IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 28, 2021, accepted January 3, 2022, date of publication January 12, 2022, date of current version January 20, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3141908

A Multi-Strategy Seeker Optimization Algorithm for Optimization Constrained Engineering Problems

SHAOMI DUAN[1,2], HUILONG LUO[1], AND HAIPENG LIU[2]
[1]Faculty of Civil Engineering and Mechanics, Kunming University of Science and Technology, Kunming 650500, China
[2]Faculty of Information Engineering and Automation, Kunming University of Science and Technology, Kunming 650500, China

Corresponding author: Huilong Luo (huilongluo@kmust.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 52166001 and Grant 51766005, and in part by the Science and Technology Project of Yunnan Tobacco Company of China under Grant 2019530000241019.

ABSTRACT This paper proposes a multi-strategy seeker optimization algorithm (MSSOA) for optimization constrained engineering problems. In this paper, three strategies were adopted to improve the poor searching capability of the seeker optimization algorithm (SOA). The first strategy was triple black hole system capture to solve the local optima issue. The second and third strategies were the multi-dimensional random interference and the precocious interference to balance the exploration and exploitation processes. These three strategies are proposed to improve respectively the SOA algorithm, and compared with the three strategies to improve together the SOA algorithm for optimizing 15 benchmark functions; the way these three strategies work together is called multi-strategy; and the efficiency of the multi-strategy is illustrated the numerical optimizing results and the convergence curves, population's positions with iterations and the search history of the benchmark functions. The proposed multi-strategy method achieved better performance in optimizing of the benchmark functions compared to other six optimization methods. The numerical and experimental results analysis were observed with respect to the optimal solution curve, the convergence curve of the fitness function, the ANOVA tests, the calculation complexity of the algorithm, the running time of the algorithm routine, the exploration and exploitation capability, the Wilcoxon's rank-sum test, the performance profile of algorithm. The results showed that the proposed multi-strategy method was efficient in the benchmark functions. The proposed multi-strategy method also achieved better performance in optimizing of the engineering problems and provided better solutions compared to other six optimization methods.

INDEX TERMS Seeker optimization algorithm, triple black hole system capture, multi-dimensional random interference, precocious interference, constrained engineering optimization problems.

I. INTRODUCTION

Recently, the heuristic algorithm has received a lot of attention. Such algorithms create random methods for many optimization problems. Since the No Free Lunch (NFL) theorem, no one optimization solution can optimize overall questions [1]. Therefore, researchers pose new algorithms or enhance the current algorithms to deal with optimization problems. The current algorithms are the genetic algorithm (GA) [2], the particle swarm optimization (PSO) [3], the simulated annealing (SA) [4], the harmony search (HS) [5], the gravitational search algorithm (GSA) [6], the moth-flame optimization (MFO) [7], the sine cosine algorithm (SCA) [8], the multi-verse optimizer (MVO) [9], the social network search (SNS) [10], and the seeker optimization algorithm (SOA) [11].

However, some optimization algorithms are still not very successful in optimization problems. The optimization problems include: being premature, issues with low optimization precision, having only a local optimal solution, slow convergence speed, unbalance the exploration and exploitation processes, and insufficient robustness. To better overcome optimization precision, prematurity, having only a local optimal solution, the slow convergence rate, unbalance the

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

exploration and exploitation processes, and poor robustness, some improved algorithms have proven to be feasible optimization algorithms and have been used in practical engineering. For instance, a discrete bat algorithm based on Levy flights is adopted to solve the Euclidean traveling salesman problem [12]. The cuckoo optimization algorithm in reverse logistics is used to design a network for COVID-19 waste management [13]. A cuckoo optimization algorithm based on introducing chaos, a levy flight, opposition learning, and disruption is used to classify the optimal feature subspace [14]. An elite symbiotic organisms search algorithm with mutually beneficial factors was adopted to optimize the functions [15]. An artificial bee colony (ABC) with dynamic Cauchy mutation is adopted to solve feature selection [16]. Polynomial variation combined genetic adaptive search is applied to optimize constrained engineering design problems [17]. The PSO based on time-varying coefficients is applied to optimize the parameters of combustion engines [18]. A new hybrid identification algorithm based on the Average Multi-Verse Optimizer and Sine Cosine Algorithm is presented for identifying the continuous-time Hammerstein system [19].

Therefore, this is a popular trend at present: on the basis of retaining the advantages of the original algorithm, add some strategies to suppress or improve the shortcomings of the algorithm, so that the algorithm is more suitable for some practical optimization problems. This article takes this popular approach to research the SOA algorithm.

Dai *et al.* propose the SOA algorithm in 2006 [20]; the goal is to mimic the seekers' behavior and the way they exchange information and solve practical application optimization problems. In the recent decade, because of the SOA's strength in fast convergence of optimization problems, the SOA algorithm has been used in many fields, such as in unconstrained optimization problems [21], optimal reactive power dispatch [22], a challenging set of benchmark problems [23], the design of a digital filter [24], optimizing parameters of artificial neural networks [25], the optimizing model and structures of fuel cell [26], the novel human group optimizer algorithm [27], and several practical applications [28]. However, in the initial stage of dealing with optimization problems, SOA converges faster than others; When all individuals are near the best individual for solving the optimization problem, the individuals will lose diversity, fall into the local optima region, affect the optimization accuracy of the algorithm, and unbalance the exploration and exploitation processes.

To improve the SOA algorithm, we choose several strategies is applied to increase individual diversity, avoid premature puberty, improve the optimization precision, and balance the exploration and exploitation processes. These strategies are the Levy variation strategy, the refraction reverse learning mechanism strategy, the mutual benefit factor strategy, the Cauchy variation strategy, the polynomial variation strategy, the time-varying compression factor strategy, the triple black hole system capture strategy, the multiple random interference strategy, and the precocious interference strategy.

We have tested the improved SOA algorithms by these above strategies on 15 high dimensional benchmark functions. Only three improvement strategies are listed in this paper, namely, the triple black hole system capture strategy, the multiple random interference strategy, and the precocious interference strategy. The SOA algorithm is improved by introducing these three strategies separately and naming the three improved algorithms as the SOA based on triple black hole system capture (TBHSOA), the SOA algorithm based on multiple interference (MISOA), the SOA algorithm based on premature interference (PISOA). These three improved strategies can increase population diversity, avoid prematurity and improve optimization accuracy while ensuring fast convergence for solving the optimization functions. So, these three strategies are introduced together into the SOA to improve the algorithm in this paper; the way these three strategies work together is called multi-strategy; and the improved SOA algorithm is named the multi-strategy seeker optimization algorithm (MSSOA).

The MSSOA involves several individuals by being captured by three separate black holes. At the same time, the multiple random interference and the precocious interference strategies are introduced into the MSSOA. In addition, the MSSOA, TBHSOA, MISOA, PISOA, and the basic SOA algorithm are tested and compared on 15 high dimensional benchmark functions. According to the experimental results, the convergence speed and the accuracy of MSSOA are higher than that of the other improved SOA algorithms, and the basic SOA. Finally, compared with the PSO, SA_GA, GSA, SCA, MVO, and the SOA, the MSSOA has been implemented and tested on 15 benchmark functions, 5 constrained problem examples, and 6 optimization constrained engineering problems taken from literature. According to the experimental results, the MSSOA have better optimization results than other algorithms in benchmark functions and constrained optimization problems.

The major contributions of this paper are summed up as follows:

- The triple black hole system capture is proposed in SOA, which will help the SOA algorithm to escape from the local optima. The merit of the triple black hole system capture is that increasing the diversity of the current individual can help any trapped individuals to jump out from the local optima region and continue a new search path.
- The multi-dimensional random interference is performed to improve the low searching capability of SOA by adding the variable random interference, which is a proper balance between exploration and exploitation.
- The precocious interference is performed to improve optimization accuracy of SOA by adding the precocious modes of interference, which is a proper balance between exploration and exploitation, while ensuring fast convergence for solving the optimization functions.

- The MSSOA outperforms the PSO, SA_GA, GSA, SCA, MVO, and the SOA in many benchmark functions, constrained problem examples, and constrained engineering optimization problems.

The rest of the article structure is as follows. Part 2 presents the SOA and the algorithm improvement strategies. Part 3 describes the MSSOA. Part 4 shows the algorithm optimization experiments, the results, and the analyses. At last, Part 5 gives some conclusions.

II. BASIC SOA ALGORITHM AND ALGORITHM IMPROVEMENT STRATEGIES

The SOA algorithm carries out in-depth research on human search behavior. It considers optimization as a search for an optimal solution by a search team in search space, taking search team as population and the site of the searcher as task method. Using "experience gradient" to determine the search direction, we use uncertain reasoning to resolve the search step measurement, through the scout direction and search step size to complete the searchers' position in the search interspace update, to attain the optimization of the solution.

A. KEY UPDATE POINTS FOR SOA ALGORITHM

SOA algorithms have three main updating steps.

1) SEARCH DIRECTION

The forward orientation of a search is defined by the experience gradient obtained from the individuals' movement and the evaluation of other individuals' search historical position. The egoistic direction $\overrightarrow{f}_{i,\mathrm{e}}(t)$, altruistic direction $\overrightarrow{f}_{i,\mathrm{a}}(t)$, and preemptive direction $\overrightarrow{f}_{i,\mathrm{p}}(t)$ of the ith individual in any dimension can be obtained.

$$\overrightarrow{f}_{i,\mathrm{e}}(t) = \vec{p}_{i,\mathrm{best}} - \vec{x}_i(t) \tag{1}$$

$$\overrightarrow{f}_{i,a}(t) = \vec{g}_{i,\mathrm{best}} - \vec{x}_i(t) \tag{2}$$

$$\overrightarrow{f}_{i,\mathrm{p}}(t) = \vec{x}_i(t_1) - \vec{x}_i(t_2) \tag{3}$$

The searcher uses the method of a random weighted average to obtain the search orientation.

$$\overrightarrow{f}_i(t) = \mathrm{sign}(\omega \overrightarrow{f}_{i,\mathrm{p}}(t) + \psi_1 \overrightarrow{f}_{i,\mathrm{e}}(t) + \psi_2 \overrightarrow{f}_{i,\mathrm{a}}(t)) \tag{4}$$

where: $t_1, t_2 \in \{t, t-1, t-2\}$, $\vec{x}_i(t_1)$ and $\vec{x}_i(t_2)$ are the best advantages of $\{\vec{x}_i(t-2), \vec{x}_i(t-1), \vec{x}_i(t)\}$ separately; $g_{i,best}$ is the historical optimal location in the neighborhood where the ith search factor is located; $p_{i,best}$ is the optimal locality from the ith search factor to the current locality; ψ_1 and ψ_2 are random numbers in [0,1]. ω is the weight of inertia.

2) SEARCH STEP SIZE

The SOA algorithm refers to the reasoning of the fuzzy approximation ability. The SOA algorithm, through the computer language, describes some of the human natural languages that can simulate human intelligence reasoning search behavior. If the algorithm expresses a simple fuzzy rule, it adapts to the best approximation of the objective optimization problems. The greater search step length is more important. However, the smaller fitness step length makes the corresponding smaller. The Gaussian distribution function is adopted to describe the search step measurement.

$$\mu(\alpha) = e^{-\frac{\alpha^2}{2\delta^2}} \tag{5}$$

where, α and δ are parameters of a membership function.

According to equation (5), the probability of the output variable exceeding $[-3\delta, 3\delta]$ is less than 0.0111. Therefore, $\mu_{\min} = 0.0111$. Under normal circumstances, the optimal position of an individual has $\mu_{\max} = 1.0$, and the worst place is 0.0111. However, to accelerate the convergence speed and get the optimal individual to have an uncertain step size, $\mu_{\max}$ is set as 0.9 in this paper. Select the following function as the fuzzy variable with a "small" target function value:

$$\mu_i = \mu_{\max} - \frac{s - I_i}{s - I}(\mu_{\max} - \mu_{\min}), \quad i = 1, 2, \ldots, s \tag{6}$$

$$\mu_{ij} = rand(\mu_i, 1), \quad j = 1, 2, \ldots, D \tag{7}$$

where: μ_{ij} is determined by equations (6) and (7), and I_i is the count of the sequence $x_i(t)$ of the current individuals arranged from high to low by function value. And the function rand $(\mu_i, 1)$ is the real number in any partition $[\mu_i, 1]$.

It can be seen from equation (6) that it simulates the random search behavior of human beings. Step measurement of j-dimensional search interspace is determined by equation (8):

$$\alpha_{ij} = \delta_{ij}(t) - \sqrt{-\ln(\mu_{ij})} \tag{8}$$

where, δ_{ij} is a parameter of the Gaussian distribution function, which is defined by equation (9):

$$\delta_{ij} = \omega x * \mathrm{abs}(\overrightarrow{x}_{\min} - \overrightarrow{x}_{\max}) \tag{9}$$

where, ω is the weight of inertia. As the evolutionary algebra increases, ω decreases linearly from 0.9 to 0.1. $\overrightarrow{x}_{\min}$ and $\overrightarrow{x}_{\max}$ are respectively the variate of the minimum value and maximum value of the function.

3) INDIVIDUAL LOCATION UPDATES

After obtaining the scout direction and scout step measurement of the individual, the location update is represented by (10):

$$x_{ij}(t+1) = x_{ij}(t) + \alpha_{ij}(t) f_{ij}(t), \quad i = 1, 2, \ldots, s; \quad j = 1, 2, \ldots, D \tag{10}$$

i denotes the ith searcher individual, j denotes the individual dimension; $f_{ij}(t)$ and $\alpha_{ij}(t)$ respectively represent the searchers' search direction and search step size at time t, $x_{ij}(t)$ and $x_{ij}(t+1)$ respectively represent the searchers' site at time t and $(t+1)$.

B. ALGORITHM IMPROVEMENT STRATEGIES

Four improved SOA algorithms based on three different strategies are listed in this paper. The three different strategies are: triple black hole system capture strategy,

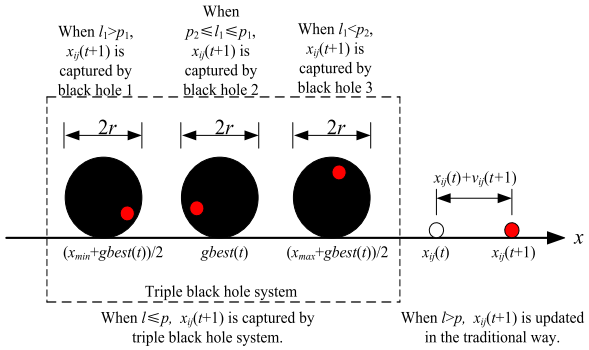

FIGURE 1. Schematic diagram of seeker position update in a triple black hole system.

multi-dimensional random interference strategy, and precocious interference strategy. The four improved algorithms are: the SOA based on triple black hole system capture (TBHSOA), the SOA algorithm based on multiple interference (MISOA), the SOA algorithm based on premature interference (PISOA), and the multi-strategy seeker optimization algorithm (MSSOA). The three improvement strategies used in this article are described below.

1) TRIPLE BLACK HOLE SYSTEM CAPTURE STRATEGY

In the merger process of cosmic galaxies, multiple binary black hole system will be generated. If the lifetime of the binary black hole system is long enough, it will merge with other galaxies and form the triple black hole systems capture. Three black holes interact with each other, and due to gravity, the seekers have a certain probability of being captured by the black hole system [29], [30]. Inspired by this, the seeker trapping mechanism of the triple black hole system capture is introduced into the SOA algorithm to get away from local optima region and enhance the global search ability.

As shown in Figure 1, the constant threshold $p \in [0,1]$ is set as the probability of seeker x_i being captured by a triple black hole system. For each seeker x_i, a random number $l \in [0,1]$ is generated in each iteration. If $l \leq p$, x_i is arrested by a triple black hole system; otherwise, it is updated in the traditional way.

If x_i is captured by a triple black hole system, triple black hole regions are formed with $gbest(t)$, $(gbest(t)+x_{\max})/2$ and $(gbest(t)+x_{\min})/2$ as the centers and r as the radius of the black hole producing a random number $l_1 \in [0,1]$. $l_1 > p_1$, x_{ij} is captured by black hole 1 in the system; $l_1 \in [p_2, p_1]$, x_{ij} is captured by black hole 2; the $l_1 < p_2$, x_{ij} is captured by black hole 3, and the position of the captured seeker is:

$$x_{ij}(t+1)=\begin{cases}(gbest(t)+x_{\min})/2+rr_3, & l_1 > p_1\\ gbest(t)+rr_3, & p_2 \leq l_1 \leq p_1\\ (gbest(t)+x_{\max})/2+rr_3, & l_1 < p_2\end{cases} \tag{11}$$

where: i represents the ith individual, j represents the individual dimension; $gbest(t)$ is the global optimal solution of generation t in the entire population; $x_{\max}/x_{\min}$ is the upper/lower limit of the seeker search region, the constant threshold $p_1, p_2 \in [0,1]$, and $p_1 < p_2$, r_3 is a random number $[-1,1]$.

Since by calculating the average distance between $gbest$ and $x_{\min}$ in (11), $x_{\min}$ can be pull out the $gbest(t)$ from trapped in local optima. In other words, by introducing this average between $gbest$ and $x_{\min}$, $x_{ij}(t+1)$ can choose different black holes, and emerge a new individual, then the diversity of the current individual is increased to avoid falling into local optimum.

2) MULTI-DIMENSIONAL RANDOM INTERFERENCE STRATEGY

Small random disturbances caused by small environmental changes can correct the developmental errors of plants, which is called developmental channelization. The combination of plasticity and the developmental channelization improves the adaptability to the environment and the stability of plant development [31], [32]. Inspired by this, the multi-dimensional random interference tactics are introduced to add the small amplitude change of the scout, and to enhance the local scout capacity. The constant threshold $pp \in [0,1]$ is a random value $k \in [0,1]$ for each dimension of each seeker. If $k \leq pp$, the interference strategy is adopted.

$$x_{ij}(t+1) = x_{ij}(t) + (1+\psi r_4) \tag{12}$$

where: ψ is the interference degree, and r_4 is the random number $[-1,1]$.

In order to understand the multi-dimensional random interference strategy, a graphical representation is showed in Figure 2. Here, a two-dimensional individual (d = 2) is considered. As shown in Figure 2, $gbest$ (red square) is global best individual; the current individual $x_{ij}(t)$ (green square) is trapped in the local optima region (red circle). Based on the conventional SOA, there is a high possibility of $x_{ij}(t)$ remaining trapped in local optima. Nevertheless, this problem can be solved if the multi-dimensional random interference strategy is introduced in $x_{ij}(t)$, which is confirmed by $x_{ij}(t+1)$ (blue square) in Figure 2. In particular, $x_{ij}(t+1)$ may pull out $x_{ij}(t)$ from trapped in local optima to the design local optima region ((blue circle), and then a new searching path can be continue explored.

3) PRECOCIOUS INTERFERENCE STRATEGY

When equation (13) is satisfied, the seekers' position is reset so that they are randomly distributed around $gbest(t)$, to potentially jump out of local optimality, namely:

$$\left|F_g(t+1) - F_g(t)\right| < 0.01 \cdot \left|F_g(t+1)\right| \tag{13}$$

$$x_{ij}(t+1) = (gbest'(t+1) + gbest(t)) \cdot r_a \tag{14}$$

where: $F_g(t)/F_g(t-1)$ are the function values corresponding to the global optimal of the $t/t-1$ generation respectively, and r_a is the random number $[-1,1]$. $gbest'(t+1)$ is the current optimal solution of generation $t+1$.

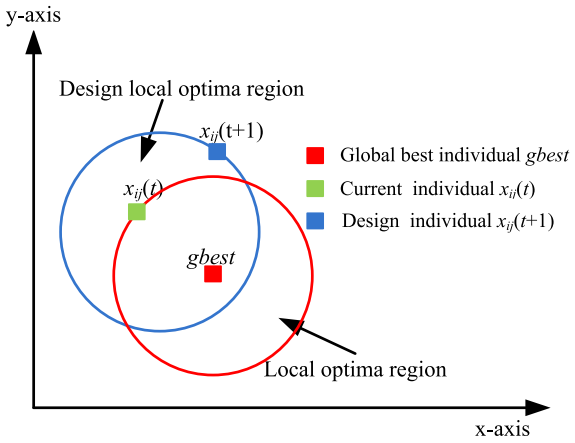

FIGURE 2. Graphical representation of the multi-dimensional random interference strategy.

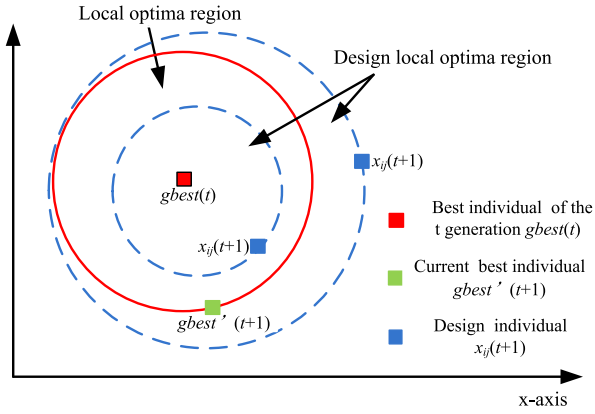

FIGURE 3. Graphical representation of the precocious interference strategy.

Figure 3 is a graphical representation of the precocious interference strategy. Here, a two-dimensional individual (d=2) is considered. As shown in Figure 3, *gbest*(t) (red square) is best individual of the t generation; the current best individual *gbest'*($t+1$) (green square) is trapped in the local optima region (red circle). Based on the conventional SOA, there is a high possibility of *gbest'*($t+1$) remaining trapped in local optima region. Since the current individual $x_{ij}(t)$ is hovering around the current best individual *gbest'*($t+1$), the current individual $x_{ij}(t)$ will also remain trapped in the local optima region. Nevertheless, this problem can be solved if the precocious interference is introduced in $x_{ij}(t+1)$, which is confirmed by $x_{ij}(t+1)$ (blue square) in Figure 3. In particular, $x_{ij}(t+1)$ may pull out *gbest'*($t+1$) and $x_{ij}(t)$ from trapped in local optima region to the design local optima region (blue circle), then a new individual can be emerged, and a new searching path can be continue explored. Here, to equation (13), since r_a is the random number $[-1,1]$, and the search step size and search direction of the SOA is a variable, then $x_{ij}(t+1)$ is the random number $[x_{min}, x_{max}]$, and the design local optima region (blue circle) can be smaller or bigger than the local optima region (red circle).

III. MSSOA ALGORITHM

The MSSOA involves some individuals in introducing the triple black hole system capture and interference strategies to improve the optimization ability of the SOA. Algorithm 1 is the primary process of the MSSOA.

IV. EXPERIMENTAL RESULTS

A. EXPERIMENTAL SETUP

The algorithms used in the experiment in this paper were running under MATLAB R2016a. The computer is configured as Intel (R) Core (TM) i7-7500U CPU @2.7GHz 2.9 GHz processor with 8 GB of memory, Windows 10 operating system.

B. ALGORITHM PERFORMANCE COMPARISON IN BENCHMARK FUNCTIONS

To ensure that the comparison of these algorithms is fair, the population number of algorithms is 30, and the evolutionary algebra is 1000. At the same time, for further ensuring the fairness of algorithm comparison and reducing the effect of randomness, the results of the algorithms after 30 independent runs were selected for comparison.

Algorithm 1 : MSSOA

```
Generate an initial population of N individuals X_MSSOA,G.
Compute the fitness. Determine the optimal solution P_best,G.
While iteration (t) limit is not satisfied.
    Generate the search direction. equation (4)
    Generate the search step size. equation (8)
    Generate a new position X_MSSOA,G. formula (10)
    If f(X_MSSOA,G) ≤ P_best,G
        P_best,G = f(X_MSSOA,G)
    end if
    if rand < P_m
        Adding the triple black hole system capture
        strategy obtain new X_MSSOA,G. equation (11)
            If f(X_MSSOA,G) ≤ P_best,G
                P_best,G = f(X_MSSOA,G)
            end if
        Adding the multi-dimensional random
        interference obtain new X_MSSOA,G. equation (12)
            If f(X_MSSOA,G) ≤ P_best,G
                P_best,G = f(X_MSSOA,G)
            end if
        Adding the precocious interference obtain new
        X_MSSOA,G. equations (13) (14)
            If f(X_MSSOA,G) ≤ P_best,G
                P_best,G = f(X_MSSOA,G)
            end if
    end if
end while
```

1) THE BENCHMARK FUNCTIONS

In this field, it is common to base the capability of algorithms on mathematic functions that are known to be globally optimal. 15 benchmark functions in the literature are used

TABLE 1. Description of benchmark function.

Name	Test functions	Search	Minimum
Sphere	$f_1(x)=\sum_{i=1}^{n}x_i^2$	[-100,100]	0
Schwefel 2.22	$f_2(x)=\sum_{i=1}^{n}\lvert x_i\rvert+\prod_{i=1}^{n}\lvert x_i\rvert$	[-10,10]	0
Rotated hyperellipsoid	$f_3(x)=\sum_{i=1}^{n}\sum_{j=1}^{i}x_j^2$	[-100,100]	0
Schwefel 2.21	$f_4(x)=\max_i\{\lvert x_i\rvert,1\le i\le n\}$	[-100,100]	0
Rosenbrock	$f_5(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	[-30,30]	0
Step	$f_6(x)=\sum_{i=1}^{n}(\lvert x_i+0.5\rvert)^2$	[-100,100]	0
Quartic	$f_7(x)=\sum_{i=1}^{n}ix_i^4+rand\mathrm{m}[0,1)$	[-1.28,1.28]	0
SumSquares	$f_8(x)=\sum_{i=1}^{n}ix_i^2$	[-10,10]	0
SumPower	$f_9(x)=\sum_{i=1}^{n}\lvert x_i\rvert^{(i+1)}$	[-1.28,1.28]	0
Schwefel	$f_{10}(x)=-\sum_{i=1}^{n}x_i\sin(\sqrt{\lvert x_i\rvert})$	[-500,500]	-418.9829*Dimension
Rastrigin	$f_{11}(x)=\sum_{i=1}^{n}[x_i^2-10\cos(2\pi x_i)+10]$	[-5.12,5.12]	0
Ackley	$f_{12}(x)=20+e-20e^{-0.2}\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}-e^{\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)}$	[-32,32]	0
Griewank	$f_{13}(x)=\frac{1}{4000}\sum_{i=1}^{n}x_i^2-\prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}})+1$	[-600,600]	0
Penalized1	$f_{14}(x)=\frac{\pi}{n}\{10\sin^2(\pi y_1)+\sum_{i=1}^{n-1}(y_i-1)^2[1+\sin^2(\pi y_{i+1})]$ $+(y_n-1)^2\}+\sum_{i=1}^{n}u(x_i,10,100,4)$ $y_i=1+\frac{1}{4}(x_i+1),u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m,x_i>a\\0,-a\le x_i\le a\\(-x_i-a)^m,x_i<-a\end{cases}$	[-50,50]	0
Penalized2	$f_{15}(x)=\frac{1}{10}\{\sin^2(3\pi x_1)+\sum_{i=1}^{n-1}(x_i-1)^2[1+\sin^2(3\pi x_{i+1})]$ $+(x_n-1)^2[1+\sin^2(2\pi x_n)]\}+\sum_{i=1}^{n}u(x_i,5,100,4)$ $u(x_i,a,k,m)=\begin{cases}k(x_i-a)^m,x_i>a\\0,-a\le x_i\le a\\(-x_i-a)^m,x_i<-a\end{cases}$	[-50,50]	0

as the comparative test platform [7], [11], [33]–[35]. The functions $f1$-$f9$ are the unimodal benchmark functions. The functions $f10$-$f15$ are the multimodal benchmark functions. Table 1 shows the functions in the experiment. Variables are set to 100.

2) ALGORITHMS PERFORMANCE COMPARISON OF THE SOA ALGORITHM WITH DIFFERENT IMPROVEMENT METHODS

There are various strategies for improving the SOA algorithms, such as the SOA algorithm based on the Levy variation, the SOA algorithm based on the refraction reverse learning mechanism, the SOA algorithm based on the mutually beneficial factor strategy, the SOA algorithm based on the Cauchy variation, the SOA algorithm based on the polynomial variation, the SOA algorithm based on the time-varying compression factor strategy, the SOA algorithm based on triple black hole system capture, the SOA algorithm based on multiple interference, the SOA algorithm based on premature interference, etc. After the experiment and comparison, four different improved SOA algorithms are selected in this paper.

The four algorithms are: the SOA based on triple black hole system capture (TBHSOA), the SOA algorithm based on multiple interference (MISOA), the SOA algorithm based on premature interference (PISOA), and the multi-strategy seeker optimization algorithm (MSSOA).

3) PARAMETER SETTING OF SOA ALGORITHM WITH DIFFERENT IMPROVEMENT METHODS

This section will introduce the parameter setting of the improved SOA algorithms used in the experiment in this paper. Dai *et al.* have done a lot of research on the parameter set of the SOA [24], and we did a lot of practice tests and comparative studies about the parameters. The same parameters of the SOA and the improved SOA algorithm are: the maximum membership degree value is 0.95; the minimum membership degree value is 0.0111; the maximum inertia weight value is 0.9; the minimum inertia weight value is 0.1; and the empirical value is: 0.2. The specific parameters of the improved SOA algorithm are shown in Table 2. The next section will use these improved algorithms for experimental comparison and choose a relatively optimal improved algorithm to compare with other advanced intelligent algorithms.

TABLE 2. Parameter settings of SOA algorithm with different improvement methods.

Algorithm	Parameters and Value
TBHSOA	The triple black hole system capture probability: 0.8.
MISOA	The multiple interference probability: 0.8.
PISOA	The premature interference probability: 0.8.
MSSOA	The multi-strategy probability: 0.8.

4) IMPROVED ALGORITHMS PERFORMANCE COMPARISON IN BENCHMARK FUNCTIONS

The SOA is improved in four different ways: using the seeker optimization algorithm based on triple black hole system capture (TBHSOA), the SOA algorithm based on multiple interference (MISOA), the SOA algorithm based on premature interference (PISOA), and the multi-strategy seeker optimization algorithm (MSSOA). To test the performance, each improved algorithm was optimized for the 15 functions in Table 1. Each algorithm and each function were run independently 30 times. The performance of the SOA and the four improved SOA in 15 function optimizations were compared by the mean (Mean), standard deviation (Std.), the best fitness (Best), the program running time (Time), and the best fitness rank (Rank) of 30 running results. The optimal fitness reflects the optimization veracity of the algorithm, the average value and Std. reflect the robustness of the algorithms, and the running time reflects the time of the program. The results of the functions $f1$-$f15$ are displayed in Table 3. The underline and boldface indicate the better result.

Based on Table 3, for the benchmark functions $f1$-$f15$, the comparison between the four improved SOA algorithms in this paper and the original SOA algorithms shows that the optimization result of the MSSOA is the best value. The mean (Mean), standard deviation (Std.), best fitness (Best), and best fitness rank (Rank) of the MSSOA algorithm were the best after 30 independent runs. The $f1$-$f15$ total program running time (Time) ranks the fifth among all of the 5 algorithms compared in this paper. To the total running time, the running time of the MSSOA is longer than other algorithms. So, the MSSOA algorithm is the one with the best optimization performance among the improved algorithms in this paper. The section IV.B.3) will compare the MSSOA algorithm with the other intelligent optimization algorithms that are widely used at present.

5) SEARCH HISTORY OF EACH IMPROVED SOA

Figure 4 shows the graph of the optimized function $f1$, the initial population's positions, the convergence curves, the population's positions with iterations and search history. Based on Figure 4, for the benchmark function $f1$, given the same evolutionary algebra, the same number of populations, and the same initial individual position, the comparison between the four improved SOA algorithms in this paper and the original SOA algorithm show that the optimization result of the MSSOA is the best value. The convergence curves of the MSSOA algorithm were the fastest. From the population's positions with iterations and the search history of the four improved SOA algorithms, along with the original SOA algorithm show that the MSSOA can close to the optima is faster and more accurate than that of other improved SOA algorithms.

Similarly, Figure 5 shows the graph of the optimized function $f10$, the initial population's positions, the convergence curves, the population's positions with iterations and the search history. Based on Figure 5, for the benchmark function $f10$, given the same evolutionary algebra, the same number of populations, and the same initial individual position, the comparison between the four improved SOA algorithms in this paper and the original SOA algorithm show that the optimization result of the MSSOA is the best value. The convergence curves of the MSSOA algorithm were the fastest. From the population's positions with iterations and the search history of the four improved SOA algorithms, along with the original SOA algorithm show that the MSSOA can close to the optima is faster and more accurate than that of other improved SOA algorithms.

6) ALGORITHMS PERFORMANCE COMPARISON OF DIFFERENT ALGORITHMS IN BENCHMARK FUNCTIONS

To test the performance of the MSSOA, the MSSOA is compared to the PSO, SA-GA, GSA, SCA, MVO, and the SOA, using 15 benchmark functions [7], [11], [33]–[35] in Table 1, which have been widely used in the test.

a: PARAMETER SETTING OF DIFFERENT ALGORITHMS

In this section, the parameters set of the PSO [36], SA_GA [37], GSA [6], SCA [8], MVO [9], SOA [20], and the MSSOA algorithm are presented. According to the references [6], [8], [9], [20], [36], [37], we did a lot of practice

TABLE 3. Performance comparison of different strategies improvement SOA of 30 independent runs for benchmark functions.

Test functions	Result	Algorithms				
		SOA	TBHSOA	MISOA	PISOA	MSSOA
f_1 (D=100)	Mean	1.0524957	0.2825531	5.2400567e-08	**0**	**0**
	Std.	0.2051906	0.3701628	6.3218468e-08	**0**	**0**
	Best	0.6215967	0.0022148	2.5821995e-09	**0**	**0**
	Time	**45.50061**	75.613686	59.635564	66.563613	87.175798
	Rank	5	4	3	**1**	**1**
f_2 (D =100)	Mean	13.002085	0.6511914	0.0014402	1.7425093e-182	**2.012e-184**
	Std.	2.0008395	0.1045537	0.0010004	**0**	**0**
	Best	9.4729280	0.4610739	6.1512225e-05	2.1326872e-209	**4.588e-213**
	Time	**54.16754**	69.699251	94.375281	91.567577	87.175798
	Rank	5	4	3	2	1
f_3 (D =100)	Mean	9.869e+03	3.4564429e+03	7.4247780e-07	**0**	**0**
	Std.	3.472e+03	3.0793769e+03	9.1072042e-07	**0**	**0**
	Best	3.318e+03	8.7199457	3.8240885e-08	**0**	**0**
	Time	**423.27283**	644.844800	594.660175	487.237550	732.835497
	Rank	5	4	3	**1**	**1**
f_4 (D =100)	Mean	21.743821	9.2885561	6.0481089e-05	1.0073412e-185	**2.540e-186**
	Std.	5.6809434	5.2325965	2.8140037e-05	**0**	**0**
	Best	2.9045088	0.6034756	2.1288138e-05	6.8815172e-211	**3.878e-216**
	Time	**47.98446**	68.785997	64.831962	65.931310	106.133452
	Rank	5	4	3	2	**1**
f_5 (D = 100)	Mean	7.093e+02	66.3920291	98.0620155	98.0684285	**0.0052165**
	Std.	1.675e+02	68.7817939	0.02777166	0.02988903	**0.0147478**
	Best	2.789e+02	14.7163775	98.0017736	97.9958827	**7.268e-05**
	Time	**54.515291**	79.939259	70.650258	72.664630	102.749566
	Rank	5	2	4	3	**1**
f_6 (D = 100)	Mean	1.1365348	1.5926423	0.0128676	0.0115968	**0.0081827**
	Std.	0.1751583	1.5709870	0.0029328	**0.0024841**	0.0028444
	Best	0.7992740	0.0080698	0.0090366	0.0076458	**6.152e-04**
	Time	**47.876927**	68.359415	62.122474	65.155501	89.505699
	Rank	5	3	4	2	**1**
f_7 (D = 100)	Mean	3.4645595	0.0042264	2.7057509e-04	7.9075623e-05	**7.033e-05**
	Std.	0.9443482	0.0028720	2.2086648e-04	6.1143005e-05	**4.849e-05**
	Best	2.0472780	8.8641644e-04	7.7950743e-06	7.5917519e-06	**4.709e-06**
	Time	**90.52067**	122.107637	117.053413	116.902097	169.388307
	Rank	5	4	3	2	**1**
f_8 (D = 100)	Mean	1.391e+02	0.2874750	2.1577791e-06	**0**	**0**
	Std.	33.925547	0.0654072	2.5515526e-06	**0**	**0**
	Best	79.288776	0.1770732	3.2599115e-08	**0**	**0**
	Time	**55.51168**	71.962022	63.516109	65.714151	118.793157
	Rank	5	4	3	**1**	**1**
f_9 (D = 100)	Mean	1.471e-05	6.9268201e-10	2.5515595e-12	**0**	**0**
	Std.	1.279e-05	1.1513070e-09	4.7865420e-12	**0**	**0**
	Best	6.318e-07	2.78443943e-12	4.0866578e-15	**0**	**0**
	Time	**102.88770**	135.670100	125.788498	120.054059	223.453628
	Rank	5	4	3	**1**	**1**
f_{10} (D = 100)	Mean	-2.324e+4	**-4.1860098e+04**	-2.4719241e+04	-2.6740043e+04	-4.185e+4
	Std.	3.396e+03	**1.6712302e+02**	4.2241689e+03	4.5932627e+03	1.998e+02
	Best	-3.1737782e+04	-4.1898286e+04	-3.6722877e+04	-3.9445108e+04	**-4.1898289e+04**
	Time	**60.474337**	83.551939	**77.701606**	81.954330	117.516091
	Rank	5	2	4	3	1

TABLE 3. *(Continued.)* **Performance comparison of different strategies improvement SOA of 30 independent runs for benchmark functions.**

f_{11} (D =100)	Mean	4.147e+02	8.3896687	9.3858180e-06	**0**	**0**
	Std.	47.133648	18.669241	9.178772e-06	**0**	**0**
	Best	3.230e+02	0.213578	4.3939895e-07	**0**	**0**
	Time	**59.59937**	85.975021	**62.415032**	57.637555	88.386537
	Rank	5	4	3	**1**	**1**
$f12$ (D = 100)	Mean	2.4523846	0.1090458	6.6760265e-05	**8.8817842e-16**	8.882e-16
	Std.	0.3106584	0.0868174	4.4294773e-05	**0**	**0**
	Best	1.9634230	0.0158898	3.2208117e-06	**8.8817842e-16**	8.882e-16
	Time	64.89326	79.786223	68.112087	**63.628471**	122.176144
	Rank	5	4	3	**1**	2
$f13$ (D = 100)	Mean	0.5354222	0.65109164	1.1369729e-09	**0**	**0**
	Std.	0.3343660	2.0970686	1.0097485e-09	**0**	**0**
	Best	0.0806019	0.0017955	7.6736506e-11	**0**	**0**
	Time	**63.67314**	87.968551	73.914279	65.874340	90.052169
	Rank	5	4	3	**1**	**1**
$f14$ (D = 100)	Mean	20.231984	3.2901206	0.0339106	0.0361956	**1.362e-04**
	Std.	8.8688013	5.7005994	0.0121061	0.0094315	**3.249e-05**
	Best	8.9451797	1.8127161e-04	0.0065775	0.0017787	**8.049e-05**
	Time	**165.94752**	211.450166	206.868663	194.971999	310.154355
	Rank	5	2	4	3	**1**
$f15$ (D = 100)	Mean	1.301e+02	40.0168746	9.8130257	9.8956288	**0.0072373**
	Std.	62.979168	66.5622861	0.4538710	0.0020314	**0.0054797**
	Best	1.7998261	0.0088686	7.4099512	9.8927602	**5.371e-06**
	Time	**166.65126**	240.353899	223.978961	182.987716	302.614372
	Rank	3	2	4	5	**1**
Average Rank		4.86666667	3.4	3.33333333	1.93333333	1.06666667
Overall Rank		5	4	3	2	1

TABLE 4. **The parameters set of different algorithms.**

Algorithm	Parameters and Value
PSO [36]	Constant inertia: 0.9~0.4, two acceleration coefficients: 1.4962.
SA_GA [37]	Select probability:0.6, crossover probability:0.7, mutation scale factor:0.05, initial temperature:100, temperature reduction parameter:0.98.
GSA [6]	The gravitational constant: G0=100, alfa=20.
SCA [8]	The random numbers: $r1$=0~2, $r2$=0~2π, $r3$=0~2, $r4$=0~1.
MVO [21]	The wormhole existence probability: WEP_Max =1, WEP_Min=0.2, travelling distance rate: TDR=0~1, the random numbers: $r1$=0~1, $r2$=0~1, $r3$=0~1.
SOA [20]	The maximum membership degree value:0.95, the minimum membership degree value:0.0111, the maximum inertia weight value:0.9, the minimum inertia weight value:0.1, the empirical value:5 (Same as the Section III.B.2)).
MSSOA	The maximum membership degree value:0.95, the minimum membership degree value:0.0111, the maximum inertia weight value:0.9, the minimum inertia weight value:0.1, the empirical value:0.2, the multi-strategy probability: 0.8 (Same as the Section III.B.2)).

tests and comparative studies for the parameters set. Table 4 is the parameters set of different algorithms.

b: RESULTS COMPARISON OF DIFFERENT ALGORITHMS IN BENCHMARK FUNCTIONS

The mean values, standard deviation, best fitness, best fitness rank between the algorithms of 30 all alone runs, and the data of the functions' $f1$-$f15$ optimization results are shown in Table 5. The underline and boldface indicate that the optimal outcome is better.

Based on Table 5, for the best value of the benchmark functions, the MSSOA is better than the others. For the standard deviation, the MSSOA is better than the others. For the mean, the MSSOA is also better than the others. According to the optimal fitness value mean rank and all rank results from Table 5, the MSSOA has a strong optimization ability to the benchmark functions.

c: THE PRECISION OF ALGORITHMS ANALYSIS

Figure 6 is a comparison of the optimal value of each function optimized by different algorithms and the theoretical optimal value when seven algorithms optimize the benchmark functions $f1$-$f15$ ($D = 100$) respectively. To ensure that the comparison of these algorithms is fair, the population number of algorithms is 30, and the evolutionary algebra is 1000. The results of the seven algorithms after 30 independent runs are selected for comparison. As seen from Figure 6, the optimal solution curves of each function obtained by the MSSOA algorithm are in maximum agreement with the theoretical optimal solution curves, and the accuracy of the MSSOA is better.

d: THE CONVERGENCE OF ALGORITHMS ANALYSIS

Figure 7 shows the fitness curves of the best values for the benchmark functions $f1$-$f15$ (D =100). As seen from Figure 7, the convergence of MSSOA is faster, and the accuracy of the MSSOA is better. The other hand is that the fitness convergence curve of MSSOA algorithm has a steep slope,

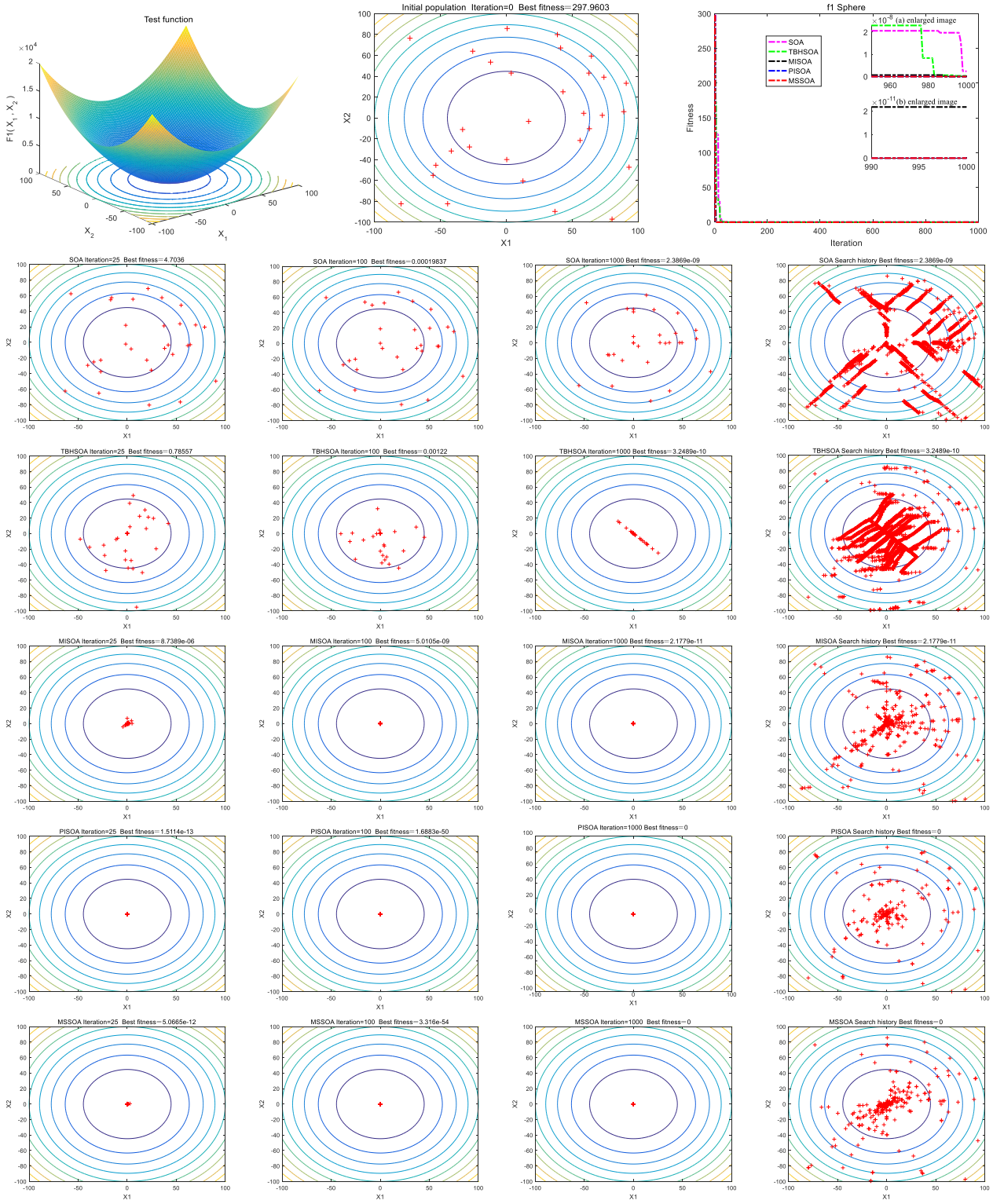

FIGURE 4. The graph of the function f1, initial population's positions, convergence curves, population's positions with iterations and search history.

and it means that the MSSOA has a fast convergence rate, and this shows that the MSSOA can exploit the search space of the problems in a very convenient behavior. This manner can show that the exploration and exploitation abilities of MSSOA algorithm is very well.

e: THE ROBUSTNESS OF ALGORITHMS ANALYSIS

Figure 8 is the analysis of variance (ANOVA) test for the benchmark functions $f1$-$f15$ ($D = 100$). To ensure that the comparison of these algorithms is fair, the population number of algorithms is 30, and the evolutionary algebra is 1000. The

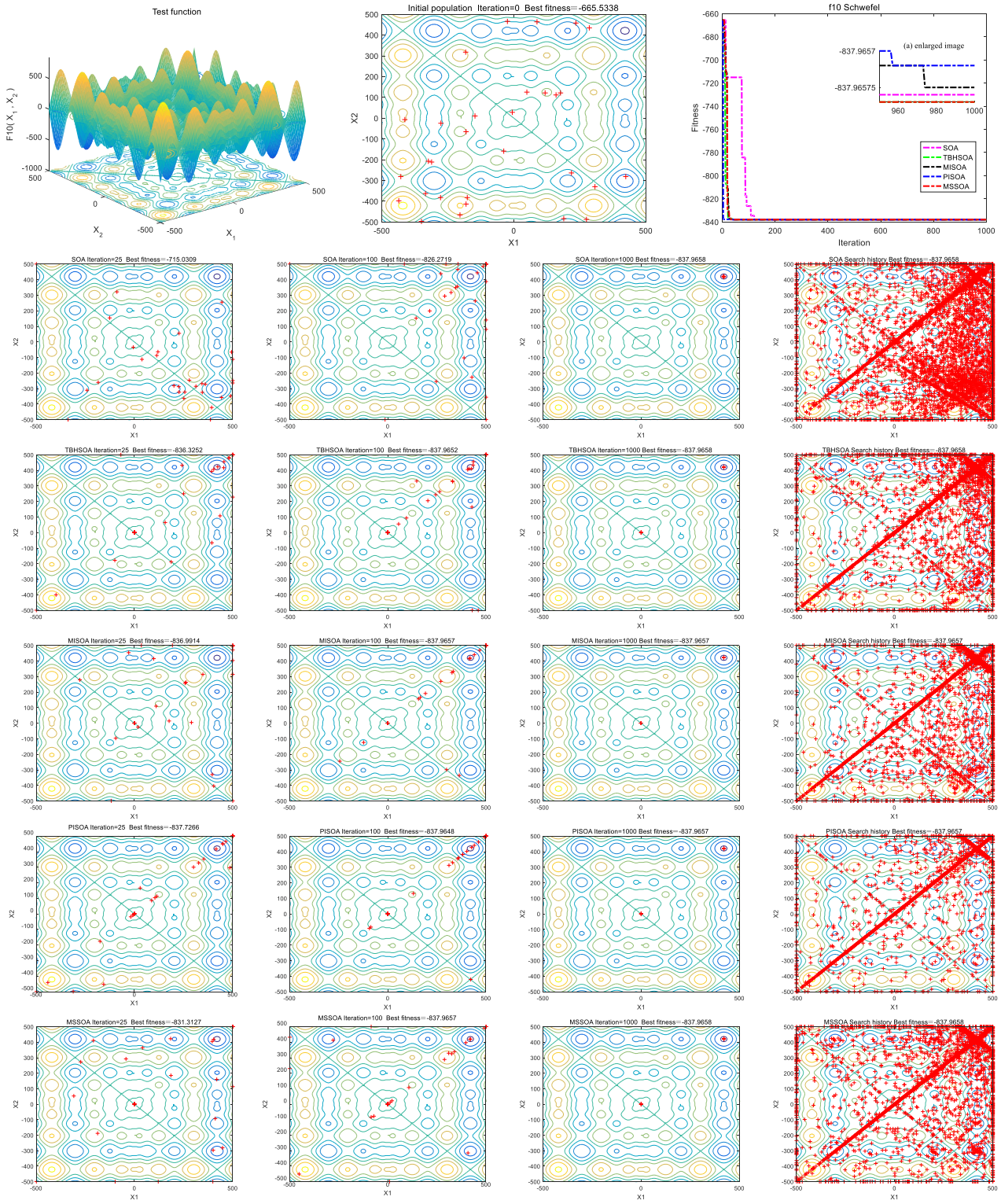

FIGURE 5. The graph of the function f10, initial population's positions, convergence curves, population's positions with iterations and search history.

best value results of the seven algorithms after 30 independent runs are selected for comparison. As seen from Figure 8, the ANOVA of MSSOA algorithm for is the benchmark functions $f1$-$f15$ ($D = 100$) is very well, and this shows that the MSSOA algorithm is very robust.

f: THE COMPLEXITY OF ALGORITHMS ANALYSIS

The calculation complexity of the algorithms' analysis is based on the principle of Big-O representation [38]. The overall calculational complexity of the PSO, SA-GA, GSA, SCA, MVO, SOA, and the MSSOA is in the following

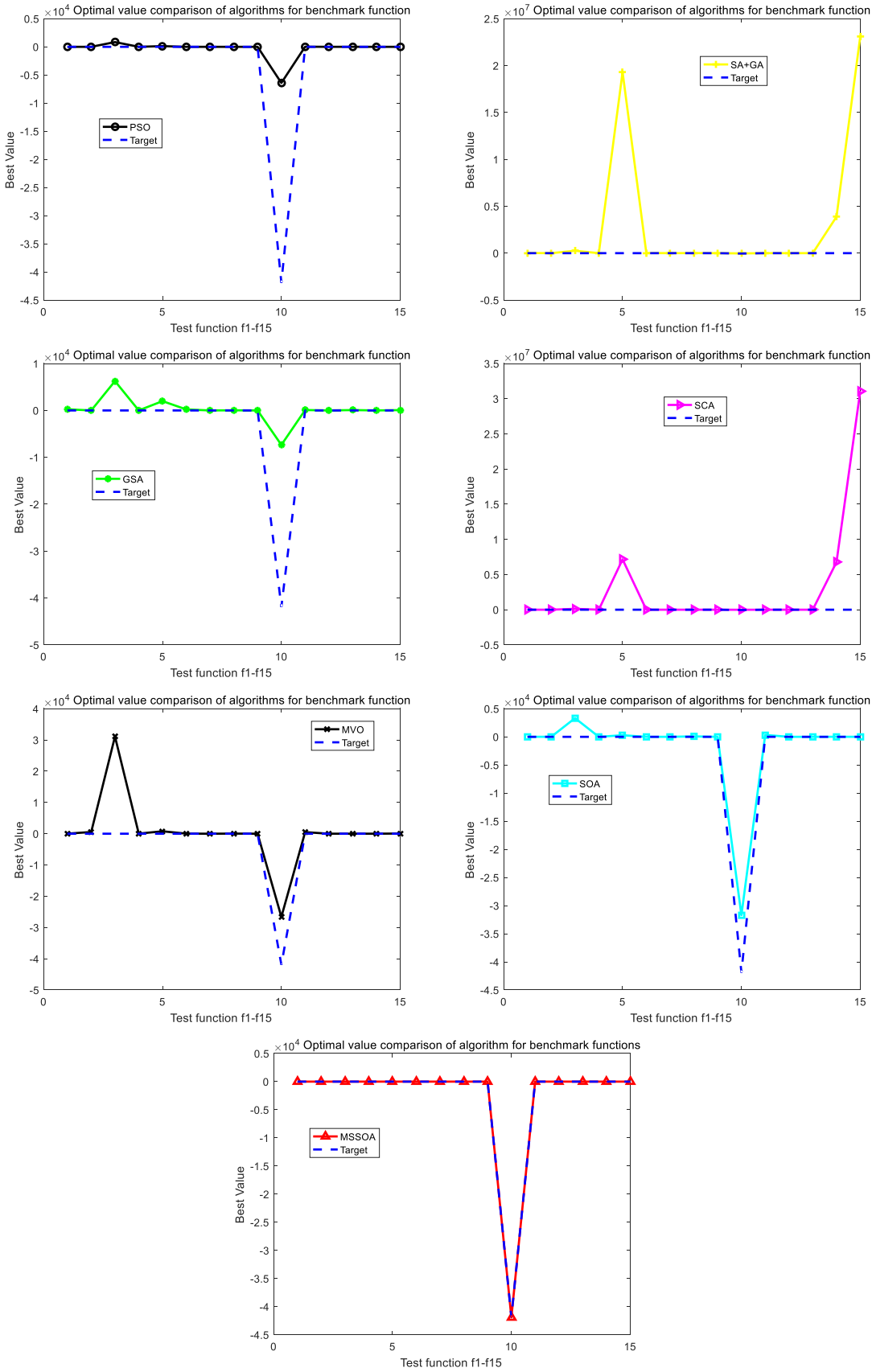

FIGURE 6. The comparison graph of the optimal solution curve of each function ($f1$-$f15$, $D = 100$) obtained by different algorithms and the optimal solution curve of theoretical value.

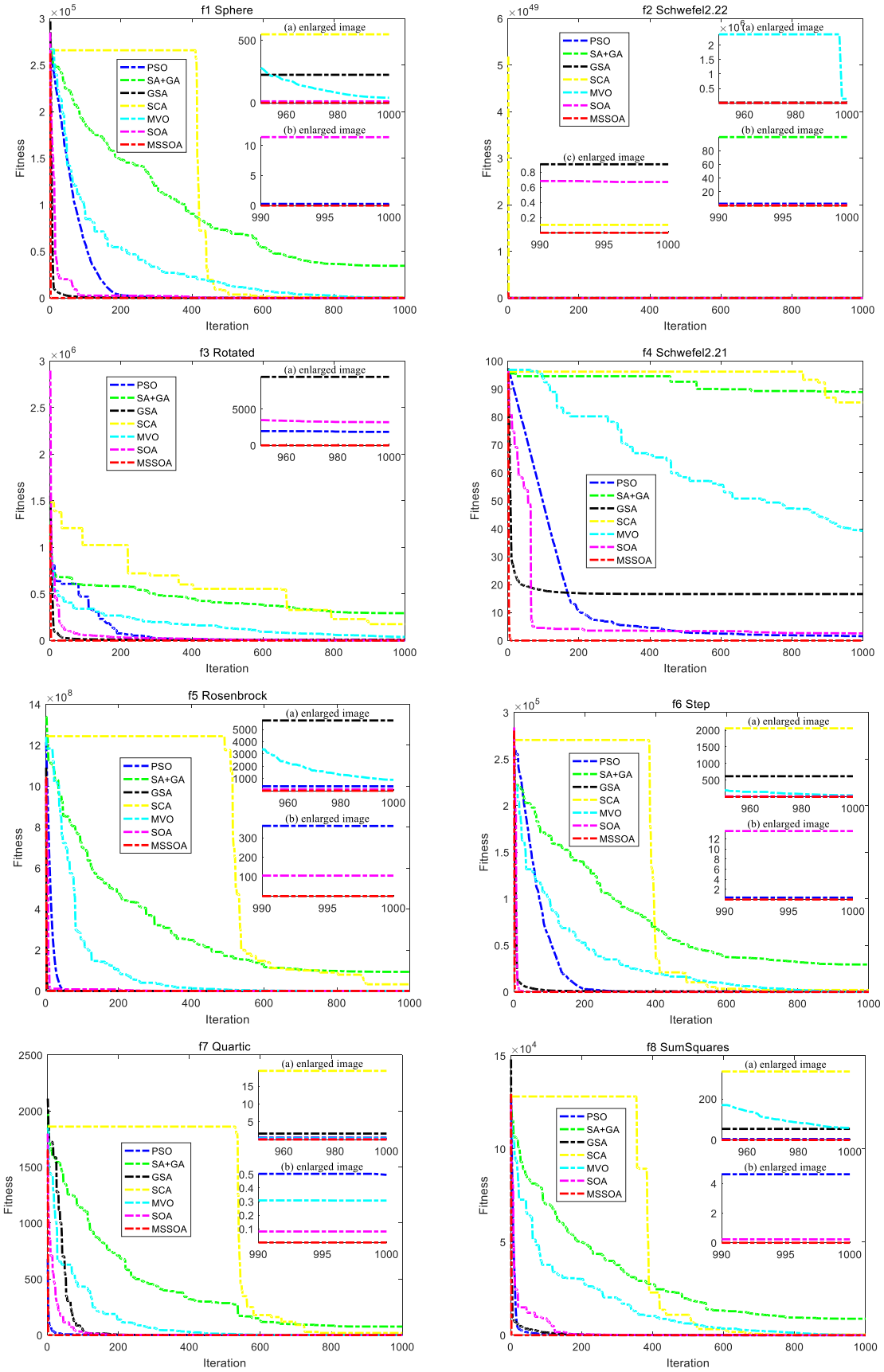

FIGURE 7. Convergence curves for benchmark functions $f1$-$f15$ ($D = 100$).

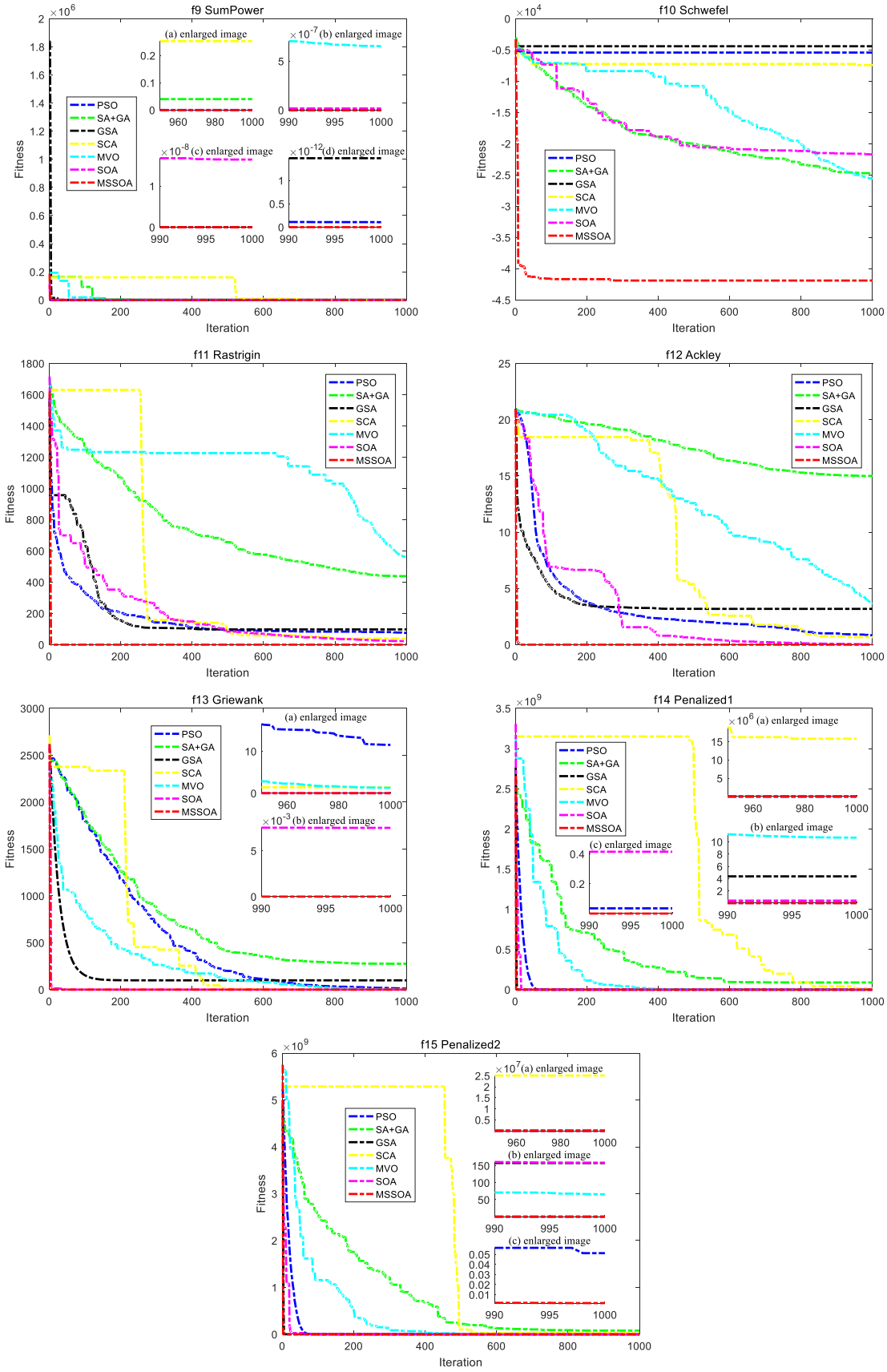

FIGURE 7. *(Continued.)* **Convergence curves for benchmark functions $f1$-$f15$ ($D = 100$).**

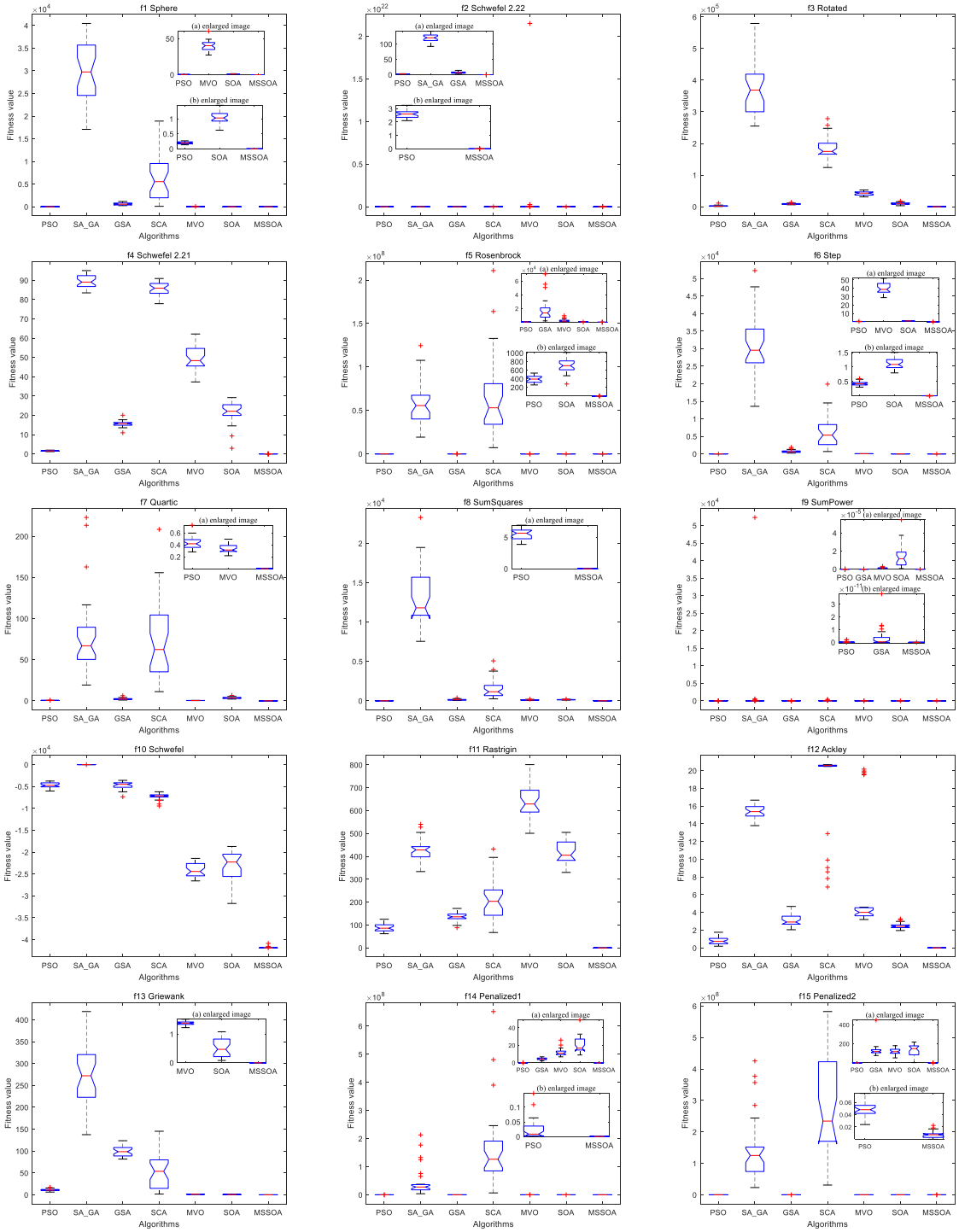

FIGURE 8. ANOVA tests for benchmark functions $f1$-$f15$ ($D = 100$).

formulas (15)-(28). n indicates the total individual count, d represents the dimension count, t represents the maximum count of algebras, c represents the cost of the function evaluation, p represents the number of offspring, m represents the number of mutated/interference populations. α represents the coefficient that shows the percentage of the number of offspring and the mutated population to the total individual count [39].

The PSO complexity:

$$\begin{aligned} O(\text{PSO}) = {} & O(\text{Problem definition}) + O(\text{Initialization}) \\ & + O(t(\text{Function evaluation})) \\ & + O(\text{t(Memory saving)}) \\ & + O(\text{t(Position update)}) = O(1) + O(\text{nd}) \\ & + O(\text{tcn}) + O(\text{tn}) \\ & + O(\text{tnd}) = O(1 + \text{nd} + \text{tcn} + \text{tn}) \end{aligned}$$

$$+ \text{tnd}) \sim= O(\text{tcn} + \text{tnd}) \quad (15)$$

$$\text{So, } O(\text{PSO}) = O(\text{tcn} + \text{tnd}). \quad (16)$$

The SA-GA complexity:

$$\begin{aligned} O(SA-GA) &= O(\text{Problem definition}) + O(\text{Initialization}) \\ &\quad + O(T(\text{Function evaluation})) \\ &\quad + O(T(\text{Temperature reduction})) \\ &\quad + O(t(\text{Temperature update})) \\ &\quad + O(T(\text{function evaluation})) \\ &\quad + O(T(\text{Selection})) + O(T(\text{Crossover})) \\ &\quad + O(T(\text{Mutation})) \\ &= O(1) + O(\text{nd}) + O(\text{Tcn}) \\ &\quad + O(\text{Tnd}) + O(\text{Tnd}) \\ &\quad + O(\text{Tcn}) + O(\text{p}) + O(\text{Tpd}) \\ &\quad + O(\text{Tmd}) \sim= O(1 + \text{nd} + \text{Tcn} + \text{Tnd} \\ &\quad + \text{Tnd} + \text{Tcn} + \text{p} + \text{Tpd} + \text{Tmd}) \quad (17) \end{aligned}$$

$$\begin{aligned} \text{p} + \text{m} &= \alpha \cdot \text{n; therefore,} \\ \text{T} &= \text{t} \cdot \alpha. \text{We consider} \alpha = 1 \text{Thus} \\ \text{T} &= \text{t and p} + \text{m} = \text{n. So, } O(SA-GA) \\ &= O(\text{tcn} + 2\text{tnd} + \text{t(m} + \text{p)d}) \\ &= O(2\text{tcn} + 3\text{tnd}). \quad (18) \end{aligned}$$

The GSA complexity:

$$\begin{aligned} O(\text{GSA}) &= O(\text{Problem definition}) + O(\text{Initialization}) \\ &\quad + O(\text{t(Function evaluation})) \\ &\quad + O(\text{t(Memory saving})) \\ &\quad + O(\text{t(Gravitational forces})) \\ &\quad + O(\text{t(Velocity})) + O(\text{t(Accelerations})) \\ &\quad + O(\text{t(Position update})) \\ &= O(1) + O(\text{nd}) + O(\text{tcn}) + O(\text{tn}) \\ &\quad + O(\text{tnd}) + O(\text{tnd}) + O(\text{tnd}) + O(\text{tnd}) \\ &= O(1 + \text{nd} + \text{tcn} + \text{tn} + \text{tnd} + \text{tnd} + \text{tnd} \\ &\quad + \text{tnd}) \sim= O(\text{tcn} + 4\text{tnd}) \quad (19) \end{aligned}$$

$$\text{So, } O(\text{GSA}) = O(\text{tcn} + 4\text{tnd}). \quad (20)$$

The SCA complexity:

$$\begin{aligned} O(\text{SCA}) &= O(\text{Problem definition}) + O(\text{Initialization}) \\ &\quad + O(\text{t(function evaluation})) \\ &\quad + O(\text{t(Memory saving})) \\ &\quad + O(\text{t(Position update})) \\ &= O(1) + O(\text{nd}) + O(\text{tcn}) + O(\text{tn}) + O(\text{tnd}) \\ &= O(1 + \text{nd} + \text{tcn} + \text{tn} + \text{tnd}) \sim \\ &= O(\text{tcn} + \text{tnd}) \quad (21) \end{aligned}$$

$$\text{So, } O(\text{SCA}) = O(\text{tcn} + \text{tnd}).(22) \quad (22)$$

The MVO complexity:

$$\begin{aligned} O(\text{MVO}) &= O(\text{Problem definition}) + O(\text{Initialization}) \\ &\quad + O(\text{t(function evaluation})) \\ &\quad + O(\text{t(Memory saving})) \\ &\quad + O(\text{t(Position update})) \\ &= O(1) + O(\text{nd}) + O(\text{tcn}) + O(\text{tn}) + O(\text{tnd}) \\ &= O(1 + \text{nd} + \text{tcn} + \text{tn} + \text{tnd}) \sim \\ &= O(\text{tcn} + \text{tnd}) \quad (23) \end{aligned}$$

$$\text{So, } O(\text{MVO}) = O(\text{tcn} + \text{tnd}). \quad (24)$$

The SOA complexity:

$$\begin{aligned} O(\text{SOA}) &= O(\text{Problem definition}) + O(\text{Initialization}) \\ &\quad + O(\text{t(function evaluation})) \\ &\quad + O(\text{t(Memory saving})) \\ &\quad + O(\text{t(Search direction})) \\ &\quad + O(\text{t(Search step size})) \\ &\quad + O(\text{t(Position update})) \\ &= O(1) + O(\text{nd}) + O(\text{tcn}) + O(\text{tn}) + O(\text{tnd}) \\ &\quad + O(\text{tnd}) + O(\text{tnd}) \\ &= O(1 + \text{nd} + \text{tcn} + \text{tn} + 3\text{tnd}) \sim \\ &= O(\text{tcn} + 3\text{tnd}) \quad (25) \end{aligned}$$

$$\text{So, } O(\text{SOA}) = O(\text{tcn} + 3\text{tnd}). \quad (26)$$

The MSSOA complexity:

$$\begin{aligned} O(\text{MSSOA}) &= O(\text{Problem definition}) \\ &\quad + O(\text{Initialization}) \\ &\quad + O(\text{t(function evaluation})) \\ &\quad + O(\text{t(Memory saving})) \\ &\quad + O(\text{t(Search direction})) \\ &\quad + O(\text{t(Search step size})) \\ &\quad + O(\text{t(triple black hole system capture}) \\ &\quad + O(\text{t(multi-dimensional random} \\ &\quad \times \text{interference}) \\ &\quad + O(\text{t(precocious interference}) \\ &\quad + O(\text{t(Position update})) \\ &= O(1) + O(\text{nd}) + O(\text{tcn}) + O(\text{tn}) \\ &\quad + O(\text{tnd}) + O(\text{tnd}) \\ &\quad + O(\text{tmd}) + O(\text{tmd}) + O(\text{tmd}) + O(\text{tnd}) \\ &= O(1 + \text{nd} + \text{tcn} \\ &\quad + \text{tn} + \text{tnd} + \text{tnd} + \text{tmd} \\ &\quad + \text{tmd} + \text{tmd} + \text{tnd}) \sim \\ &= O(\text{tcn3tmd} + 3\text{tnd}) \quad (27) \end{aligned}$$

$$\text{So, } O(\text{MSSOA}) = O(\text{tcn} + 3\text{tmd} + 3\text{tnd}). \quad (28)$$

As shown in the complexity analysis of the above seven algorithms, although the calculational complexity of the MSSOA algorithm analysis is more complicated than the PSO, MVO, SCA, and SOA, the complexity of the MSSOA algorithm is also a polynomial. Therefore, the MSSOA is considered to be an effective algorithm.

TABLE 5. Performance comparison of algorithms for benchmark functions.

Test functions	Result	Algorithms						
		PSO	SA-GA	GSA	SCA	MVO	SOA	MSSOA
f_1 (D=100)	Mean	0.0031	3.018e+04	6.21E+02	6.42E+03	40.269355	1.0524957	**0**
	Std.	6.31E-04	6.691e+03	2.71E+02	5.62E+03	6.8812839	0.2051906	**0**
	Best	0.0022	1.708e+04	2.50E+02	96.29169	27.3763	0.6216	**0**
	Rank	2	7	6	5	4	3	**1**
f_2 (D=100)	Mean	0.2192801	118.6375	7.1387617	1.3795842	7.34E+20	13.002085	**2.012e-184**
	Std.	0.0260935	13.168469	3.2135126	1.4335667	3.92E+21	2.0008395	**0**
	Best	0.1732089	92.513924	2.0654838	0.0310204	5.43E+02	9.4729280	**4.588e-213**
	Rank	3	7	4	2	6	5	**1**
f_3 (D=100)	Mean	2.27E+03	3.77e+05	8.80E+03	1.84E+05	4.24E+04	9.87E+03	**0**
	Std.	1.30E+03	8.95e+04	1.84E+03	3.48E+04	6.09E+03	3.47E+03	**0**
	Best	8.43E+02	2.55e+05	6.26E+03	1.24E+05	3.11E+04	3.32E+03	**0**
	Rank	2	7	4	6	5	3	**1**
f_4 (D=100)	Mean	1.4624169	89.626880	15.671857	85.539422	49.557442	21.743821	**2.540e-186**
	Std.	0.2035038	3.454423	1.5605923	3.5980134	6.2884298	5.6809434	**0**
	Best	1.0929345	83.496264	10.958917	77.909721	37.299715	2.9045088	**3.878e-216**
	Rank	2	7	4	6	5	3	**1**
f_5 (D=100)	Mean	2.515e+02	5.783e+07	1.784e+04	6.427e+07	2.440e+03	7.093e+02	**0.00521650**
	Std.	59.018749	2.620e+07	1.578e+04	4.671e+07	1.959e+03	1.675e+02	**0.01474780**
	Best	1.236e+02	1.925e+07	1.987e+03	7.162e+06	7.755e+02	2.789e+02	**7.268e-05**
	Rank	2	7	5	6	4	3	**1**
f_6 (D=100)	Mean	0.0119147	3.074e+04	6.995e+02	6.131e+03	40.0892130	1.1365348	**0.00818272**
	Std.	0.0020782	8.323e+03	3.890e+02	4.651e+03	5.9162672	0.1751583	0.00284437
	Best	0.008661	1.356e+04	2.169e+02	6.729e+02	28.9167207	0.799274	**6.152e-04**
	Rank	2	7	5	6	4	3	**1**
f_7 (D=100)	Mean	0.2211229	77.487242	2.3136998	73.024146	0.3355664	3.4645595	**0.00818272**
	Std.	0.0419681	49.238470	1.1506156	46.772681	0.0733789	0.9443482	**4.849e-05**
	Best	0.1180966	19.170953	0.8371509	11.108763	0.2176315	2.0472780	**4.709e-06**
	Rank	2	7	4	6	3	5	**1**
f_8 (D=100)	Mean	0.0624146	1.29e+04	1.32E+02	1.52E+03	1.05E+02	1.39E+02	**0**
	Std.	0.0164020	3.57e+03	74.414138	1.16E+03	43.821833	33.925549	**0**
	Best	0.0291246	7.55e+03	31.981022	2.47E+02	34.296414	79.288776	**0**
	Rank	2	7	3	6	4	5	**1**
f_9 (D=100)	Mean	2.08E-26	1.78e+03	3.67E-12	35.077327	1.21E-06	1.47E-05	**0**
	Std.	3.99E-26	9.54e+03	7.66E-12	80.100469	5.23E-07	1.28E-05	**0**
	Best	2.07E-29	0.002883	4.23E-16	0.0268174	4.55E-07	6.32E-07	**0**
	Rank	2	6	3	7	4	5	**1**
f_{10} (D=100)	Mean	-4.746e+3	-2.475e+04	-4.729e+3	-7.256e+3	-2.408e+4	-2.324e+4	**-4.185e+4**
	Std.	6.889e+02	8.845e+02	8.702e+02	6.962e+02	1.533e+03	3.396e+03	**1.998e+02**
	Best	-6.440e+3	-2.659e+04	-7.354e+3	-9.457e+3	-2.658e+4	-3.174e+4	**-4.190e+04**
	Rank	7	3	6	5	4	2	**1**
f_{11} (D=100)	Mean	33.896488	425.8326	1.36E+02	2.06E+02	6.41E+02	4.15E+02	**0**
	Std.	6.2350851	48.908770	18.532649	92.249647	69.443553	47.133647	**0**
	Best	23.655297	3.34 E+02	88.161003	67.314606	5.01E+02	3.30E+02	**0**
	Rank	2	6	4	3	7	5	**1**
f_{12} (D=100)	Mean	0.0220555	15.3479726	3.1378392	18.309269	6.5910720	2.4523845	**8.882e-16**
	Std.	0.0032932	0.7816167	0.6404753	4.7183569	6.0322718	0.3106583	**0**
	Best	0.0161767	13.7962876	2.0521342	6.872015	3.2031355	1.9634230	**8.882e-16**
	Rank	2	7	4	6	5	3	**1**
f_{13} (D=100)	Mean	9.1115292	274.5795	98.761595	53.145123	1.3775687	0.5354221	**0**
	Std.	1.8646352	66.687333	11.527169	38.195663	0.0626040	0.3343659	**0**
	Best	6.1937260	137.5120	81.688324	1.7219609	1.2241138	0.0806019	**0**
	Rank	5	**7**	6	4	3	2	**1**
f_{14} (D=100)	Mean	0.0167129	4.418e+07	4.5497727	1.582e+08	11.6567511	20.2319839	**1.362e-04**
	Std.	0.0241246	5.110e+07	1.21396147	1.352e+08	4.2010589	8.8688013	**3.249e-05**
	Best	4.838e-05	3.883e+06	2.1673113	6.757e+06	6.7785747	8.9451797	**8.049e-05**
	Rank	2	6	3	7	4	5	**1**
f_{15} (D=100)	Mean	0.0010358	1.435e+08	1.309e+02	2.674e+08	1.209e+02	1.301e+02	**0.0072373**
	Std.	0.002800	1.008e+08	64.254054	1.527e+08	30.717732	62.9791683	**0.0054797**
	Best	1.898e-04	2.309e+07	76.4103682	3.106e+07	51.2656139	1.7998261	**5.371e-06**
	Rank	2	6	5	7	4	3	**1**
Average Rank		2.6	6.4666667	4.4	5.4666667	4.4	3.6666667	**1**
Overall Rank		2	7	4	6	4	3	**1**

TABLE 6. Run time comparison of 30 independent runs for benchmark functions $f1$-$f15$ ($D = 100$).

Test functions (D= 100)	Run time of algorithms						
	PSO	SA_GA	GSA	SCA	MVO	SOA	MSSOA
f_1	19.2614	542.3931	150.8607	**15.6790**	37.1642	45.5006	87.175798
f_2	22.358433	521.8564	154.253122	**16.774635**	22.505080	54.167535	88.718713
f_3	**131.065635**	3509.5453	276.851090	157.841395	150.595822	423.272832	732.835497
f_4	21.427574	642.475529	154.476221	**17.295688**	40.792862	47.984464	106.133452
f_5	19.859456	651.808449	188.478202	**18.794625**	41.892321	54.515291	102.749566
f_6	20.542143	480.833828	153.064086	**17.318195**	39.282568	47.876927	89.505699
f_7	34.792130	890.981756	168.550287	**30.316487**	51.654684	90.520669	169.388307
f_8	23.367437	481.638596	153.526963	**17.055855**	39.908609	55.511679	118.793157
f_9	35.632177	975.1530	173.578067	**32.396346**	45.258449	102.887695	223.453628
f_{10}	26.354619	562.020557	158.499353	**20.396761**	22.749008	60.474337	117.516091
f_{11}	**21.033824**	648.6857	184.680588	22.147268	41.646913	59.599374	88.386537
f_{12}	23.395074	624.441362	173.135274	**21.299206**	43.723453	64.893256	122.176144
f_{13}	29.505900	615.5540	169.814804	**23.426107**	46.092607	63.673136	90.052169
f_{14}	60.288823	1654.958657	193.881593	**52.751386**	74.326149	165.947519	310.154355
f_{15}	59.640283	1629.496415	196.205228	**53.109655**	75.622178	166.651257	302.614372
The total time	548.5249	14431.84	2649.856	**516.6026**	773.2149	1503.477	2749.653
Overall Rank	**2**	7	5	**1**	3	4	6

g: RUN TIME COMPARISON OF ALGORITHMS IN BENCHMARK FUNCTIONS

We recorded the running time of each algorithm for each function under the same conditions: population number 30, evolution algebra 1000, and 30 independent runs of the above 15 benchmark functions $f1$-$f15$ (D=100). Then, the running time of the 15 functions is added to obtain the sum of the 30 independent running times of each algorithm for the 15 functions listed in this paper, and the ranking of the total time, as shown in Table 6. As seen from Table 6, the SCA algorithm has the more minor program running time, followed by the PSO algorithm, which has less program running time. The MSSOA algorithm ranks sixth, which has a relatively longer program running time. At the bottom of the list is the SA_GA algorithm, which takes the most running time.

To learn more traits about the program running time of the seven algorithms in the 15 functions, a bar chart in Figure 9 was made for the total time of each algorithm after 30 independent runs. From Figure 9 to the running time, the SCA is the least; the SA_GA is the most; the MSSOA is less than the SA_GA; the MSSOA is less than one in six of SA_GA, and the MSSOA is nearly four times of the SCA, which is relatively large.

h: EXPLORATION AND EXPLOITATION IN BENCHMARK FUNCTIONS

According to the literature [40]–[42], the formula (29)-(32) represents the exploration and exploitation capability of

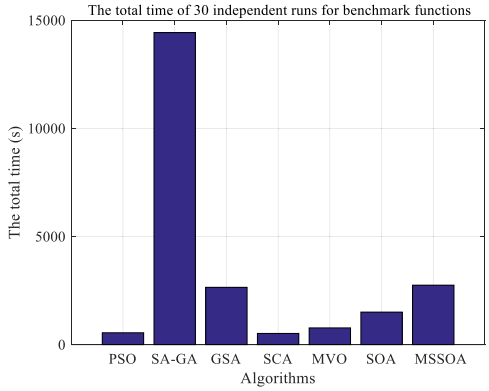

FIGURE 9. The total time of 30 independent runs of 7 algorithms on 15 benchmark functions.

an algorithm.

$$Div_j = \frac{1}{n}\sum_{i=1}^{n} \text{median}\left(x^j\right) - x_i^j \tag{29}$$

$$Div = \frac{1}{D}\sum_{i=1}^{D} Div_j \tag{30}$$

$$Xpl\% = \frac{Div}{Div_{\max}} \times 100 \tag{31}$$

$$Xpt\% = \frac{|Div - Div_{\max}|}{Div_{\max}} \times 100 \tag{32}$$

where, median x^j is the median of dimension j in whole swarm. x_i^j is the dimension j of the swam individual i. n is the size of swarm. Div_j is the average for all the individuals.

TABLE 7. p-values of the Wilcoxon rank-sum test.

Test functions (D= 100)	p test values of various algorithms						
	PSO	SA_GA	GSA	SCA	MVO	SOA	MSSOA
f_1	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	N/A
f_2	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	N/A
f_3	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	N/A
f_4	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	N/A
f_5	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	N/A
f_6	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	N/A
f_7	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	N/A
f_8	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	N/A
f_9	1.7203e-12	1.7203e-12	1.7203e-12	1.7203e-12	1.7203e-12	1.7203e-12	N/A
f_{10}	3.1602e-12	3.1602e-12	3.1602e-12	3.1602e-12	3.1602e-12	3.1602e-12	N/A
f_{11}	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	N/A
f_{12}	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	N/A
f_{13}	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	1.2118e-12	N/A
f_{14}	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	N/A
f_{15}	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	3.0199e-11	N/A

Div is the diversity of swarm in an iteration and $Div_{\max}$ is the maximum diversity in all iterations. Xpl% and Xpt% are the exploration and exploitation percentages for an iteration, respectively.

Figure 10 shows the exploration and exploitation abilities of the MSSOA as the number of iterations increases in the benchmark functions *f5, f7, f8, f9, f10, f11, f14, f*15. As observed from the plotted curves shown in Figure 10, the MSSOA maintain good balance between the exploration and exploitation ratios as the number of iterations increases.

i: STATISTICAL TESTING OF ALGORITHMS IN BENCHMARK FUNCTIONS

Using the Wilcoxon's rank-sum test [43] can discover the important differentia between the two algorithms. This test gives the value $p < 0.05$.

The mean values are test in the Wilcoxon's rank-sum. To ensure that the comparison of these algorithms is fair, the population number of algorithms is 30, and the evolutionary algebra is 1000. The mean results of the seven algorithms after 30 independent runs are selected for comparison. Table 7 is the results of statistical testing. N/A represents the best algorithm. From Table 7, the MSSOA results are statistically significant in optimization high dimension benchmark functions.

j: PERFORMANCE PROFILES OF ALGORITHMS IN BENCHMARK FUNCTIONS

The average fitness was selected as the capability index. The algorithmic capability is expressed in performance profiles, which is calculated by the formulas (33)(34).

$$r_{f,g} = \mu_{f,g} / \min\{\mu_{f,g} : g \in G\} \tag{33}$$

$$\rho_g(\tau) = \text{size}\{f \in F : r_{f,g} \leq \tau\}/n_f \tag{34}$$

where, g represents an algorithm; G is the algorithms set; f means a function; F represents the function set; n_g represents the count of algorithms in the experiment; n_f is the number of functions in the experiment; $\mu_{f,g}$ is the average fitness after the algorithm g solving function f; $r_{f,g}$ is the capability ratio; ρ_g is the algorithmic capability; τ is a factor of the best probability [44].

Figure 11 shows the capability ratios of the average value for the seven algorithms on the benchmark functions $f1$-$f15$ ($D = 100$). The consequences are revealed by a log scale 2. As shown in Figure 11, the MSSOA has the highest probability. When $\tau = 1$, the MSSOA is about 0.667, which is better than that of the others. When $\tau = 3$ the MSSOA is the winner on the given test functions is about 1, the PSO is 0.667, SA_GA is 0.067, GSA is 0.067, SCA is 0.067, MVO is 0.2, and the SOA is 0.2. When $\tau = 11$ the MSSOA is the winner on the given test functions is about 1, the PSO is 0.73, SA_GA is 0.067, GSA is 0.267, SCA is 0.133, MVO is 0.267, and the SOA is 0.267. Regarding the performance curve, the MSSOA is the best; the MSSOA can achieve 100% when $\tau \geq 3$. Thus, the property of the MSSOA is better than that of the other algorithms.

C. ALGORITHM PERFORMANCE COMPARISON IN CONSTRAINED OPTIMIZATION PROBLEMS

We are using 5 constrained problem examples and 6 constrained engineering design problems to test the capability

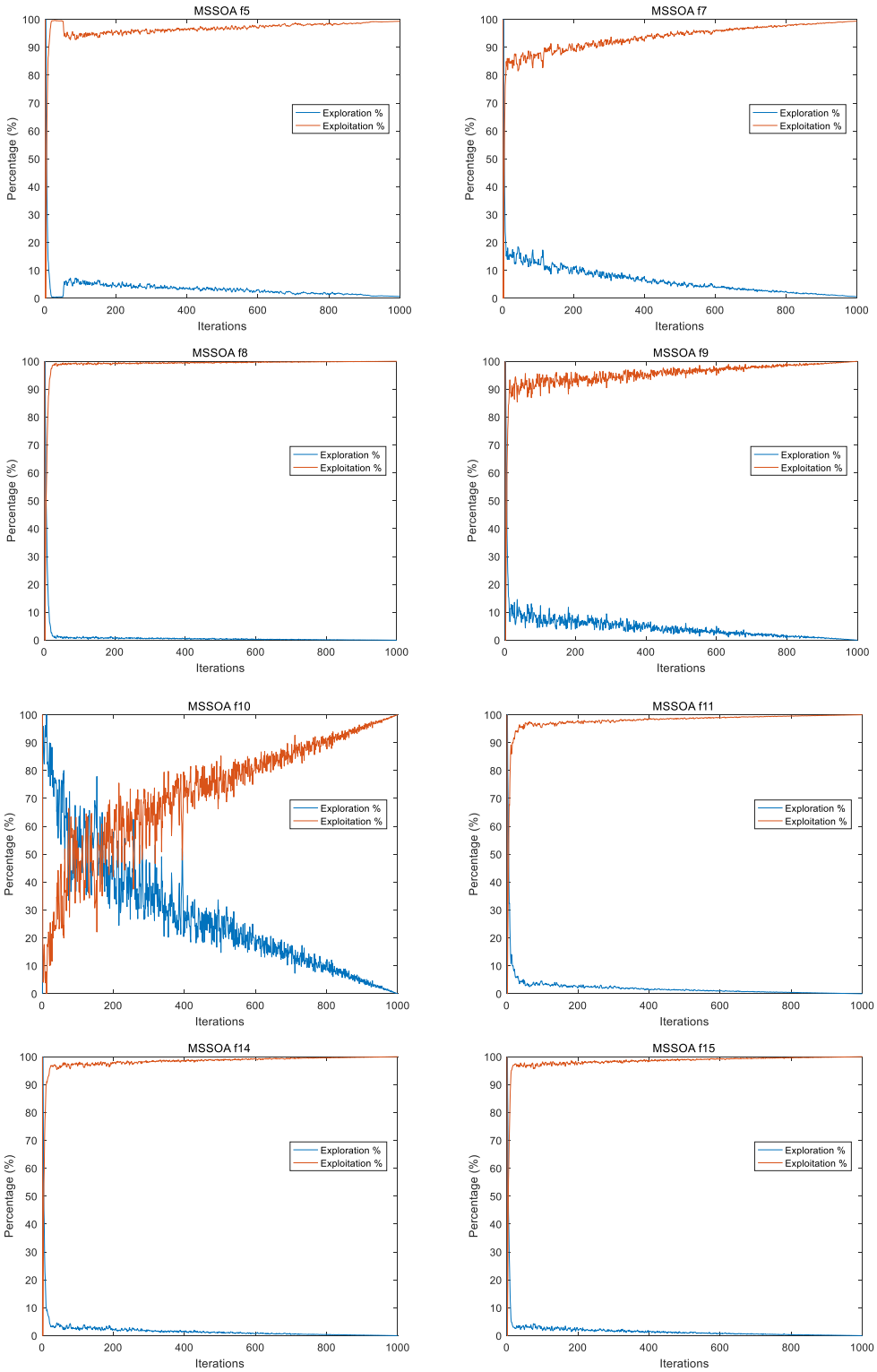

FIGURE 10. The exploration and exploitation abilities of the MSSOA in benchmark functions.

of the MSSOA further. These problems are very popular in the literature. The penalty function is used to calculate the constrained problem. The parameters set for all of the heuristic algorithms still adopts the parameter setting form Table 4 of Section IV.B.3).

Regarding the penalty function, in this paper, the constraint problem is defined as formula (35):

$$\begin{aligned} &\text{Minimize } f(x), \quad \vec{x} \in R^d \\ &\text{Subject to } g_i(x) \leq 0, \quad i = 1, 2, \ldots, p \end{aligned}$$

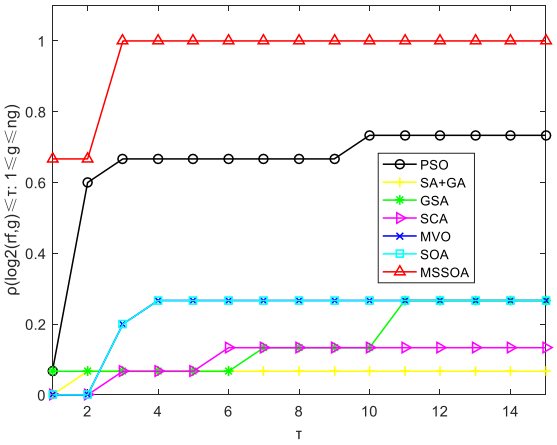

FIGURE 11. Performance profile of 7 algorithms on 15 benchmark functions.

$$h_j(x) = 0, j = 1, 2, \ldots, q \tag{35}$$

where g_i represents the inequality constraints, h_i represents the equality constraints, and R^d represents the d dimensional real numbers. Using the heuristic algorithm to find the best $\vec{x} = \{x_1, x_2, \ldots, x_d\}$ minimizes $f(\vec{x})$.

When using the penalty function to deal with the constrained problems in the heuristic algorithm. A large penalty value proportional to the values of the violated constraints is added to the objective function. Therefore, the optimization of the constrained problems is defined as formula (36):

$$\text{Minimize } F(\vec{x}) = \begin{cases} f(\vec{x}), & \vec{x} \in S \\ f(\vec{x}) + \lambda(\sum_{i=1}^{p}\sum_{i=1}^{p} g_i(\vec{x}) & \\ \quad + \sum_{i=1}^{q} h_j(\vec{x})), & \vec{x} \notin S \end{cases} \tag{36}$$

where S represents the search space. When individuals violate a constraint, they are assigned a big fitness value.

1) CONSTRAINED PROBLEM EXAMPLES

We are using 5 constrained problem examples to test the capability of the MSSOA. The formulations of these problems are available in Appendix A.

a: CONSTRAINED PROBLEM 1

For the constrained problem, the MSSOA compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA. The mean values, standard deviation, worst fitness, best fitness, and the run time between the algorithms of 30 all alone runs, and the data of the constrained problem 1 optimization results are shown in Table 8. The underline and boldface indicate that the optimal outcome is better.

Based on Table 8, for the best value of the constrained problem, the MSSOA is better than the others. For the worst value, the standard deviation, and the mean, except the MVO and the SOA, the MSSOA is better than the PSO, SA-GA, GSA, and the SCA. For the time, the SCA is the shortest; the MSSOA is only shorter than the SA-GA.

TABLE 8. Performance comparison of algorithms for the constrained problem 1.

Algorithm	Optimal values for variables				
	Mean	Std.	Worst	Best	Time
PSO	0.75001	1.843e-05	0.75008	0.74999783	16.82
SA-GA	0.79428	0.05981	0.98903	0.75003	653.5
GSA	0.87413	0.07407	1	0.75235541	23.72
SCA	0.75083	5.160e-04	0.75244	0.75005837	**8.482**
MVO	**0.750004**	**6.772e-06**	**0.75002**	0.74999776	9.378
SOA	0.750007	8.225e-06	0.75003	0.749997543	64.00
MSSOA	0.750007	9.637e-06	0.75003	**0.749997542**	115.8

TABLE 9. Comparison results for constrained problem 1.

Algorithm	Optimal values for variables		Optimum weight	Rank
	x_1	x_2		
MBA [42]	-0.706958	0.499790	0.750000	5
PSO	0.706734627542	0.499478081331	0.74999782901508	3
SA-GA	0.702965083028	0.494164907940	0.75003154829935	6
GSA	0.679880262201	0.461373744130	0.75235541444374	8
SCA	0.711413165129	0.506087378323	0.75005836936988	7
MVO	-0.70677036617	0.499529976302	0.74999776008488	4
SOA	-0.70721653313	0.500159804058	0.74999754323344	2
MSSOA	0.705824137042	0.498173561503	**0.74999754237257**	**1**

TABLE 10. Performance comparison of algorithms for the constrained problem 2.

Algorithm	Optimal values for variables				
	Mean	Std.	Worst	Best	Time
PSO	**13.59111**	3.799e-04	13.5921016	13.590742	15.20
SA-GA	24.22872	22.625406	89.4492809	13.599119	562.7
GSA	18.13747	8.7743096	60.4248018	13.662536	24.44
SCA	68.32585	72.913954	1.625e+02	13.608482	**7.001**
MVO	13.59195	0.0012790	13.595743	13.590751	9.543
SOA	13.59133	6.184e-04	13.593641	13.590731	50.85
MSSOA	13.59114	**3.166e-04**	**13.591883**	**13.590723**	151.3

The MSSOA compared to MBA [45], PSO, SA_GA, GSA, SCA, MVO, and SOA, and provided the best-obtained values for variables and the best-obtained values in Table 9. According to Table 9, the MSSOA algorithm is better than the MBA, PSO, SA_GA, GSA, SCA, MVO, and the SOA.

b: CONSTRAINED PROBLEM 2

For the constrained problem, the MSSOA compared to PSO, SA_GA, GSA, SCA, MVO, and SOA. The mean values, standard deviation, worst fitness, best fitness, and the run time between the algorithms of 30 all alone runs, and the data of the constrained problem 2 optimization results are shown in Table 10. The underline and boldface indicate that the optimal outcome is better.

Based on Table 10, for the best value of the constrained problem, the MSSOA is better than the others. For the worst value, the MSSOA is better than the others. For the standard

TABLE 11. Comparison results for constrained problem 2.

Algorithm	Optimal values for variables		Optimum weight	Rank
	x_1	x_2		
MBA [45]	2.246833	2.381997	13.590842	5
HS [46]	2.246840	2.382136	13.590845	6
PSO	2.24679362	2.38112814	13.59074151	3
SA-GA	2.24762961	2.39765431	13.59911899	7
GSA	2.24365506	2.34999866	13.66253631	9
SCA	2.24635687	2.38393098	13.60848206	8
MVO	2.24687306	2.38263657	13.59075144	4
SOA	2.24659360	2.37795776	13.59073090	2
MSSOA	2.24677100	2.38115228	**13.59072342**	**1**

TABLE 12. Performance comparison of algorithms for the constrained problem 3.

Algorithm	Optimal values for variables				
	Mean	Std.	Worst	Best	Time
PSO	-0.0958248	3.351e-07	-0.095823	**-0.095825**	16.65
SA-GA	-0.0623719	0.0340304	-0.025768	-0.095825	561.5
GSA	1.1413e+05	2.854e+05	1.562e+06	-0.077561	24.76
SCA	-0.0948443	9.383e-04	-0.091295	-0.095810	**8.62**
MVO	-0.0958247	3.664e-07	-0.095823	-0.095825	9.20
SOA	-0.09582491	1.495e-07	-0.095824	-0.095825	57.89
MSSOA	**-0.09582492**	**9.900e-08**	**-0.095825**	-0.095825	270.6

deviation, the MSSOA is also better than the others. For the mean, the PSO is the best; the MSSOA is better than the SA-GA, GSA, SCA, MVO, and the SOA. For the time, the SCA is the shortest; the MSSOA is only shorter than that of the SA-GA.

The MSSOA compared to MBA [45], HS [46], PSO, SA_GA, GSA, SCA, MVO, and SOA, and provided the best-obtained values for variables and the best-obtained values in Table 11. According to Table 11, the MSSOA algorithm is better than the HS, MBA, PSO, SA_GA, GSA, SCA, MVO, and the SOA.

c: CONSTRAINED PROBLEM 3

For the constrained problem, the MSSOA compared to PSO, SA_GA, GSA, SCA, MVO, and SOA. The mean values, standard deviation, worst fitness, best fitness, and the run time between the algorithms of 30 all alone runs, and the data of the constrained problem 3 optimization results are shown in Table 12. The underline and boldface indicate that the optimal outcome is better.

Based on Table 12, for the best value of the constrained problem, the MSSOA is better than the GSA and the SCA; except the GSA and the SCA, the MSSOA is not much of a difference between the optimal value of the MSSOA algorithm and that of the others. For the worst value, the MSSOA is better than the others. For the standard deviation, the MSSOA is also better than the others. For the mean, the MSSOA is also better than the others. For the time, the SCA is the shortest; the MSSOA is only shorter than that of the SA-GA.

The MSSOA compared to ABC [45], MBA [45], PSO, SA_GA, GSA, SCA, MVO, and SOA, and provided the best-obtained values for variables and the best-obtained

TABLE 13. Comparison results for constrained problem 3. "N.A" stands for not available.

Algorithm	Optimal values for variables		Optimum weight	Rank
	x_1	x_2		
ABC [45]	N.A	N.A	−0.095825	6
MBA [45]	N.A	N.A	−0.095825	6
PSO	1.22798168	4.24538499	**-0.0958250406**	**1**
SA-GA	1.22797135	4.24537337	-0.0958250414	3
GSA	1.20702106	4.15205317	-0.0775611591	9
SCA	1.22853872	4.24800720	-0.0958101010	8
MVO	1.22796786	4.24542282	-0.0958250367	5
SOA	1.22803036	4.24525088	-0.0958250397	4
MSSOA	1.22809352	4.24564249	-0.0958250408	2

TABLE 14. Performance comparison of algorithms for the constrained problem 4.

Algorithm	Optimal values for variables				
	Mean	Std.	Worst	Best	Time
PSO	**-6.969e+3**	**0.0764693**	**-6.968e+3**	**-6.969e+3**	20.30
SA-GA	-1.760e+3	2.1187e+4	1.1023e+5	-6.865e+3	877.2
GSA	-6.776e+3	4.7591e+2	-4.620e+3	-6.968e+3	35.52
SCA	3.1731e+4	1.0453e+4	3.4517e+4	-6.737e+3	**9.53**
MVO	-6.938e+3	33.997171	-6.824e+3	-6.966e+3	13.38
SOA	-6.682e+3	1.0271e+3	-1.266e+3	-6.967e+3	54.79
MSSOA	-6.967e+3	2.1694785	-6.957e+3	-6.969e+3	161.01

values in Table 13. In Table 13, the MSSOA algorithm proves to be better than the ABC, MBA, PSO, GSA, SCA, MVO, and the SOA. Although the optimum of the MSSOA is worse than that of the PSO, there is not much of a difference between the optimal value of the MSSOA algorithm and that of the SA-GA and PSO algorithm.

d: CONSTRAINED PROBLEM 4

For the constrained problem, the MSSOA compared to PSO, SA_GA, GSA, SCA, MVO, and SOA. The mean values, standard deviation, worst fitness, best fitness, and the run time between the algorithms of 30 all alone runs, and the data of the constrained problem 4 optimization results are shown in Table 14. The underline and boldface indicate that the optimal outcome is better.

Based on Table 14, for the best value of the constrained problem, there is not much of a difference between the optimal value of the MSSOA algorithm and that of the PSO algorithm. For the worst value, the standard deviation, and the mean, the MSSOA is worse than the PSO; the MSSOA is better than the others except the PSO. For the time, the SCA is the shortest; the MSSOA is only shorter than that of the SA-GA.

The MSSOA is compared to the ABC [45], MBA [45], PSO, SA_GA, GSA, SCA, MVO, and SOA, and provides the best-obtained values for variables and the best-obtained values in Table 15. According to Table 15, the MSSOA algorithm is better than the ABC, MBA, SA_GA, GSA, SCA, MVO, and the SOA. Although the optimum of the MSSOA is worse than the PSO, there is not much of a difference between the optimal value of the MSSOA algorithm and that of the PSO algorithm.

TABLE 15. Comparison results for constrained problem 4. "N.A" stands for not available.

Algorithm	Optimal values for variables		Optimum weight	Rank
	x_1	x_2		
ABC [45]	N.A	N.A	-6961.814	6
MBA [45]	N.A	N.A	-6961.813875	7
PSO	14.08919351	0.83093718	**-6.968587e+03**	**1**
SA-GA	14.06645200	0.78442061	-6.86487e+03	8
GSA	14.08940978	0.83119240	-6.96807e+03	3
SCA	14.16266815	0.97398070	-6.73678e+03	9
MVO	14.08928314	0.83151360	-6.96645e+03	5
SOA	14.11657823	0.89089956	-6.96723e+03	4
MSSOA	14.09144709	0.83565663	-6.968585e+03	2

TABLE 16. Performance comparison of algorithms for the constrained problem 5.

Algorithm	Optimal values for variables				
	Mean	Std.	Worst	Best	Time
PSO	-0.99655	0.002037	**-0.98892**	-0.99932	41.45
SA-GA	-0.19653	0.178712	-0.00153	-0.56083	1302.5
GSA	-0.222051	0.389941	-2.723e-05	-0.99285	63.07
SCA	-0.071263	0.165932	0	-0.53873	**27.03**
MVO	-0.998647	**9.3632e-04**	-0.99569	-0.99989	33.26
SOA	-0.998463	8.02951	-0.99666	-0.99936	124.9
MSSOA	**-0.999310**	4.8379e-04	-0.998257	**-1.00004**	300.8

e: CONSTRAINED PROBLEM 5

For the constrained problem, the mean values, standard deviation, worst fitness, best fitness, and the run time between the algorithms of 30 all alone runs, and the data of the constrained problem 5 optimization results are shown in Table 16. The underline and boldface indicate that the optimal outcome is better.

Based on Table 16, for the best value of the constrained problem, the MSSOA is better than the others. For the worst value, the MSSOA is better than the others except the PSO. For the standard deviation, the MSSOA is better than the others except the MVO. For the mean, the MSSOA is the best. For the time, the SCA is the shortest; the MSSOA is only shorter than that of the SA-GA.

The MSSOA is compared to the MBA [45], ABC [46], PSO, SA_GA, GSA, SCA, MVO, and the SOA, and provides the best-obtained values for variables and the best-obtained values in Table 17. According to Table 17, the MSSOA algorithm is better than the MBA, PSO, SA_GA, GSA, SCA, MVO, and the SOA. There is not much of a difference between the optimal value of the MSSOA algorithm and that of the ABC algorithm.

2) PRACTICAL CONSTRAINED ENGINEERING PROBLEMS

We are using 6 constrained engineering problems to test the capability of the MSSOA further. The formulations of these problems are available in Appendix B.

a: WELDED BEAM DESIGN PROBLEM

This is a least fabrication cost problem, which has four parameters and seven constraints. The parameters of the structural system are shown in Figure 12 [9].

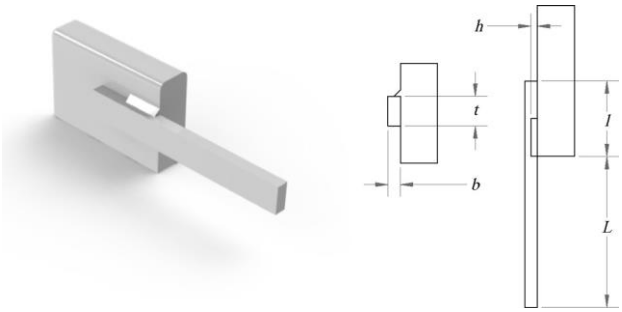

FIGURE 12. Design parameters of the welded beam design problem.

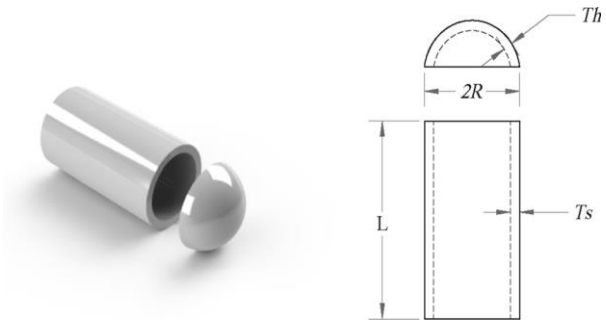

FIGURE 13. Pressure vessel design problem.

For the problem in this paper, the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA. The mean values, standard deviation, worst fitness, best fitness, and the run time between the algorithms of 30 all alone runs, and the data of the welded beam design problem optimization results are shown in Table 18. The underline and boldface indicate that the optimal outcome is better.

Based on Table 18, for the best value of the constrained problem, the MSSOA is better than the others algorithm. For the worst value, the standard deviation, and the mean, the MSSOA is also better than the others. For the time, the SCA is the shortest; the MSSOA is only shorter than that of the SA-GA.

Some of the works come from these kinds of literature: GSA [6], MFO [7], MVO [9], CPSO [47], and HS [48]. And the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA, and provides the best-obtained values for variables and the best-obtained values in Table 19.

In Table 19, the results of MSSOA algorithm are better than the GSA, MFO, MVO, CPSO, and the HS in other kinds of literature. The results of MSSOA are also better than the PSO, SA_GA, GSA, SCA, MVO, and the SOA.

b: PRESSURE VESSEL DESIGN PROBLEM

This is also the least fabrication cost problem of four parameters and four constraints. The parameters of the structural system are shown in Figure 13 [9].

For the problem in this paper, the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA. The mean values, standard deviation, worst fitness, best fitness, and the

TABLE 17. Comparison results for constrained problem 5, "N.A" stands for not available.

Algorithm	Optimal values for variables										Optimum weight	Rank
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}		
MBA [45]	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	-0.9998	5
ABC [46]	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	N.A	**-1**	**1**
PSO	0.31701	0.31878	0.31517	0.31416	0.31487	0.31436	0.31510	0.31442	0.32260	0.31571	-0.9993	7
SA-GA	0.28101	0.36264	0.32543	0.20339	0.24639	0.29019	0.43072	0.34228	0.39289	0.20078	-0.5608	9
GSA	0.31168	0.31549	0.32409	0.31275	0.31831	0.31277	0.32446	0.29530	0.32217	0.32418	-0.9929	8
SCA	0.32290	0.23659	0.34231	0.30454	0.38043	0.27937	0.42222	0.18004	0.25397	0.32969	-0.5387	10
MVO	0.31547	0.31567	0.31642	0.31740	0.31720	0.31416	0.31661	0.31546	0.31885	0.31507	-0.9999	4
SOA	0.31861	0.31810	0.31281	0.32214	0.31960	0.31128	0.31554	0.31306	0.31351	0.31753	-0.9994	6
MSSOA	0.31586	0.31569	0.31495	0.31623	0.31588	0.31599	0.31746	0.31695	0.31749	0.31575	-1.00004	2

TABLE 18. Performance comparison of algorithms for the welded beam design problem.

Result	Algorithms						
	PSO	SA-GA	GSA	SCA	MVO	SOA	MSSOA
Mean	1.788157235746	2.755187402106	3.088532351176	1.839667997861	1.719443638856	1.701777355905	**1.698400133899**
Std.	0.103424982300	0.617943995354	0.497165740387	0.045305420774	0.021705183473	0.006210252944	**0.002127438839**
Worst	2.046413310679	4.003256393174	3.925665010955	1.929612704440	1.777793458511	1.721858419034	**1.706257255757**
Best	1.697564416220	1.745541336597	2.022338397989	1.771438206326	1.697401277111	1.696109278951	**1.695936222372**
Time	16.258530	660.530622	28.804379	**8.418633**	10.941200	69.129909	183.911207

TABLE 19. Comparison results of the welded beam design problem.

Algorithm	Optimal values for variables				Optimal cost	Rank
	h	l	t	b		
GSA [6]	0.182129	3.856979	10.0000	0.202376	1.87995	10
MFO [7]	0.2057	3.4703	9.0364	0.2057	1.72452	5
MVO [9]	0.205463	3.473193	9.044502	0.205695	1.72645	6
CPSO [47]	0.202369	3.544214	9.048210	0.205723	1.72802	7
HS [48]	0.2442	6.2231	8.2915	0.2443	2.3807	12
PSO	0.205143375132900	3.260999556438029	9.050891106782855	0.205686317248074	1.697564416220149	4
SA-GA	0.184769549442834	3.707539689062885	8.913293291156665	0.211463867785461	1.745541336597454	8
GSA	0.188691249382521	4.518431717869246	9.083806192624566	0.227928590077372	2.022338397989212	11
SCA	0.206264473596472	3.491396540750882	8.881354620750962	0.215064864696517	1.771438206326408	9
MVO	0.205008451832884	3.274310401271678	9.035573526674412	0.205798767764750	1.697401277111472	3
SOA	0.201987971216458	3.327944107096977	9.037256929932264	0.205750385361675	1.696109278951345	2
MSSOA	0.205476896038301	3.262045628652172	9.041791068471218	0.205704435075519	**1.695936222371749**	**1**

run time between the algorithms of 30 all alone runs, and the data of the pressure vessel design problem optimization results are shown in Table 20. The underline and boldface indicate that the optimal outcome is better.

Based on Table 20, for the best value of the constrained problem, the MSSOA is better than the PSO, SA-GA, GSA, SCA, MVO, and the SOA. For the worst value, the standard deviation, and the mean, the MSSOA is better than the others. For the time, the SCA is the shortest; the MSSOA is only shorter than that of the SA-GA.

Some of the works come from the literature: MFO [7], ES [49], DE [50], ACO [51], and GA [52]. And the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA, and provides the best-obtained values for variables and the best-obtained values in Table 21.

For the problem, the MSSOA algorithm is better than the MFO, ES, DE, ACO, and the GA algorithms in other kinds of literature. The MSSOA is also better than the PSO, SA_GA, GSA, SCA, MVO, and the SOA.

C: CANTILEVER BEAM DESIGN PROBLEM

This is a problem that is determined by 5 parameters and is only applied to the scope of the variables of constraints. The parameters of the structural system are shown in Figure 14 [7].

For the problem in this paper, the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA. The mean values, standard deviation, worst fitness, best fitness, and the run time between the algorithms of 30 all alone runs, and the data of the cantilever beam design problem optimization results are shown in Table 22. The underline and boldface indicate that the optimal outcome is better.

Based on Table 22, for the best value of the constrained problem, the MSSOA is better than the PSO, SA-GA, GSA, SCA, MVO and the SOA. For the worst value, the standard deviation, and the mean, the MSSOA is better than the others. For the time, the SCA is the shortest; the MSSOA is only shorter than that of the SA-GA.

TABLE 20. Performance comparison of algorithms for the pressure vessel design problem.

Result	Algorithms						
	PSO	SA-GA	GSA	SCA	MVO	SOA	MSSOA
Mean	1.42839441e+04	6.93682335e+03	6.49222884e+03	4.25987350e+04	6.41381975e+03	5.99980731e+03	**5.82746316e+03**
Std.	1.85910050e+04	3.38090560e+02	3.88660018e+02	2.63296156e+04	4.43351485e+02	3.20646873e+02	**2.18668682e+02**
Worst	8.93441840e+04	7.19285121e+03	7.40712578e+03	7.54857662e+04	7.14161898e+03	6.81153704e+03	**6.65135067e+03**
Best	6.61263274e+03	5.90783779e+03	6.02589090e+03	5.82113657e+03	5.75119625e+03	5.73534518e+03	**5.73510204e+03**
Time	16.486317	688.353289	26.375727	**10.706477**	12.903910	81.821338	219.366853

TABLE 21. Comparison results for the pressure vessel design problem.

Algorithm	Optimal values for variables				Optimal cost	Rank
	T_s	T_h	R	L		
MFO [7]	0.8125	0.4375	42.098445	176.636596	6059.7143	8
ES [49]	0.8125	0.4375	42.098087	176.640518	6059.7456	10
DE [50]	0.8125	0.4375	42.098411	176.637690	6059.7340	9
ACO [51]	0.8125	0.4375	42.103624	176.572656	6059.0888	7
GA [52]	0.8125	0.4375	42.097398	176.654050	6059.9463	11
PSO	0.8855303751578	0.4524306944924	47.3710175872595	139.4774510560814	6612.632735153998	12
SA-GA	0.8286437203582	0.4076337806232	44.5808343023210	148.1276798336512	5907.837786470595	5
GSA	0.8413964541417	0.4134042204535	45.2230377372840	144.8944326676142	6025.890901037682	6
SCA	0.7676969577481	0.3576083643660	40.6299488066177	196.3985187764563	5821.136566031909	4
MVO	0.7454084468015	0.3669001422895	40.6040426419039	196.1357865457513	5751.196248760631	3
SOA	1.042698229610387	0.502612582123923	55.209890788137130	61.748997269949271	5735.345181442674	2
MSSOA	0.7420518229763	0.3715914566788	40.3329487051406	199.8628982503359	**5735.102040019222**	**1**

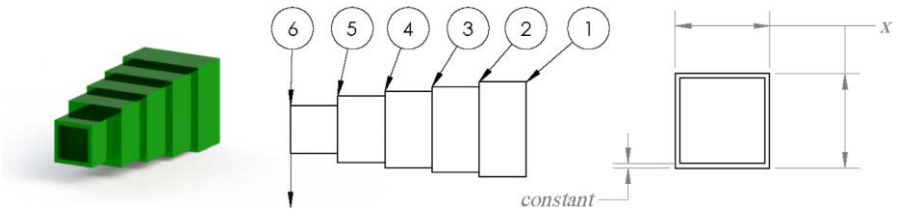

FIGURE 14. Cantilever beam design problem.

Some of the works come from these kinds of literature: MFO [7], CS [53], GCA [54], MMA [54], and SOS [55]. And the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA, and provides the best-obtained values for variables and the best-obtained values in Table 23.

In Table 23, the MSSOA algorithm proves to be better than the MFO, CS, GCA, MMA, and the SOS algorithms in other kinds of literature. The MSSOA is also better than the PSO, SA_GA, GSA, SCA, MVO, and the SOA.

d: TUBULAR COLUMN DESIGN

This is also a minimum cost problem of two parameters and six constraints. The parameters of the structural system are shown in Figure 15 [56].

For the problem in this paper, the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA. The mean values, standard deviation, worst fitness, best fitness, and the run time between the algorithms of 30 all alone runs, and the data of the tubular column design problem optimization results are shown in Table 24. The underline and boldface indicate that the optimal outcome is better.

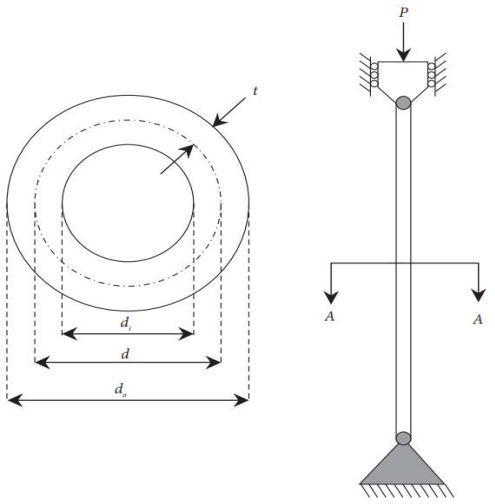

FIGURE 15. Tubular column design problem.

Based on Table 24, for the best value of the constrained problem, the MVO is the best; the MSSOA is better than that of other algorithms. For the worst value and the standard

TABLE 22. Performance comparison of algorithms for the cantilever beam design problem.

Result	Algorithms						
	PSO	SA-GA	GSA	SCA	MVO	SOA	MSSOA
Mean	1.340044656035	1.64899842786	1.34024454022	1.3910341586	1.34059792791	1.33998704816	**1.33997715057**
Std.	8.79249987e-05	0.27193805858	3.5214752e-04	0.0172732002	3.75586210e-04	2.10772933e-05	**1.9652352e-05**
Worst	1.34034669200	2.26096076762	1.34173921342	1.429682073511	1.34130109731	1.34005939342	**1.3400548758**
Best	1.33996051444	1.35443574643	1.33995803663	1.361411514762	1.34001137692	1.33995352050	**1.3399522567**
Time	19.322597	598.869855	28.586624	**9.175697**	12.077241	52.943335	164.983097

TABLE 23. Comparison results for the cantilever beam design problem.

Algorithm	Optimal values for variables					Optimum weight	Rank
	x_1	x_2	x_3	x_4	x_5		
MFO [7]	5.9848717732	5.3167269243	4.4973325858	3.5136164677	2.1616202934	1.339988086	6
CS [53]	6.0089	5.3049	4.5023	3.5077	2.1504	1.33999	7
GCA [54]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400	8
MMA [54]	6.0100	5.3000	4.4900	3.4900	2.1500	1.3400	8
SOS [55]	6.01878	5.30344	4.49587	3.49896	2.15564	1.33996	4
PSO	6.0053406124	5.3177844495	4.4879695880	3.5090481883	2.1535708454	1.339960514	5
SA-GA	6.2396310580	5.7001619772	4.6524696443	3.2778615899	1.8354638207	1.354435746	11
GSA	6.0083043234	5.3157704728	4.5033485026	3.4939663933	2.1522404325	1.339958037	3
SCA	6.3738759865	4.9903030313	4.3841035306	3.3449830328	2.7242266426	1.361411515	12
MVO	6.0169985301	5.3216277767	4.4830302436	3.5165667788	2.1354271906	1.340011377	10
SOA	6.0110848387	5.2989655437	4.4903004818	3.5245272179	2.1489317851	1.339953520	2
MSSOA	6.0170977409	5.2965179429	4.4949907330	3.4942666705	2.1710731191	**1.339952257**	**1**

TABLE 24. Performance comparison of algorithms for the tubular column design problem.

Result	Algorithms						
	PSO	SA-GA	GSA	SCA	MVO	SOA	MSSOA
Mean	26.48817581983	27.50685171232	27.25847209185	26.61047972329	26.48795645950	26.48851288085	**26.48744720697**
Std.	0.001266135080	0.906067717414	1.036218793187	0.066795689421	**0.001204566021**	0.001570913109	0.001633617916
Worst	26.49111192810	29.90764214511	31.77298392287	26.80758004649	**26.49096384150**	26.492298210474	26.49531524954
Best	26.48643791207	26.54702124371	26.59900262493	26.50433585534	**26.48627239710**	26.486647729114	26.48631993554
Time	16.634990	593.690108	23.016568	**7.732375**	11.092001	77.181567	120.929135

deviation, the MSSOA is better than the SA-GA, GSA, and the SCA, and the MSSOA is worse than PSO, MVO, and the SOA. For the mean value, the MSSOA is the MVO is the best. For the time, the SCA is the shortest; the MSSOA is only shorter than that of the SA-GA.

Some of these works come from the kinds of literature: CS [57], ISA [58], FA [59], ASO [60], SNS [56]. And the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA, and provides the best-obtained values for variables and the best obtained values in Table 25.

In Table 25, the MSSOA algorithm proves to be better than the CS, ISA, FA, ASO, and the SNS algorithms in other kinds of literature. Except for the MVO, the MSSOA is also better than the PSO, SA-GA, GSA, SCA, and the SOA. The result of the MSSOA has reached the theoretical best solution, although the optimum of the MSSOA is worse than that of the MVO.

e: PISTON LEVER PROBLEM

This is a locating the piston components problem, which has four variables and four constraints. Figure 16 is the schematic diagram [56].

TABLE 25. Comparison results of the tubular column design problem.

Algorithm	Optimal values for variables		Optimal cost	Rank
	$x_1(d)$	$x_2(t)$		
CS [57]	5.45139	0.29196	26.53217	10
ISA [58]	5.45115623	0.29196547	26.5313	8
FA [59]	N/A	N/A	26.4994969	5
ASO [60]	N/A	N/A	26.53137828	9
SNS [56]	26.5313	26.5313	26.4994969	5
PSO	5.45241248	0.29161380	26.48643791	3
SA-GA	5.48259506	0.28999485	26.54702124	11
GSA	5.46443535	0.29261817	26.59900262	12
SCA	5.45179801	0.29199765	26.50433586	7
MVO	5.45225365	0.29161486	**26.48627240**	**1**
SOA	5.45386190	0.29153427	26.48664773	4
MSSOA	5.45219456	0.29162428	26.48631994	2

For the problem in this paper, the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA. The mean values, standard deviation, worst fitness, best fitness, and the run time between the algorithms of 30 all alone runs, and the data of the piston lever problem optimization results are shown in Table 26. The underline and boldface indicate that the optimal outcome is better.

Based on Table 26, for the best value of the constrained problem, the MSSOA is best. For the worst value, the

TABLE 26. Performance comparison of algorithms for the piston lever problem.

Result	Algorithms						
	PSO	SA-GA	GSA	SCA	MVO	SOA	MSSOA
Mean	1.8924714e+04	3.8395338e+02	8.0900632e+04	9.190761525	1.5870761e+02	**8.99463714**	1.1855562e+02
Std.	7.1700790e+04	7.0949773e+02	1.6243533e+05	**0.347684151**	1.2582486e+02	2.1857150	1.2356288e+02
Worst	2.8416627e+05	3.1853816e+03	7.0125150e+05	**9.922422079**	2.9937715e+02	18.75755139	4.0213748e+02
Best	10.24259065	8.41373929	3.29028058e+02	8.52096198	8.47942385	8.41347665	**8.4134112**
Time	16.627484	613.857016	25.635308	**8.816001**	11.093057	66.245904	113.090778

TABLE 27. Comparison results of the piston lever problem.

Algorithm	Optimal values for variables				Optimal value	Rank
	$x_1(H)$	$x_2(B)$	$x_3(D)$	$x_4(X)$		
DE [61]	129.4	2.43	119.80	4.75	159	9
GA [61]	250.0	3.96	60.03	5.91	161	11
HPSO [61]	135.5	2.48	116.62	4.75	161	11
CS [57]	0.050	2.043	120.000	4.085	8.427	5
SNS [56]	0.050	2.042	120.000	4.083	**8.412698349**	**1**
PSO	0.0500000000000	2.2004458009526	4.3429316143480	110.6668235984532	10.24259065154840	8
SA-GA	0.0500000000000	2.0417619164866	4.0830416789885	119.9999715756912	8.413739285922011	4
GSA	215.9646294854686	344.6817103997157	03.2179597335794	60.1907279614403	329.0280579311141	12
SCA	0.0589732446248	2.0456822965266	4.0848813955281	120.0000000000000	8.520961983238227	7
MVO	0.0500000000000	2.0502088605801	4.0908333358488	119.9640112340799	8.479423852552021	6
SOA	0.7649083155587	2.0351385415011	4.0554693386457	120.0000000000000	8.413476646923973	3
MSSOA	0.0500000000000	2.0433232146874	4.0832945913726	120.0000000000000	8.413411204004042	2

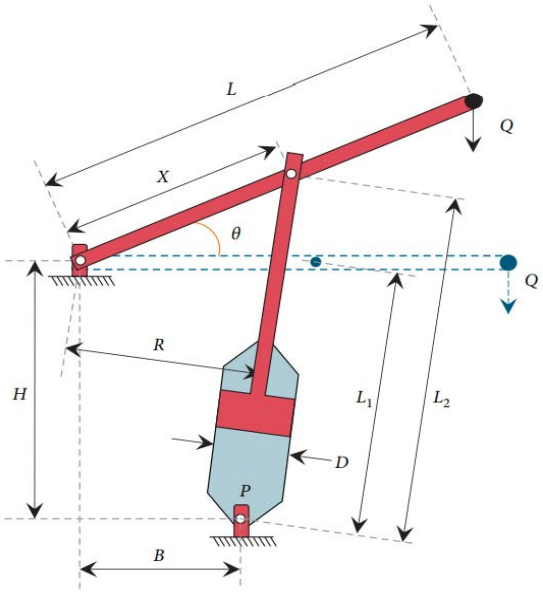

FIGURE 16. Piston lever problem.

MSSOA is better than the PSO, SA-GA, and the GSA, and the MSSOA is worse than the SCA, MVO and the SOA. For the standard deviation and the mean, the MSSOA is better than the PSO, SA-GA, GSA, and the MVO, and the MSSOA is worse than the SCA and the SOA. For the time, the SCA is the shortest; the MSSOA is only shorter than that of the SA-GA.

Some of these works come from the kinds of literature: DE [61], GA [61], HPSO [61], CS [57], SNS [56]. And the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA, and provides the best-obtained values for variables and the best obtained values in Table 27.

In Table 27, except for the SNS, the MSSOA algorithm proves to be better than the DE, GA, HPSO, and the CS algorithm in other kinds of literature. The MSSOA is also better than the PSO, SA-GA, SCA, GSA, MVO, and the SOA. The result of the MSSOA has reached the theoretical best solution, although the optimum of the MSSOA is worse than that of the SNS algorithm.

f: REINFORCED CONCRETE BEAM DESIGN PROBLEM

This is an optimization problem of designing a reinforced concrete beam, which has three variables and two constraints. Figure 17 is the schematic diagram [56].

For the problem in this paper, the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA. The mean values, standard deviation, worst fitness, best fitness, and the run time between the algorithms of 30 all alone runs, and the data of the reinforced concrete beam design problem optimization results are shown in Table 28. The underline and boldface indicate that the optimal outcome is better.

Based on Table 28, for the best value of the constrained problem, the PSO and the SA-GA are the best; the MSSOA is same the SOA; and the MSSOA is better than the SCA, GSA, and the MVO. For the worst value, the standard deviation, and the mean, the MSSOA is better than the SA-GA, GSA, SCA, and the GSA, and the MSSOA is worse than the PSO and the SOA. For the time, the SCA is the shortest; the MSSOA is only shorter than that of the SA-GA.

Some of these works come from the kinds of literature: GHN-EP [61], FA [62], CS [57], ASO [60], SNS [56]. And

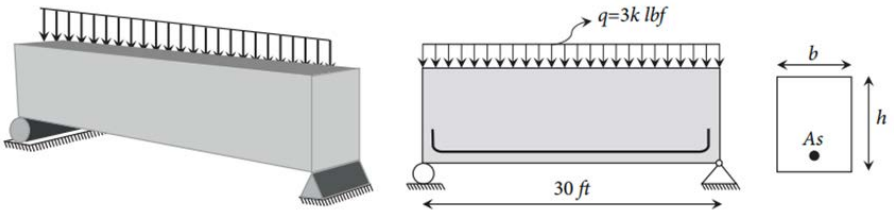

FIGURE 17. Reinforced concrete beam design problem.

TABLE 28. Performance comparison of algorithms for the reinforced concrete beam design problem.

Result	Algorithms						
	PSO	SA-GA	GSA	SCA	MVO	SOA	MSSOA
Mean	**359.20330245**	362.93372827	364.15942375	3.6169104797	361.21732296	359.20330848	359.20331897
Std.	**9.7088183e-13**	3.309756009202	3.15927997028	1.261266108362	1.563144445	8.9104475e-06	3.4545089e-05
Worst	**359.20330245**	378.8853225476	372.68688107	362.6203630796	362.6336325	359.20334720	359.20348620
Best	**359.20330245**	**359.20330245**	359.62412635	359.23116167	359.20335774	359.20330246	359.20330246
Time	16.194841	717.071326	24.905824	**7.269785**	11.001462	65.146201	122.401593

TABLE 29. Comparison results of the reinforced concrete beam design problem.

Algorithm	Optimal values for variables			Optimal value	Rank
	$x_1(A_s)$	$x_2(b)$	$x_3(h)$		
GHN-EP [61]	6.32	34	8.637180	362.00648	12
FA [62]	6.32	34	8.5000	359.2080	6
CS [57]	6.32	34	8.5000	359.2080	6
ASO [60]	6.32	34	8.5000	359.2080	6
SNS [56]	6.32	34	8.5000	359.2080	6
PSO	0.21973978	0.48774465	8.49953948	**359.20330245**	**1**
SA-GA	0.22271591	0.49673391	8.49953948	**359.20330245**	**1**
GSA	0.27021862	0.46197772	8.52039835	359.62412635	11
SCA	0.23315802	0.50078494	8.49841820	359.23116167	10
MVO	0.22893546	0.48404040	8.49958944	359.20335774	5
SOA	0.21878243	0.50307295	8.49954126	359.20330246	3
MSSOA	0.24071606	0.53598239	8.49954508	359.20330246	3

the MSSOA is compared to the PSO, SA_GA, GSA, SCA, MVO, and the SOA, and provides the best-obtained values for variables and the best obtained values in Table 29.

In Table 29, the MSSOA algorithm proves to be better than the GHN-EP, FA, CS, ASO, and the SNS algorithm in other kinds of literature. Except for the PSO, SA_GA, and the SOA, the MSSOA is also better than the GSA, SCA, and the MVO. The result of the MSSOA has reached the theoretical best solution, although the optimum of the MSSOA is worse than that of the PSO and the SA_GA algorithm.

V. CONCLUSION

A MSSOA algorithm is presented, with a triple black hole system capture and interference methods. According to the three phases to test and analyze the MSSOA from different perspectives. The three phases include improvement of SOA algorithms, optimization of 15 benchmark functions, optimization of 5 constraint problem examples and optimization of 6 constraint engineering problems respectively.

In the first phase, the SOA is improved in four different ways: the TBHSOA, MISOA, PISOA, and the MSSOA. Each improved algorithm was optimized for the 15 functions. In the phase, we consider the ranking values of 30 all alone running between MSSOA mean values, standard deviation values, best fitness values, and best fitness values rank, convergence curves of function $f1$ and $f10$, and the population's positions with iterations and search history of function $f1$ and $f10$. From the comparative study, the results of MSSOA are better than the other improved SOA algorithms.

In the second phase, 15 benchmark functions optimization problems are used to test the MSSOA further. The MSSOA is compared to the PSO, SA-GA, GSA, SCA, MVO, and the SOA for verification. In the benchmark functions optimization problems, the precision, the complexity, the Wilcoxon's rank-sum test, the run time, the exploration and exploitation abilities, and the results of the performance ratios of the average solution of the MSSOA are researched. From the comparative analyzing, the performance of MSSOA is better than the other algorithms.

In the last phase, 5 constrained problem examples and 6 engineering optimization problems further tested the MSSOA. The MSSOA was compared to various algorithms. The results demonstrate that the MSSOA can achieve very competitive performance.

The further improving and application of MSSOA should be incorporated into future studies. The improved SOA algorithm and the heuristic algorithms base on those improved strategies can not only be applied to engineering optimization problems, but also to path planning problems, pattern recognition, intelligent control and other fields. In addition to these improvement strategies mentioned in the paper, other improvement methods such as binary coding, complex coding and hybrid other algorithms should also be introduced to further improve the algorithm.

APPENDIX A

I – Constrained problem 1

$$\begin{aligned}&\text{Minimize } f(x) = x_1^2 + (x_2 - 1)^2,\\ &\text{Subject to } h(x) = x_2 - x_1^2 = 0,\\ &\text{Variable range } -1 \le x_i \le 1, i = 1, 2,\end{aligned}$$

II – Constrained problem 2

$$\begin{aligned}&\text{Minimize } f(x) = (x_2^2 + x_1 - 11)^2 + (x_1 + x_2^2 - 7)^2,\\ &\text{Subject to } g_1(x) = 4.84 - (x_1 - 0.05)^2 - (x_2 - 2.5)^2 \ge 0,\\ &\qquad g_2(x) = x_1^2 + (x_2 - 2.5)^2 - 4.84 \ge 0,\\ &\text{Variable range } 0 \le x_i \le 6, \quad i = 1, 2.\end{aligned}$$

III – Constrained problem 3

$$\begin{aligned}&\text{Maximum } f(x) = \frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1 + x_2)},\\ &\text{Subject to } g_1(x) = x_1^2 - x_2 + 1 \le 0,\\ &\qquad g_2(x) = 1 - x_1 + (x_2 - 4)^2 \le 0,\\ &\text{Variable range } 0 \le x_i \le 10, \quad i = 1, 2.\end{aligned}$$

IV – Constrained problem 4

$$\begin{aligned}&\text{Minimize } f(x) = (x_1 - 10)^3 + (x_2 - 20)^3,\\ &\text{Subject to } g_1(x) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \le 0,\\ &\qquad g_2(x) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \le 0,\\ &\text{Variable range } 13 \le x_1 \le 100, \quad 0 \le x_2 \le 100.\end{aligned}$$

V–Constrained problem 5

$$\begin{aligned}&\text{Minimize } f(x) = -(\sqrt{n})^n \cdot + \prod_{i=1}^{n} x_i,\\ &\text{Subject to } h(x) = \sum_{i=1}^{n} x_i^2 = 1,\\ &\text{Variable range } 0 \le x_i \le 1, \quad i = 1, 2, \ldots, n.\end{aligned}$$

APPENDIX B

I – Welded beam design problem

$$\begin{aligned}&\text{Consider } \vec{x} = [x_1, x_2, x_3, x_4] = [h, l, t, b],\\ &\text{Minimize } f(\vec{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14 + x_2),\\ &\text{Subject to } g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \le 0,\\ &\qquad g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \le 0,\\ &\qquad g_3(\vec{x}) = x_1 - x_4 \le 0,\\ &\qquad g_4(\vec{x}) = 1.10471x_1^2\\ &\qquad\quad + 0.04811x_3x_4(14 + x_2) - 5 \le 0,\\ &\qquad g_5(\vec{x}) = 0.125 - x_1 \le 0,\\ &\qquad g_6(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \le 0,\\ &\qquad g_7(\vec{x}) = P - P_c(\vec{x}) \le 0,\\ &\text{Variable range } 0.1 \le x_1 \le 2, \quad 0.1 \le x_2 \le 10,\\ &\qquad 0.1 \le x_3 \le 10, \quad 0.1 \le x_4 \le 2\end{aligned}$$

where

$$\begin{aligned}\tau(\vec{x}) &= \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2},\\ \tau' &= \frac{P}{\sqrt{2}\,x_1x_2}, \quad \tau'' = \frac{MP}{J}, \; M = P(L + \frac{x_2}{2}),\\ R &= \sqrt{\frac{x_2^2}{4} + (\frac{x_1 + x_3}{2})^2},\\ J &= 2\left\{\sqrt{2}\,x_1x_2\left[\frac{x_2^2}{4} + (\frac{x_1 + x_3}{2})^2\right]\right\},\\ \sigma(\vec{x}) &= \frac{6PL}{x_4x_3^2}, \quad \delta(\vec{x}) = \frac{6PL^3}{Ex_4x_3^3},\\ P_c(\vec{x}) &= \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right),\end{aligned}$$

$P = 6000$ lb, $L = 14$ in, $E = 30 \times 10^6$ psi, $G = 12 \times 10^6$ psi, $\tau_{\max} = 136000$ psi, $\sigma_{\max} = 30000$ psi, $\delta_{\max} = 0.25$ in.

II – Pressure vessel design problem

$$\begin{aligned}&\text{Consider } \vec{x} = [x_1, x_2, x_3] = [T_s, T_h, R, L],\\ &\text{Minimize } f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2\\ &\qquad + 3.1661x_1^2x_4 + 19.84x_1^2x_3,\\ &\text{Subject to } g_1(\vec{x}) = -x_1 + 0.0193x_3 \le 0,\\ &\qquad g_2(\vec{x}) = -x_2 + 0.00954x_3 \le 0,\\ &\qquad g_3(\vec{x}) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0,\\ &\qquad g_4(\vec{x}) = x_4 - 240 \le 0,\\ &\text{Variable range } 0 \le x_1 \le 99, \quad 0 \le x_2 \le 99,\\ &\qquad 10 \le x_3 \le 200, \quad 10 \le x_4 \le 200.\end{aligned}$$

III – Cantilever design problem

$$\begin{aligned}&\text{Consider } \vec{x} = [x_1, x_2, x_3, x_4, x_5],\\ &\text{Minimize } f(\vec{x}) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5),\\ &\text{Subject to } g(\vec{x}) = \frac{61}{x_1^3} + \frac{37}{x_2^3} + \frac{19}{x_3^3} + \frac{7}{x_4^3} + \frac{1}{x_5^3} - 1 \le 0,\\ &\text{Variable range } 0.01 \le x_1, x_2, x_3, x_4, x_5 \le 100.\end{aligned}$$

IV – Tubular column design problem

$$\begin{aligned}&\text{Consider } \vec{x} = [x_1, x_2] = [d, t]\\ &\text{Minimize } f(\vec{x}) = 9.8\,x_1x_2 + 2x_1\end{aligned}$$

$$\text{Subject to } g_1(\vec{x}) = \frac{P}{\pi x_1 x_2 \sigma_y} - 1 \le 0$$
$$g_2(\vec{x}) = \frac{8PL^2}{\pi^3 \, \mathrm{E} x_1 \, x_2 \left(x_1^2 + x_2^2\right)} - 1 \le 0$$
$$g_3(\vec{x}) = \frac{2 \cdot 0}{x_1} - 1 \le 0$$
$$g_4(\vec{x}) = \frac{x_1}{14} - 1 \le 0$$
$$g_5(\vec{x}) = \frac{0.2}{x_2} - 1 \le 0$$
$$g_6(\vec{x}) = \frac{x_2}{8} - 1 \le 0$$
$$\text{Variable range } 2 \le x_1 \le 14, \quad 0.2 \le x_2 \le 0.8$$

where $\sigma_y = 500$ kgf /cm^2, E $= 0.85 \times 10^6$ kgf /cm^2.

V –Piston lever problem

$$\text{Consider } \vec{x} = [x_1, x_2, x_3, x_4] = [H, B, D, X]$$
$$\text{Minimize } f(\vec{x}) = \frac{1}{4}\pi x_3^2 \left(L_2 - L_1\right)$$
$$\text{Subject to } g_1(\vec{x}) = QL\cos\theta - R \times F \le 0,$$
$$g_2(\vec{x}) = Q\left(L - x_4\right) - M_{\max} \le 0,$$
$$g_3(\vec{x}) = 1.2\left(L_2 - L_1\right) - L_1 \le 0,$$
$$g_4(\vec{x}) = \frac{x_3}{0} - x_2 \le 0,$$

where:

$$R = \frac{\left|-x_4\left(x_4\sin\theta + x_1\right) + x_1\left(x_2 - x_4\cos\theta\right)\right|}{\sqrt{\left(x_4 - x_2\right)^2 + x_1^2}},$$
$$F = \frac{\pi P x_3^2}{4},$$
$$L_1 = \sqrt{\left(x_4 - x_2\right)^2 + x_1^2},$$
$$L_2 = \sqrt{\left(x_4\sin\theta + x_1\right)^2 + \left(x_2 - x_4\cos\theta\right)^2},$$
$$\theta = 45^\circ, \quad Q = 100001\text{bs}, \; L = 240\text{in},$$
$$M_{\max} = 1.8 \times 10^6 1\text{bsin}, \quad P = 1500\text{psi},$$

$$\text{Variable range } 0.05 \le x_1, x_2, x_4 \le 500, \; 0.05 \le x_3 \le 120.$$

VI –Reinforced concrete beam design problem

$$\text{Consider } \vec{x} = [x_1, x_2, x_3] = [A_s, b, h]$$
$$\text{Minimize } f(\vec{x}) = 2.9\, x_1 + 0.6\, x_2 x_3$$
$$\text{Subject to } g_1(\vec{x}) = \frac{x_2}{x_3} - 4 \le 0,$$
$$g_2(\vec{x}) = 180 + 7.375\frac{x_1^2}{x_3} - x_1 x_2 \le 0,$$
$$\text{Variable range } x_1 \in \{6, 6.16, 6.32, 6.6,$$
$$7, 7.11, 7.2, 7.8, 7.9, 8, 8.4\},$$
$$x_2 \in \{28, 29, 30, \ldots, 40\}, \quad 5 \le x_3 \le 10.$$

REFERENCES

[1] D. H. Wolper and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.
[2] J. H. Holland, "Genetic algorithms," *Sci. Amer.*, vol. 267, no. 1, pp. 66–72, Jul. 1992.
[3] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. 6th Int. Symp. Micro Mach. Hum. Sci. (MHS)*, Apr. 1995, pp. 39–43.
[4] E. H. L. Aarts and P. J. M. Laarhoven, "Simulated annealing: An introduction," *Stat Neerl*, vol. 43, pp. 31–52, Mar. 1989.
[5] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: Harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, Feb. 2001.
[6] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi, "GSA: A gravitational search algorithm," *J. Inf. Sci.*, vol. 179, no. 13, pp. 2232–2248, 2009.
[7] S. Mirjalili, "Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm," *Knowl.-Based Syst.*, vol. 89, pp. 228–249, Nov. 2015.
[8] S. Mirjalilia, "SCA: A sine cosine algorithm for solving optimization problems," *Knowl.-Based Syst.*, vol. 96, no. 27, pp. 1–14, Mar. 2016.
[9] S. Mirjalilia, S. M. Mirjalili, and A. Hatamlou, "Multi-Verse Optimizer: A nature-inspired algorithm for global optimization," *Neural Comput. Appl.*, vol. 17, pp. 16–19, Feb. 2015.
[10] S. Talatahari, H. Bayzidi, and M. Saraee, "Social network search for global optimization," *IEEE Access*, vol. 9, pp. 92815–92863, 2021.
[11] M. Tuba, I. Brajevic, and R. Jovanovic, "Hybrid seeker optimization algorithm for global optimization," *Appl. Math. Inf. Sci.*, vol. 7, no. 3, pp. 867–875, May 2013.
[12] Y. S. Barkatou, "A discrete bat algorithm based on Levy flights for Euclidean traveling salesman problem," *Expert Syst. Appl.*, vol. 172, pp. 1–17, Jun. 2021.
[13] E. Shadkam, "Cuckoo optimization algorithm in reverse logistics: A network design for COVID-19 waste management," *Waste Manage. Res., J. Sustain. Circular Economy*, vol. 3, pp. 1–12, Mar. 2021.
[14] M. Kelidari and J. Hamidzadeh, "Feature selection by using chaotic cuckoo optimization algorithm with Levy flight, opposition-based learning and disruption operator," *Soft Comput.*, vol. 25, no. 4, pp. 2911–2933, 2020.
[15] Y. Wang and Z. Ma, "Elite symbiotic organisms search algorithm based on subpopulation stretching operation," *Control Decis.*, vol. 34, no. 7, pp. 1355–1364, 2019.
[16] X. Chu, S. Li, D. Gao, W. Zhao, J. Cui, and L. Huang, "A binary superior tracking artificial bee colony with dynamic Cauchy mutation for feature selection," *Complexity*, vol. 2020, pp. 6–25, Nov. 2020.
[17] K. M. Deb, "A combined genetic adaptive search (GeneAS) for engineering design," *Comput. Sci. Inform.*, vol. 26, no. 4, pp. 30–45, 1996.
[18] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson, "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 240–255, Jun. 2004.
[19] J. J. Jui and M. A. Ahmad, "A hybrid Metaheuristic algorithm for identification of continuous-time Hammerstein systems," *Appl. Math. Model.*, vol. 95, pp. 339–360, Jul. 2021.
[20] C. Dai, W. Chen, and Y. Zhu, "Seeker optimization algorithm," in *Proc. Int. Conf. Comput. Intell. Secur.*, Nov. 2006, pp. 225–229.
[21] C. Dai, Y. Zhu, and W. Chen, "Seeker optimization algorithm," in *Computational Intelligence and Security* (Lecture Notes in Computer Science), vol. 4456. Guangzhou, China: IEEE Press, 2007, pp. 167–176.
[22] C. Dai, W. Chen, Y. Zhu, and X. Zhang, "Seeker optimization algorithm for optimal reactive power dispatch," *IEEE Trans. Power Syst.*, vol. 24, no. 3, pp. 1218–1231, May 2009.
[23] C. Dai, W. Chen, Y. Song, and Y. Zhu, "Seeker optimization algorithm: A novel stochastic search algorithm for global numerical optimization," *J. Syst. Eng. Electron.*, vol. 21, no. 2, pp. 300–311, Apr. 2010.
[24] C. Dai, W. Chen, and Y. Zhu, "Seeker optimization algorithm for digital IIR filter design," *IEEE Trans. Ind. Electron.*, vol. 57, no. 5, pp. 1710–1718, May 2010.
[25] C. Dai, W. Chen, Y. Zhu, Z. Jiang, and Z. You, "Seeker optimization algorithm for tuning the structure and parameters of neural networks," *Neurocomputing*, vol. 74, no. 6, pp. 876–883, 2011.
[26] C. Dai, W. Chen, Z. Cheng, Q. Li, Z. Jiang, and J. Jia, "Seeker optimization algorithm for global optimization: A case study on optimal modelling of proton exchange membrane fuel cell (PEMFC)," *Int. J. Electr. Power Energy Syst.*, vol. 33, no. 3, pp. 369–376, Mar. 2011.
[27] C. Dai, W. Chen, L. Ran, Y. Zhang, and Y. Du, "Human group optimizer with local search," in *Advances in Swarm Intelligence* (Lecture Notes in Computer Science), vol. 6728. Berlin, Germany: Springer, 2011, pp. 310–320.
[28] Y. Zhu, C. Dai, and W. Chen, "Seeker optimization algorithm for several practical applications," *Int. J. Comput. Intell. Syst.*, vol. 7, no. 2, pp. 353–359, 2014.
[29] M. J. Valtonen, "Merging galaxies and black hole ejections," *Int. Astronomical Union Colloq.*, vol. 124, pp. 497–502, Nov. 1990.

[30] M. J. Valtonen, "Triple black hole systems formed in mergers of galaxies," *Monthly Notices Roy. Astronomical Soc.*, vol. 278, no. 1, pp. 186–190, Jan. 1996.
[31] S. E. Sultan, "An emerging focus on plant ecological development," *New Phytologist*, vol. 166, no. 1, pp. 1–5, Apr. 2005.
[32] C. H. Waddington, *The Strategy of the Genes*. London, U.K.: Routledge, 2014.
[33] L. Li, Y. Zhou, and J. Xie, "A free search krill herd algorithm for functions optimization," *Math. Problems Eng.*, vol. 2014, pp. 1–21, Jan. 2014.
[34] X. Li, J. Zhang, and M. Yin, "Animal migration optimization: An optimization algorithm inspired by animal migration behavior," *Neural Comput. Appl.*, vol. 24, nos. 7–8, pp. 1867–1877, 2014.
[35] M. Molga and C. Smutnicki. (2005). *Test Functions for Optimization Needs*. [Online]. Available: http://www.robertmarks.org/Classes/ENGR5358/Papers/functions.pdf
[36] J. Kennedy, *Particle Swarm Optimization in Encyclopedia of Machine Learning*. New York, NY, USA: Springer, 2010, pp. 760–766.
[37] H. Yu, H. Fang, P. Yao, and Y. Yuan, "A combined genetic algorithm/simulated annealing algorithm for large scale system energy integration," *Comput. Chem. Eng.*, vol. 24, no. 8, pp. 2023–2035, Sep. 2000.
[38] S. Dasgupta, C. H. Papadimitriou, and U. V. Vazirani, *Algorithms*. New York, NY, USA: McGraw-Hill, 2009, pp. 15–18.
[39] A. Faramarzi, M. Heidarinejad, B. Stephens, and S. Mirjalilia, "Equilibrium optimizer: A novel optimization algorithm," *Knowl.-Based Syst.*, vol. 191, pp. 1–21, 2020.
[40] M. Črepinšek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: A survey," *ACM Comput. Surv.*, vol. 45, no. 3, pp. 1–33, Jun. 2013.
[41] K. Hussain, M. N. M. Salleh, S. Cheng, and Y. Shi, "On the exploration and exploitation in popular swarm-based Metaheuristic algorithms," *Neural Comput. Appl.*, vol. 31, no. 11, pp. 7665–7683, Nov. 2019.
[42] B. Morales-Castaeda, D. Zaldivar, E. Cuevas, F. Fausto, and A. Rodríguez, "A better balance in metaheuristic algorithms: Does it exist?" *Swarm Evol. Comput.*, vol. 54, pp. 1–23, May 2020.
[43] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.
[44] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Math. Program.*, vol. 91, no. 2, pp. 201–213, 2002.
[45] A. Sadollah, A. Bahreininejad, H. Eskandar, and M. Hamdi, "Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems," *Appl. Soft Comput.*, vol. 13, no. 5, pp. 2592–2612, May 2013.
[46] K. Deb, "An efficient constraint handling method for genetic algorithms," *Comput. Methods Appl. Mech. Eng.*, vol. 186, nos. 2–4, pp. 311–338, 2000.
[47] R. A. Krohling and L. D. S. Coelho, "Coevolutionary particle swarm optimization using Gaussian distribution for solving constrained optimization problems," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36, no. 6, pp. 1407–1416, Dec. 2006.
[48] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: Harmony search theory and practice," *Comput. Methods Appl. Mech. Eng.*, vol. 194, no. 3, pp. 3902–3933, 2005.
[49] E. Mezura-Montes and C. A. C. Coello, "An empirical study about the usefulness of evolution strategies to solve constrained optimization problems," *Int. J. Gen. Syst.*, vol. 37, no. 4, pp. 443–473, 2008.
[50] L. J. Li, Z. B. Huang, F. Liu, and Q. H. Wu, "A heuristic particle swarm optimizer for optimization of pin connected structures," *Comput. Struct.*, vol. 85, pp. 340–349, Apr. 2007.
[51] A. Kaveh and S. Talatahari, "An improved ant colony optimization for constrained engineering design problems," *Eng. Comput.*, vol. 27, pp. 155–182, Jan. 2010.
[52] K. Deb, "Optimal design of a welded beam via genetic algorithms," *AIAA J.*, vol. 29, no. 11, pp. 2013–2015, Nov. 1991.
[53] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Cuckoo search algorithm: A Metaheuristic approach to solve structural optimization problems," *Eng. With Comput.*, vol. 29, no. 1, pp. 17–35, Jan. 2013.
[54] H. Chickermane and H. C. Gea, "Structural optimization using a new local approximation method," *Int. J. Numer. Methods Eng.*, vol. 39, no. 5, pp. 829–846, 1996.
[55] M.-Y. Cheng and D. Prayogo, "Symbiotic Organisms Search: A new metaheuristic optimization algorithm," *Comput. Struct.*, vol. 139, pp. 98–112, Jul. 2014.
[56] H. Bayzidi, S. Talatahari, M. Saraee, and C.-P. Lamarche, "Social network search for solving engineering optimization problems," *Comput. Intell. Neurosci.*, vol. 2021, pp. 1–32, Sep. 2021.
[57] I. Rechenberg, *Evolutionsstrategien, Medizinische Informatik Und Statistik*. Berlin, Germany: Springer, 1978.
[58] A. H. Gandomi and D. A. Roke, "Engineering optimization using interior search algorithm," in *Proc. IEEE Symp. Swarm Intell.*, Dec. 2014, p. 2014.
[59] J. Wu, Y.-G. Wang, K. Burrage, Y.-C. Tian, B. Lawson, and Z. Ding, "An improved firefly algorithm for global continuous optimization problems," *Expert Syst. Appl.*, vol. 149, Jul. 2020, Art. no. 113340.
[60] S. Talatahari, M. Azizi, and A. H. Gandomi, "Material generation algorithm: A novel Metaheuristic algorithm for optimization of engineering problems," *Processes*, vol. 9, no. 5, p. 859, May 2021.
[61] P. Kim and J. Lee, "An integrated method of particle swarm optimization and differential evolution," *J. Mech. Sci. Technol.*, vol. 23, no. 2, pp. 426–434, Feb. 2009.
[62] A. H. Gandomi, X.-S. Yang, and A. H. Alavi, "Mixed variable structural optimization using firefly algorithm," *Comput. Struct.*, vol. 89, pp. 2325–2336, Dec. 2011.