IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received March 9, 2019, accepted April 5, 2019, date of publication April 23, 2019, date of current version May 1, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2911629

Crow Search Algorithm Based on Neighborhood Search of Non-Inferior Solution Set

CHIWEN QU[1] AND YANMING FU[2]
[1]School of Information Engineering, Baise University, Baise 533000, China
[2]Computer and Electronic Information College, Guangxi University, Nanning 530004, China

Corresponding authors: Chiwen Qu (quchiwen@163.com) and Yanming Fu (fym2005@126.com)

This work was supported in part by the Natural Science Foundation of Guangxi Province under Grant 2014GXNSFBA118283, and in part by the Ability Enhancement Project of Young Teachers in Guangxi Universities under Grant 2018KY0579.

ABSTRACT Characterized by less parameter settings, easy implementation, and strong optimization capacity, the crow search algorithm has been successfully applied to solve the optimization problem. As the basic crow search algorithm is a new kind of swarm intelligent algorithm only based on the crow's memory foraging mode, it also contains defects like slow search speed and low optimization precision in later iterations, which are especially obvious for the optimization of high-dimensional functions. In order to overcome these shortcomings, a new crow search algorithm based on neighborhood search of non-inferior solution set (NICSA) is proposed. The proposed algorithm makes the crow individual choose the memory search mode or neighborhood search mode automatically in the course of evolution by the determination factor of non-inferior solution. With this strategy, the local exploitation and the global exploration of the algorithm became more balanced. In the neighborhood search, the selectivity factor is used to guide non-inferior solutions to adaptively execute neighborhood search of Levy flight or Gaussian flight, to enhance the neighborhood searchability of the algorithm and improve the optimization precision. The result of simulation experiments with 23 benchmark test functions verifies that the proposed algorithm has good optimization effect in the aspects of search veracity, convergence rate, and robustness.

INDEX TERMS Crow search algorithm, determination factor of non-inferior solution, neighborhood search, selectivity factor.

I. INTRODUCTION

In the fields of scientific research and engineering practice, most problems encountered by people boil down to the issue of solution optimization. As a computational process to find the optimal solution according to the features and requirements of the problems, it has always been the hot research topic. The common optimization algorithms for solving the optimization problem include traditional exact solution method, structural algorithm and swarm intelligent algorithm. The traditional exact solution method [1], [2] can get the exact solution of the problem, but it is only suitable for the fields of small-scale problem due to its complexity. Meanwhile, it requires the problem to be continuous and differentiable when solving the optimization problem, and lacks global optimization ability for multimodal, strong-nonlinearity and dynamically changing problems [3].

The associate editor coordinating the review of this manuscript and approving it for publication was Bora Onat.

The structural algorithm [4] can acquire the solution rapidly, but it has poor solution quality and can hardly meet the actual engineering requirements. The swarm intelligent algorithms are the global optimization methods designed through simulating the cooperative behavior mechanism among gregarious biological individuals in natural world. Such method can search the optimum solution or quasi-optimal solution of complicated optimization problems faster than traditional exact methods. Compared with traditional optimization methods, the swarm intelligent algorithm has simple principles, few adjustment parameters and relatively strong global optimization ability, and does not need gradient information of the problems. At present, common swarm intelligent algorithms include particle swarm optimization [5], differential evolution algorithm [6], artificial bee colony algorithm [7], cuckoo algorithm [8], flower pollination algorithm [9], krill swarm optimization algorithm [10], [11], and sine & cosine search algorithm [12]–[14]. Moreover, it is extensively applied to the engineering fields of function optimization [15]–[17], combination optimization [18], flow shop scheduling [19]–[21] and image processing. Therefore, it has practical and application significance to get the optimum solution through the swarm intelligent algorithm.

The crow search algorithm (CSA) is a new swarm intelligent algorithm proposed by Askarzadeh [22] according to the crow's intelligent behaviors in 2016. Characterized by easy implementation, less parameter setting and relatively strong development capacity in the searching process, it has already been successfully applied to solve the problems of function optimization and engineering application fields, and gained a very good effect [22]–[26]. Askarzadeh [22] studied the crow's superhigh memory ability when seeking and hiding food, and designed the crow search algorithm. According to the results of solving 6 constrained engineering design problems, the crow search algorithm has better solution results than genetic algorithm and particle swarm optimization. In literature [23], the crow search algorithm was used to solve the optimum size and position of capacitors in power distribution network. From the experimental results, CSA possesses more accurate solutions than other search methods. Oliva et al. [24] utilized the crow search algorithm to optimize the threshold value in image segmentation technology. Compared with other separation optimization techniques, CSA can avoid premature convergence of sub-optimal solutions, and obtain an excellent effect in complicated MR image automatic segmentation. Aleem et al. [25] optimized the design of resonance damping capacity in filters through the crow search algorithm. Compared with the genetic algorithm, this algorithm presents a higher convergence rate and can effectively solve the optimal design problem of third-order passive filters in power distribution network. Besides, Liu et al. [26] optimized the input weight of extreme learning machine (ELM) and threshold value of hidden layer neuron with the crow search algorithm, and proposed a groundwater quality evaluation model (CSA-ELM) of extreme learning machine (ELM) based on crow search algorithm. In comprehensive evaluation of underground water quality, the evaluated precision and generalization ability of CSA-ELM model reach a very high level. However, the primary crow search algorithm is a swarm intelligent search algorithm established according to the crow's life habit of searching food via its memory ability, so this algorithm has defects shown by other swarm intelligent algorithms including low search precision, high possibility of getting into local optimum, and premature convergence, especially for multi-dimensional optimization problems.

In order to overcome the defects of primary crow search algorithm, many scholars proposed various improvement strategies [27]–[35]. Jain et al. [27] introduced experience factor to balance the exploration and development abilities of the algorithm, promoted the algorithm to conduct global search in the whole solution space with the Levy flight mode, prevented the algorithm to fall into the local optimum, and avoided premature convergence. Through verification with high-dimensional nonlinear benchmark function, the improved CSA has very strong competitiveness and is not sensitive to the dimensions. Sayed et al. [28] proposed a sine chaotic crow search algorithm, and applied this improved algorithm to solve feature selection problems. This algorithm has greatly improved the classification performance and reduced the number of characteristic values selected. Dos et al. [29] adjusted the control parameters via diversity information and Gaussian distribution, and verified the effectiveness of improved CSA with benchmark problems of solenoid. Mohammadi and Abdi [30] proposed an improved crow search algorithm by introducing the new crow tracking target and adaptive adjustment of flight length, and used it to solve economic load dispatch problems. Díaz et al. [31] improved the awareness probability (AP) of primary crow search algorithm and the method of producing new solutions with stochastic disturbance. The proposed algorithm has maintained population diversity, and improved the convergence rate of solving complicated multimodal optimization problems. Gupta et al. [32] introduced an availability feature extraction algorithm based on hierarchical model, so as to extract and predict the availability feature. In literature [33]–[35], the chaos theory was utilized to improve the primary crow search algorithm, and used it to solve fractional optimization problems, multi-objective optimization problems and Parkinson's disease prediction. The above measures have improved the optimization performance of the crow search algorithm to some extent, but all improvement work focuses on standard CSA based on single memory search mode. They do not consider other search behaviors in the crow's intelligent behaviors. When solving the complicated and high-dimensional problems, they still have defects like slow convergence rate, low solution accuracy, and insufficient robustness.

In fact, except a few varieties including collared crow, other crows in natural world are the gregarious animal of high gregariousness. With an extremely high intelligence quotient, a crow individual has creative thinking ability [51]. For instance, the crow will fly down when the ice fisherman in Sweden leaves, and it will eat the fish or bait by drawing back the fishline. Under the enlightenment of the crow's foraging mode, a new crow search algorithm based on neighborhood search of non-inferior solution was proposed. According to the determination factor of non-inferior solution set, the crow individual will automatically choose the foraging search mode. The comparison between the crow's fitness value and the current global optimum, the neighborhood search mode of Levy flight or Gaussian flight is selected self-adaptively, to strengthen the neighborhood search ability of the algorithm. Compared with other swarm intelligent algorithms, the proposed algorithm has better performance in search veracity, convergence rate and stability.

II. ANALYSIS ON THE PRIMARY CROW SEARCH ALGORITHM AND ITS DEFECTS

A. PRIMARY CROW SEARCH ALGORITHM

As a bird living a gregarious life, the crow has very intelligent foraging behaviors: (i) hide superfluous food (the hiding place of food is called memory location), which can be taken out if necessary; (ii) follow other crows to take their food, while other crows will prevent their food from being stolen with a certain awareness probability(AP). Suppose that n crows are distributed in a D-dimensional space at random, and x_i^t is the location of crow i in the tth iteration. The update mode of individuals is as follows:

$$x_i^{t+1} = \begin{cases} x_i^t + r_1 {}^* fl_i^t {}^* (mem_j^t - x_i^t), & r \geq AP_j^t \\ random\ position, & other \end{cases} \tag{1}$$

where r_1 and r obey uniform distribution between 0 and 1, mem_j^t means the memory location of crow j in the tth iteration, AP_j^t denotes the awareness probability of crow j in the tth iteration, and fl_i^t indicates the flying range crow i in the tth iteration.

B. ANALYSIS ON THE DEFECTS OF PRIMARY CROW SEARCH ALGORITHM

As a swarm intelligent algorithm, the crow search algorithm has relatively strong global search capability. The primary crow search algorithm conduct search simply according to Eq. (1), it has limitations as following:

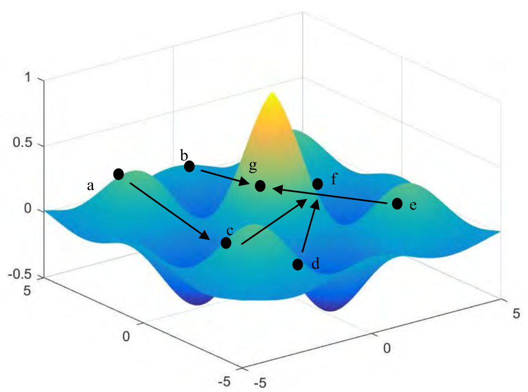

FIGURE 1. Schematic diagram for the individual crow's movement.

(i) In the search process, the crow individual will start random search in the sector area centering on the current position. The sector area is constructed by the other individual's historical optimum position (memory location), current position, and their difference value. Owing to such search method of single mode, the crow's flight activity lacks motility and flexibility. Figure 1 is the search schematic diagram of the basic crow search algorithm for the case of finding the maximum. Because the basic crow search algorithm does not have the ability to search non-inferior solutions, it is unable to search other possible global optimum regions (e.g. g) effectively, which leads to easy falling into local optimum. The probability that the algorithm can search for better values will be greatly increased if the local search can be carried out in the field of individuals whose memory values are similar to the current optimal memory values.

(ii) The primary CSA enhances the algorithm diversity by producing new random solutions in probability, but ignores the fine gene of advantageous positions in the learning community. Therefore, the algorithm has defects shown by other swarm intelligent algorithms including low search veracity, high possibility of getting into local optimum, and early-maturing, especially for multi-dimensional optimization problems.

III. CROW SEARCH ALGORITHM BASED ON NEIGHBORHOOD SEARCH OF NON-INFERIOR SOLUTION SET

According to the above analysis, the basic crow search algorithm has relatively strong global search capability, but its local search capability is not good. In order to balance the local exploitation and the global exploration capability of the algorithm, an independent global search and local search mode should be designed. As you can see in figure 1, if the crow individual conducts neighborhood search near the current optimum individual position, the algorithm's probability of finding a better value will be increased greatly. Under the inspiration of this idea, this algorithm makes the crow individual choose the memory search mode or neighborhood search mode automatically in the evolution process by introducing the determination factor of non-inferior solution. The principle of the proposed algorithm is as followings: When the difference between the optimum value searched by a crow and the current global best value in the population is small, the solution searched by this crow will be the non-inferior solution. As there might be better solutions in the field, this crow will carry out neighborhood search around, rather than executing the mode of memory search. Neighborhood search of Levy flight is executed for the crow individuals far away from the global optimum, to enhance the global exploration of the algorithm. Neighborhood search of Gaussian flight is executed for the crow individuals bear the global optimum, so as to enhance the local exploitation capability of the algorithm. This mechanism possesses mutability when searching the nearby areas in detail, so it can fully traverse all solution areas.

A. SELF-ADAPTIVE DETERMINATION FACTOR OF NON-INFERIOR SOLUTION SET

In the primary crow search algorithm, the crow will execute memory search mode or stochastic search mode with a fixed awareness probability (AP). However, this search strategy does not consider features of the current crow, and the search process has a certain blindness. Neighborhood search should be conducted for superior individuals with a relatively high probability, so as to acquire a better solution. The search mode of fixed probability will miss a comparatively excellent solution in the current field. The primary crow search algorithm prevents the food from being stolen in a random way. Though the early-maturing risk is increased, but the convergence rate is lowered. In order to realize full neighborhood search around non-inferior solutions, we designed a self-adaptive judgment method of non-inferior solutions. The self-adaptive judgment method of non-inferior solutions is as follows:

$$fit^t(mem_i) - fit^t(mem_{best}) > \omega \cdot |fit^t(mem_{avg}) - fit^t(mem_{best})| \quad (2)$$

where $fit^t(mem_i)$, $fit^t(mem_{best})$ and $fit^t(mem_{avg})$ are the memory value of crow i in the tth iteration, the memory value about the best position in the population and the average memory value respectively. ω means the coordinating parameter used to adjust the scale of non-inferior solution set.

Neighborhood search of non-inferior solutions in the population can help to excavate possible better solutions around the non-inferior solutions, and to improve the search veracity of the algorithm. But if too many non-inferior solutions are chosen, the algorithm might fall into local optimum easily and show insufficient global development ability. In this paper, a self-adaptive non-inferior solution adjustment parameter ω was designed, and its definition is as follows:

$$\omega = \log_{0.5}(t/Titer) \quad (3)$$

where t and $Titer$ are the current number of iterations and maximum number of iterations. The value of parameter ω decreases with the increase of iterations, and the corresponding number of non-inferior solution sets rises. In early iterations, most crow individuals execute the mode of memory search, so as to enhance the global search capability. In later iterations, more and more crows will execute neighborhood search of non-inferior solutions, so as to improve the local search capability and increase the search veracity of the algorithm.

B. SELF-ADAPTIVE NEIGHBORHOOD SEARCH STRATEGY GUIDED BY SELECTIVITY FACTOR

In order to give the algorithm sufficient ergodicity and mutability during neighborhood search of non-inferior solution set, a neighborhood search strategy of dynamic self-adaptive selection is designed in this paper.

1) NEIGHBORHOOD SEARCH MODE OF LEVY FLIGHT

Levy flight originates from the French statistician Paul Lévy, and many colonial organisms' life styles can be described with Levy flight. Levy flight is often reflected as the combination of long-term short-distance wander and occasional long-distance jump. Such long-distance jump is characterized by directional variability. The neighborhood search mode based on Levy flight can not only guarantee sufficient ergodicity near non-inferior solutions, but also has mutation performance, which will help to balance the local development capacity and global exploration capability of the algorithm. The neighborhood search mode is as follows:

$$x_i^{t+1} = mem_i^t + levy(D).^*mem_i^t \quad (4)$$

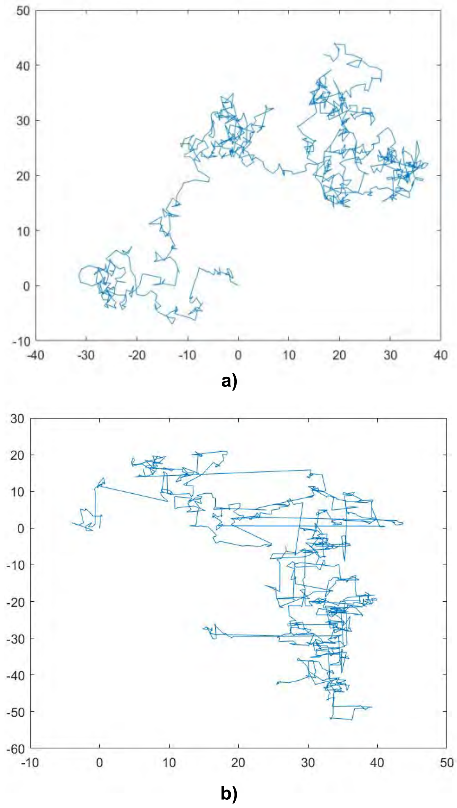

FIGURE 2. Comparison chart of Gaussian flight and Levy flight. (a) Gaussian flight. (b) Levy flight.

where mem_i^t is the memory value of crow j in the tth iteration. $levy(D)$ means the D-dimensional Levy flight mode. Figure 2 is the effect picture when Gaussian flight and Levy flight are executed for 1,000 times in the two-dimensional space of initial position (0,0).

As can be seen from the figure 2, Levy flight is appropriate for the crow individuals far away from the global optimum, and a strong development capacity is presented. Gaussian flight can enhance the local development capability of the algorithm, and it is suitable for individuals near the optimum individual.

2) NEIGHBORHOOD SEARCH MODE OF GAUSSIAN MUTATION

The neighborhood search mode of Gaussian mutation can make the crow individual fully traverse the surrounding position, and improve the search veracity of the algorithm. The traversal mode is as follows:

$$x_i^{t+1} = mem_i^t + normrnd(0, 1, 1, D).^*mem_i^t \quad (5)$$

where mem_i^t is the memory value of crow j in the tth iteration. $normrnd(0, 1, 1, D)$ are the D random numbers which obey standard Gaussian distribution.

It can be seen from Eq.(5): The Gaussian disturbance term is controlled by the crow individual's current position, so it can not only effectively prevent the crow individual from

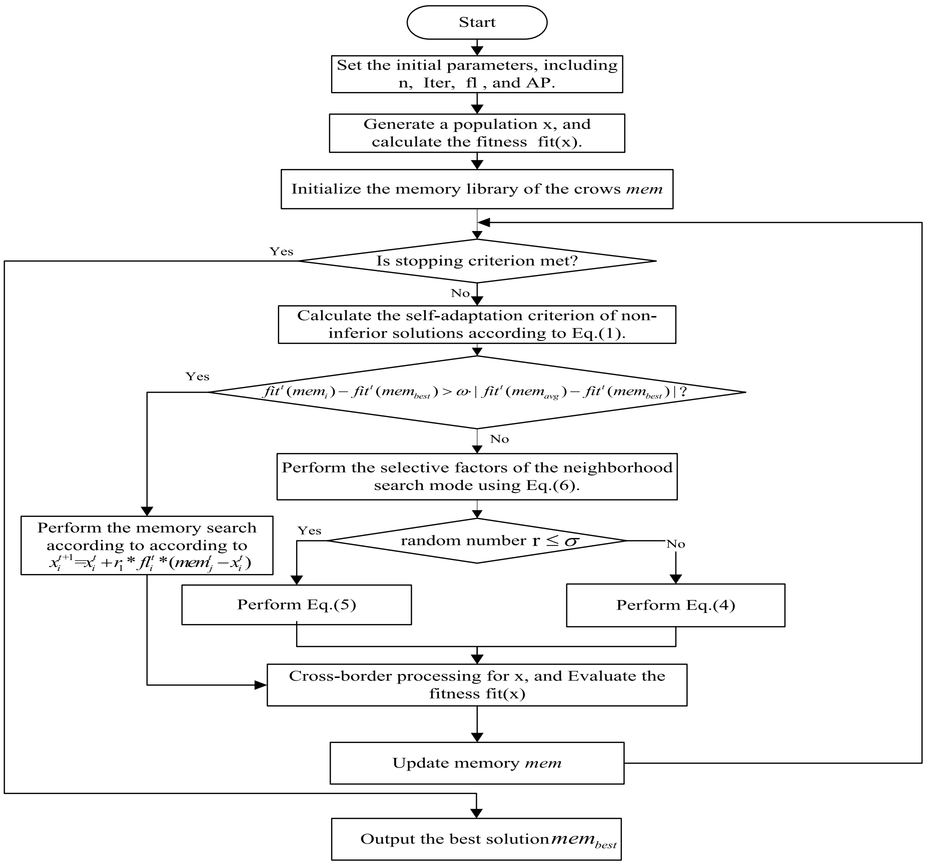

FIGURE 3. The flowchart of the proposed algorithm.

falling into local extreme point but also guarantee the crow individual's self-learning ability and increase the convergence rate. This improvement strategy provides a mechanism for the crow individual in the population to converge quickly and avoid early-maturing.

3) SELECTIVITY FACTOR OF NEIGHBORHOOD SEARCH MODE

In order to make the crow individual choose the suitable neighborhood search strategy according to its' evolution state, neighborhood search of Gaussian flight is executed for non-inferior solutions near the optimum, which can help to develop better solutions and increase the convergence rate. Neighborhood search of Levy flight is executed for non-inferior solutions far away from the optimum, so as to increase the development capacity of the algorithm. Therefore, a self-adaptive selectivity factor was introduced in this paper, to make the crow individual in this algorithm choose a reasonable neighborhood search mode. The selectivity factor is shown in Eq. (6).

$$\sigma = 1 - \exp(-|\mathrm{fit}^t(\mathrm{mem}_i) - \mathrm{fit}^t(\mathrm{mem}_{best})|) \tag{6}$$

where $fit^t(mem_i)$ and $fit^t(mem_{best})$ are the memory value of crow i in the tth iteration and the memory value of the best position in the population. The self-adaptive neighborhood search mode guided by selectivity factor is shown in Eq. (7).

$$x_i^{t+1} = \begin{cases} mem_i^t + levy(d).{}^*mem_i^t & \text{rand} \le \sigma \\ mem_i^t + normrnd(0, 1, 1, d).{}^*mem_i^t & \text{else} \end{cases} \tag{7}$$

From Eq. (6) and (7), the difference between the crow individual's fitness value and the optimum fitness value is relatively huge at starting period, and most crow individuals in the population execute neighborhood search of Levy flight. With the increase of iterations, the crow individuals approach the optimum crow individual gradually, so the algorithm focuses on development near the optimum solution.

C. ALGORITHM FLOW

The flowchart of the proposed algorithm discussed above is presented in figure 3 and the procedure is shown as follow.

IV. EXPERIMENTAL SIMULATION

In order to verify the performance of NICSA, the experiment will be conducted from the following four aspects:(1) The NICSA is compared with bat algorithm (BA) [36], [37], cuckoo optimization algorithm (CS) [8], differential evolution (DE) [38], flower pollination algorithm (FPA) [9], [39], grey wolf optimizer (GWO) [40], particle swarm

TABLE 1. The pseudocode of the proposed algorithm.

(1). Set the initial parameters, including the total population size n, the maximum number of generations $Iter$, flight distance fl, and the awareness probability(AP) .
(2). Generate a population $x=(x_1,x_2,...,x_i,...x_n)$.
(3). Calculate the fitness $fit(x_i)$ and find the best solution x_{best} of the population.
(4). Initialize the memory library of the crows $mem_i^0=x_i$.
(5). for t=1: $Iter$
(6). Calculate the self-adaptation criterion of non-inferior solutions according to Eq.(1).
(7). for i=1:n
(8). if $fit^t(mem_i)-fit^t(mem_{best})>\omega\cdot|fit^t(mem_{avg})-fit^t(mem_{best})|$
Perform the memory search according to
$x_i^{t+1}=x_i^t+r_1*fl_i^t*(mem_j^t-x_i^t)$.
(9). else
(10). Perform the selective factors of the neighborhood search mode using Eq.(6).
(11). Generate a random number r;
(12). if $r\leq\sigma$
(13). $x_i^{t+1}=mem_i^t+levy(d).*mem_i^t$
(14). else
(15). $x_i^{t+1}=mem_i^t+normrnd(0,1,1,d).*mem_i^t$
(16). end if
(17). end if
(18). Cross-border processing for x_i^t.
(19). Evaluate the fitness $fit(x_i^t)$ of the new crow i.
(20). If $fit(x_i^t)<fit(mem_i)$
(21). $mem_i^{t+1}=x_i^t$
(22). else
(23). $mem_i^{t+1}=mem_i^t$
(24). end if
(25). end for
(26). end for
(27). Output the best solution mem_{best}.

optimization (PSO) [5], teaching-learning-based optimization (TLBO) [41] and crow search algorithm (CSA) [22] for solving the optimal problems with dimensions lower than 30. (2) A contrastive analysis is made on the search performance of NICSA and other swarm intelligent algorithms with 50 dimensions and 100 dimensions. (3) The significance of NICSA's performance is analyzed via wilcoxon rank sum test. (4) Different values are selected for the determination factor ω of non-inferior solution and selectivity factor σ of neighborhood search mode.

A. BENCHMARK FUNCTION AND TEST PLATFORM

1) TEST PLATFORM

For providing a full test environment, the simulation experiment is conducted in the test environment with operating system of Windows 7, CPU of Intel (R) Core (TM) i5-7400 (4 cores), dominant frequency of 3.0GHZ and internal memory of 8GB, and programming tool of Matlab 2016b.

2) BENCHMARK FUNCTION

In order to validate the effectiveness of the proposed algorithm, 23 benchmark functions in literature [42]–[44] are selected as the experiment object, and the function type, serial number, expression, dimension, search scope and theoretically optimum value are shown in table 2. This test functions have diversity, and can reflect the search performance of the algorithm more objectively, fairly and comprehensively. The 23 benchmark functions can be categorized into three types: $f_1 \sim f_7$ are unimodal high-dimensional functions (I) to investigate the search veracity of the algorithm; $f_8 \sim f_{13}$ are the multimodal high-dimensional functions (II) with many local extreme points used to test the global search performance of the algorithm; $f_{14} \sim f_{23}$ are the multimodal low-dimensional functions (III). f_5 is a mutated function, the global optimum is at valley bottom of a parabola (the change of the fitness value near the valley bottom is quite small); moreover, it is hard to search the global optimum of this function. A large number of local minimum values exist for f_9 in the multimodal high-dimensional functions, which increases the search difficulty of the algorithm. In the multimodal low-dimensional functions, most functions have strong oscillation features.

B. EXPERIMENTAL RESULTS AND ANALYSIS ABOUT THE COMPARISON OF NICSA WITH OTHER SWARM INTELLIGENT ALGORITHMS

In experimental comparison, the selection of parameters has a huge influence on the algorithm performance. For fairness and reasonability, the parameters of the contrast algorithm are consistent with that of the original literature. Parameters of the algorithms are presented in table 3. Meanwhile, for each test function, evaluation is conducted for 50,000 times, and each algorithm runs independently for 30 times. The four indexes of optimum value, average value, worst value and variance are used to measure the performance of various algorithms, and the experimental statistical results are shown in table 4-table 6.

It can be seen from table 4 that NICSA has relatively good performance for unimodal high-dimensional functions. For functions f_1, f_2, f_3, f_5 and f_7, the search results of NICSA are superior to that of the other 8 algorithms in optimum value, average value, worst value and standard deviation, presenting very strong robustness. Especially for f_5, NICSA shows strong search performance. For f_4, NICSA is a little inferior to GWO and TLBO, but it is much better that other algorithms in search performance. This means that GWO and TLBO are applicable to solve such functions. For f_6, the search result of NICSA is a little inferior to that of DE algorithm, but superior to that of the other 7 algorithms. For the comparison algorithms, the overall performance of BA is the worst, BA and PSO algorithm shows relatively weak stability and big variance.

As seen from table 5, in terms of multimodal high-dimensional functions, NICSA can search the global optimum of three functions including f_8, f_9 and f_{11}, and presents very good stability. Especially for f_8 which is a typical deceptive function, the distance from the position of global optimum to the position of another local optimum is long. Once the algorithm falls into the local optimum, it can hardly

TABLE 2. Standard test functions.

Type	serial number	Benchmark test functions	Dimension	Scope	Optimum
I	$f_1(x)$	$f(x)=\sum_{i=1}^{D} x_i^2$	30	[-100,100]	0
	$f_2(x)$	$f(x)=\sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	30	[-10,10]	0
	$f_3(x)$	$f(x)=\sum_{i=1}^{D} (\sum_{j=1}^{i} x_i)^2$	30	[-100,100]	0
	$f_4(x)$	$f(x)=\max_{i=1}^{D}\{\lvert x_i \rvert\}$	30	[-100,100]	0
	$f_5(x)$	$f(x)=\sum_{i=1}^{n\text{-}1}\left[100(\mathrm{x}_{i+1}\text{-}\mathrm{x}_i^2)^2+(1\text{-}\mathrm{x}_i)^2\right]$	30	[-30,30]	0
	$f_6(x)$	$f(x)=\sum_{i=1}^{D}(\lfloor x_i+0.5 \rfloor)^2$	30	[-100,100]	0
	$f_7(x)$	$f(x)=\sum_{i=1}^{D} i \cdot x_i^4 + random(0,1)$	30	[-1.28,1.28]	0
II	$f_8(x)$	$f(x)=\sum_{i=1}^{D} -x_i \cdot \sin(\sqrt{\lvert x_i \rvert})$	30	[-500,500]	-418.9829*n
	$f_9(x)$	$f(x)=\sum_{i=1}^{n}(x_i^2\text{-}10 \cdot \cos(2 \cdot \pi \cdot \mathrm{x}_i)+10)$	30	[-5.12,5.12]	0
	$f_{10}(x)$	$f(x)=-20\exp(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2})-\exp(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i))+20+e$	30	[-32,32]	0
	$f_{11}(x)$	$f(x)=\frac{1}{4000}\cdot\sum_{i=1}^{n} x_i^2\text{-}\prod_{i=1}^{n}\cos\frac{x_i}{\sqrt{i}}+1$	30	[-600,600]	0
	$f_{12}(x)$	$f(x)=\frac{\pi}{D}\left\{10\sin^2(\pi \cdot y_1)+\sum_{i=1}^{D-1}(y_i-1)^2[1+10\sin^2(\pi \cdot y_{i+1})]+(y_n-1)^2\right\}$ $+\sum_{i=1}^{n} u(x_i,10,100,4)$ $y_i=1+\frac{x_i+1}{4}$ $u(x_i,\alpha,k,m)=\begin{cases} k(x_i-\alpha)^m, x_i>\alpha \\ 0, -\alpha \le x_i \le \alpha \\ k(-x_i-\alpha)^m, x_i \le \alpha \end{cases}$	30	[-50,50]	0
	$f_{13}(x)$	$f(x)=0.1\left\{10\sin^2(3\pi \cdot x_1)+\sum_{i=1}^{D-1}(x_i-1)^2[1+10\sin^2(3\pi \cdot x_{i+1})]+(x_n-1)^2\right\}$ $+\sum_{i=1}^{n} u(x_i,10,100,4)$	30	[-50,50]	0
III	$f_{14}(x)$	$f(x)=\left[\frac{1}{500}+\sum_{j=25}^{25}\frac{1}{j+\sum_{i=1}^{2}(x_i-a_{ij})^6}\right]^{-1}$	2	[-65.56,65.56]	0.9980
	$f_{15}(x)$	$f(x)=\sum_{i=1}^{11}\left[a_i-\frac{x_1(b_i^2+b_i x_2)}{b_i^2+b_i x_3+x_4}\right]^2$	4	[-5,5]	0.0003075
	$f_{16}(x)$	$f(x)=4x_1^2-2.1x_1^4+x_1^6/3+x_1x_2-4x_2^2+4x_2^4$	2	[-5,5]	-1.031629
	$f_{17}(x)$	$f(x)=(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1-6)^2+10(1-\frac{1}{8\pi})\cos x_1+10$	2	[-5,10;0,15]	0.398
	$f_{18}(x)$	$f(x)=[1+(x_1+x_2+1)^2(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)]$ $[30+(2x_1-3x_2)^2(18-32x_1+12x_1^2+48x_2-36x_1x_2+27x_2^2)]$	2	[-2,2]	3
	$f_{19}(x)$	$f(x)=-\sum_{i=1}^{4} c_i \cdot \exp\left[-\sum_{j=1}^{3} a_{ij}(x_j-p_{ij})^2\right]$	3	[0,1]	-3.8628
	$f_{20}(x)$	$f(x)=-\sum_{i=1}^{4} c_i \cdot \exp\left[-\sum_{j=1}^{6} a_{ij}(x_j-p_{ij})^2\right]$	6	[0,1]	-3.32
	$f_{21}(x)$	$f(x)=-\sum_{i=1}^{5}\left[(x-a_i)(x-a_i)^T+c_i\right]^{-1}$	4	[0,10]	-10.1532
	$f_{22}(x)$	$f(x)=-\sum_{i=1}^{7}\left[(x-a_i)(x-a_i)^T+c_i\right]^{-1}$	4	[0,10]	-10.4029
	$f_{23}(x)$	$f(x)=-\sum_{i=1}^{10}\left[(x-a_i)(x-a_i)^T+c_i\right]^{-1}$	4	[0,10]	-10.5364

TABLE 3. The parameters set of the algorithms.

Algorithms	Parameters
BA	the population size is 50,Qmin=0,Qmax=2, $R^0=0.1$, $A=0.9$, $\alpha=0.95$, $\gamma=0.9$.
CS	the population size is 50, discovery rate of alien eggs *pa* is 0.25.
DE	the population size is 50, pCR=0.2, $\beta_{\min}=0.2$, $\beta_{\max}=0.8$.
FPA	the population size is 50, probability switch *p* is 0.8.
GWO	the population size is 50.
PSO	the population size is 50,c1 = 1.49445,c2 = 1.49445, $\varpi=0.729$.
TLBO	the population size is 50, *TF* is randomly selected with 1 or 2.
CSA	the population size is 10, awareness probability *AP* is 0.2,and flight length *fl* is 2.
NICSA	the population size is 10 ,and flight length *fl* is 2.

TABLE 4. Test statistical results of function$f_1 \sim f_7$.

Benchmark function		BA	CS	DE	FPA	GWO	PSO	TLBO	CSA	NICSA
$f_1(x)$	Best	8.07154E+03	5.84693E+00	1.81806E-12	2.27835E+02	8.86632E-73	7.77048E+01	5.50209E-88	8.77126E-05	**5.44220E-261**
	Mean	1.28469E+04	1.47276E+01	5.74286E-12	5.31297E+02	1.88445E-70	2.98215E+02	9.40763E-87	3.38482E-04	**7.21320E-210**
	Worst	2.31105E+04	3.17336E+01	1.32869E-11	7.80119E+02	2.14177E-69	6.74886E+02	3.67128E-86	7.43627E-04	**2.16200E-208**
	Std	3.46491E+03	5.39555E+00	2.50351E-12	1.37176E+02	4.47695E-70	1.50655E+02	9.89714E-87	1.56006E-04	**0**
$f_2(x)$	Best	9.07062E+00	6.77554E+00	3.00964E-08	1.58711E+01	3.79814E-42	8.57450E+00	2.33445E-44	5.39687E+00	**8.11000E-215**
	Mean	8.04129E+01	1.08963E+01	5.16045E-08	2.56253E+01	5.89125E-41	1.71013E+01	1.59798E-43	1.87895E+01	**2.74360E-191**
	Worst	9.63393E+02	1.49832E+01	8.50427E-08	3.98884E+01	2.71300E-40	3.02752E+01	5.16495E-43	3.48761E+01	**8.22420E-190**
	Std	1.69120E+02	2.20621E+00	1.36665E-08	5.47876E+00	6.31089E-41	5.69461E+00	1.05459E-43	6.87753E+00	**0**
$f_3(x)$	Best	1.26536E+04	1.92777E+03	1.78369E+04	1.76558E+02	3.31339E-25	1.18689E+03	2.35068E-18	1.82642E+00	**1.53857E-47**
	Mean	3.41976E+04	2.95978E+03	2.39658E+04	4.47046E+02	2.54522E-19	2.52699E+03	1.27516E-16	4.89778E+00	**1.61584E-21**
	Worst	1.06206E+05	4.07593E+03	3.12445E+04	7.33714E+02	2.43563E-18	5.02924E+03	6.15546E-16	7.81543E+00	**2.18939E-20**
	Std	1.87025E+04	5.28750E+02	3.35239E+03	1.14518E+02	6.10088E-19	1.09847E+03	1.69407E-16	1.57783E+00	**5.09004E-21**
$f_4(x)$	Best	3.05197E+01	8.56850E+00	1.46740E+00	1.17106E+01	6.70742E-19	8.70281E+00	**1.17299E-35**	2.47061E+01	4.95432E-14
	Mean	4.52879E+01	1.11093E+01	2.00157E+00	1.67896E+01	1.93352E-17	1.43200E+01	**3.87286E-35**	5.27968E+01	1.29906E-04
	Worst	6.88328E+01	1.60052E+01	2.63504E+00	2.25055E+01	8.22734E-17	2.08613E+01	**1.04594E-34**	2.46625E+02	8.20484E-04
	Std	6.59024E+00	1.54076E+00	2.60063E-01	2.60257E+00	2.29672E-17	2.96473E+00	**2.43478E-35**	5.45146E+01	2.47276E-04
$f_5(x)$	Best	2.57772E+05	3.90603E+02	2.60570E+01	8.44411E+03	2.51758E+01	2.47961E+03	2.15580E+01	1.02591E-04	**1.24479E-05**
	Mean	5.20020E+06	6.83190E+02	4.51097E+01	3.50347E+04	2.63923E+01	1.82747E+04	2.30890E+01	3.20455E-04	**1.32343E-04**
	Worst	3.24417E+07	1.32314E+03	1.00007E+02	6.63404E+04	2.79015E+01	5.78775E+04	2.43256E+01	9.02797E-04	**7.04931E-04**
	Std	5.67074E+06	2.06929E+02	2.20033E+01	1.65081E+04	7.51441E-01	1.10659E+04	7.86560E-01	1.64139E-04	**1.35865E-04**
$f_6(x)$	Best	4.81290E+03	8.17890E+00	**2.79734E-12**	1.94898E+02	8.20012E-06	4.72411E+01	2.17525E-11	9.93531E-03	1.49318E-10
	Mean	1.18043E+04	1.43148E+01	**5.14675E-12**	5.18777E+02	3.66968E-01	3.43127E+02	2.06329E-09	1.91013E-02	6.69778E-10
	Worst	2.09629E+04	1.97574E+01	**9.81516E-12**	9.32918E+02	9.94811E-01	7.27500E+02	1.73203E-08	4.01609E-02	1.66068E-09
	Std	4.57641E+03	3.36828E+00	**1.74688E-12**	1.71868E+02	2.76152E-01	1.73491E+02	3.58576E-09	7.46737E-03	3.59720E-10
$f_7(x)$	Best	6.00374E-01	4.20477E-02	1.22125E-02	4.45896E-02	1.68131E-04	3.02112E-01	4.27566E-04	5.39687E+00	**2.25397E-05**
	Mean	2.85574E+00	8.06440E-02	2.33801E-02	1.21472E-01	5.68317E-04	7.34756E-01	1.05109E-03	1.87895E+01	**2.71506E-04**
	Worst	6.08099E+00	1.28901E-01	3.20297E-02	2.00965E-01	1.49561E-03	2.25510E+00	1.70801E-03	3.48761E+01	**9.83674E-04**
	Std	1.30947E+00	2.32909E-02	5.05492E-03	3.52022E-02	3.05922E-04	4.05018E-01	2.91967E-04	6.87753E+00	**2.47235E-04**

jump out. NICSA has searched the global optimum in 30, 50 and 100 dimensions (Section 4.3). For the comparison algorithms, only TLBO algorithm has searched the global optimum for f_{11}. For f_{10}, NICSA is superior to the other 8 algorithms in optimum value, average value, worst value and variance. For f_{12} and f_{13}, NICSA is a little inferior to DE algorithm, but superior to the other 7 algorithms. This means that DE algorithm is applicable to such multimodal high-dimensional functions. But generally speaking, NICSA has a better search effect.

Table 6 shows the statistical results of 9 algorithms when multimodal low-dimensional functions are solved. It is shown that NICSA can search the global optimum of all 10 test functions, and shows good stability and robustness.

figure 4–figure 26 are the convergence curves of optimal results for 9 algorithms when solving 23 test functions, and figure 27 – figure 49 are the variance analysis charts. It can be seen from figure 4 – figure 26, BA shows the lowest convergence rate, and the convergence rate of the other 7 algorithms varies as the characteristics of test functions change. For f_6, f_{12} and f_{13}, DE algorithm shows the highest convergence rate and search veracity, and TLBO algorithm also presents good convergence performance. But for other test functions, NICSA shows obviously higher convergence rate than the other 8 algorithms. Therefore, different algorithms have different advantages in convergence rate, but NICSA has more comprehensive advantages. From the variance analysis charts, NICSA has the smallest variance except for f_6 and f_{19}, and it shows good stability for different types of test functions. On the contrary, the test results of other comparison algorithms are unstable, and NICSA has very strong robustness.

TABLE 5. Test statistical results of function $f_8 \sim f_{13}$.

Benchmark function		BA	CS	DE	FPA	GWO	PSO	TLBO	CSA	NICSA
$f_8(x)$	Best	-5196.2	-8677.6	**-12569.5**	-8260.4	-7365.3	-7369.7	-9951.4	-10036.1	**-12569.5**
	Mean	-3205.0	-8195.1	-12553.3	-7577.2	-6179.0	-6060.3	-7848.0	-7643.95	**-12569.5**
	Worst	-2292.8	-7842.5	-12429.3	-7059.2	-3312.3	-4286.7	-5565.2	-5309.14	**-12569.5**
	Std	812.5	200.4	38.9	267.4	750.9	759.7	1049.1	1235.51	**0**
$f_9(x)$	Best	2.48769E+01	6.00766E+01	5.26591E+01	1.05670E+02	**0**	7.29271E+01	**0**	2.38791E+01	**0**
	Mean	7.85379E+01	1.07896E+02	6.10728E+01	1.31387E+02	5.68434E-15	1.44104E+02	1.04036E+01	4.90517E+01	**0**
	Worst	2.15908E+02	1.37146E+02	7.35561E+01	1.64148E+02	5.68434E-14	2.43434E+02	2.62432E+01	8.85514E+01	**0**
	Std	3.87595E+01	1.48800E+01	4.85934E+00	1.29234E+01	1.73446E-14	3.56633E+01	6.56653E+00	1.30465E+01	**0**
$f_{10}(x)$	Best	1.24919E+01	5.81139E+00	4.71154E-07	4.01453E+00	7.99361E-15	5.18331E+00	4.44089E-15	1.64632E+00	**8.88178E-16**
	Mean	1.49313E+01	8.95929E+00	6.38600E-07	5.34853E+00	1.37964E-14	7.89693E+00	6.80937E-15	3.71975E+00	**3.73035E-15**
	Worst	1.72078E+01	1.36754E+01	9.29271E-07	7.00990E+00	2.22045E-14	1.01234E+01	7.99361E-15	6.32806E+00	**4.44089E-15**
	Std	1.17821E+00	1.98751E+00	1.19152E-07	7.69467E-01	2.87314E-15	1.12453E+00	1.70340E-15	1.14351E+00	**1.45039E-15**
$f_{11}(x)$	Best	7.16985E+01	1.06420E+00	1.06479E-11	4.16891E+00	**0**	1.72070E+00	**0**	4.04997E-03	**0**
	Mean	1.39455E+02	1.15604E+00	5.30328E-11	5.85309E+00	2.21854E-03	4.03778E+00	**0**	1.77231E-02	**0**
	Worst	2.48865E+02	1.27604E+00	3.21346E-10	1.00272E+01	3.04707E-02	7.71118E+00	**0**	5.00918E-02	**0**
	Std	4.39194E+01	5.00261E-02	6.24705E-11	1.32619E+00	6.18942E-03	1.56712E+00	**0**	1.19795E-02	**0**
$f_{12}(x)$	Best	1.13505E+02	1.77548E+00	**2.30376E-13**	6.20875E+00	6.48000E-03	3.25486E+00	4.85971E-13	1.32266E+00	5.68181E-10
	Mean	2.93943E+06	3.25869E+00	**5.71409E-13**	1.02036E+01	2.35032E-02	1.01703E+01	2.76691E-11	5.09167E+00	4.87848E-09
	Worst	2.23418E+07	4.60098E+00	**1.50431E-12**	1.58354E+01	5.52255E-02	2.17416E+01	2.91293E-10	8.28434E+00	2.12224E-08
	Std	5.60381E+06	7.25969E-01	**2.79092E-13**	2.44288E+00	1.22219E-02	4.36040E+00	5.81830E-11	1.81494E+00	5.38263E-09
$f_{13}(x)$	Best	3.99761E+05	3.66447E+00	**1.57921E-12**	2.10399E+01	1.88807E-05	2.19679E+01	4.17506E-10	1.69966E-03	2.90656E-09
	Mean	1.28537E+07	8.21545E+00	**3.81172E-12**	1.78786E+02	3.26447E-01	5.11438E+01	2.83021E-02	4.31052E+00	2.11787E-08
	Worst	4.81967E+07	1.93343E+01	**1.20399E-11**	1.32474E+03	8.31130E-01	1.01034E+02	9.73712E-02	4.99597E+01	9.90463E-08
	Std	1.33038E+07	3.12897E+00	**2.11554E-12**	3.17286E+02	1.82779E-01	1.80726E+01	2.48752E-02	1.21877E+01	2.32719E-08

TABLE 6. Test statistical results of function $f_{14} \sim f_{23}$.

Benchmark function		BA	CS	DE	FPA	GWO	PSO	TLBO	CSA	NICSA
$f_{14}(x)$	Best	**0.998004**	**0.998004**	**0.998004**	**0.998004**	**0.998004**	**0.998004**	**0.998004**	**0.998004**	**0.998004**
	Mean	9.450481	**0.998004**	1.031138	**0.998004**	3.485311	1.262551	**0.998004**	**0.998004**	**0.998004**
	Worst	23.809434	**0.998004**	1.992031	**0.998004**	10.763181	2.982105	**0.998004**	**0.998004**	**0.998004**
	Std	6.99E+00	5.59E-16	1.81E-01	9.33E-11	3.44E+00	6.86E-01	**0**	1.60E-16	2.49E-16
$f_{15}(x)$	Best	0.000723	0.000310	0.000442	0.000308	**0.000307**	0.000313	**0.000307**	**0.000307**	**0.000307**
	Mean	0.012420	0.000400	0.000678	0.000338	0.003749	0.003207	0.000350	0.000945	**0.000307**
	Worst	0.020687	0.000548	0.001223	0.000433	0.020363	0.020363	0.000669	0.001223	**0.000308**
	Std	9.55E-03	7.14E-05	1.36E-04	2.67E-05	7.56E-03	5.98E-03	9.46E-05	4.11E-04	**3.86E-09**
$f_{16}(x)$	Best	**-1.031628**	**-1.031628**	**-1.031628**	**-1.031628**	**-1.031628**	**-1.031628**	**-1.031628**	**-1.031628**	**-1.031628**
	Mean	-1.004423	**-1.031628**	**-1.031628**	**-1.031628**	**-1.031628**	-1.031627	**-1.031628**	**-1.031628**	**-1.031628**
	Worst	-0.215464	**-1.031628**	**-1.031628**	**-1.031628**	**-1.031628**	-1.031622	**-1.031628**	**-1.031628**	**-1.031628**
	Std	1.49E-01	5.08E-16	6.78E-16	1.16E-10	3.37E-09	1.78E-06	6.78E-16	**4.97E-16**	6.71E-16
$f_{17}(x)$	Best	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397888**	**0.397887**	**0.397887**	**0.397887**
	Mean	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397888**	0.397895	**0.397887**	**0.397887**	**0.397887**
	Worst	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397890**	0.397920	**0.397887**	**0.397887**	**0.397887**
	Std	1.83E-10	2.97E-14	0.00E+00	2.58E-08	5.23E-07	7.32E-06	**0**	**0**	5.89E-16
$f_{18}(x)$	Best	**3**	**3**	**3**	**3**	**3**	3.000006	**3**	**3**	**3**
	Mean	**3**	**3**	**3**	**3**	3.000003	3.000139	**3**	**3**	**3**
	Worst	**3**	**3**	**3**	**3**	3.000021	3.000913	**3**	**3**	**3**
	Std	2.50E-08	1.54E-15	**1.15E-15**	4.05E-15	4.93E-06	1.93E-04	1.31E-15	2.92E-15	8.73E-09
$f_{19}(x)$	Best	**-3.862782**	**-3.862782**	**-3.862782**	**-3.862782**	**-3.862782**	**-3.862782**	**-3.862782**	**-3.862782**	**-3.862782**
	Mean	**-3.862782**	**-3.862782**	**-3.862782**	**-3.862782**	-3.861300	-3.862781	**-3.862782**	**-3.862782**	-3.862385
	Worst	**-3.862782**	**-3.862782**	**-3.862782**	**-3.862782**	-3.854900	-3.862779	**-3.862782**	**-3.862782**	-3.854901
	Std	2.40E-08	2.39E-15	2.71E-15	1.01E-14	2.94E-03	8.03E-07	2.71E-15	2.42E-15	1.57E-03
$f_{20}(x)$	Best	-3.321994	**-3.321995**	**-3.321995**	-3.321901	**-3.321995**	-3.321922	**-3.321995**	**-3.321995**	**-3.321995**
	Mean	-3.254620	**-3.321995**	**-3.321995**	-3.320793	-3.279817	-3.204598	-3.305866	**-3.321995**	-3.273171
	Worst	-3.203094	**-3.321995**	-3.321977	-3.318925	-3.086677	-2.896747	-3.203102	**-3.321995**	-3.133436
	Std	5.99E-02	1.17E-07	3.38E-06	8.00E-04	6.90E-02	1.07E-01	3.84E-02	2.00E-14	6.72E-02
$f_{21}(x)$	Best	**-10.153200**	**-10.153200**	**-10.153200**	**-10.153200**	-10.153180	-10.148810	**-10.153200**	**-10.153200**	**-10.153200**
	Mean	-5.473106	-10.153199	-10.135711	-10.153199	-9.474729	-6.198967	**-10.153200**	-7.114147	**-10.153200**
	Worst	-2.630471	-10.153197	-9.915133	-10.153191	-5.055192	-2.616143	**-10.153200**	-5.055198	**-10.153200**
	Std	3.26E+00	4.64E-07	5.61E-02	1.92E-06	1.76E+00	3.55E+00	1.46E-14	2.52E+00	**5.84E-15**
$f_{22}(x)$	Best	-10.402940	**-10.402941**	**-10.402941**	-10.402940	-10.402884	-10.401963	**-10.402941**	**-10.402941**	**-10.402941**
	Mean	-5.256186	-10.402940	-10.402929	-10.402842	-10.402634	-7.272706	-10.223097	-7.781414	**-10.402941**
	Worst	-1.837593	-10.402939	-10.402641	-10.401640	-10.402275	-2.736282	-5.007628	-2.765897	**-10.402941**
	Std	3.29E+00	5.22E-07	5.51E-05	2.60E-04	1.62E-04	3.55E+00	9.85E-01	2.91E+00	**1.98E-15**
$f_{23}(x)$	Best	-10.536409	**-10.536410**	**-10.536410**	-10.536404	-10.536360	-10.535326	**-10.536410**	**-10.536410**	**-10.536410**
	Mean	-5.142954	-10.536405	**-10.536410**	-10.536197	-10.357348	-7.388032	-10.325986	-9.017735	**-10.536410**
	Worst	-1.676553	-10.536361	**-10.536410**	-10.534857	-5.175303	-1.859323	-4.223693	-2.806631	**-10.536410**
	Std	3.90E+00	1.19E-05	7.00E-15	3.11E-04	9.79E-01	3.65E+00	1.15E+00	2.57E+00	**3.97E-15**

Table 7 shows the average running time (in second) spent by each comparison algorithm for solving 23 benchmark functions in table 2. The total average running time of NICSA algorithm is 0.93 s, which is less than that of DE and TLBO, and more than that of other algorithms, but it is also within the time range of 1s.

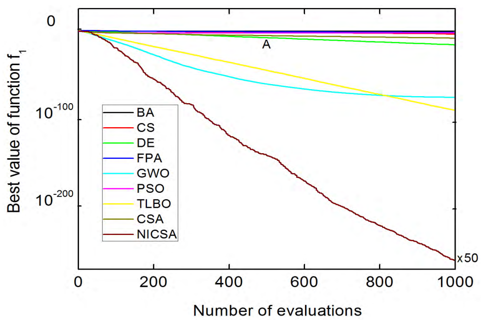

FIGURE 4. Evolution curves of fitness value for f_1.

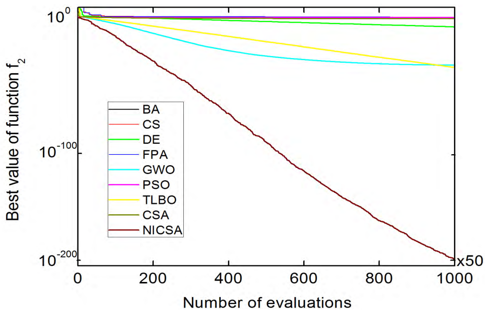

FIGURE 5. Evolution curves of fitness value for f_2.

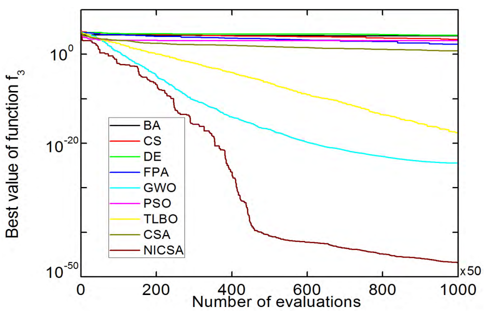

FIGURE 6. Evolution curves of fitness value for f_3.

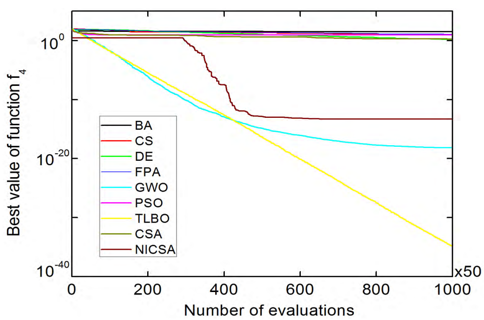

FIGURE 7. Evolution curves of fitness value for f_4.

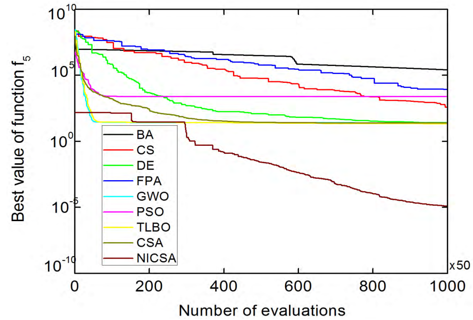

FIGURE 8. Evolution curves of fitness value for f_5.

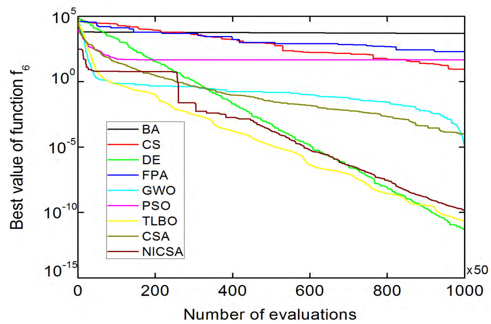

FIGURE 9. Evolution curves of fitness value for f_6.

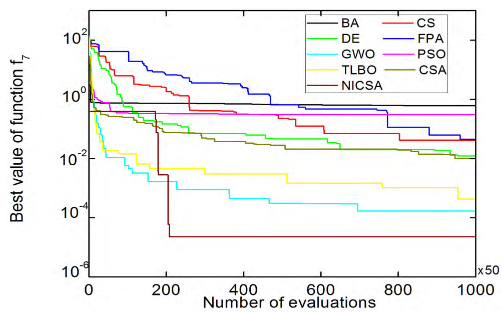

FIGURE 10. Evolution curves of fitness value for f_7.

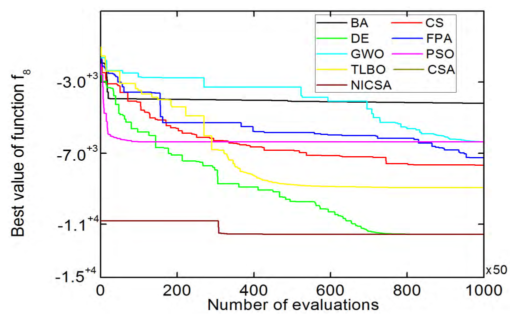

FIGURE 11. Evolution curves of fitness value for f_8.

C. TEST OF HIGH-DIMENSIONAL FUNCTIONS

In order to verify the search performance of NICSA for high-dimensional functions, independent tests are also conducted for 13 high-dimensional test functions including $f_1 \sim f_{13}$ with 50 dimensions and 100 dimensions. Meanwhile, every test function is evaluated for 50,000 times, and each algorithm runs independently for 30 times. Other parameters of various algorithms are consistent with parameters in table 3. The performance of various algorithms is measured through

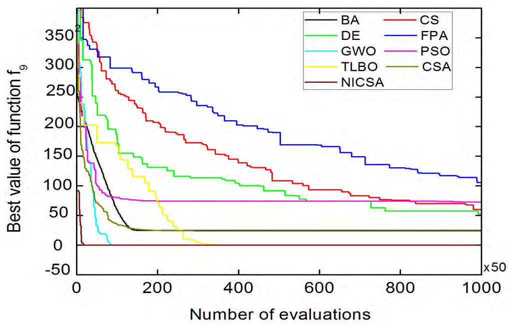

FIGURE 12. Evolution curves of fitness value for f_9.

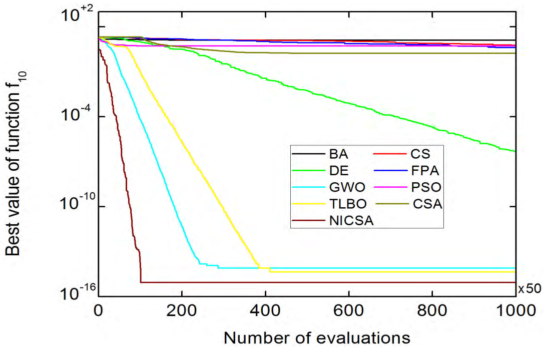

FIGURE 13. Evolution curves of fitness value for f_{10}.

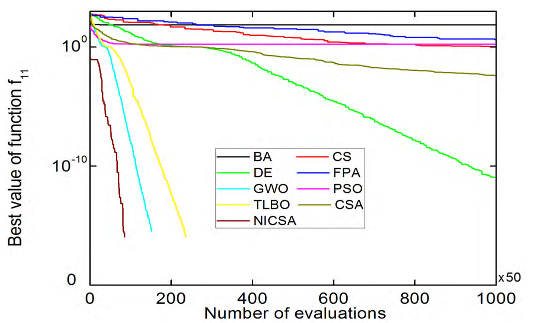

FIGURE 14. Evolution curves of fitness value for f_{11}.

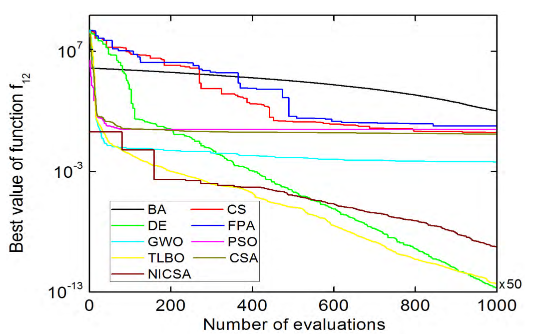

FIGURE 15. Evolution curves of fitness value for f_{12}.

four indexes covering optimum value, average value, worst value and variance, and the experimental statistical results are shown in table 8.

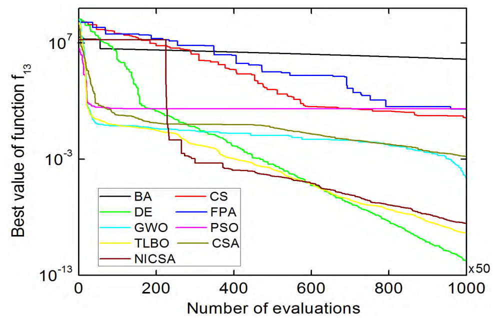

FIGURE 16. Evolution curves of fitness value for f_{13}.

TABLE 7. The average running time of the different algorithms.

Fun	BA	CS	DE	FPA	GWO	PSO	TLBO	CSA	NICSA
f_1	0.64	0.54	1.52	0.57	0.44	0.31	0.72	0.45	0.65
f_2	0.62	0.57	1.58	0.65	0.48	0.32	0.77	0.48	0.70
f_3	1.69	1.65	2.64	1.79	1.55	1.47	1.75	1.53	1.69
f_4	0.57	0.53	1.44	0.58	0.43	0.28	0.70	0.42	0.65
f_5	0.62	0.58	1.60	0.64	0.48	0.34	0.76	0.48	0.74
f_6	0.56	0.52	1.51	0.57	0.43	0.28	0.69	0.43	0.64
f_7	0.80	0.76	1.69	0.83	0.66	0.52	0.83	0.66	0.73
f_8	0.67	0.64	1.59	0.71	0.52	0.39	0.72	0.49	0.54
f_9	0.60	0.59	1.47	0.64	0.45	0.34	0.68	0.45	0.72
f_{10}	0.66	0.65	1.63	0.67	0.49	0.38	0.69	0.50	0.66
f_{11}	0.71	0.69	1.70	0.72	0.53	0.40	0.74	0.54	1.13
f_{12}	1.33	1.26	2.53	1.35	1.19	1.04	1.50	1.16	1.23
f_{13}	1.36	1.28	2.51	1.37	1.17	1.04	1.50	1.16	1.31
f_{14}	2.53	2.42	3.72	2.57	2.20	2.44	2.71	2.47	2.60
f_{15}	0.54	0.42	1.40	0.53	0.23	0.29	0.63	0.45	0.70
f_{16}	0.46	0.35	1.31	0.47	0.17	0.22	0.57	0.42	0.73
f_{17}	0.44	0.33	1.28	0.44	0.15	0.21	0.53	0.40	0.82
f_{18}	0.44	0.33	1.26	0.43	0.15	0.20	0.53	0.39	0.52
f_{19}	0.62	0.50	1.50	0.62	0.32	0.37	0.71	0.55	0.85
f_{20}	0.63	0.52	1.54	0.65	0.37	0.38	0.73	0.53	0.68
f_{21}	0.82	0.70	1.74	0.81	0.53	0.56	0.92	0.73	0.85
f_{22}	0.95	0.84	1.90	0.95	0.65	0.70	1.03	0.87	1.06
f_{23}	1.15	1.02	2.18	1.16	0.84	0.91	1.26	1.05	1.27
avage	0.84	0.77	1.79	0.86	0.63	0.58	0.94	0.72	0.93

It can be seen from table 8, for high-dimensional functions with 50 dimensions and 100 dimensions, NICSA shows better search results than the other 8 algorithms except for f_4 and f_{11}. For f_{11}, both TLBO and NICSA have searched the global optimum. For f_4, the results of NICSA is inferior to GWO in the research results of 30 and 50 dimensions, but NICSA shows better results than GWO in optimum value, average value, worst value and variance in 100 dimensions. Therefore, NICSA has better performance for high-dimensional functions.

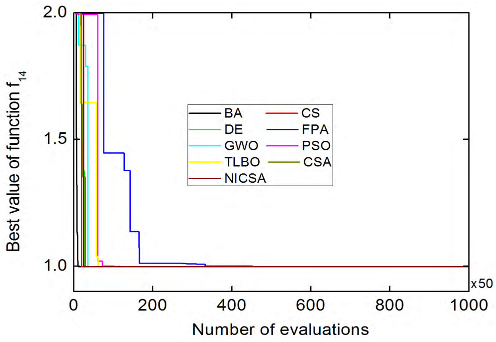

FIGURE 17. Evolution curves of fitness value for f_{14}.

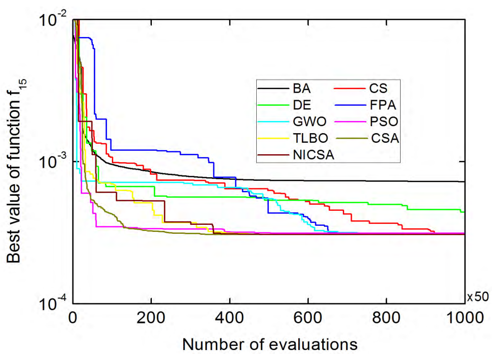

FIGURE 18. Evolution curves of fitness value for f_{15}.

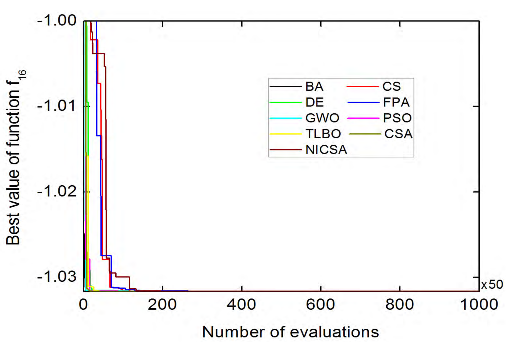

FIGURE 19. Evolution curves of fitness value for f_{16}.

D. NONPARAMETRIC STATISTICAL ANALYSIS

In order to better verify the significance of the search effect of the proposed algorithm, Wilcoxon rank sum test was conducted for the results in low-dimensional (D <= 30), 50-dimensional and 100-dimensional functions. The value of effective level p is 0.05. Table 9 – table 11 show the results of Wilcoxon rank sum test in low-dimensional (D <= 30), 50-dimensional and 100-dimensional functions respectively. "+" means that **NICSA** has obvious advantages when compared with other algorithms, and "≈" indicates that **NICSA** has no obvious difference from other algorithms.

It can be seen from table 9, in the comparison with 5 algorithms including BA, CS, FPA, PSO and CSA, the p values of 22 functions are smaller than 0.05, showing that NICSA

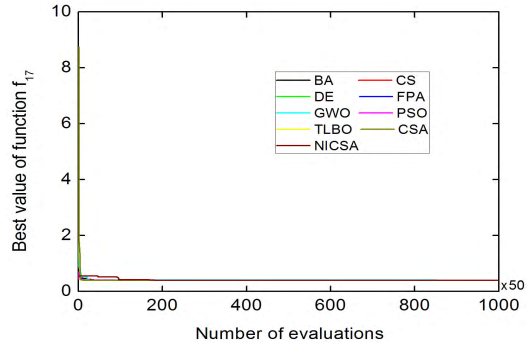

FIGURE 20. Evolution curves of fitness value for f_{17}.

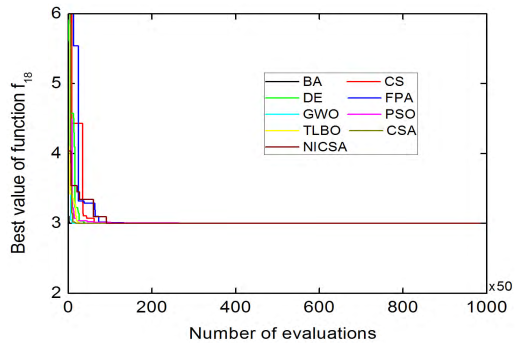

FIGURE 21. Evolution curves of fitness value for f_{18}.

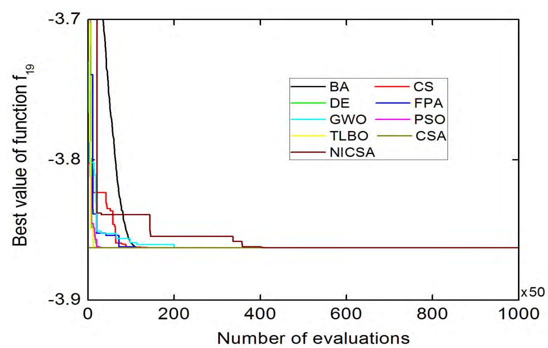

FIGURE 22. Evolution curves of fitness value for f_{19}.

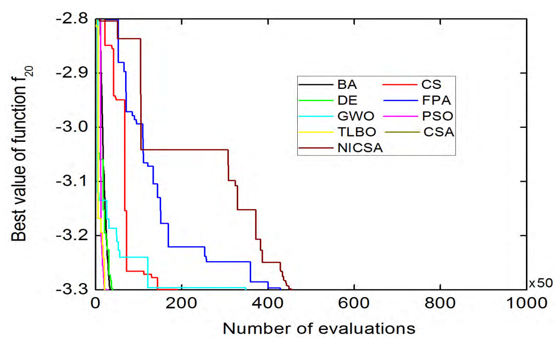

FIGURE 23. Evolution curves of fitness value for f_{20}.

is superior to the above 5 algorithms. In the comparison with DE, GWO and TLBO, the p values of 20, 21 and 17 functions are smaller than 0.05. According to the analysis from the

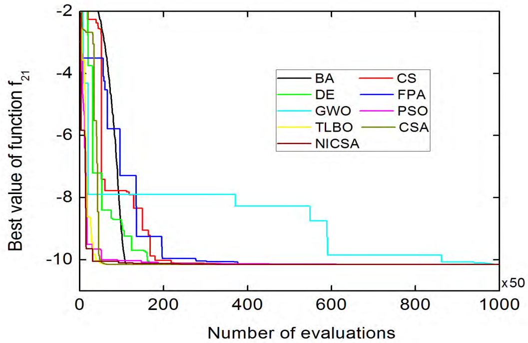

FIGURE 24. Evolution curves of fitness value for f_{21}.

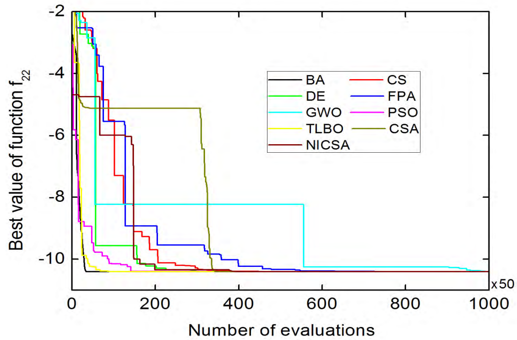

FIGURE 25. Evolution curves of fitness value for f_{22}.

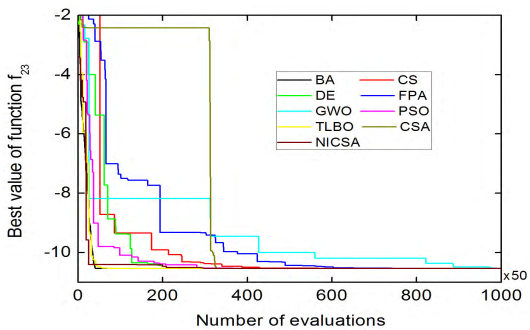

FIGURE 26. Evolution curves of fitness value for f_{23}.

angle of statistics, NICSA has obvious search effects for low-dimensional functions.

Table 10 and table 11 show 50-dimensional (D = 50) and 100-dimensional (D = 100) Wilcoxon rank sum test results. In the comparison with BA, CS, DE, FPA, PSO and CSA, the p values of 13 high-dimensional functions are smaller than 0.05, showing that NICSA is superior to the above 6 algorithms. In the comparison with GWO and TLBO, the p values of 12 and 11 functions are smaller than 0.05. This further proves that NICSA has obvious search effects for high-dimensional functions.

E. PARAMETER SENSITIVITY ANALYSIS

1) PARAMETER ANALYSIS ON THE SELF-ADAPTIVE DETERMINATION FACTOR ω OF NON-INFERIOR SOLUTIONS

In order to discuss the influence of the self-adaptive determination factor ω of non-inferior solutions on the algorithm

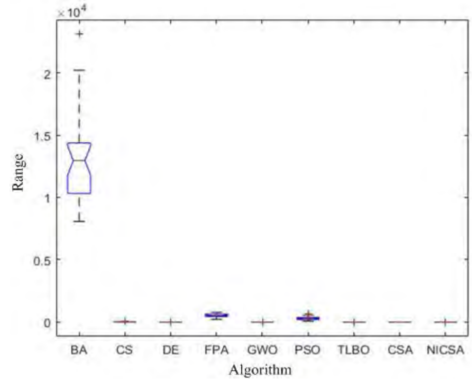

FIGURE 27. ANOVA test of global optimum for f_1.

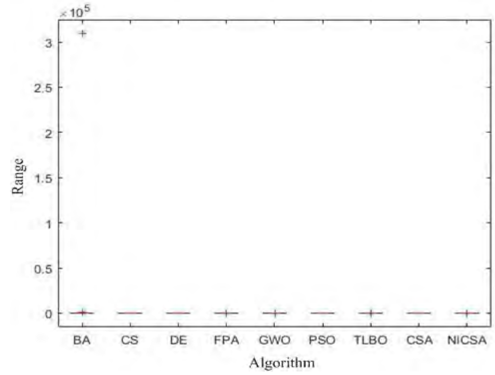

FIGURE 28. ANOVA test of global optimum for f_2.

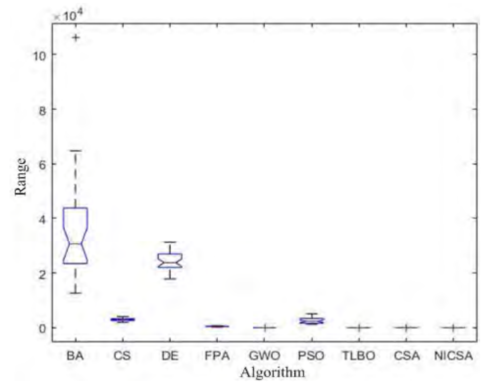

FIGURE 29. ANOVA test of global optimum for f_3.

performance, 23 standard test functions in table 1 are selected for verification. The value of ω is 0.1~0.9 with step 0.1, and independent experiments are conducted for the proposed strategy in this paper under 10 situations. Other parameters are consistent with that in table 2. Table 12 shows the results when different values of ω were chosen. The black bold represents the winner in the comparison. According to the last column of table 12, the number of average optimum gained by way of determination factor is 16, obviously better than the results with other situations. In order to further calculate the advantages and disadvantages when different values of

TABLE 8. Test statistical results of benchmark functions for different algorithm (D = 50 and D = 100).

Test function		D=50				D=100			
		Min	Mean	Worst	Std	Min	Mean	Worst	Std
$f_1(x)$	BA	1.23E+04	2.31E+04	3.79E+04	6.44E+03	3.42E+04	5.19E+04	8.92E+04	1.34E+04
	CS	1.46E+02	2.38E+02	3.52E+02	5.97E+01	1.85E+03	2.77E+03	3.61E+03	4.83E+02
	DE	2.43E-05	3.76E-05	5.35E-05	7.46E-06	1.31E+01	2.14E+01	2.81E+01	3.50E+00
	FPA	1.29E+03	1.91E+03	3.42E+03	4.06E+02	5.10E+03	6.67E+03	8.97E+03	1.03E+03
	GWO	7.20E-54	7.68E-52	7.18E-51	1.33E-51	1.73E-35	1.59E-34	1.06E-33	2.04E-34
	PSO	7.97E+01	3.59E+02	9.71E+02	1.69E+02	1.64E+03	2.46E+03	3.50E+03	4.67E+02
	TLBO	3.10E-83	7.61E-82	3.51E-81	7.40E-82	4.29E-79	1.05E-77	6.42E-77	1.15E-77
	CSA	1.20E-02	3.08E-02	8.10E-02	1.53E-02	1.10E+02	1.43E+02	1.87E+02	1.83E+01
	NICSA	**1.22E-146**	**8.11E-84**	**2.43E-82**	**4.44E-83**	**6.36E-93**	**4.75E-38**	**9.79E-37**	**1.93E-37**
$f_2(x)$	BA	1.96E+01	1.33E+07	3.30E+08	6.11E+07	7.52E+01	1.28E+17	3.85E+18	7.03E+17
	CS	2.30E+01	3.44E+01	4.55E+01	5.80E+00	6.42E+01	8.16E+01	1.26E+02	1.32E+01
	DE	3.59E-04	5.45E-04	7.44E-04	7.98E-05	1.84E+00	2.23E+00	2.69E+00	2.11E-01
	FPA	2.90E+01	4.31E+01	5.97E+01	6.45E+00	6.40E+01	8.30E+01	1.05E+02	1.04E+01
	GWO	3.64E-31	1.32E-30	2.69E-30	7.97E-31	2.09E-21	6.99E-21	1.78E-20	4.03E-21
	PSO	1.67E+01	2.48E+01	3.86E+01	5.19E+00	4.75E+01	5.99E+01	8.46E+01	8.93E+00
	TLBO	6.67E-42	3.55E-41	1.74E-40	3.32E-41	6.97E-40	2.32E-39	5.80E-39	1.17E-39
	CSA	1.47E+00	3.82E+00	7.05E+00	1.44E+00	1.21E+01	1.53E+01	1.88E+01	1.78E+00
	NICSA	**7.40E-182**	**6.75E-153**	**2.02E-151**	**3.69E-152**	**4.28E-143**	**7.46E-123**	**2.24E-121**	**4.08E-122**
$f_3(x)$	BA	3.58E+04	1.02E+05	2.54E+05	5.16E+04	1.49E+05	4.40E+05	9.98E+05	2.10E+05
	CS	1.20E+04	1.57E+04	2.19E+04	2.13E+03	7.30E+04	9.55E+04	1.15E+05	1.06E+04
	DE	7.10E+04	9.02E+04	1.05E+05	7.42E+03	3.21E+05	3.91E+05	4.59E+05	3.33E+04
	FPA	1.21E+03	2.04E+03	2.93E+03	4.12E+02	4.48E+03	8.26E+03	1.25E+04	2.08E+03
	GWO	9.56E-14	8.11E-09	1.14E-07	2.79E-08	8.54E-04	3.86E-01	2.90E+00	7.17E-01
	PSO	1.80E+03	4.32E+03	8.30E+03	1.90E+03	1.18E+04	2.15E+04	4.05E+04	7.46E+03
	TLBO	9.11E-13	1.49E-10	1.20E-09	2.28E-10	3.58E-07	1.19E-05	5.44E-05	1.64E-05
	CSA	1.23E+02	2.13E+02	3.36E+02	6.05E+01	2.36E+03	3.58E+03	6.30E+03	7.43E+02
	NICSA	**1.22E-18**	**2.45E-09**	**4.72E-08**	**8.72E-09**	**3.18E-15**	**7.04E-09**	**6.83E-08**	**1.63E-08**
$f_4(x)$	BA	3.98E+01	5.11E+01	6.89E+01	7.32E+00	3.99E+01	5.34E+01	6.70E+01	6.32E+00
	CS	8.86E+00	1.11E+01	1.34E+01	1.38E+00	2.04E+01	2.59E+01	3.01E+01	2.53E+00
	DE	1.89E+01	2.24E+01	2.73E+01	1.63E+00	7.59E+01	8.22E+01	8.56E+01	2.47E+00
	FPA	1.90E+01	2.28E+01	2.65E+01	1.95E+00	2.31E+01	2.77E+01	3.42E+01	2.90E+00
	GWO	**9.24E-13**	**2.93E-11**	**1.35E-10**	**3.52E-11**	1.12E-06	8.23E-04	1.64E-02	3.02E-03
	PSO	9.12E+00	1.29E+01	1.71E+01	2.15E+00	1.24E+01	1.70E+01	2.07E+01	2.21E+00
	TLBO	1.53E-33	5.63E-33	1.58E-32	3.22E-33	1.31E-31	3.77E-31	1.11E-30	2.42E-31
	CSA	4.06E+00	8.42E+00	1.08E+01	1.41E+00	7.64E+00	9.72E+00	1.15E+01	1.14E+00
	NICSA	5.52E-06	2.00E-04	9.35E-04	2.33E-04	**1.26E-07**	**6.45E-04**	**4.36E-03**	**8.25E-04**
$f_5(x)$	BA	2.33E+06	1.19E+07	4.54E+07	9.93E+06	7.58E+06	3.65E+07	9.49E+07	2.00E+07
	CS	5.33E+03	1.46E+04	3.58E+04	7.42E+03	1.70E+05	3.27E+05	4.84E+05	9.02E+04
	DE	6.49E+01	2.09E+02	2.88E+02	5.92E+01	1.62E+04	2.91E+04	3.81E+04	5.25E+03
	FPA	9.31E+04	2.23E+05	3.82E+05	7.27E+04	4.61E+05	1.16E+06	2.20E+06	3.89E+05
	GWO	4.51E+01	4.68E+01	4.84E+01	8.57E-01	9.50E+01	9.73E+01	9.84E+01	8.60E-01
	PSO	4.11E+03	1.56E+04	5.63E+04	1.23E+04	4.99E+04	1.69E+05	3.32E+05	7.45E+04
	TLBO	4.24E+01	4.43E+01	4.55E+01	7.38E-01	9.34E+01	9.55E+01	9.81E+01	1.18E+00
	CSA	4.65E+01	9.60E+01	2.29E+02	5.37E+01	1.93E+03	3.31E+03	4.75E+03	7.03E+02
	NICSA	**5.52E-06**	**2.00E-04**	**9.35E-04**	**2.33E-04**	**2.85E-02**	**1.29E+01**	**9.64E+01**	**3.32E+01**
$f_6(x)$	BA	1.25E+04	2.44E+04	3.99E+04	7.30E+03	2.92E+04	5.59E+04	9.65E+04	1.55E+04
	CS	1.75E+02	2.76E+02	5.21E+02	8.44E+01	2.34E+03	2.88E+03	4.02E+03	4.01E+02
	DE	1.94E-05	3.78E-05	6.77E-05	1.12E-05	1.49E+01	2.17E+01	3.08E+01	3.64E+00
	FPA	1.16E+03	1.84E+03	2.69E+03	4.34E+02	4.47E+03	6.74E+03	8.58E+03	1.09E+03

TABLE 8. ***(Continued.)*** **Test statistical results of benchmark functions for different algorithm (D = 50 and D = 100).**

	GWO	5.02E-01	1.79E+00	2.76E+00	5.40E-01	5.49E+00	7.66E+00	9.26E+00	9.39E-01
	PSO	1.83E+02	4.11E+02	8.93E+02	1.47E+02	1.28E+03	2.25E+03	3.00E+03	4.50E+02
	TLBO	6.74E-05	2.38E-03	3.34E-02	6.29E-03	4.87E-01	1.20E+00	2.37E+00	4.82E-01
	CSA	1.15E-02	2.97E-02	5.58E-02	1.04E-02	9.81E+01	1.45E+02	1.95E+02	1.88E+01
	NICSA	**6.50E-08**	**2.85E-07**	**8.08E-07**	**1.77E-07**	**1.51E-05**	**4.55E-05**	**1.68E-04**	**3.22E-05**
$f_7(x)$	BA	2.27E+00	6.42E+00	1.20E+01	2.48E+00	8.12E+00	1.43E+01	2.33E+01	3.61E+00
	CS	9.83E-02	2.15E-01	3.61E-01	5.91E-02	6.62E-01	1.15E+00	2.07E+00	3.91E-01
	DE	4.41E-02	7.63E-02	1.05E-01	1.38E-02	4.75E-01	5.70E-01	7.56E-01	6.10E-02
	FPA	1.65E-01	3.68E-01	8.58E-01	1.48E-01	1.24E+00	2.15E+00	3.26E+00	5.41E-01
	GWO	2.19E-04	8.35E-04	2.44E-03	4.54E-04	9.10E-04	1.65E-03	3.88E-03	7.33E-04
	PSO	3.46E-01	9.28E-01	1.79E+00	4.16E-01	3.24E+00	8.40E+00	3.64E+01	6.99E+00
	TLBO	5.10E-04	1.02E-03	1.80E-03	3.36E-04	7.14E-04	1.32E-03	2.22E-03	3.33E-04
	CSA	2.30E-02	4.85E-02	7.61E-02	1.41E-02	1.04E-01	2.00E-01	2.77E-01	3.60E-02
	NICSA	**3.68E-05**	**4.20E-04**	**2.31E-03**	**5.19E-04**	**1.89E-05**	**2.89E-04**	**7.80E-04**	**2.07E-04**
$f_8(x)$	BA	-5.74E+03	-4.16E+03	-2.92E+03	8.08E+02	-1.13E+04	-5.97E+03	-3.62E+03	1.74E+03
	CS	-1.28E+04	-1.22E+04	-1.18E+04	2.45E+02	-2.11E+04	-1.98E+04	-1.90E+04	4.58E+02
	DE	-1.44E+04	-1.32E+04	-1.25E+04	4.25E+02	-1.94E+04	-1.83E+04	-1.73E+04	5.38E+02
	FPA	-1.12E+04	-1.05E+04	-9.98E+03	2.82E+02	-1.67E+04	-1.60E+04	-1.54E+04	3.32E+02
	GWO	-1.18E+04	-9.36E+03	-5.86E+03	1.18E+03	-2.04E+04	-1.64E+04	-5.86E+03	2.44E+03
	PSO	-1.18E+04	-9.99E+03	-7.89E+03	9.31E+02	-1.96E+04	-1.58E+04	-1.19E+04	1.74E+03
	TLBO	-1.39E+04	-1.16E+04	-6.62E+03	2.02E+03	-2.49E+04	-1.72E+04	-9.84E+03	4.19E+03
	CSA	-1.43E+04	-1.12E+04	-5.71E+03	2.44E+03	-2.64E+04	-1.96E+04	-1.02E+04	4.49E+03
	NICSA	**-2.09E+04**	**-2.09E+04**	**-2.09E+04**	**1.61E-03**	**-4.19E+04**	**-4.07E+04**	**-3.01E+04**	**3.61E+03**
$f_9(x)$	BA	6.47E+01	1.25E+02	2.41E+02	4.55E+01	1.29E+02	2.46E+02	3.92E+02	6.62E+01
	CS	1.89E+02	2.29E+02	2.54E+02	1.73E+01	5.45E+02	6.26E+02	7.15E+02	4.38E+01
	DE	1.81E+02	1.97E+02	2.17E+02	8.12E+00	6.31E+02	6.96E+02	7.35E+02	2.66E+01
	FPA	2.39E+02	2.90E+02	3.32E+02	2.24E+01	6.76E+02	7.29E+02	8.13E+02	3.10E+01
	GWO	0.00E+00	3.02E-01	5.68E+00	1.19E+00	0.00E+00	3.43E-01	6.71E+00	1.37E+00
	PSO	1.82E+02	2.29E+02	2.71E+02	2.51E+01	5.39E+02	6.38E+02	7.20E+02	5.39E+01
	TLBO	0.00E+00	1.62E+01	4.01E+01	1.20E+01	0.00E+00	6.21E+00	1.01E+02	2.37E+01
	CSA	4.58E+01	8.49E+01	1.62E+02	2.68E+01	1.29E+02	1.75E+02	2.22E+02	2.42E+01
	NICSA	**0**	**0**	**0**	**0**	**0**	**1.33E+01**	**9.95E+01**	**3.44E+01**
$f_{10}(x)$	BA	1.42E+01	1.55E+01	1.76E+01	9.84E-01	1.39E+01	1.62E+01	2.00E+01	1.73E+00
	CS	8.25E+00	1.22E+01	1.71E+01	2.29E+00	1.05E+01	1.48E+01	1.82E+01	1.95E+00
	DE	8.03E-04	1.25E-03	1.67E-03	1.97E-04	1.65E+00	2.00E+00	2.41E+00	2.20E-01
	FPA	4.94E+00	5.88E+00	8.98E+00	7.74E-01	5.15E+00	6.24E+00	7.81E+00	5.98E-01
	GWO	1.87E-14	2.67E-14	3.29E-14	3.95E-15	5.77E-14	6.92E-14	7.55E-14	4.82E-15
	PSO	5.30E+00	7.54E+00	9.49E+00	9.14E-01	7.35E+00	8.69E+00	1.03E+01	6.83E-01
	TLBO	4.44E-15	7.76E-15	7.99E-15	9.01E-16	7.99E-15	7.99E-15	7.99E-15	0.00E+00
	CSA	3.48E+00	5.64E+00	2.00E+01	3.97E+00	4.04E+00	5.19E+00	2.00E+01	2.83E+00
	NICSA	**8.88E-16**	**4.32E-15**	**7.99E-15**	**1.47E-15**	**4.44E-15**	**7.05E-15**	**7.99E-15**	**1.60E-15**
$f_{11}(x)$	BA	1.05E+02	2.35E+02	4.79E+02	8.28E+01	3.16E+02	5.32E+02	7.32E+02	1.12E+02
	CS	2.23E+00	3.32E+00	4.51E+00	5.51E-01	1.67E+01	2.44E+01	3.13E+01	2.97E+00
	DE	2.49E-05	8.94E-05	3.12E-04	5.97E-05	1.12E+00	1.19E+00	1.25E+00	2.52E-02
	FPA	1.18E+01	1.89E+01	2.62E+01	3.73E+00	4.53E+01	6.45E+01	9.20E+01	9.40E+00
	GWO	0.00E+00	2.83E-03	1.84E-02	5.51E-03	0.00E+00	1.09E-03	1.17E-02	3.33E-03
	PSO	1.59E+00	4.31E+00	8.56E+00	1.66E+00	9.95E+00	2.35E+01	3.46E+01	5.32E+00
	TLBO	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	CSA	8.80E-02	1.49E-01	2.11E-01	3.36E-02	1.92E+00	2.36E+00	2.73E+00	2.24E-01
	NICSA	**0**	**0**	**0**	**0**	**0**	1.11E-17	3.33E-16	6.08E-17
$f_{12}(x)$	BA	2.28E+04	1.07E+07	6.73E+07	1.48E+07	9.55E+05	3.18E+07	1.50E+08	3.32E+07
	CS	5.53E+00	7.27E+00	9.09E+00	1.09E+00	1.40E+01	1.80E+01	2.71E+01	3.21E+00

TABLE 8. ***(Continued.)*** **Test statistical results of benchmark functions for different algorithm (D = 50 and D = 100).**

	DE	8.97E-06	2.25E-05	5.62E-05	9.47E-06	2.35E+01	4.18E+01	1.30E+02	2.17E+01
	FPA	9.80E+00	1.79E+01	2.57E+01	3.56E+00	1.65E+01	3.99E+02	3.92E+03	8.67E+02
	GWO	3.07E-02	6.15E-02	1.18E-01	2.28E-02	1.27E-01	2.01E-01	3.18E-01	4.50E-02
	PSO	4.65E+00	8.59E+00	1.51E+01	2.85E+00	8.12E+00	1.31E+01	2.07E+01	3.26E+00
	TLBO	3.64E-07	2.10E-03	6.22E-02	1.14E-02	4.89E-03	1.26E-02	2.48E-02	4.89E-03
	CSA	1.86E+00	5.80E+00	1.04E+01	2.05E+00	2.57E+00	5.38E+00	8.04E+00	1.27E+00
	NICSA	**1.72E-08**	**1.33E-07**	**4.69E-07**	**1.20E-07**	**7.52E-07**	**5.76E-06**	**3.59E-05**	**7.60E-06**
$f_{13}(x)$	BA	4.64E+06	2.96E+07	8.02E+07	2.20E+07	2.12E+07	9.61E+07	3.21E+08	6.71E+07
	CS	2.74E+01	5.15E+01	1.21E+02	2.00E+01	2.97E+03	2.02E+04	5.95E+04	1.63E+04
	DE	5.45E-05	1.14E-04	3.02E-04	5.22E-05	6.40E+02	4.94E+03	1.12E+04	3.04E+03
	FPA	8.85E+02	2.61E+04	1.06E+05	2.62E+04	3.74E+04	3.74E+05	1.16E+06	3.01E+05
	GWO	8.33E-01	1.39E+00	2.56E+00	3.89E-01	4.75E+00	5.55E+00	6.78E+00	4.93E-01
	PSO	3.62E+01	7.70E+01	1.60E+02	2.70E+01	1.44E+02	7.46E+02	1.15E+04	2.06E+03
	TLBO	3.13E-03	1.72E-01	6.31E-01	1.65E-01	2.67E+00	4.32E+00	5.87E+00	8.34E-01
	CSA	8.54E+00	5.48E+01	9.11E+01	2.28E+01	1.50E+01	5.47E+01	1.78E+02	4.66E+01
	NICSA	**2.67E-07**	**1.17E-06**	**3.66E-06**	**9.32E-07**	**1.62E-05**	**4.50E-05**	**1.28E-04**	**2.24E-05**

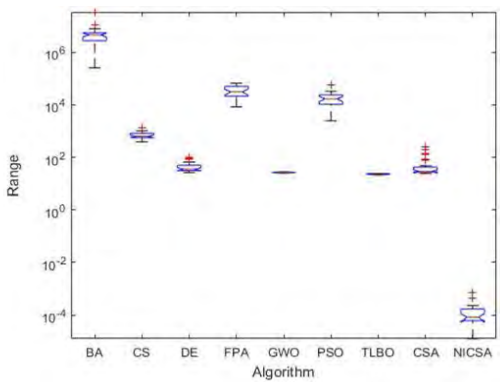

FIGURE 30. **ANOVA test of global optimum for f_4.**

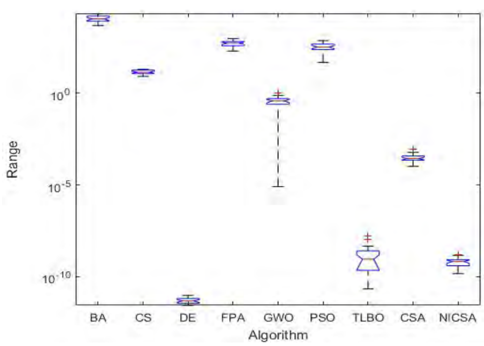

FIGURE 31. **ANOVA test of global optimum for f_5.**

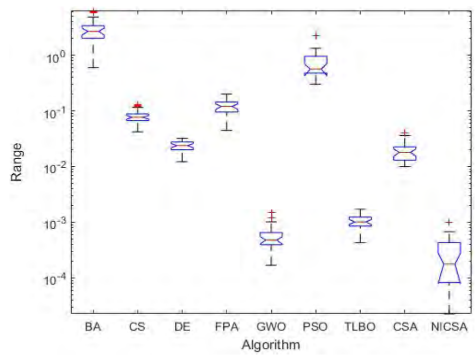

FIGURE 32. **ANOVA test of global optimum for f_6.**

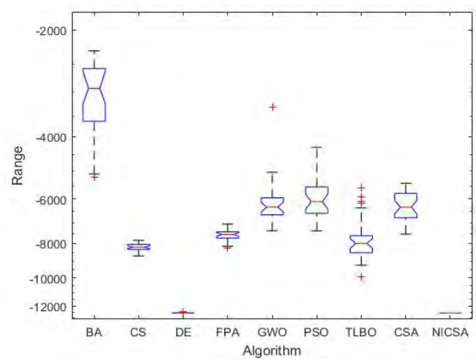

FIGURE 33. **ANOVA test of global optimum for f_7.**

ω are adopted, tests are conducted with Friedman. Friedman test is a nonparametric test method to analyze whether population distribution has significant differences via the rank. The smaller the average value of rank is, the better the results will be. According to the ranking situations of Friedman test when different values of ω are adopted in table 13, the lowest rank by using Eq. (3) is 3.24, further showing that the search effect of the proposed strategy in this paper is better than that under other ω values.

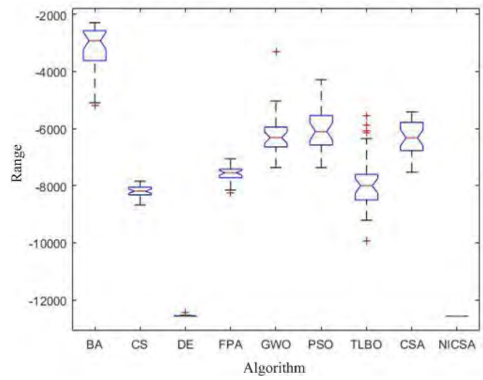

FIGURE 34. ANOVA test of global optimum for f_8.

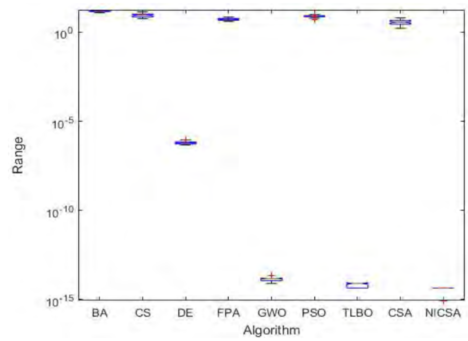

FIGURE 35. ANOVA test of global optimum for f_9.

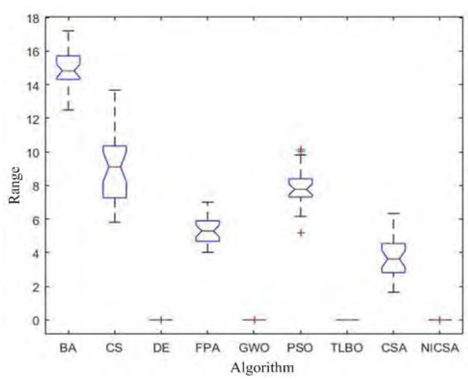

FIGURE 36. ANOVA test of global optimum for f_{10}.

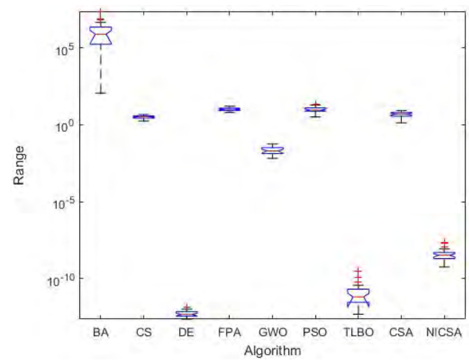

FIGURE 37. ANOVA test of global optimum for f_{11}.

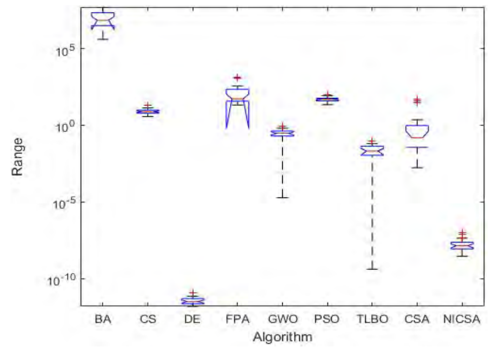

FIGURE 38. ANOVA test of global optimum for f_{12}.

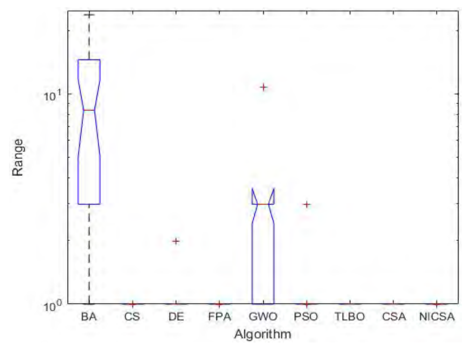

FIGURE 39. ANOVA test of global optimum for f_{13}.

2) PARAMETER ANALYSIS ON THE SELECTIVITY FACTOR σ OF NEIGHBORHOOD SEARCH MODE

In the neighborhood search mode, the value of the selectivity factor σ has a big effect on the algorithm performance. In order to discuss the influence of parameter on the search performance of the algorithm, the standard test functions in table 2 are selected for verification. The value of σ is 0.1∼0.9 with step 0.1, and independent experiments are conducted for the proposed strategy in this paper with 10 situations. Other parameters are consistent with that in table 3.

Table 14 shows the results that NICSA adopts different values of the selectivity factor σ. The optimum results are bold. It can be seen from table 14, when the σ value is adopted according to Eq. (6), 15 values are acquired on average optimal, better than the situation when a fixed value is adopted for σ. Therefore, it is reasonable to choose neighborhood search according to Eq.(6). According to the ranking situations of Friedman test when different values of σ are adopted in table 15, the lowest rank of choosing σ value according to Eq.(6) is 4.39, further showing that the search effect of the proposed strategy in this paper is better than that under other fixed σ values.

TABLE 9. Test statistical results of wilcoxon rank sum test (D<= 30).

Fun	NICSAvs BA	CS	DE	FPA	GWO	PSO	TLBO	CSA
f_1	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)
f_2	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)
f_3	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.3E-06(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)
f_4	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)
f_5	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)
f_6	3.0E-11(+)	3.0E-11(+))	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	4.3E-01(≈)	3.0E-11(+)
f_7	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	6.4E-05(+)	3.0E-11(+)	3.5E-10(+)	3.0E-11(+)
f_8	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)
f_9	1.2E-12(+)	1.2E-12(+)	1.2E-12(+)	1.2E-12(+)	8.1E-02(+)	1.2E-12(+)	4.6E-12(+)	1.2E-12(+)
f_{10}	6.3E-12(+)	6.3E-12(+)	6.3E-12(+)	6.3E-12(+)	2.6E-12(+)	6.3E-12(+)	3.5E-08(+)	6.3E-12(+)
f_{11}	1.2E-12(+)	1.2E-12(+)	1.2E-12(+)	1.2E-12(+)	2.7E-02(+)	1.2E-12(+)	NaN(≈)	1.2E-12(+)
f_{12}	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)
f_{13}	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.4E-04(+)	3.0E-11(+)
f_{14}	1.2E-11(+)	1.3E-04(+)	6.4E-09(+)	1.2E-11(+)	1.2E-11(+)	1.2E-11(+)	2.4E-10(+)	1.6E-03(+)
f_{15}	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	3.0E-11(+)	9.1E-08(+)	3.0E-11(+)	4.5E-01(≈)	6.7E-10(+)
f_{16}	1.7E-12(+)	1.1E-08(+)	3.3E-01(≈)	1.7E-12(+)	1.7E-12(+)	1.7E-12(+)	3.3E-01(≈)	6.0E-10(+)
f_{17}	3.6E-12(+)	2.8E-06(+)	8.1E-02(≈)	3.2E-12(+)	3.6E-12(+)	3.2E-12(+)	8.1E-02(≈)	8.1E-02(≈)
f_{18}	5.4E-10(+)	4.5E-01(+)	1.9E-06(+)	2.7E-02(+)	4.8E-11(+)	2.6E-11(+)	3.2E-05(+)	4.3E-02(+)
f_{19}	5.6E-04(+)	8.8E-12(+)	1.2E-12(+)	2.9E-11(+)	7.5E-02(≈)	2.3E-01(≈)	1.2E-12(+)	1.6E-11(+)
f_{20}	7.7E-02(≈)	1.7E-01(≈)	2.4E-11(+)	7.7E-02(≈)	1.2E-02(+)	2.4E-05(+)	8.6E-05(+)	2.4E-09(+)
f_{21}	2.1E-11(+)	2.1E-11(+)	4.8E-01(≈)	2.1E-11(+)	2.1E-11(+)	2.1E-11(+)	4.8E-01(≈)	1.1E-09(+)
f_{22}	2.3E-11(+)	2.3E-11(+)	3.3E-05(+)	2.3E-11(+)	2.3E-11(+)	2.3E-11(+)	6.2E-08(+)	8.1E-08(+)
f_{23}	1.6E-11(+)	1.6E-11(+)	9.1E-09(+)	1.6E-11(+)	1.6E-11(+)	1.6E-11(+)	1.9E-09(+)	4.3E-05(+)
number of winners(+/≈)	22/1	22/1	20/3	22/1	21/2	22/1	17/6	22/1

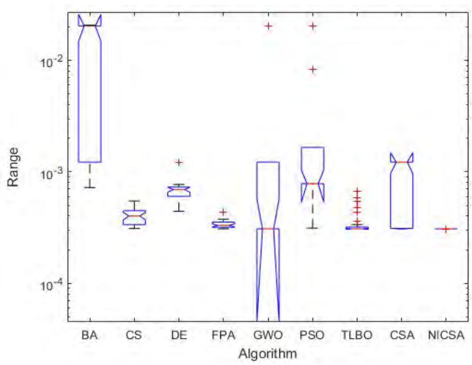

FIGURE 40. ANOVA test of global optimum for f_{14}.

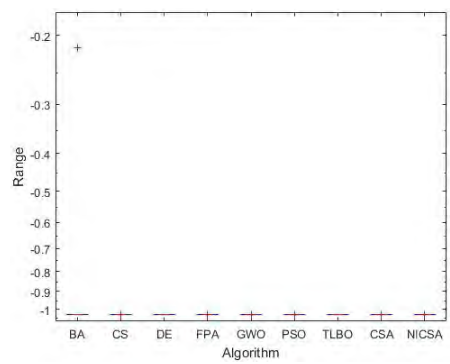

FIGURE 41. ANOVA test of global optimum for f_{15}.

F. COMPARISON OF NICSA WITH OTHER ENHANCED CSA METHODS

In order to verify the superiority of NICSA, we compare the proposed algorithm with other improved CSA, including improved crow search algorithm based on variable-factor weighted learning and adjacent-generations dimension crossover strategy (C^4SA) [45], sine cosine crow search algorithm (SCCSA) [46], and ICSA [27]. For the comparative analysis between ICSA [27] and NICSA, the test functions including $f_1 \sim f_{13}$ with 50 dimensions are conducted.

TABLE 10. Test statistical results of wilcoxon rank sum test (dim = 50).

Fun	NICSAvs							
	BA	CS	DE	FPA	GWO	PSO	TLBO	CSA
f_1	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	7.39E-11(+)	3.02E-11(+)
f_2	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
f_3	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	7.06E-01(≈)	3.02E-11(+)	8.88E-01(≈)	3.02E-11(+)
f_4	3.01E-11(+)	3.01E-11(+)	3.01E-11(+)	3.01E-11(+)	3.01E-11(+)	3.01E-11(+)	3.01E-11(+)	3.01E-11(+)
f_5	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
f_6	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
f_7	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	6.74E-06(+)	3.02E-11(+)	1.47E-07(+)	3.02E-11(+)
f_8	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
f_9	1.21E-12(+)	1.21E-12(+)	1.21E-12(+)	1.21E-12(+)	5.54E-03(+)	1.21E-12(+)	1.70E-08(+)	1.21E-12(+)
f_{10}	5.20E-12(+)	5.20E-12(+)	5.20E-12(+)	5.20E-12(+)	3.26E-12(+)	5.20E-12(+)	5.61E-11(+)	5.20E-12(+)
f_{11}	1.21E-12(+)	1.21E-12(+)	1.21E-12(+)	1.21E-12(+)	5.58E-03(+)	1.21E-12(+)	NaN(≈)	1.21E-12(+)
f_{12}	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	4.08E-11(+)	3.02E-11(+)
f_{13}	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
number of winners(+/≈)	13/0	13/0	13/0	13/0	12/1	13/0	11/2	13/0

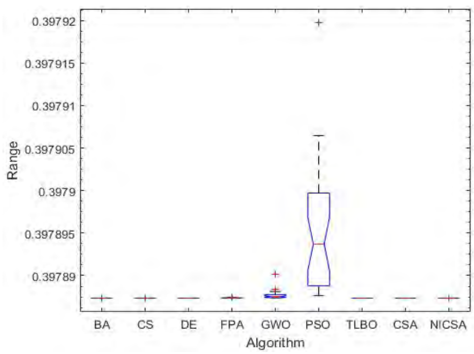

FIGURE 42. ANOVA test of global optimum for f_{16}.

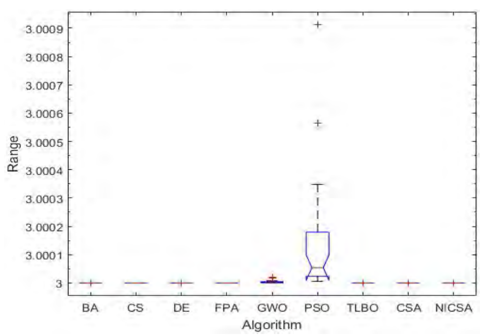

FIGURE 43. ANOVA test of global optimum for f_{17}.

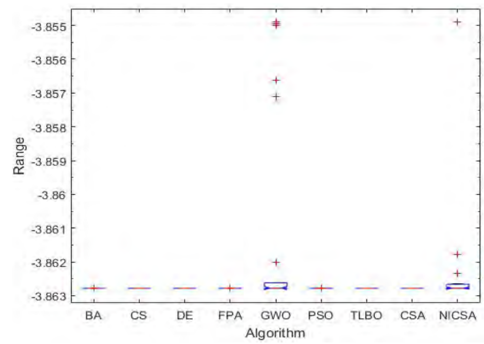

FIGURE 44. ANOVA test of global optimum for f_{18}.

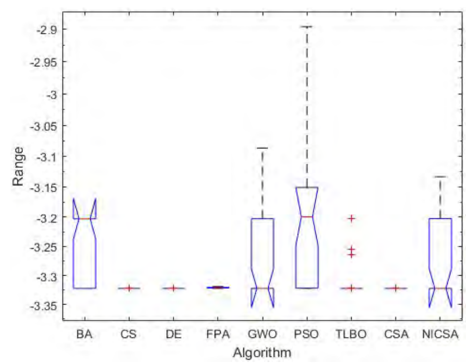

FIGURE 45. ANOVA test of global optimum for f_{19}.

For the comparative analysis between other algorithms (SCCSA and C^{4}SA) and NICSA, the test functions including $f_1 \sim f_{13}$ with 30 dimensions are conducted. Table 16 shows the experimental statistical results of various comparison algorithms. "-" means that the comparative algorithms did not test the function. Meanwhile, for each test function, evaluation is conducted for 50,000 times, and

TABLE 11. Test statistical results of wilcoxon rank sum test (dim = 100).

Fun	NICSAvs							
	BA	CS	DE	FPA	GWO	PSO	TLBO	CSA
f_1	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	5.57E-10(+)	3.02E-11(+)
f_2	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
f_3	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
f_4	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	2.53E-04(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
f_5	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	1.33E-10(+)	3.02E-11(+)	1.73E-07(+)	3.02E-11(+)
f_6	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
f_7	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.69E-11(+)	3.02E-11(+)
f_8	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
f_9	4.11E-12(+)	4.11E-12(+)	4.11E-12(+)	4.11E-12(+)	1.95E-05(+)	4.11E-12(+)	4.02E-01(≈)	4.11E-12(+)
f_{10}	8.86E-12(+)	8.87E-12(+)	8.87E-12(+)	8.87E-12(+)	6.67E-12(+)	8.87E-12(+)	2.70E-03(+)	8.87E-12(+)
f_{11}	1.72E-12(+)	1.72E-12(+)	1.72E-12(+)	1.72E-12(+)	2.89E-01(≈)	1.72E-12(+)	3.34E-01(≈)	1.72E-12(+)
f_{12}	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
f_{13}	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)	3.02E-11(+)
number of winners(+/≈)	13/0	13/0	13/0	13/0	12/1	13/0	11/2	13/0

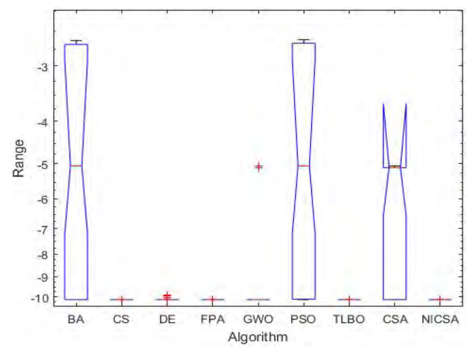

FIGURE 46. ANOVA test of global optimum for f_{20}.

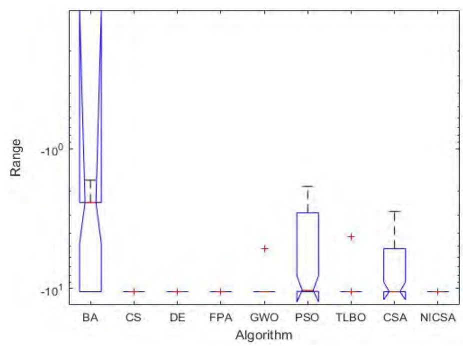

FIGURE 48. ANOVA test of global optimum for f_{22}.

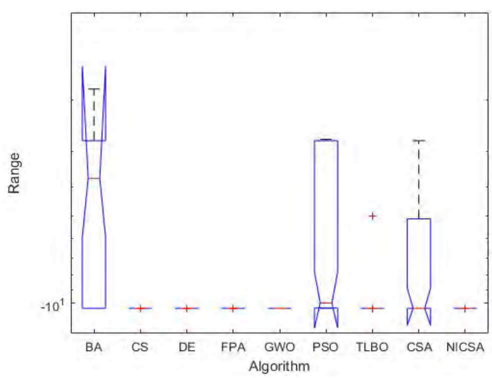

FIGURE 47. ANOVA test of global optimum for f_{21}.

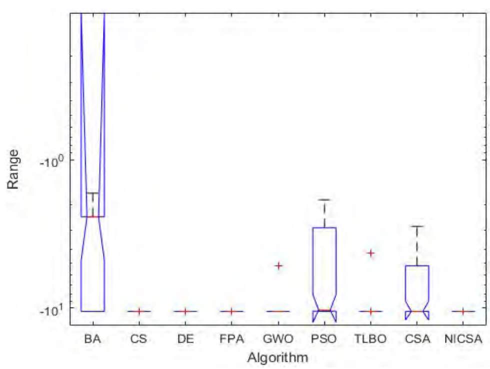

FIGURE 49. ANOVA test of global optimum for f_{23}.

the experimental results are the average of 30 independent test. It can be seen from Table 16, the search results of NICSA are better than those of ICSA [27] except function f_{10} (the result of the two algorithms is basically the same). Compared with SCCSA [46], the search results of NICSA for all test functions are better than SCCSA [46]. To solve

TABLE 12. Statistical results with different ω values.

Fun	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	Eq.(3)
f_1	1.60E-171	7.70E-185	3.90E-200	8.30E-204	1.30E-216	3.40E-218	3.10E-222	**5.00E-233**	2.30E-228	7.21E-210
f_2	1.20E-163	2.20E-175	3.10E-179	4.60E-187	1.60E-189	7.90E-193	**1.60E-196**	6.80E-196	3.60E-200	2.74E-191
f_3	2.94E-01	2.18E-01	1.19E-01	3.95E-02	3.93E-01	2.10E-01	2.52E+00	1.85E+01	1.41E+02	**1.62E-21**
f_4	1.77E+01	1.29E+01	7.71E+00	2.86E+00	3.93E+00	9.01E-01	3.27E-01	2.39E-02	1.83E-01	**1.30E-04**
f_5	1.09E+01	4.37E+00	8.04E+00	1.76E+00	1.80E+00	8.49E-01	1.75E+00	8.61E-01	1.73E+00	**1.32E-04**
f_6	1.99E-01	1.10E-01	4.08E-13	2.59E-13	**3.90E-13**	6.41E-13	1.33E-12	2.18E-12	1.76E-11	6.70E-10
f_7	6.43E-02	4.46E-02	5.70E-02	5.23E-03	4.16E-03	3.55E-03	1.82E-03	7.52E-04	3.56E-04	**2.72E-04**
f_8	-9.51E+03	-9.21E+03	-9.20E+03	-1.05E+04	-9.22E+03	-1.04E+04	-1.02E+04	-1.07E+04	-1.07E+04	**-1.26E+04**
f_9	3.79E+01	4.58E+01	2.49E+01	3.86E+01	3.38E+01	2.18E+01	2.09E+01	9.95E+00	8.92E+00	**0**
f_{10}	1.24E+00	1.06E+00	2.24E+00	2.57E+00	1.23E+00	4.19E-01	5.38E-01	4.19E-01	4.44E-15	**3.73E-15**
f_{11}	5.98E-03	4.04E-03	4.59E-11	**0**	**0**	**0**	3.97E-03	**0**	**0**	**0**
f_{12}	5.45E+00	7.57E-01	2.94E-06	8.35E-01	**2.52E-11**	1.04E-01	6.31E-11	2.41E-04	3.68E+00	4.88E-09
f_{13}	1.08E+01	6.31E+00	7.01E-04	3.66E-04	**1.32E-10**	3.66E-04	3.22E-10	7.26E-10	2.61E-09	2.12E-08
f_{14}	2.43E+00	1.91E+00	1.97E+00	1.97E+00	4.18E+00	1.59E+00	1.65E+00	1.06E+00	2.50E+00	**9.98E-01**
f_{15}	3.46E-04	3.74E-04	3.08E-04	3.15E-04	4.00E-04	3.39E-04	3.50E-04	3.18E-04	3.08E-04	**3.07E-04**
f_{16}	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**
f_{17}	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.39789**
f_{18}	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**
f_{19}	-3.86252	**-3.86278**	-3.86262	-3.86252	-3.8625	-3.86259	-3.86268	-3.86200	-3.86278	-3.86238
f_{20}	**-3.27792**	-3.26413	-3.2672	-3.26035	-3.25868	-3.24155	-3.25936	-3.26043	-3.27196	-3.27317
f_{21}	-7.54951	-8.54753	-7.8138	-7.20965	-7.63664	-6.46937	-8.30027	-8.38238	-8.04822	**-10.15320**
f_{22}	-8.30024	-8.35721	-9.04215	-7.74783	-9.27139	-8.48005	-9.0876	-9.344	-9.51706	**-10.40294**
f_{23}	-8.7027	-8.01908	-7.32397	-7.94023	-7.97081	-8.80512	-8.8727	-9.66801	-8.87485	**-10.53641**
number of winners	5	4	4	5	6	4	4	5	4	16

TABLE 13. Rankings of friedman test with different ω values.

ω	mean rank
Eq.(3)	3.24
0.9	4.09
0.8	4.11
0.7	4.78
0.6	5.20
0.5	6.07
0.4	6.30
0.3	6.39
0.2	6.98
0.1	7.85

unimodal high-dimensional functions, the results of NICSA is better than C4SA. But the results of C4SA is better than NICSA for solving multimodal high-dimensional functions, which shows that the C4SA is suitable for solving these functions.

G. ENGINEERING OPTIMIZATION PROBLEMS

In order to validate the performance of NICSA for constraint problems, it is tested against the following three engineering structure design optimization.

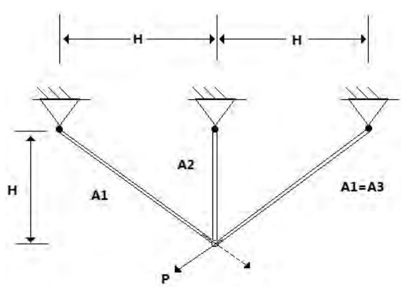

FIGURE 50. Threebar truss design.

1) THREEBAR TRUSS DESIGN

This test case considers a three-bar planar truss design problem shown in figure 50. The problem has 2 continuous variables and 3 nonlinear inequality constraints

TABLE 14. Statistical results under different σ values.

Fun	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	Eq.(6)
f_1	8.00E-272	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	7.21E-210
f_2	6.50E-256	7.70E-279	**1.00E-296**	6.70E-290	1.10E-286	1.40E-280	2.30E-270	1.10E-249	8.60E-221	2.74E-191
f_3	5.37E-19	8.80E-50	**2.20E-120**	3.80E-101	6.71E-97	1.55E-95	8.05E-72	4.96E-20	2.53E-17	1.62E-21
f_4	3.89E-01	1.31E-04	9.24E-05	3.50E-04	3.22E-04	6.58E-05	1.18E-06	8.10E-06	**1.06E-07**	1.30E-04
f_5	6.13E+00	3.50E+00	3.49E+00	1.79E+00	3.59E+00	1.79E+00	2.70E+00	8.92E-01	1.83E+00	**1.32E-04**
f_6	8.64E-10	9.09E-10	5.73E-09	**4.15E-11**	2.95E-09	2.21E-09	1.25E-09	1.46E-10	7.99E-10	6.70E-10
f_7	5.07E-04	8.35E-04	5.69E-04	5.19E-05	3.34E-04	5.96E-04	6.48E-04	5.92E-04	6.18E-04	**2.72E-04**
f_8	-1.15E+04	-1.22E+04	-1.21E+04	-1.22E+04	-1.23E+04	-1.25E+04	-1.25E+04	-1.21E+04	-1.21E+04	**-1.26E+04**
f_9	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_{10}	1.48E-15	1.24E-15	1.48E-15	1.36E-15	1.48E-15	1.48E-15	**1.01E-15**	1.36E-15	1.72E-15	3.73E-15
f_{11}	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
f_{12}	8.34E-09	1.05E-09	9.63E-09	3.46E-11	3.28E-11	1.72E-11	1.05E-11	5.04E-12	**2.46E-12**	4.88E-09
f_{13}	3.66E-08	9.87E-08	**1.19E-09**	0.000367	4.58E-08	2.37E-08	1.16E-09	7.99E-08	4.95E-08	2.12E-08
f_{14}	1.45E+00	**9.98E-01**	1.06E+00	1.13E+00	1.39E+00	1.06E+00	**9.98E-01**	1.39E+00	**9.98E-01**	**9.98E-01**
f_{15}	**3.07E-04**	**3.07E-04**	**3.07E-04**	3.08E-04	3.08E-04	3.08E-04	3.08E-04	3.08E-04	3.10E-04	**3.07E-04**
f_{16}	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**	**-1.03163**
f_{17}	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**	**0.397887**
f_{18}	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**
f_{19}	-3.85842	-3.85967	-3.8595	-3.85882	-3.85956	-3.85967	-3.85917	-3.85976	-3.86134	**-3.86238**
f_{20}	-3.21379	-3.24289	-3.22602	-3.2223	-3.22642	-3.2193	-3.21316	-3.23168	-3.22367	**-3.27317**
f_{21}	-9.81333	-9.98479	**-10.1532**	**-10.1532**	**-10.1532**	-9.98479	**-10.1532**	-9.98479	**-10.1532**	**-10.1532**
f_{22}	-10.0486	**-10.4029**	**-10.4029**	-10.2271	-10.4029	-10.4029	-10.2271	-10.2258	**-10.4029**	**-10.40294**
f_{23}	-10.5364	-10.5364	-10.5364	-10.3577	-10.5364	-10.5364	-10.5364	-10.5364	-10.3577	**-10.53641**
number of winners	6	8	12	8	7	6	8	6	10	15

TABLE 15. Rankings of Friedman test under different σ values.

σ	mean rank
Eq.(6)	4.39
0.7	5.13
0.3	5.22
0.6	5.28
0.2	5.48
0.4	5.48
0.8	5.48
0.5	5.50
0.9	5.72
0.1	7.33

TABLE 16. The result obtained by NICS and other enhanced CSA methods.

Fun	C4SA[46]	SCCSA[47]	NICSA	ICSA[27]	NICSA
	D=30	D=30	D=30	D=50	D=50
f_1	1. 26E-27	9.22E-69	7.21E-210	4.06E-30	8.11E-84
f_2	4. 19E-14	8.25E-41	2.74E-191	1.03E-14	2.02E-151
f_3	-	4.31E-31	1.62E-21	-	2.45E-09
f_4	-	1.06E-16	1.30E-04	-	2.00E-04
f_5	-	5.907939	1.32E-04	7.98	2.00E-04
f_6	1. 58E-08	4.14E-08	6.70E-10	3.99E-3	2.85E-07
f_7	-	0.001336	2.72E-04	-	4.20E-04
f_{10}	1. 47E- 21	-	3.73E-15	3.02E-15	4.32E-15

as follows:

$$\min\ f(x) = (2\sqrt{2}\,x_1 + x_2) \times l$$
$$\text{s.t. } g_1(x) = \frac{\sqrt{2}\,x_1 + x_2}{\sqrt{2}\,x_1^2 + 2x_1x_2}P - \sigma \le 0$$
$$g_2(x) = \frac{x_2}{\sqrt{2}\,x_1^2 + 2x_1x_2}P - \sigma \le 0$$
$$g_3(x) = \frac{1}{\sqrt{2}\,x_2 + x_1}P - \sigma \le 0$$

TABLE 17. Comparison of the best solution obtained by different algorithm for three-bar truss design problem.

	NCCO [51]	CSA[22]	HCPS[50]	NICSA
x_1	0.8699	0.7886751284	0.78867515	0.788675135
x_2	0.2164	0.4082483080	0.40824826	0.4082482903
g_1	-5.8363e-08	-1.687539e-14	-7.55×10-14	0
g_2	-1.7008	-1.4641015952	-1.46410165	-1.4641016162
g_3	-0.2992	-0.5358984048	-0.53589835	-0.53589838376
f(x)	279.7245	263.89584338	263.895843	263.895843
rank	3	2	1	1

$$g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \leq 0$$

where $0 \leq x_1 \leq 1, 0 \leq x_2 \leq 1, l = 100cm, P = kN/cm^2, \sigma = 2kN/cm^2$.

The comparison of obtained by other methods such as NCCO [50], CSA [22], and HCPS [49]. The best results of the various methods for solving the three-bar truss design problem are shown in table 17. It can be seen from table 17 that the optimal solution of the NICSA algorithm is better than NCCO and CSA. Compared with HCPS, NICSA provides similar results.

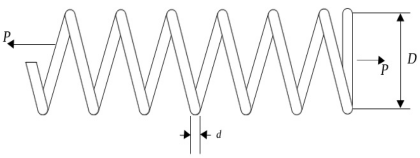

FIGURE 51. Tension-compression spring.

2) TENSION COMPRESSION SPRING DESIGN PROBLEM

As figure 51 indicates, the objective of the tension-compression spring problem is to minimize the weight with four inequality constraints and three continuous variables. The approaches has previously been applied to solve this problem including many different numerical optimization techniques, such as IAPSO [47], CSA [22], and HCPS [49].

$$\begin{aligned}
\min\ & f(x) = (x_3 + 2)x_2 x_1^2 \\
\text{s.t. } & g_1(x) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0 \\
& g_2(x) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} \leq 0 \\
& g_3(x) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0 \\
& g_4(x) = \frac{x_1 + x_2}{1.5} - 1 \leq 0
\end{aligned}$$

where $0.05 \leq x_1 \leq 2, 0.25 \leq x_2 \leq 1.3, 2 \leq x_3 \leq 15$.

Table 18 shows the best solution and the values of the constraints obtained by NICSA and other algorithm for this design problem. According to the table 18, the optimal value of NICSA is superior to that of other 3 methods.

TABLE 18. Comparison of the best solution obtained by different algorithm for tension-compression spring problem.

	IAPSO[48]	CSA[22]	HCPS[50]	NICSA
x_1	0.051685	0.0516890284	0.0516891	0.05
x_2	0.356629	0.3567169544	0.3567178	0.374432870717
x_3	11.294175	11.2890117993	11.288955	8.546569316461
g_1	−1,97E-10	-4.44089e-16	3.46e-6	0
g_2	-4,64E-10	-4.10783e-15	-1.98e-6	-1.1102230E-16
g_3	-4.053610	-4.0537841	-4.053793	-4.86073376802
g_4	-1.091686	-0.7277293	-0.72773	-0.7170447528
f(x)	0.0126653	0.01266523	0.012665	0.00987245556
rank	4	3	2	1

TABLE 19. Comparison of the best solution obtained by different algorithm for pressure vessel design problem.

	IAPSO[48]	CSA[22]	HCS-LSAL [49]	NICSA
x_1	0.8125	0.812500	0.8125	0.7781686413
x_2	0.4375	0.437500	0.4375	0.39034742182
x_3	42.0984	42.098445	42.09844	40.3196187240
x_4	176.6366	176.636598	176.6366	200
g_1	-4.09E-13	-4.024098e-9	−2.01e−09	-4.4408921E-16
g_2	-3.58E-2	-0.0358808	-0.0331588	-0.005698259
g_3	-1.39E-07	-7.122662e-4	-0.002495	-1.3969838E-08
g_4	-63.3634	-63.3634014	-63.3634	-40
f(x)	6059.71433	6059.71436	6059.714	5901.80420020
rank	3	4	2	1

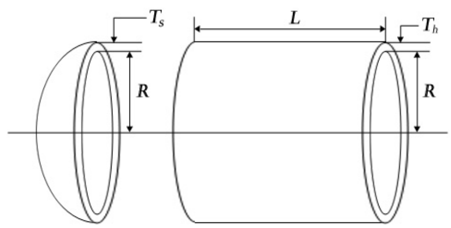

FIGURE 52. The pressure vessel design problem.

3) PRESSURE VESSEL DESIGN PROBLEM

The pressure vessel design problem proposed by Kannan and Kramer is often used to validate the performance of the method for solving constraint problems. Figure 52 shows a pressure vessel with four variables and four nonlinear inequality constraints. The objective function of the problem can be expressed as follows:

$$\begin{aligned}
\min\ & f(x) = 0.6224 x_1 x_3 x_4 + 1.7881 x_2 x_3^2 + 3.1661 x_1^2 x_4 \\
& \quad + 19.84 x_1^2 x_3 \\
\text{s.t. } & g_1(x) = -x_1 + 0.0193 x_3 \leq 0 \\
& g_2(x) = -x_2 + 0.00954 x_3 \leq 0 \\
& g_3(x) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0 \\
& g_4(x) = x_4 - 240 \leq 0
\end{aligned}$$

where $0 \leq x_1, x_2 \leq 100, 10 \leq x_3, x_4 \leq 200$.

The comparison results obtained for the pressure vessel design problem with previous studies including IAPSO [47], CSA [22],and HCS-LSAL [48]. From table 19, it is clear that the best solution obtained by NICSA is better than those by IAPSO, CSA,and HCS-LSAL.

V. CONCLUSIONS

Aiming at the defects of the crow search algorithm based on memory search mode, a new crow search algorithm based on neighborhood search of non-inferior solution is proposed. The new algorithm can make the crow individual choose a suitable foraging search mode through the self-adaptive determination factor of non-inferior solutions. Hence, the algorithm can conduct optimum search more reasonably, and sufficient neighborhood search can be conducted around non-inferior solutions. In neighborhood search, the determination factor of neighborhood search mode was designed. The crow individuals choose the neighborhood search mode of Levy flight or Gaussian flight self-adaptively, so as to enhance the neighborhood search capacity of the algorithm. According to the experimental results about the comparison with other swarm intelligent algorithms in convergence rate, search veracity, variance analysis, nonparametric statistical analysis and parameter sensitivity, the proposed algorithm is effective.

REFERENCES

[1] K. Mao, Q.-K. Pan, X. Pang, and T. Chai, "A novel Lagrangian relaxation approach for a hybrid flowshop scheduling problem in the steelmaking-continuous casting process," *Eur. J. Oper. Res.*, vol. 235, no. 1, pp. 51–60, Jul. 2014.
[2] P. Fattahi, S. M. H. Hosseini, F. Jolai, and R. Tavakkoli-Moghaddam, "A branch and bound algorithm for hybrid flow shop scheduling problem with setup time and assembly operations," *Appl. Math. Model.*, vol. 38, no. 1, pp. 119–134, Jan. 2014.
[3] C. A. Floudas and C. E. Gounaris, "A review of recent advances in global optimization," *J. Global Optim.*, vol. 45, no. 1, pp. 3–38, 2009.
[4] M. S. Nagano, H. H. Miyata, and D. C. Araújo, "A constructive heuristic for total flowtime minimization in a no-wait flowshop with sequence-dependent setup times," *J. Manuf. Syst.*, vol. 36, no. 1, pp. 224–230, Jul. 2015.
[5] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, Nov./Dec. 1995, pp. 1942–1948.
[6] S. Biswas, S. Kundu, and S. Das, "Inducing niching behavior in differential evolution through local information sharing," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 246–263, Apr. 2015.
[7] D. Karaboga, B. Gorkemli, C. Ozturk, and N. Karaboga, "A comprehensive survey: Artificial bee colony (ABC) algorithm and applications," *Artif. Intell. Rev.*, vol. 42, no. 1, pp. 21–57, 2014.
[8] R. Rajabioun, "Cuckoo optimization algorithm," *Appl. Soft Comput.*, vol. 11, no. 8, pp. 5508–5518, Aug. 2011.
[9] X.-S. Yang, M. Karamanoglu, and X. He, "Flower pollination algorithm: A novel approach for multiobjective optimization," *Eng. Optim.*, vol. 46, no. 9, pp. 1222–1237, Apr. 2013.
[10] A. H. Gandomi and A. H. Alavi, "Krill herd: A new bio-inspired optimization algorithm," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 17, pp. 4831–4845, May 2012.
[11] G.-G. Wang, L. H. Guo, A. H. Gandomi, G.-S. Hao, and H. Q. Wang, "Chaotic krill herd algorithm," *Inf. Sci.*, vol. 274, pp. 17–34, Aug. 2014.
[12] S. Mirjalili, "SCA: A sine cosine algorithm for solving optimization problems," *Knowl.-Based Syst.*, vol. 96, pp. 120–133, Mar. 2016.
[13] C. Qu, Z. Zeng, J. Dai, Z. Yi, and W. He, "A modified sine-cosine algorithm based on neighborhood search and greedy levy mutation," *Comput. Intell. Neurosci.*, vol. 2018, Jul. 2018, Art. no. 4231647.
[14] S. Li, H. Fang, and X. Liu, "Parameter optimization of support vector regression based on sine cosine algorithm," *Expert Syst. Appl.*, vol. 91, pp. 63–77, Jan. 2018.
[15] Q.-K. Pan, H.-Y. Sang, J.-H. Duan, and L. Gao, "An improved fruit fly optimization algorithm for continuous function optimization problems," *Knowl.-Based Syst.*, vol. 62, pp. 69–83, May 2014.
[16] M. A. Wei and Z. X. Sun, "A Global Cuckoo Optimization Algorithm Using Coarse-to-Fine Search," *Acta Electronica Sinica*, vol. 43, no. 12, pp. 2429–2439, Dec. 2015.
[17] H. C. Liu and W. U. Zhi-Jian, "Differential evolution algorithm using rotation-based learning," *Acta Electronica Sinica*, vol. 43, no. 10, pp. 2040–2046, Oct. 2015.
[18] A. Baykaso?lu, A. Hamzadayi, and S. Y. Köse, "Testing the performance of teaching–learning based optimization (TLBO) algorithm on combinatorial problems: Flow shop and job shop scheduling cases," *Inf. Sci.*, vol. 276, no. 20, pp. 204–218, Aug. 2014.
[19] S. Abdollahpour and J. Rezaeian, "Minimizing makespan for flow shop scheduling problem with intermediate buffers by using hybrid approach of artificial immune system," *Appl. Soft Comput.*, vol. 28, pp. 44–56, Mar. 2015.
[20] C. Zhang, Z. Shi, Z. Huang, Y. Wu, and L. Shi, "Flow shop scheduling with a batch processor and limited buffer," *Int. J. Prod. Res.*, vol. 55, no. 11, pp. 3217–3233, Nov. 2017.
[21] G. Moslehi and D. Khorasanian, "A hybrid variable neighborhood search algorithm for solving the limited-buffer permutation flow shop scheduling problem with the makespan criterion," *Comput. Oper. Res.*, vol. 52, pp. 260–268, Dec. 2014.
[22] A. Askarzadeh, "A novel metaheuristic method for solving constrained engineering optimization problems: Crow search algorithm," *Comput. Struct.*, vol. 169, pp. 1–12, Jun. 2016.
[23] A. Askarzadeh, "Capacitor placement in distribution systems for power loss reduction and voltage improvement: A new methodology," *IET Gener. Transmiss. Distrib.*, vol. 10, no. 14, pp. 3631–3638, Dec. 2016.
[24] D. Oliva, S. Hinojosa, E. Cuevas, G. Pajares, O. Avalos, and J. Gálvez, "Cross entropy based thresholding for magnetic resonance brain images using crow search algorithm," *Expert Syst. Appl.*, vol. 79, pp. 164–180, Aug. 2017.
[25] S. H. E. A. Aleem, A. F. Zobaa, and M. E. Balci, "Optimal resonance-free third-order high-pass filters based on minimization of the total cost of the filters using Crow Search Algorithm," *Electr. Power Syst. Res.*, vol. 151, pp. 381–394, Oct. 2017.
[26] D. Liu *et al.*, "ELM evaluation model of regional groundwater quality based on the crow search algorithm," *Ecological Indicators*, vol. 81, pp. 302–314, Oct. 2017.
[27] M. Jain, A. Rani, and V. Singh, "An improved crow search algorithm for high-dimensional problems," *J. Intell. Fuzzy Syst.*, vol. 33, no. 6, pp. 3597–3614, Nov. 2017.
[28] G. I. Sayed, A. E. Hassanien, and A. T. Azar, "Feature selection via a novel chaotic crow search algorithm," *Neural Comput. Appl.*, no. 1, pp. 171–188, Apr. 2017.
[29] L. dos Santos Coelho, C. Richter, V. C. Mariani, and A. Askarzadeh, "Modified crow search approach applied to electromagnetic optimization," in *Proc. IEEE Conf. Electromagn. Field Comput. (CEFC)*, Miami, FL, USA, Nov. 2016, p. 1.
[30] F. Mohammadi and H. Abdi, "A modified crow search algorithm (MCSA) for solving economic load dispatch problem," *Appl. Soft Comput.*, vol. 71, pp. 51–65, Oct. 2018.
[31] P. Díaz *et al.*, "An improved crow search algorithm applied to energy problems," *Energies*, vol. 11, no. 3, p. 571, Mar. 2018.
[32] D. Gupta, J. J. P. C. Rodrigues, S. Sundaram, A. Khanna, V. Korotaev, and V. H. C. Albuquerque, "Usability Feature Extraction Using Modified Crow Search Algorithm: A novel approach," in *Neural Computing and Applications*, 2018, pp. 1–11. doi: 10.1007/s00521-018-3688-6.
[33] R. M. Rizk-Allah, A. E. Hassanien, and S. Bhattacharyya, "Chaotic crow search algorithm for fractional optimization problems," *Appl. Soft Comput.*, vol. 71, pp. 1161–1175, Oct. 2018.
[34] S. Hinojosa, D. Oliva, E. Cuevas, G. Pajares, O. Avalos, and J. Gálvez, "Improving multi-criterion optimization with chaos: A novel multi-objective chaotic crow search algorithm," *Neural Comput. Appl.*, vol. 29, no. 8, pp. 319–335, Oct. 2018.
[35] D. Gupta, S. Sundaram, A. Khanna, A. E. Hassanien, and V. H. C. de Albuquerque, "Improved diagnosis of Parkinson's disease using optimized crow search algorithm," *Comput. Electr. Eng.*, vol. 68, pp. 412–424, May 2018.
[36] X. S. Yang, "A new metaheuristic bat-inspired algorithm," in *Nature Inspired Cooperative Strategies for Optimization (NICSO)*. Berlin, Germany: Springer, 2010, pp. 65–74.

[37] X.-S. Yang and A. H. Gandomi, "Bat algorithm: A novel approach for global engineering optimization," *Eng. Comput.*, vol. 29, no. 5, pp. 464–483, 2012.
[38] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.
[39] X.-S. Yang, "Flower pollination algorithm for global optimization," in *Proc. Int. Conf. Unconventional Comput. Natural Comput.* Berlin, Germany: Springer, 2012, pp. 240–249.
[40] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.
[41] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching–learning–based optimization: A novel method for constrained mechanical design optimization problems," *Comput.-Aided Des.*, vol. 43, no. 3, pp. 303–315, Mar. 2011.
[42] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.
[43] K. Tang *et al.*, "Benchmark functions for the CEC'2008 special session and competition on large scale global optimization," in *Nature Inspired Computation and Applications Laboratory*. Anhui Sheng, China: USTC, 2007, pp. 1–24.
[44] L.-Y. Tseng and C. Chen, "Multiple trajectory search for large scale global optimization," in *Proc. IEEE Congr. Evol. Comput. (IEEE World Congr. Comput. Intell.)*, Hong Kong, Jun. 2008, pp. 3052–3059.
[45] Z. Shi-jie *et al.*, "Improved crow search algorithm based on variable-factor weighted learning and adjacent-generations dimension crossover strategy," *Acta Elect Ronica Sinica*, vol. 47, no. 1, pp. 40–48, Jan. 2018.
[46] S. H. R. Pasandideh and S. Khalilpourazari. (2018). "Sine cosine crow search algorithm: A powerful hybrid meta heuristic for global optimization." [Online]. Available: https://arxiv.org/abs/1801.08485
[47] N. B. Guedria, "Improved accelerated PSO algorithm for mechanical engineering optimization problems," *Appl. Soft Comput.*, vol. 40, pp. 455–467, Mar. 2016.
[48] W. Long, X. Liang, Y. Huang, and Y. Chen, "An effective hybrid cuckoo search algorithm for constrained global optimization," *Neural Comput. Appl.*, vol. 25, no. 3, pp. 911–926, Sep. 2014.
[49] W. Long, W. Zhang, Y. Huang, and Y. Chen, "A hybrid cuckoo search algorithm with feasibility-based rule for constrained structural optimization," *J. Central South Univ.*, vol. 21, no. 8, pp. 3197–3204, Aug. 2014.
[50] J. Gálvez, E. Cuevas, S. Hinojosa, O. Avalos, and M. Pérez-Cisneros, "A reactive model based on neighborhood consensus for continuous optimization," *Expert Syst. Appl.*, vol. 121, pp. 115–141, May 2019.
[51] P. Rincon, "Science/nature/crows and jays top bird IQ scale," BBC News.

CHIWEN QU received the M.S. degree in computer software and theory from Southwest Jiaotong University, in 2008. He is currently an Associate Professor with Baise University. He has published over 20 journals and conference papers. His research interests include computation intelligence, neural networks, and intelligence optimization. He has also hosted or participated in many National Natural Science Fund Projects and Provincial Nature Fund Projects. He is a member of the Chinese Computer Society.

YANMING FU received the Ph.D. degree from Sichuan University, in 2011. He is currently an Associate Professor with Guangxi University. He has published a number of journals and conference papers. His research interests include data mining, computation intelligence, and network security.