly act as the artificial ferocious wolves and gather towards the lead wolf according to (3); Y_i is the smell concentration of prey perceived by wolf i; if $Y_i \geq Y_{\text{lead}}$, go to Step 2; otherwise the wolf i continues running until $L(i, l) \leq L_{\text{near}}$; then go to Step 4.

Step 4. The position of artificial wolves who take besieging behavior is updated according to (4).

Step 5. Update the position of lead wolf under the winner-take-all generating rule and update the wolf pack under the population renewing rule according to (6).

Step 6. If the program reaches the precision requirement or the maximum number of iterations, the position and function value of lead wolf, the problem optimal solution, will be outputted; otherwise go to Step 2.

So the flow chart of WPA can be shown as Figure 1.

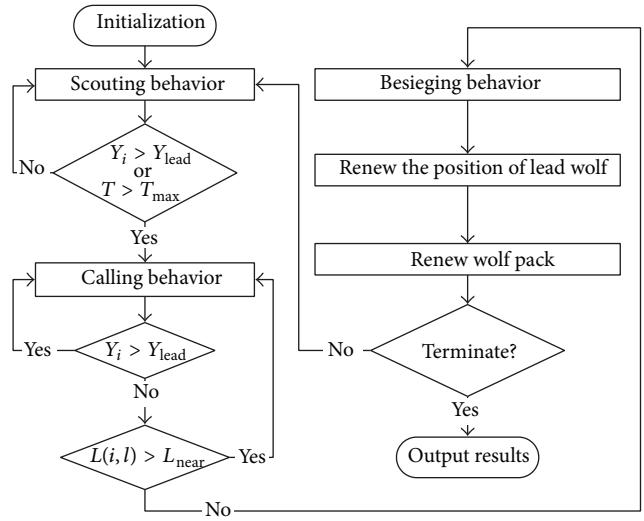


Figure 1: The flow chart of WPA.

4. Experimental Results

The ingredients of the WPA method have been described in Section 3. In this section, the design of experiments is explained, sensitivity analysis of parameters on WPA is explored, and the empirical results are reported, which compare the WPA approach with those of GA, PSO, ASFA, ABC, and FA.

4.1. Design of the Experiments

4.1.1. Benchmark Functions. In order to evaluate the performance of these algorithms, eight classical benchmark functions are presented in Table 1. Though only eight functions are used in this test, they are enough to include some different kinds of problems such as unimodal, multimodal, regular, irregular, separable, nonseparable and multidimensional.

If a function has more than one local optimum, this function is called multimodal. Multimodal functions are used to test the ability of algorithms to get rid of local minima.

Table 2: The list of various methods used in the paper.

Method	Authors and references
Genetic algorithm (GA)	Goldberg [14]
Particle swarm optimization algorithm (PSO)	Kennedy and Eberhart [7]
Artificial fish school algorithm (ASFA)	Li et al. [9]
Artificial bee colony algorithm (ABC)	Karaboga [10]
Firefly algorithm (FA)	Yang [11]

Table 3: Six criteria and their abbreviations.

Criteria	Abbreviation
The best value of optima found in 50 runs	Best
The worst value of optima found in 50 runs	Worst
The average value of optima found in 50 runs	Mean
The standard deviations	StdDev
The success rate of the results	SR
The average reaching time	Art

Another group of test problems is separable or nonseparable functions. A p-variable separable function can be expressed as the sum of p functions of one variable, such as Sumsquares and Rastrigin. Nonseparable functions cannot be written in this form, such as Bridge, Rosenbrock, Ackley, and Griewank. Because nonseparable functions have interrelation among their variable, these functions are more difficult than the separable functions.

In Table 1, characteristics of each function are given under the column titled C. In this column, M means that the function is multimodal, while U means that the function is unimodal. If the function is separable, abbreviation S is used to indicate this specification. Letter N refers to that the function is nonseparable. As seen from Table 1, 4 functions are multimodal, 4 functions are unimodal, 3 functions are separable, and 5 functions are nonseparable.

The variety of functions forms and dimensions make it possible to fairly assess the robustness of the proposed algorithms within limit iteration. Many of these functions allow a choice of dimension, and an input dimension ranging from 2 to 200 for test functions is given. Dimensions of the problems that we used can be found under the column titled D. Besides, initial ranges, formulas, and global optimum values of these functions are also given in Table 1.

4.1.2. Experimental Settings. In this subsection, experimental settings are given. Firstly, in order to fully compare the performance of different algorithms, we take the simulation under the same situation. So the values of the common parameters used in each algorithm such as population size and evaluation number were chosen to be the same. Population size was 100 and the maximum evaluation number was 2000 for all algorithms on all functions. Additionally, we follow the parameter settings in the original paper of GA, PSO, AFSA, ABC, and FA; see Table 2.

For each experiment, 50 independent runs were conducted with different initial random seeds. To evaluate the performance of these algorithms, six criteria are given in Table 3.

Accelerating convergence speed and avoiding the local optima have become two important and appealing goals in swarm intelligent search algorithms. So, as seen in Table 3, we adopted criteria best, mean, and standard deviation to evaluate efficiency and accuracy of algorithms and adopted criteria Art, Worst, and SR to evaluate convergence speed, effectiveness, and robustness of six algorithms.

Specifically speaking, SR provides very useful information about how stable an algorithm is. Success is claimed if an algorithm successfully gets a solution below a prespecified threshold value with the maximum number of function evaluations [15]. So, to calculate the success rate, an error accuracy level $\varepsilon = 10^{-6}$ must be set ($\varepsilon = 10^{-6}$ also used in [16]). Thus, we compared the result F with the known analytical optima F^* and consider F to be "successful" if the following inequality holds:

$$\begin{aligned} \frac{\left|F - F^*\right|}{F^*} &< \varepsilon, \quad F^* \neq 0, \\ \left|F - F^*\right| &< \varepsilon, \quad F^* = 0. \end{aligned} \tag{7}$$

The SR is a percentage value that is calculated as

$$\mathrm{SR} = \frac{\#\text{successful runs}}{\#\text{runs}}. \tag{8}$$

Art is the average value of time once an algorithm gets a solution satisfying the formula (7) in 50-run computations. Art also provides very useful information about how fast an algorithm converges to certain accuracy or under the same termination criterion, which has important practical significance.

All algorithms have been tested in Matlab 2008a over the same Lenovo A4600R computer with a Dual-Core 2.60 GHz processor, running Windows XP operating system over 1.99 Gb of memory.

4.2. Experiments 1: Effect of Distance Measurements and Four Parameters on WPA. In order to study the effect of two distance measures and four parameters on WPA, different measures and values of parameters were tested on typical functions listed in Table 1. Each experiment, WPA algorithm that runs 50 times on each function, and several criteria described in Section 4.1.2 are used. The experiment is conducted with the original coefficients shown in Table 9.

4.2.1. Effect of Distance Measurements on the Performance of WPA. This subsection will investigate the performance of different distance measurements using functions with different characteristics. As is known to all, Euclidean distance (ED) and Manhattan distance (MD) are the two most common distance metrics in practical continuous optimization. In the proposed WPA, MD or ED can be adopted to measure the distance between two wolves in the candidate solution

Table 4: Sensitivity analysis of distance measurements.

Function	Global extremum	D	Distance	Best	Worst	Mean	StdDev	SR/%	Art/s
Rosenbrock	$f_{\min}(\vec{x}) = 0$	2	MD	$9.21e-11$	$3.24e-8$	$1.12e-8$	$1.18e-8$	100	10.5165
			ED	$4.26e-9$	$2.71e-7$	$1.27e-7$	$6.81e-8$	100	37.1053
Colville	$f_{\min}(\vec{x}) = 0$	4	MD	$5.62e-8$	$5.28e-7$	$2.49e-7$	$2.23e-7$	100	46.8619
			ED	$1.74e-7$	$1.70e-6$	$5.74e-7$	$3.70e-7$	90	68.3220
Sphere	$f_{\min}(\vec{x}) = 0$	200	MD	$3.20e-161$	$3.29e-144$	$2.07e-145$	$7.49e-145$	100	11.5494
			ED	$1.76e-160$	$3.36e-143$	$1.68e-144$	$7.51e-144$	100	11.6825
Sumsquares	$f_{\min}(\vec{x}) = 0$	150	MD	$1.56e-161$	$3.09e-144$	$1.79e-145$	$6.95e-145$	100	8.5565
			ED	$3.97e-160$	$2.24e-144$	$1.13e-145$	$5.00e-145$	100	8.7109
Booth	$f_{\min}(\vec{x}) = 0$	2	MD	$5.63e-12$	$1.15e-10$	$4.19e-11$	$3.32e-11$	100	11.1074
			ED	$1.08e-9$	$2.64e-8$	$1.16e-8$	$6.93e-9$	100	40.5546
Bridge	$f_{\max}(\vec{x}) = 3.0054$	2	MD	3.0054	3.0054	3.0054	$4.56e-16$	100	1.1093
			ED	3.0054	3.0054	3.0054	$4.56e-16$	100	1.9541
Ackley	$f_{\min}(\vec{x}) = 0$	50	MD	$8.88e-16$	$8.88e-16$	$8.88e-16$	0	100	19.3648
			ED	$8.88e-16$	$8.88e-16$	$8.88e-16$	0	100	43.6884
Griewank	$f_{\min}(\vec{x}) = 0$	100	MD	0	0.1507	$3.01e-3$	0.0213	98	$>8.77e3$
			ED	0	0.8350	0.0167	0.1181	92	$>1.35e+4$

space. Therefore, a discussion about their impacts on the performance of WPA is needed.

There are two wolves: $\mathbf{X}_p = (x_{p1}, x_{p2}, \ldots, x_{pD})$ is the position of wolf $p, \mathbf{X}_q = (x_{q1}, x_{q2}, \ldots, x_{qD})$ is the position of wolf q, and the ED and MD between them can be, respectively, calculated as formula (9). D is the dimension number of solution space

$$L_{\mathrm{ED}}(p,q) = \sum_{d=1}^{D} \left(x_{pd} - x_{qd}\right)^2,$$
$$L_{\mathrm{MD}}(p,q) = \sum_{d=1}^{D} \left|x_{pd} - x_{qd}\right|. \tag{9}$$

The statistical results obtained by WPA after 50-run computation are shown in Table 4. Firstly, we note that WPA with Euclidean distance (WPA_ED) does not get 100% success rate on Colville ($D = 4$) and Griewank functions ($D = 100$), while WPA with Manhattan distance (WPA_MD) does not get 100% success rate on Griewank functions ($D = 100$), which means that WPA_ED and WPA_MD with original coefficients still have the risk of premature convergence to local optima.

As seen from Table 4, WPA is not very sensitive to two distance measurements on most functions (Rosenbrock, Sphere, Sumsquares, Booth, and Ackley), and no matter which metric is used, WPA can always get a good result with SR = 100%. But, for these functions, comparing the results between WPA_MD and WPA_ED in detail, we can find that WPA_MD has shorter average reaching time (ARt), which means faster convergence speed to a certain accuracy. The reason may be that ED has the higher computational complexity. Meanwhile, WPA_MD has better performance on other four criteria (best, worst, mean, and StdDev), which means better solution accuracy and robustness.

Naturally, because of its better efficiency, precision, and robustness, WD is more suitable for WPA. So the WPA algorithm used in what follows is WPA_MD.

4.2.2. Effect of Four Parameters on the Performance of WPA. In this subsection, we investigate the impact of the parameters S, L_{near}, $T_{\max}$, and β on the new algorithm. S is the step coefficient, L_{near} is the distance determinant coefficient, $T_{\max}$ is the maximum number of repetitions in scouting behavior, and β is the population renewing proportional coefficient. The parameters selection procedure is performed in a one-factor-at-a-time manner. For each sensitivity analysis in this section, only one parameter is varied each time, and the remaining parameters are kept at the values suggested by the original estimate listed in Table 9. The interaction relation between parameters is assumed unimportant.

Each time one of the WPA parameters is varied in a certain interval to see which value within this internal will result in the best performance. Specifically, the WPA algorithm also runs 50 times on each case.

Table 5 shows the sensitivity analysis of the step coefficient S. All results are shown in the form of Mean ± Std (SR/%). The choice of interval $[0.04, 0.16]$ used in this analysis was motivated by the original Nelder-Mead simplex search procedure, where a step coefficient greater than 0.04 was suggested for general usage.

Meanwhile, based on detailed comparison of the results, on Rosenbrock, Sphere, and Bridge functions, step coefficient is not sensitive to WPA, and for Booth function there is a tendency of better results with larger S. From Table 5, it is found that a step coefficient setting at 0.12 returns the best result which has better Mean, small Std, and SR = 100% for all functions.

Tables 6–8 analyze sensitivity of L_{near}, $T_{\max}$, and β. Generally speaking, L_{near}, $T_{\max}$, and β are not sensitive to most functions except Griewank function, since Griewank not only

Table 5: Sensitivity analysis of step coefficient (S).

Functions	Mean ± Std (SR/%) (the default of SR is 100%)						
	0.04	0.06	0.08	0.10	0.12	0.14	0.16
Rosenbrock	$6.9e-8 \pm 4.3e-8$	$2.7e-8 \pm 3.5e-8$	$1.1e-8 \pm 9.1e-9$	$3.2e-9 \pm 2.7e-9$	$5.0e-9 \pm 5.7e-9$	$3.2e-9 \pm 3.7e-9$	$1.2e-9 \pm 1.6e-9$
Colville	$1.3e-7 \pm 7.1e-8$	$3.3e-7 \pm 2.8e-7$ (90)	$2.6e-7 \pm 1.9e-7$	$2.3e-7 \pm 1.4e-7$	$3.5e-7 \pm 2.5e-7$	$9.5e-7 \pm 1.0e-6$ (80)	$1.4e-6 \pm 1.5e-6$ (50)
Sphere	$2.3e-145 \pm 7.1e-145$	$6.6e-152 \pm 2.1e-151$	$2.1e-146 \pm 4.5e-146$	$3.9e-146 \pm 1.2e-145$	$1.2e-145 \pm 3.4e-145$	$1.7e-146 \pm 5.3e-146$	$2.2e-149 \pm 6.8e-149$
Sumsquares	$9.8e-145 \pm 3.1e-144$	$3.1e-146 \pm 8.4e-146$	$8.1e-147 \pm 2.6e-146$	$4.8e-146 \pm 1.0e-145$	$3.8e-152 \pm 7.9e-152$	$3.4e-147 \pm 1.1e-146$	$1.2e-147 \pm 3.9e-147$
Booth	$5.4e-7 \pm 3.3e-7$	$1.6e-9 \pm 1.1e-9$	$3.2e-11 \pm 1.6e-11$	$1.3e-12 \pm 9.1e-13$	$1.3e-13 \pm 1.2e-13$	$3.9e-15 \pm 1.8e-15$	$1.2e-16 \pm 5.8e-17$
Bridge	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$
Ackley	$8.9e-16 \pm 0$	0.25 ± 0.53 (80)	$1.2e-15 \pm 1.1e-15$	$8.9e-16 \pm 0$	$1.2e-15 \pm 1.1e-15$	$8.9e-16 \pm 0$	$8.9e-16 \pm 0$
Griewank	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0.06 ± 0.19 (92)	0.20 ± 0.42 (86)

Table 6: Sensitivity analysis of distance determinant coefficient (L_{near}).

Functions	Mean ± Std (SR/%) (the default of SR is 100%)						
	0.04	0.06	0.08	0.10	0.12	0.14	0.16
Rosenbrock	$4.4e-8 \pm 6.5e-8$	$2.3e-8 \pm 3.7e-8$	$3.4e-9 \pm 4.8e-9$	$3.0e-8 \pm 2.9e-8$	$1.9e-8 \pm 2.4e-8$	$2.4e-8 \pm 4.7e-8$	$2.9e-8 \pm 5.3e-8$
Colville	$2.0e-7 \pm 9.9e-8$	$2.6e-7 \pm 1.6e-7$	$3.5e-7 \pm 2.6e-7$	$2.3e-7 \pm 1.5e-7$	$1.2e-7 \pm 3.4e-8$	$2.8e-7 \pm 1.9e-7$	$1.4e-7 \pm 6.9e-8$
Sphere	$6.8e-146 \pm 2.0e-145$	$1.9e-146 \pm 6.2e-146$	$1.7e-145 \pm 4.3e-145$	$2.6e-148 \pm 8.3e-148$	$3.6e-146 \pm 1.1e-145$	$3.7e-151 \pm 1.1e-150$	$5.3e-149 \pm 1.7e-148$
Sumsquares	$1.1e-147 \pm 3.4e-147$	$1.0e-146 \pm 3.3e-146$	$3.7e-151 \pm 8.9e-151$	$6.2e-146 \pm 1.9e-145$	$6.2e-152 \pm 1.9e-151$	$1.22e-145 \pm 2.9e-145$	$1.3e-148 \pm 4.0e-148$
Booth	$2.6e-11 \pm 1.3e-11$	$2.9e-11 \pm 1.9e-11$	$2.4e-11 \pm 1.6e-11$	$3.1e-11 \pm 1.8e-011$	$2.4e-11 \pm 1.3e-11$	$3.1e-11 \pm 2.1e-11$	$1.0e-10 \pm 1.3e-10$
Bridge	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$
Ackley	0.14 ± 0.43 (90)	$1.2e-15 \pm 1.1e-15$	$8.9e-16 \pm 0$	$1.2e-15 \pm 1.1e-15$	$8.9e-16 \pm 0$	$1.59e-15 \pm 1.49e-15$	$8.9e-16 \pm 0$
Griewank	0.08 ± 0.26 (90)	$1.0e-3 \pm 0.02$ (96)	0 ± 0	0 ± 0	0 ± 0	0.10 ± 0.33 (92)	0 ± 0

Table 7: Sensitivity analysis of the maximum number of repetitions in scouting behavior (T_{max}).

Functions	Mean ± Std (SR/%) (the default of SR is 100%)						
	6	8	10	12	14	16	18
Rosenbrock	$2.4e-8 \pm 2.6e-8$	$8.4e-9 \pm 8.0e-9$	$1.3e-8 \pm 1.3e-8$	$1.4e-8 \pm 1.0e-8$	$2.0e-8 \pm 1.9e-8$	$2.1e-8 \pm 2.5e-8$	$1.2e-8 \pm 8.9e-9$
Colville	$4.8e-7 \pm 2.2e-7$	$3.4e-7 \pm 1.8e-7$	$1.5e-7 \pm 1.2e-7$	$3.8e-7 \pm 2.0e-7$	$3.6e-7 \pm 3.7e-7$ (96)	$3.4e-7 \pm 2.5e-7$	$2.6e-7 \pm 1.5e-7$
Sphere	$7.1e-147 \pm 2.2e-146$	$4.5e-146 \pm 9.0e-146$	$7.8e-146 \pm 2.3e-145$	$1.9e-148 \pm 5.3e-148$	$5.7e-148 \pm 1.3e-147$	$6.9e-145 \pm 2.2e-144$	$3.6e-147 \pm 1.1e-146$
Sumsquares	$4.1e-146 \pm 1.3e-145$	$2.4e-149 \pm 4.8e-149$	$4.2e-149 \pm 1.3e-148$	$8.3e-150 \pm 2.6e-149$	$8.5e-147 \pm 2.7e-146$	$5.4e-146 \pm 9.0e-146$	$1.4e-151 \pm 4.4e-151$
Booth	$3.2e-11 \pm 2.9e-11$	$4.2e-11 \pm 2.7e-11$	$2.5e-11 \pm 1.5e-11$	$2.1e-11 \pm 1.5e-11$	$3.2e-11 \pm 2.5e-11$	$2.6e-11 \pm 1.8e-11$	$2.6e-11 \pm 2.7e-11$
Bridge	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$
Ackley	$8.9e-16 \pm 0$	$8.9e-16 \pm 0$	$8.9e-16 \pm 0$	$8.9e-16 \pm 0$	$1.2e-15 \pm 1.1e-15$	$8.9e-16 \pm 0$	$8.9e-16 \pm 0$
Griewank	0.10 ± 0.33 (92)	0 ± 0	$1.0e-3 \pm 0.02$ (98)	0.09 ± 0.31 (88)	0 ± 0	0.09 ± 0.29 (94)	$8.3e-4 \pm 0.02$ (98)

Table 8: Sensitivity analysis of population renewing proportional coefficient (β).

Functions	Mean ± Std (SR/%) (the default of SR is 100%)						
	2	3	4	5	6	7	8
Rosenbrock	$1.0e-8 \pm 9.2e-9$	$8.7e-9 \pm 7.6e-9$	$1.2e-8 \pm 1.0e-8$	$8.6e-9 \pm 8.3e-9$	$1.4e-8 \pm 1.3e-8$	$9.9e-9 \pm 9.8e-9$	$1.1e-8 \pm 1.2e-9$
Colville	$3.2e-8 \pm .1.8e-8$	$1.4e-7 \pm 1.3e-7$	$1.2e-7 \pm 5.9e-8$	$1.4e-7 \pm 9.4e-8$	$3.0e-7 \pm 6.9e-8$	$3.9e-7 \pm 1.7e-7$	$8.6e-7 \pm 4.0e-7$ (80)
Sphere	$1.9e-166 \pm 0$	$5.2e-158 \pm 1.6e-157$	$2.9e-153 \pm 9.2e-153$	$4.3e-149 \pm 1.3e-148$	$7.9e-139 \pm 2.5e-138$	$8.3e-134 \pm 1.8e-133$	$3.4e-126 \pm 8.0e-126$
Sumsquares	$2.8e-167 \pm 0$	$1.4e-157 \pm 4.3e-157$	$2.8e-155 \pm 4.5e-155$	$8.3e-146 \pm 1.8e-145$	$6.9e-143 \pm 1.7e-142$	$5.3e-143 \pm 1.3e-142$	$3.3e-127 \pm 1.0e-126$
Booth	$8.1e-11 \pm 1.3e-10$	$2.5e-11 \pm 1.7e-11$	$1.9e-11 \pm 1.2e-11$	$2.5e-11 \pm 1.7e-011$	$2.5e-11 \pm 1.5e-11$	$2.3e-11 \pm 1.5e-11$	$2.3e-11 \pm 1.4e-11$
Bridge	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$	$3.0054 \pm 4.7e-16$
Ackley	$8.9e-16 \pm 0$	$8.9e-16 \pm 0$	$8.9e-16 \pm 0$	$8.9e-16 \pm 0$	$8.9e-16 \pm 0$	$8.9e-16 \pm 0$	$8.9e-16 \pm 0$
Griewank	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0.19 ± 0.41 (86)	0 ± 0	$1.2e-3 \pm 0.31$ (96)

Table 9: Best suggestions for WPA parameters.

No.	WPA parameters name	Original	Best-suggested
1	Step coefficient (S)	0.08	0.12
2	Distance determinant coefficient (L_{near})	0.12	0.08
3	The maximum number of repetitions in scouting (T_{max})	10	8
4	Population renewal coefficient (β)	5	2

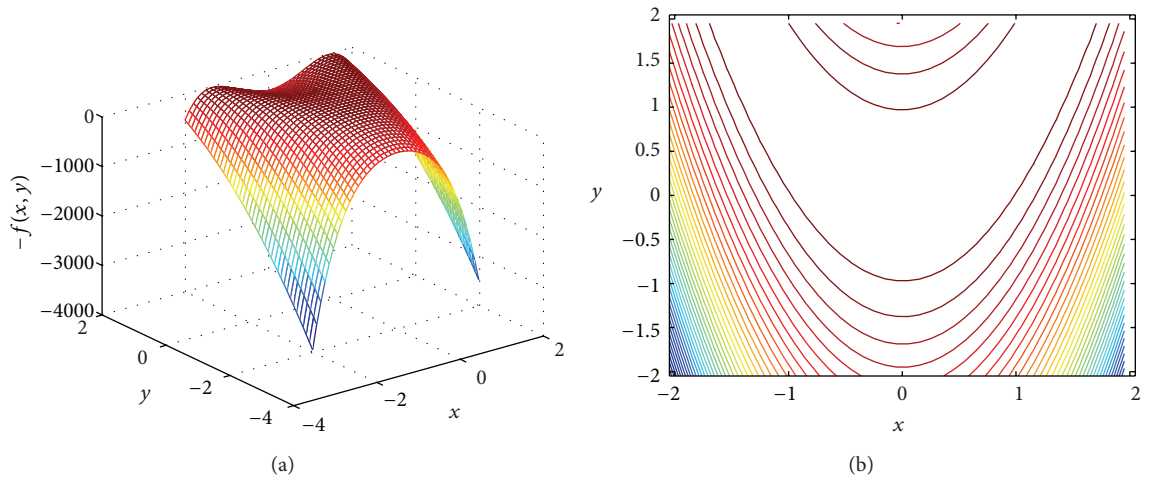


Figure 2: Rosenbrock function ($D = 2$): (a) surface plot and (b) contour lines.

is a high-dimensional function for its 100 parameters, but also has very large search space for its interval of $[-600, 600]$, which is hard to optimized.

Table 6 illustrates the sensitivity analysis of L_{near}, and from this table it is found that setting L_{near} at 0.08 returns the best results with the best mean, smaller standard deviations, and 100% success rate for all functions.

Tables 7-8 indicate that T_{max} and β, respectively, setting at 8 and 2 return the best results on eight functions.

So we summarize the above findings in Table 9 and apply these parameter values in our approach for conducting experimental comparisons with other algorithms listed in Table 2.

4.3. Experiments 2: WPA versus GA, PSO, AFSA, ABC, and FA. In this section, we compared GA, PSO, AFSA, ABC, FA, and WPA algorithms on eight functions described in Table 1. Each of the experiments was repeated for 50 runs with different random seeds and the best, worst, and mean values, standard deviations, success rates, and average reaching time are given in Table 10. The best results for each case are highlighted in boldface.

As can clearly be seen from Table 10, when solving the unimodal nonseparable problems (Rosenbrock, Colville), although the results of WPA are not good enough as FA or ASFA algorithm, WPA also achieves 100% success rate. Firstly, with respect to Rosenbrock function, its surface plot and contour lines are shown in Figure 2.

As seen in Figure 2, Rosenbrock function is well known for its Rosenbrock valley. Global minimum value for this function is 0 and optimum solution is $(x_1, x_2) = (1, 1)$. But the global optimum is inside a long, narrow, parabolic-shaped flat valley. Since it is difficult to converge to the global optimum of this function, the variables are strongly dependent, and the gradients generally do not point towards the optimum; this problem is repeatedly used to test the performance of the algorithms [17]. As shown in Table 10, PSO, AFSA, FA, and WPA achieve 100% success rate, and PSO shows the fastest convergence speed; AFSA gets the value $1.10e-13$ with the best accuracy. FA also shows good performance because of its robustness on Rosenbrock function.

On the Colville function, its surface plot and contour lines are shown in Figure 3. Colville function also has a narrow curving valley and it is hard to be optimized if the search space cannot be explored properly and the direction changes cannot be kept up with. Its global minimum value is 0 and optimum solution is $(x_1, x_2, x_3, x_4) = (1, 1, 1, 1)$.

Although the best accurate solution is obtained by AFSA, WPA outperforms the other algorithms in terms of the worst, mean, std., SR, and Art on Colville function.

Sphere and Sumsquares are convex, unimodal, and separable functions. They are all high-dimensional functions for their 200 and 150 parameters, respectively, and the global minima are all 0 and optimum solution is $(x_1, x_2, \ldots, x_m) = (0, 0, \ldots, 0)$. Surface plot and contour lines of them are, respectively, shown in Figures 4 and 5.

As seen from Table 10, when solving the unimodal separable problems, we note that WPA outperforms other five algorithms both on convergence speed and solution accuracy. In particular, WPA offers the highest accuracy and improves the precision by about 170 orders of magnitude on Sphere and

Table 10: Statistical results of 50 runs obtained by GA, PSO, AFSA, ABC, FA, and WPA algorithms.

Function	Global extremum	D	C	Algorithms	Best	Worst	Mean	StdDev	SR/%	Art/s
Rosenbrock	$f_{\min}(\vec{x}) = 0$	2	UN	GA	$1.78e-10$	0.0373	0.0091	0.0092	10	>759.8323
				PSO	$2.26e-11$	$5.89e-7$	$1.07e-7$	$1.30e-7$	**100**	**0.7444**
				AFSA	**1.10e − 13**	$1.11e-9$	$2.34e-10$	$2.62e-10$	**100**	2.0578
				ABC	$5.99e-6$	0.0099	$8.61e-4$	0.0015	0	>391.0297
				FA	$6.28e-13$	**6.29e − 10**	**1.86e − 10**	**1.62e − 10**	**100**	33.1256
				WPA	$3.49e-11$	$2.34e-8$	$5.09e-9$	$4.34e-9$	**100**	6.6333
Colville	$f_{\min}(\vec{x}) = 0$	4	UN	GA	0.0022	0.3343	0.1272	0.1062	0	$>1.22e+3$
				PSO	$1.29e-6$	$3.46e-4$	$5.06e-5$	$6.71e-5$	0	>114.0869
				AFSA	**3.66e − 8**	$8.91e-7$	$3.16e-7$	$2.32e-7$	**100**	40.1807
				ABC	0.0103	0.5337	0.1871	0.1232	0	>384.4193
				FA	$2.41e-7$	$3.69e-5$	$6.62e-6$	$8.07e-6$	8	$>3.14e+3$
				WPA	$4.71e-8$	**3.72e − 7**	**1.25e − 7**	**6.97e − 8**	**100**	**27.4054**
Sphere	$f_{\min}(\vec{x}) = 0$	200	US	GA	$1.56e+5$	$1.81e+5$	$1.71e+5$	$5.78e+3$	0	$>4.44e+4$
				PSO	1.0361	1.5520	1.2883	0.1206	0	>271.9201
				AFSA	$5.12e+5$	$5.79e+5$	$5.51e+5$	$1.63e+4$	0	$>7.41e+3$
				ABC	0.0041	1.2521	0.0444	0.1773	0	>442.9045
				FA	0.1432	0.2327	0.1865	0.0199	0	$>8.34e+3$
				WPA	**1.49e − 172**	**2.41e − 165**	**1.56e − 166**	**0**	**100**	**6.1729**
Sumsquares	$f_{\min}(\vec{x}) = 0$	150	US	GA	$5.93e+4$	7.15 + 4	$6.63e+4$	$2.88e+3$	0	$>3.16e+4$
				PSO	39.7098	91.1145	55.9050	10.4165	0	>232.5464
				AFSA	$1.43e+5$	$1.79e+5$	$1.64e+5$	$9.58e+3$	0	$>7.36e+3$
				ABC	$1.71e-5$	0.0017	$1.99e-4$	$3.36e-4$	0	>435.1848
				FA	8.9920	99.8861	40.5721	19.2743	0	$>6.88e+3$
				WPA	**2.68e − 172**	**5.47e − 166**	**2.62e − 167**	**0**	**100**	**6.5954**
Booth	$f_{\min}(\vec{x}) = 0$	2	MS	GA	$4.55e-11$	$4.55e-11$	$4.55e-11$	0	**100**	1.2621
				PSO	$1.22e-12$	$2.41e-8$	$2.80e-9$	$4.52e-9$	**100**	**0.2079**
				AFSA	$3.02e-12$	$1.45e-9$	$4.61e-10$	$4.08e-10$	**100**	4.4329
				ABC	**6.05e − 20**	**1.41e − 17**	**4.63e − 18**	**4.14e − 18**	**100**	0.4175
				FA	$1.80e-12$	$4.39e-9$	$1.18e-9$	$1.11e-9$	**100**	37.9191
				WPA	$8.22e-15$	$7.05e-13$	$1.21e-13$	$1.19e-13$	**100**	6.9339
Bridge	$f_{\max}(\vec{x}) = 3.0054$	2	MN	GA	**3.0054**	**3.0054**	**3.0054**	$1.35e-15$	**100**	0.1927
				PSO	**3.0054**	**3.0054**	**3.0054**	$4.84e-8$	**100**	**0.0929**
				AFSA	**3.0054**	3.0047	3.0052	$1.69e-4$	12	$>8.01e+3$
				ABC	**3.0054**	**3.0054**	**3.0054**	$3.59e-15$	**100**	0.0932
				FA	**3.0054**	**3.0054**	**3.0054**	$3.11e-10$	**100**	22.7230
				WPA	**3.0054**	**3.0054**	**3.0054**	**3.58e − 15**	**100**	0.1742
Ackley	$f_{\min}(\vec{x}) = 0$	50	MN	GA	11.4570	12.6095	12.1612	0.2719	0	$>1.04e+4$
				PSO	0.0469	1.7401	0.6846	0.6344	0	>192.5522
				AFSA	20.1600	20.6009	20.4229	0.1009	0	$>9.80e+3$
				ABC	20.0085	20.0025	20.0061	0.0014	0	>596.3841
				FA	0.0101	0.0209	0.0160	0.0021	0	$>4.28e+3$
				WPA	**8.88e − 16**	**4.44e − 15**	**1.10e − 15**	**8.52e − 16**	**100**	**7.9476**

Table 10: Continued.

Function	Global extremum	D	C	Algorithms	Best	Worst	Mean	StdDev	SR/%	Art/s
Griewank	$f_{\min}(\vec{x}) = 0$	100	MN	GA	317.4525	399.6376	363.4174	17.2922	0	$>2.07e+4$
				PSO	0.0029	0.0082	0.0052	0.0011	0	>367.0080
				AFSA	$2.05e+3$	$2.55e+3$	$2.33e+3$	109.6821	0	$>6.51e+3$
				ABC	$8.95e-7$	0.0043	$2.26e-4$	$7.81e-4$	2	>620.9561
				FA	0.0068	0.0118	0.0091	0.0011	0	$>5.72e+3$
				WPA	**0**	**0**	**0**	**0**	**100**	**14.5338**

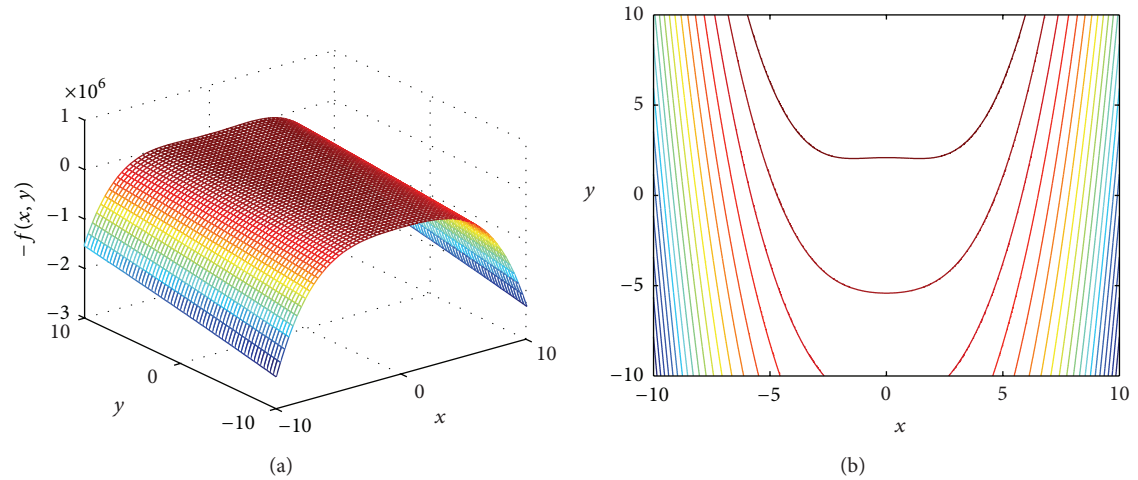


Figure 3: Colville function ($x_1 = x_3$, $x_2 = x_4$): (a) surface plot and (b) contour lines.

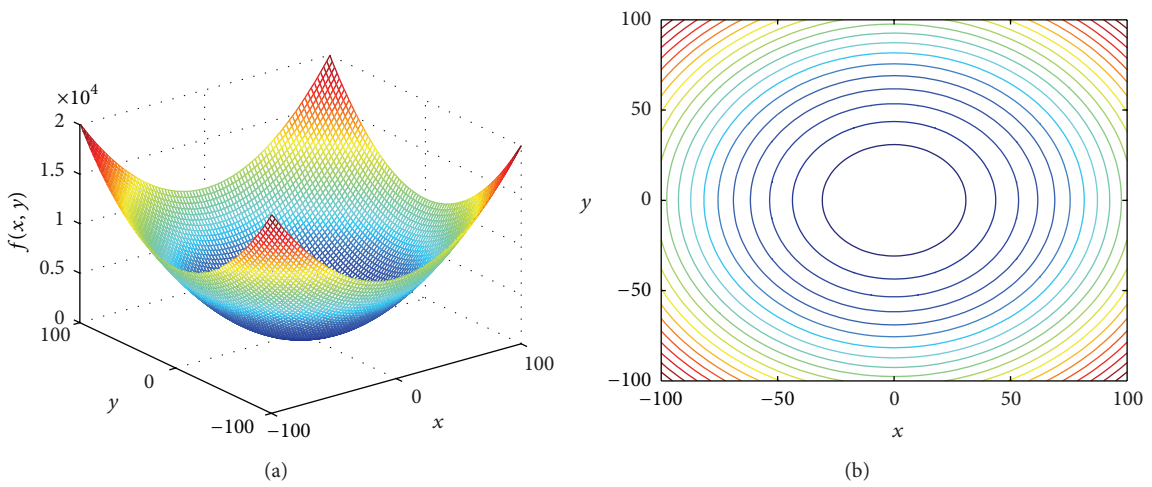


Figure 4: Sphere function ($D = 2$): (a) surface plot and (b) contour lines.

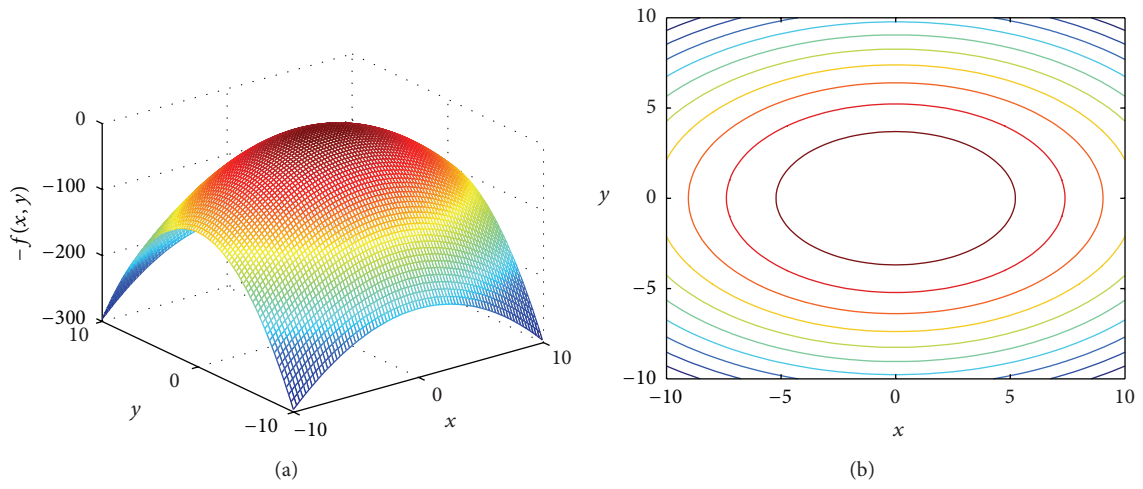


Figure 5: Sumsquares function ($D = 2$): (a) surface plot and (b) contour lines.

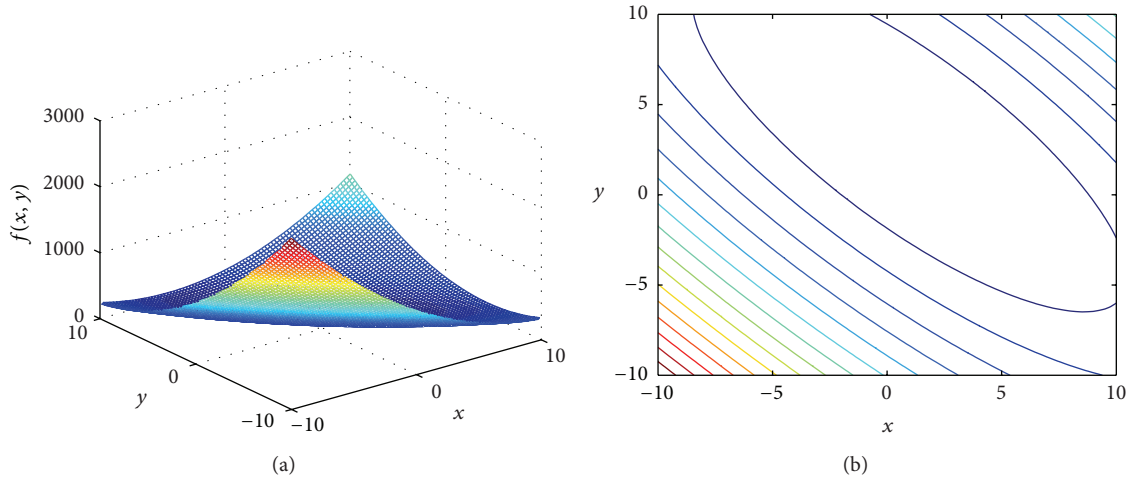


FIGURE 6: Booth function ($D = 2$): (a) surface plot and (b) contour lines.

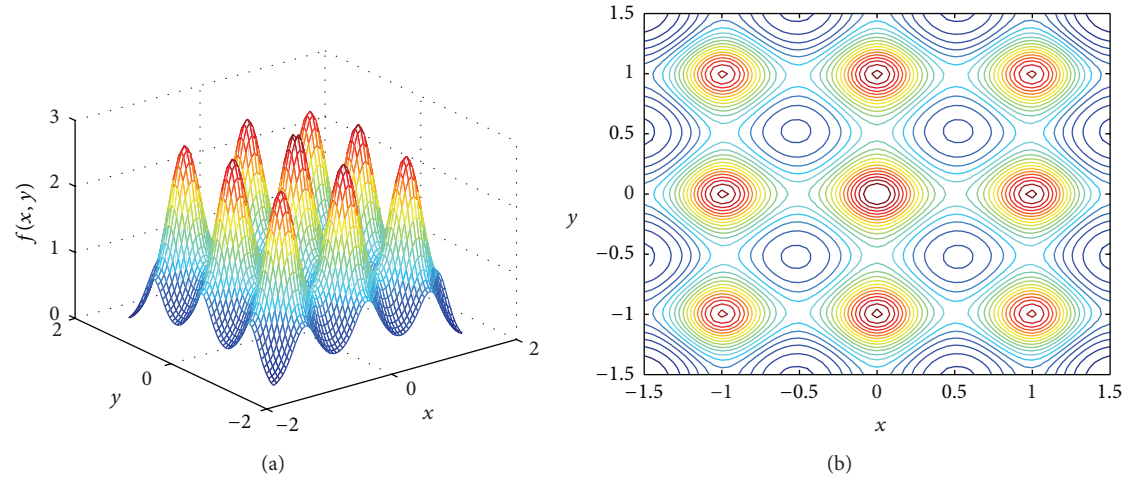


FIGURE 7: Bridge function ($D = 2$): (a) surface plot and (b) contour lines.

Sumsquares functions, when compared with the best results of the other algorithms.

Booth is a multimodal and separable function. Its global minimum value is 0 and optimum solution is $(x_1, x_2) = (1, 3)$. When handing Booth function, ABC can get the closer-to-optimal solution within shorter time. Surface plot and contour lines of Booth are shown in Figure 6.

As shown in Figure 6, Booth function has flat surfaces and is difficult for algorithms since the flatness of the function does not give the algorithm any information to direct the search process towards the minima. So WPA does not get the best value as good as ABC, but it can also find good solution and achieve 100% success rate.

Bridge and Ackley are multimodal and nonseparable functions. The global maximum value of Bridge function is 3.0054 and optimum solution is $(x_1, x_2) \to (0, 0)$. The global minimum value of Ackley function is 0 and optimum solution is $(x_1, x_2, \ldots, x_m) = (0, 0, \ldots, 0)$. Surface plot and contour lines of them are separately shown in Figures 7 and 8.

As seen in Figures 7 and 8, the locations of the extremum are regularly distributed, and there are many local extremums near the global extremum. The difficult part of finding optima is that algorithms may easily be trapped in local optima on their way towards the global optimum or oscillate between these local extremums. From Table 10, all algorithms except ASFA show equal performance and achieve 100% success rate on Bridge function. While with respect to Ackley ($D = 50$), only WPA achieves 100% success rate and improves the precision by 13 or 15 orders of magnitude when compared with the best results of other algorithms.

Otherwise, the dimensionality and size of the search space are important issues in the problem [18]. Griewank function, an multimodal and nonseparable function, has the global minimum value of 0 and its corresponding global optimum solution is $(x_1, x_2, \ldots, x_m) = (0, 0, \ldots, 0)$. Moreover, the increment in the dimension of function increases the difficulty. Since the number of local optima increases with the dimensionality, the function is strongly multimodal. Surface

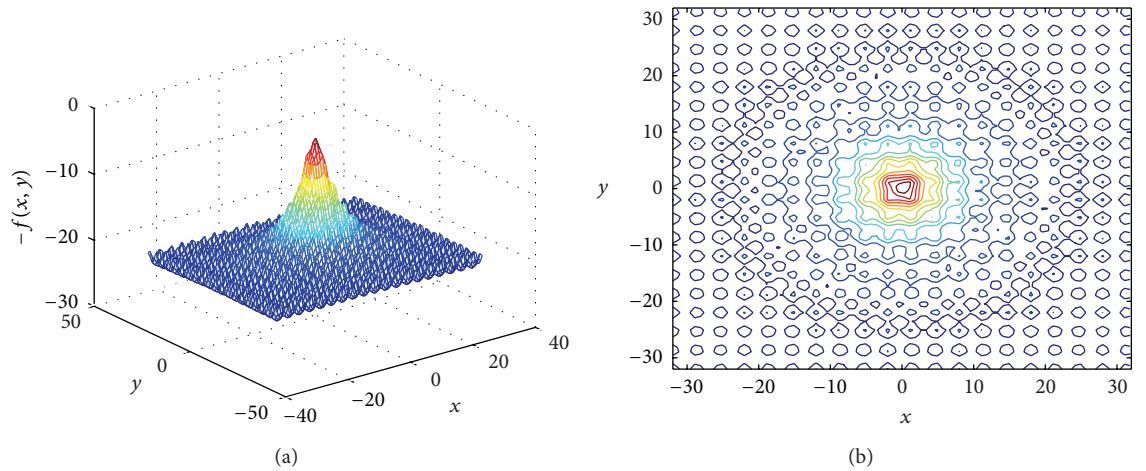


Figure 8: Ackley function ($D = 2$): (a) surface plot and (b) contour lines.

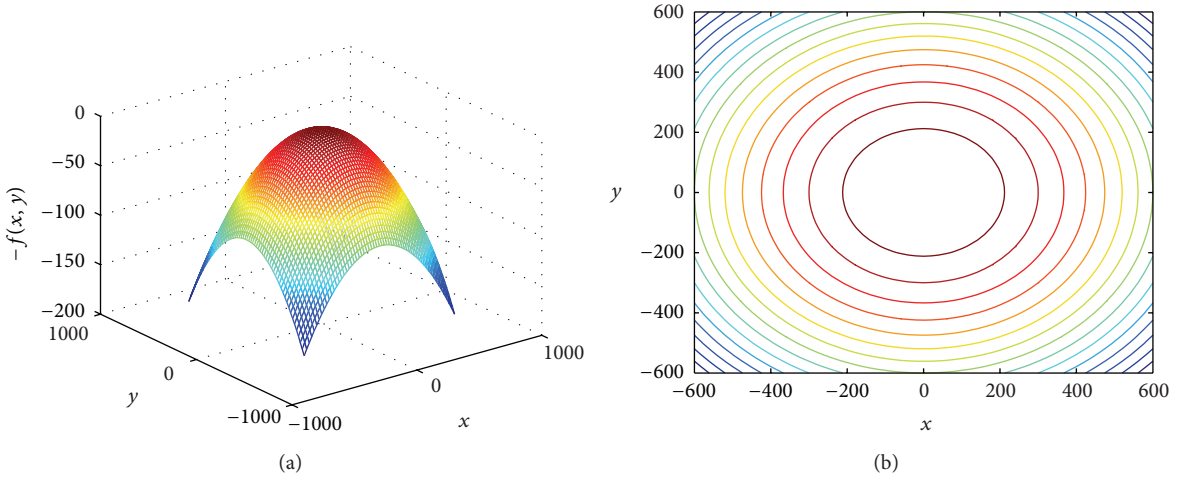


Figure 9: Griewank function ($D = 2$): (a) surface plot and (b) contour lines.

plot and contour lines of Griewank function are shown in Figure 9.

WPA with optimized coefficients has good performance in high-dimensional functions. Griewank function ($D = 100$) is a good example. In such a great search space, as shown in Table 10, other algorithms present serious flaws such as premature convergence and difficulty to overcome local minima, while WPA successfully gets the global optimum 0 in 50 runs computation.

As is shown in Table 10, SR shows the robustness of every algorithm, and it means how consistently the algorithm achieves the threshold during all runs performed in the experiments. WPA achieves 100% success rate for functions with different characteristics, which shows its good robustness.

In the experiments, there are 8 functions with variables ranging from 2 to 200. WPA statistically outperforms GA on 6, PSO on 5, ASFA on 6, ABC on 6, and FA on 7 of these 8 functions. Six of the functions on which GA and ABC are unsuccessful are two unimodal nonseparable functions (Rosenbrock and Colville) and four high-dimensional functions (Sphere, Sumsquares, Ackley, and Griewank). PSO and FA are unsuccessful on 1 unimodal nonseparable function and four high-dimensional functions. But WPA is also not perfect enough for all functions; there are many problems that need to be solved for this new algorithm. From Table 10, on the Rosenbrock function, the accuracy and convergence speed obtained by WPA are not the best ones. So ameliorating WPA inspired by intelligent behaviors of wolves for these special problems is one of our future works. However, so far, it seems to be difficult to simultaneously achieve both fast convergence speed and avoiding local optima for every complex function [19].

It can be drawn that the efficiency of WPA becomes much clearer as the number of variables increases. WPA performs statistically better than the five other state-of-the-art algorithms on high-dimensional functions. Nowadays, high-dimensional problems have been a focus in evolutionary computing domain, since many recent real-world problems (biocomputing, data mining, design, etc.) involve

optimization of a large number of variables [20]. It is convincing that WPA has extensive application in science research and engineering practices.

5. Conclusions

Inspired by the intelligent behaviors of wolves, a new swarm intelligent optimization method, wolf pack algorithm (WPA), is presented for locating the global optima of continuous unconstrained optimization problems. We testify the performance of WPA on a suite of benchmark functions with different characteristics and analyze the effect of distance measurements and parameters on WPA. Compared with PSO, ASFA, GA, ABC, and FA, WPA is observed to perform equally or potentially more powerful. Especially for high-dimensional functions such as Sphere ($D = 200$), Sumsquares ($D = 150$), Ackley ($D = 50$), and Griewank ($D = 100$), WPA may be a better choice, since WPA possesses superior performance in terms of accuracy, convergence speed, stability, and robustness.

After all, WPA is a new attempt and achieves some success for global optimization, which can provide new ideas for solving engineering and science optimization problems. In future, different improvements can be made on the WPA algorithm and tests can be made on more different test functions. Meanwhile, practical applications in areas of classification, parameters optimization, engineering process control, and design and optimization of controller would also be worth further studying.

Conflict of Interests

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

References

[1] F. Kang, J. Li, and Z. Ma, "Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions," *Information Sciences*, vol. 181, no. 16, pp. 3508–3531, 2011.

[2] C. Grosan and A. Abraham, "A novel global optimization technique for high dimensional functions," *International Journal of Intelligent Systems*, vol. 24, no. 4, pp. 421–440, 2009.

[3] Y. Yang, Y. Wang, X. Yuan, and F. Yin, "Hybrid chaos optimization algorithm with artificial emotion," *Applied Mathematics and Computation*, vol. 218, no. 11, pp. 6585–6611, 2012.

[4] W. S. Gao and C. Shao, "Pseudo-collision in swarm optimization algorithm and solution: rain forest algorithm," *Acta Physica Sinica*, vol. 62, no. 19, Article ID 190202, pp. 1–15, 2013.

[5] Y. Celik and E. Ulker, "An improved marriage in honey bees optimization algorithm for single objective unconstrained optimization," *The Scientific World Journal*, vol. 2013, Article ID 370172, 11 pages, 2013.

[6] E. Cuevas., D. Zaldívar, and M. Pérez-Cisneros, "A swarm optimization algorithm for multimodal functions and its application in multicircle detection," *Mathematical Problems in Engineering*, vol. 2013, Article ID 948303, 22 pages, 2013.

[7] 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.

[8] M. Dorigo, *Optimization, learning and natural algorithms [Ph.D. thesis]*, Politecnico di Milano, Milano, Italy, 1992.

[9] X.-L. Li, Z.-J. Shao, and J.-X. Qian, "Optimizing method based on autonomous animats: Fish-swarm Algorithm," *System Engineering Theory and Practice*, vol. 22, no. 11, pp. 32–38, 2002.

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

[11] X.-S. Yang, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications*, vol. 5792 of *Lecture Notes in Computer Science*, pp. 169–178, Springer, Berlin, Germany, 2009.

[12] J. A. Ruiz-Vanoye, O. Díaz-Parra, F. Cocón et al., "Meta-Heuristics algorithms based on the grouping of animals by social behavior for the travelling sales problems," *International Journal of Combinatorial Optimization Problems and Informatics*, vol. 3, no. 3, pp. 104–123, 2012.

[13] C.-G. Liu, X.-H. Yan, and C.-Y. Liu, "The wolf colony algorithm and its application," *Chinese Journal of Electronics*, vol. 20, no. 2, pp. 212–216, 2011.

[14] D. E. Goldberg, *Genetic Algorithms in Search, Optimisation and Machine Learning*, Addison-Wesley, Reading, Mass, USA, 1989.

[15] S.-K. S. Fan and E. Zahara, "A hybrid simplex search and particle swarm optimization for unconstrained optimization," *European Journal of Operational Research*, vol. 181, no. 2, pp. 527–548, 2007.

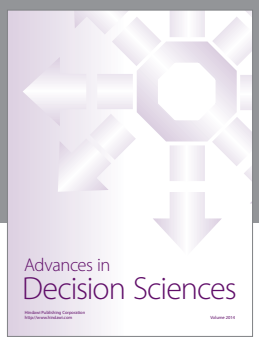
[16] P. Caamaño, F. Bellas, J. A. Becerra, and R. J. Duro, "Evolutionary algorithm characterization in real parameter optimization problems," *Applied Soft Computing*, vol. 13, no. 4, pp. 1902–1921, 2013.

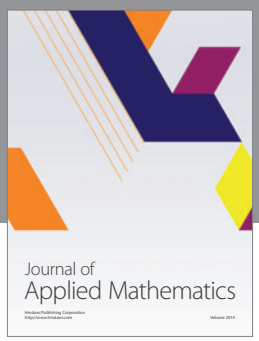
[17] D. Ortiz-Boyer, C. Hervás-Martínez, and N. García-Pedrajas, "CIXL2: a crossover operator for evolutionary algorithms based on population features," *Journal of Artificial Intelligence Research*, vol. 24, pp. 1–48, 2005.

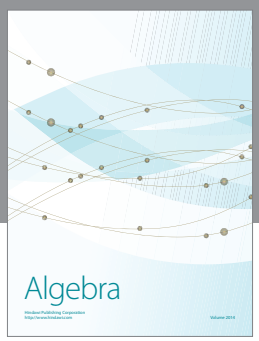
[18] M. S. Kıran and M. Gündüz, "A recombination-based hybridization of particle swarm optimization and artificial bee colony algorithm for continuous optimization problems," *Applied Soft Computing*, vol. 13, no. 4, pp. 2188–2203, 2013.

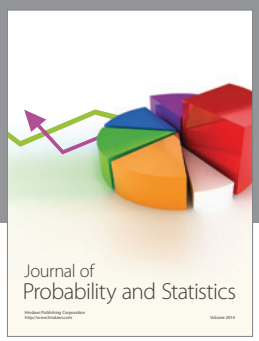
[19] W. Gao and S. Liu, "Improved artificial bee colony algorithm for global optimization," *Information Processing Letters*, vol. 111, no. 17, pp. 871–882, 2011.

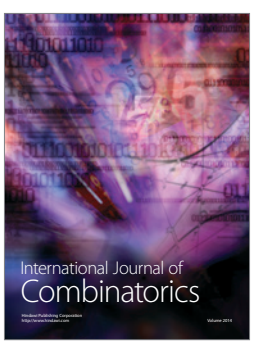
[20] Y. F. Ren and Y. Wu, "An efficient algorithm for high-dimensional function optimization," *Soft Computing*, vol. 17, no. 6, pp. 995–1004, 2013.

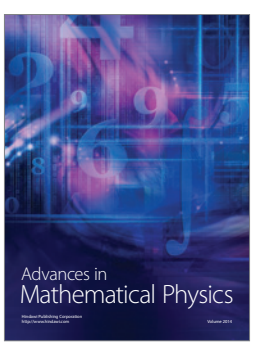


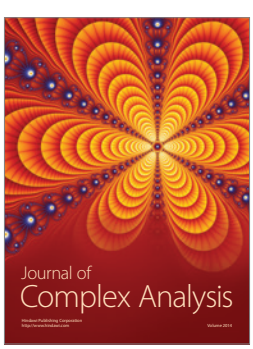


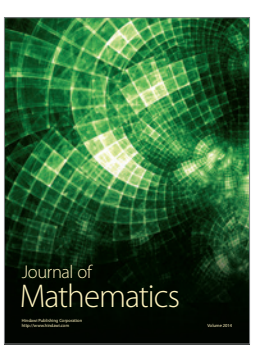


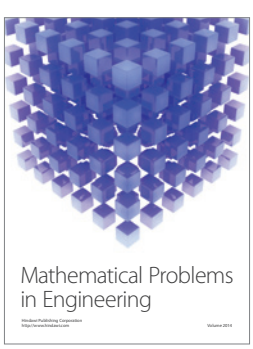


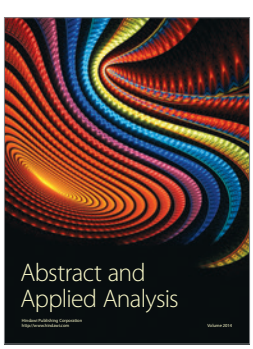


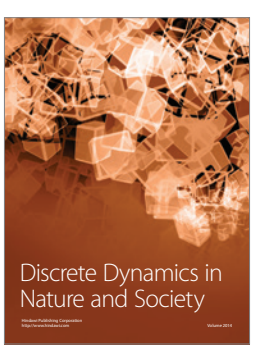


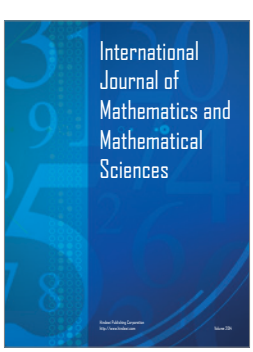












Hindawi

Submit your manuscripts at
http://www.hindawi.com