WILEY | Hindawi

Review Article

An Improved Grasshopper Optimizer for Global Tasks

Hanfeng Zhou,[1] **Zewei Ding,**[1] **Hongxin Peng,**[1] **Zitao Tang,**[1] **Guoxi Liang** [iD]**,**[2] **Huiling Chen** [iD]**,**[1] **Chao Ma,**[3] **and Mingjing Wang**[4]

[1]*College of Computer Science and Artificial Intelligence, Wenzhou University, Wenzhou, Zhejiang 325035, China*
[2]*Department of Information Technology, Wenzhou Polytechnic, Wenzhou 325035, China*
[3]*School of Digital Media, Shenzhen Institute of Information Technology, Shenzhen 518172, China*
[4]*Institute of Research and Development, Duy Tan University, Da Nang 550000, Vietnam*

Correspondence should be addressed to Guoxi Liang; guoxiliang2017@gmail.com and Huiling Chen; chenhuiling.jlu@gmail.com

The grasshopper optimization algorithm (GOA) is a metaheuristic algorithm that mathematically models and simulates the behavior of the grasshopper swarm. Based on its flexible, adaptive search system, the innovative algorithm has an excellent potential to resolve optimization problems. This paper introduces an enhanced GOA, which overcomes the deficiencies in convergence speed and precision of the initial GOA. The improved algorithm is named MOLGOA, which combines various optimization strategies. Firstly, a probabilistic mutation mechanism is introduced into the basic GOA, which makes full use of the strong searchability of Cauchy mutation and the diversity of genetic mutation. Then, the effective factors of grasshopper swarm are strengthened by an orthogonal learning mechanism to improve the convergence speed of the algorithm. Moreover, the application of probability in this paper greatly balances the advantages of each strategy and improves the comprehensive ability of the original GOA. Note that several representative benchmark functions are used to evaluate and validate the proposed MOLGOA. Experimental results demonstrate the superiority of MOLGOA over other well-known methods both on the unconstrained problems and constrained engineering design problems.

1. Introduction

The swarm intelligence technique is used to simulate the behavior of different groups that cooperatively search for food in nature. By learning from each other, each member of these groups continually shifts the direction of searching [1–8]. Compared with the traditional optimization methods [9], they show unique advantages in optimizing practical problems [10–17]. Some typical algorithms and as particle swarm optimization (PSO) [18], differential evolution (DE) [19], artificial bee colony optimization (ABC) [20], firefly algorithm (FA) [21], fruit fly optimization algorithm (FOA) [22], Harris hawks optimization (HHO) [23], slime mould algorithm (SMA) [24], and grasshopper optimization algorithm (GOA) [25]. Because of its unique advantages, GOA has outstanding performance in solving practical problems and has been widely concerned in recent years.

Since its introduction, GOA has been applied to tackling many problems. Aljarah et al. [26] introduced GOA to find the variables in the design of the support vector machine (SVM) and to evaluate the feature functions of the ideal subsets. This technique minimizes the amount of chosen characteristics and is superior to other techniques in classification precision. The optimized GOA was introduced to the electrical field by Barman et al. [27]. A regional hybrid short-term load forecast (STLF) model based on GOA with SVM was suggested to assess the relevant parameters in order to enhance precision. Hamad et al. [28] applied enhanced GOA combined with SVM for automatic detection of EEG epilepsy. The results showed that this method has better classification accuracy. Ibrahim et al. [29] combined SVM with improved GOA for feature selection and applied the method to Iraqi cancer patients and biomedical datasets (California Irvine datasets) to achieve high-performance accuracy in real-world datasets. In order to accelerate its

global convergence rate, Arora and Anand. [30] proposed improved GOA by using chaos theory to balance the optimization process of GOA. Saxena et al. [31] used chaotic mapping to enhance GOA and applied it to the parameter estimation of three bar truss layout and FM acoustic synthesis. El-Fergany [][32] took the modeling of the proton exchange membrane fuel cell reactor as an example and the findings highlighted the feasibility and validity of the GOA-based program in steady-state and dynamic simulation. Fathy [33] used the objective function to maximize the set of slaves and matched GOA with the complete cross-connection (TCT), the Su Do Ku connection, and the GA set up to boost the output capacity of the photovoltaic array. Ewees et al. [34] proposed the modified GOA by utilizing a learning strategy, named opposition-based learning strategy (OBLGOA). Heidari et al. [35] used hybrid random training to improve GOA in terms of avoiding falling into the local optimal solution and balancing the trends of exploration and exploitation and applied this enhanced method to the medical field. Liu et al. [36] suggested a detailed approach mixing linear weighted sum (LWS) with GOA to address the energy management issue, improve exhaustive energy efficiency, and ensure regional coordination and optimization. Luo et al. [37] put forward an enhanced GOA system combining three approaches to achieve a more suitable development-exploration equilibrium. Mafarja et al. [38] chose the enhanced binary GOA to subject it to the selection of the ideal subset of features and that of the wrapper-based classification field, which made the algorithm's output superior. Mirjalili et al. [39] introduced a selection strategy into GOA to make it have the ability to deal with multiobjective issues, which made the algorithm archive a good maintenance in terms of high convergence and adaptability. Tharwat et al. [40] integrated an external archive with GOA to preserve the optimal solution; then, this archive was used in multiobjective search space to define the social behavior of GOA and apply it to the field of multiobjective problems. Wu et al. [41] combined comprehensive empirical mode decomposition (EMD) with extreme learning machines and GOA and applied them to STLF in power management. The improvement in the algorithm can find the most appropriate parameters of the extreme learning machine. Wu et al. [42] improved the GOA by developing a natural selection approach, a democratic selection-making mechanism, and a vibrant 1/5-based feedback mechanism. Zakeri and Hokmabadi [43] used statistical measures to complement the iteration process that replaced repetitive features with the most promising ones, and they were applied to the field of feature selection in a final. Zhang et al. [44] suggested a GOA-based parameter-adaptive variational mode decomposition (VMD) technique for analyzing vibration signals of rotating machinery. Zhao et al. [45] merged the technique of decomposition of Beveridge-Nelson (B-N) with the least squares support vector machine (LSSVM) and GOA and applied it to the prediction model of wind power.

This paper introduces two approaches to GOA and proposes an enhanced version called MOLGOA. First, taking into account the benefits and disadvantages of Cauchy mutation and genetic mutation, we incorporated the probabilistic combination mutation system into GOA to enhance the algorithm's worldwide search capability. Then we applied the orthogonal learning mechnism (OL) to GOA, which significantly enhanced the algorithm's convergence velocity. In order to check the effectiveness of the enhanced MOLGOA, the algorithm was evaluated on some representative benchmark functions, including single-mode, multimode, and fixed-dimensional models, and the test results were compared with other competitors including mature metaheuristic algorithms and variations of several sophisticated algorithms. Experiments show that the mechanism proposed in this paper effectively alleviates the problems of local optimization and slow convergence of the original GOA and solves the balance the diversification and intensification.

The remainder of this paper is structured as follows. The original GOA is presented in Section 2. Section 3 shows the details of the enhanced MOLGOA. The fourth section presents and discusses the experimental outcomes. MOLGOA is defined for engineering issues in Section 5. Finally, conclusions and future works are summarized.

2. An Overview of GOA

GOA was proposed by Saremi et al. [25] in 2017, which simulates the population behavior of grasshopper. Grasshoppers are pests whose life cycle consists of eggs, nymphs, and adults that affect crop production and agriculture. GOA simulates grasshopper's behavior in nymph and adult stages. In the nymph stage, grasshopper's primary motion features are jumping and moving in the manner of rolling cylinders (tiny steps, slow motion) and eat crops discovered along with the manner, while in the phase of adulthood, grasshoppers migrate in groups over lengthy distances (quick steps).

GOA simulates grasshopper population behavior according to the following formula:

$$X_i = S_i + G_i + A_i, \tag{1}$$

where X_i is the position of the ith grasshopper and Si means social interaction. G_i denotes the gravity of the i grasshopper and A_i denotes the gravity wind advection of the i grasshopper.

Firstly, the S component in equation (1) is calculated according to the following equation:

$$S_i = a_0 + \sum_{j=1, j \neq i}^{N} s(d_{ij}) \cdot \widehat{d_{ij}}. \tag{2}$$

dij represents the distance between grasshopper i and grasshopper j, which is defined as $d_{ij} = |x_j - x_i|$, and $\widehat{d_{ij}}$ represents the distance unit vector from grasshopper i to grasshopper j.

The population power function $s(r)$ is described as follows:

$$s(r) = f \cdot e^{-(r/1)} - e^r, \tag{3}$$

where f reflects the intensity of the appeal and the duration of the attraction.

Secondly, the G component in equation (1) is obtained as

$$G_i = -g \cdot \widehat{e_g}, \tag{4}$$

where g denotes the gravitational constant, and $\widehat{e_g}$ is a unit vector pointing to the origin.

Thirdly, the component A in equation (3) is determined by the following equation:

$$A_i = u \cdot \widehat{e_w}, \tag{5}$$

where u and $\widehat{e_w}$ represent a unit vector of constant drift and wind direction, respectively. Then combining S, G, and A into one formula, the expression is changed as follows:

$$X_i^d = c \left(\sum_{j=1, j \neq i}^{N} c \cdot \frac{ub_d - lb_d}{2} \cdot s\left(\left| x_j^d - x_i^d \right| \right) \cdot \frac{x_j - x_i}{d_{ij}} \right) + \widehat{T_d}, \tag{6}$$

where N represents the number of grasshoppers. The coefficient c of each iteration is determined according to the following equation:

$$c = c\max - l \cdot \frac{c\max - c\min}{L}, \tag{7}$$

where $c\max$ and $c\min$ represent the upper bound and lower bound of c, respectively, l is the current iteration, and L represents the maximum iteration number.

3. Improved MOLGOA Method

The improved MOLGOA has two effective strategies. Firstly, the orthogonal mechanism is successfully integrated with GOA to accelerate the convergence speed of it. Then, for the advantages of Gaussian mutation and nonlinear programming mutation, these two mutation strategies are adopted in the improved MOLGOA.

3.1. Orthogonal Learning Mechanism.
The orthogonal learning mechanism has been proved to be effective in lifting the optimization capability of the intelligent algorithms [46–53]. The orthogonal table is obtained by listing out the horizontal, vertical, and mixed orthogonal table selected in the orthogonal test design. Based on the in-depth study of mathematical combination theory, a set of test design table with complete rules is constructed by using Latin letters. The symbol of the orthogonal table is $L_M(Q^F)$:

L—The sign of an orthogonal table

M—Number of rows in the orthogonal table (number of tests, number of test schemes)

Q—Number of digits in an orthogonal table (level of factors)

F—Number of columns in orthogonal tables (number of experimental factors)

According to the data structure of the orthogonal table, it can be seen that its distribution is a "crisscross" table, where "vertical" refers to the horizontal distribution of the factors selected in the experiment, and "horizontal" refers to the

number of factors examined in the implementation. The total number of "longitudinal" is the exact amount of experimental schemes of the orthogonal test. As an orthogonal table of row F and column Q, its characteristics are mainly described as in each column; different numbers appear the same number of times and the numbers in any two columns are arranged in a complete and balanced manner. This fully reflects the two advantages of the orthogonal table, namely, "uniform dispersion, neatly comparable". In general, each level of each factor touches each level of the other factors, and that's orthogonal. Based on the orthogonal combination and factor analysis, the highest combination can be readily discovered, which can be proved in the literature [54].

The orthogonal experimental structure is efficient for solving the [55] multi-factor and multi-level experimental issues. By arranging a few tests with the orthogonal table $L_M(Q^F)$, it can discover the finest way for implementation circumstances like optimization. $L_M(Q^F)$ is used for orthogonal table where F and Q represented as factor and levels respectively. Similarly, L and M denoted as Latin square and number of horizontal combinations, respectively. For the experimental problems of F factor and Q level, there are Q^F groups of experiments if the comprehensive combination experiment is carried out. However, it is impossible to do all groups of experiments when Q and F are large. So, the orthogonal table $L_M(Q^F)$ is applied to arrange the orthogonal experiment, which makes only M combinations needed for the experiment. Certainly, M is usually much less than Q^F. We take orthogonal table $L_9(3^3)$ as an example, which is shown in Table 1. For problems of 3 factors and 3 levels, only 9 experiments are needed to conduct orthogonal experiments according to orthogonal table $L_9(3^3)$, but $3^3 = 27$ experiments are needed for comprehensive combined experiments in total. It can be seen that orthogonal experimental design greatly reduces the number of experiments, and the greater the factors and level, the more obvious advantages of this method.

For convenience, mark $L_M(Q^F)$ as $[a_{i,j}]_{M \times F}$ where the ith combination of the level of j is arranged as $a_{i,j} \in \{1, 2, \ldots, Q\}$. The jth column of the orthogonal table $[a_{i,j}]_{M \times F}$ illustrated as a_j, if $j = 1, 2, ((Q^3 - 1)/(Q - 1)) + 1, \ldots, ((Q^{J-1} - 1)/(Q - 1)) + 1$, then a_j is called the basic column; the other columns are called the nonbasic columns. A previous study [55] demonstrated an algorithm for generating the orthogonal table $L_M(Q^F)$ that first generates fundamental columns and then generates nonbasic columns. As described in Table 2, Q is the primary column and $M = Q^J$, J satisfies the formula $F \leq (Q^J - 1)/(Q - 1)$.

Through the orthogonal table, we can find the optimal combination between different levels of various factors with fewer test times. By comparing the characteristics of uniform dispersion and the neatness of the orthogonal table, the algorithm is not only implemented on a smaller scale but also accelerates the convergence rate of itself.

3.2. Genetic Mutation.
Genetic algorithm [56] is an evolutionary algorithm whose fundamental concept is to imitate the evolutionary law of the biological world's "survival of the

TABLE 1: Example of orthogonal table. $L_9(3^3)$.

The serial number	The level of		
	1	2	3
1	1	1	1
2	1	2	2
3	1	3	3
4	2	1	2
5	2	2	3
6	2	3	1
7	3	1	3
8	3	2	1
9	3	3	2

TABLE 2: The pseudocode of construction of the orthogonal table $L_M(Q^F)$.

```
(1) Choose the smallest J fulfilling (Q^J − 1)/(Q − 1) ≤ F;
(2) if (Q^J − 1)/(Q − 1) = F, then F' = F
    else F' = (Q^J − 1)/(Q − 1);
(3) Construct the basic columns as follows:
    for k = 1 to J do
        j = ((Q^(k−1) − 1)/(Q − 1)) + 1;
        for i = 1 to QJ do
            a_i,j = |(i − 1)/(Q^J − 1)|mod Q;
        end for;
    end for;
(4) Construct the following nonbasic columns:
    for k = 2 to J do
        j = ((Q^(k−1) − 1)/(Q − 1)) + 1;
        for s = 1 to j − 1 j − 1 do
            for t = 1 to Q − 1 do
                a_(j+(s−1)(q−1)+t) = (a_s × t + a_j) mod Q;
            end for;
        end for;
    end for;
(5) Increase a_i,j by one for all 1 ≤ i ≤ M and 1 ≤ j ≤ F';
(6) Delete the last F' − F columns of L_QJ (Q^F') to get L_M(Q^F') where M = Q^J
```

fittest in natural selection". Genetic algorithms encode issue parameters as chromosomes and use iterative selection, crossover, mutation, and other procedures to exchange chromosome data in the population in order to produce chromosomes in line with optimization goals.

Nonlinear programming is a new discipline formed in the 1950s that is used under a set of constraints of equality or inequality to study the extreme problem of the n-element real function.

The constraint conditions are

$$
\min f(x) \longrightarrow
\begin{cases}
c(x) \leq 0, \\
c_{eq}(x) = 0, \\
A \cdot x \leq b, \\
A_{eq} \cdot x = b_{eq}, \\
lb \leq x \leq ub,
\end{cases}
\tag{8}
$$

where, x, b, b_{eq}, lb, and ub are vectors. A and A_{eq} are matrices. $c(x)$ and $c_{eq}(x)$ return functions of vectors. $f(x)$, $c(x)$, and $c_{eq}(x)$ are nonlinear functions.

The mutation procedure appears in the process of using a genetic algorithm to fix this issue, whose primary aim is to maintain the diversity of the population. This mutation procedure chooses a person from a population randomly and then chooses a point in the person to mutate to create a better person. The mutation operation of the jth gene in the ith individual a_{ij} is formulated as follows:

$$
a_{ij} =
\begin{cases}
a_{ij} + (a_{ij} - a_{max}) \times f(g), & r \geq 0.5, \\
a_{ij} + (a_{min} - a_{ij}) \times f(g), & r < 0.5,
\end{cases}
\tag{9}
$$

where a_{max} and a_{min} are the maximum and minimum of a_{ij}, respectively:

$$
f(g) = r_2 \left(1 - \frac{g}{G_{max}}\right)^2.
\tag{10}
$$

In equation (10), r_2 is a random number, g represents the current iteration number, G_{max} is the maximum evolution number, and r is a randomly generated value between 0 and 1.

By referring to the GA-based mutation system for solving nonlinear programming problems, new mutant individuals can be acquired according to a certain variety of probability; the potential individual can be acquired

according to a certain variety of probability, which enhances the algorithm's search capability and retains population diversity.

3.3. Cauchy Mutation.

The Cauchy mutation is used to enhance the search capabilities of algorithms in the context of searching and also to avoid the grasshopper population from falling into local optimum. the cauchy mutation is to produce a set of random variables that are consistent with the Cauchy distribution to change the population's position to increase population searchability. The graph of Cauchy density function [57] is quite similar to that of the Gaussian variation density function. The difference between the two is that Cauchy density function slides slowly on both sides of the Y-axis. Moreover, the Cauchy distribution with this feature means that the Cauchy distribution can produce a large disturbance to avoid falling into local optimum and have a stronger global search ability.

The Cauchy mutation is formulated as the following formula:

$$\text{mut}_{\text{SalpPositions}_C} = \text{SalpPositions} \times (1 + \text{Cauchy}(0, 1)). \tag{11}$$

The one-dimensional density function of Cauchy mutation is determined as

$$f(x, \delta, \mu) = \frac{1}{\pi} \frac{\delta}{\delta^2 + (x - \mu)^2}, \quad -\infty < x < \infty. \tag{12}$$

So when $\delta = 1$, $\mu = 0$, the standard form of the one-dimensional Cauchy density function at the center of the origin is obtained as follows:

$$f(x) = \frac{1}{\pi} \frac{1}{x^2 + 1}, \quad -\infty < x < \infty. \tag{13}$$

Correspondingly, the standard Cauchy distribution function is illustrated as follows:

$$\text{Cauchy}(0, 1) = \tan[(\xi - 0.5)\pi], \quad \xi \in U[0, 1]. \tag{14}$$

The advantage of Cauchy mountain is to generate high-probability disturbance to enhance global search capability. The Cauchy mutation is widely used in various optimization algorithms, such as PSO [15], which uses a Cauchy mutation to mutate the optimal particles obtained and achieved the optimal global solution. Another instance is the krill herd optimization algorithm (KHA) [58] that utilizes the Cauchy mutation to achieve better alternatives by optimizing the present situation.

3.4. Combined Mutation Strategy.

In this subsection, the technique of variability in probability choice is implemented to enhance GOA's search precision and worldwide search capability. Because of the benefits of Cauchy mutation in improving worldwide searchability and the features of potential individuals produced by genetic algorithm, Cauchy mutation is first performed, and then the iteration genetic mutation is feasible. The steps are given as follows:

Step 1: Judge the times of the present iteration. If it is not a multiple of 5, create a one-dimensional row vector that is compatible with equation (11) and perform Cauchy mutation on each individual's present place in the population. At the same moment, determine whether the position is superior to the original position after Cauchy mutation. If so, there will be a Cauchy mutation; otherwise, the original position will be retained. If the current number of iterations is a multiple of 5, then jump to Step 2 directly.

Step 2: Generate a number n randomly $(0 < n < 1)$ and continue to Step 3 to the current genetic mutation limit.

Step 3: Equation (9) carries out genetic mutation on each individual's present place in the population. (Note that the mutated position does not need to be compared with the original position to join the following iteration.) The proposed MOLGOA steps are illustrated in Algorithm 1, and the procedure of MOLGOA is presented in the form of a flowchart given in Figure 1.

In Figure 1, l is the current number of iterations, T is the maximum number of iterations, $X\_$new is the population at the random initial position, $X\_$newnut is the population by carrying out mutation operation, $X\_$newcause is the population after Cauchy mutation, and F_{best} is the best fitness.

In this paper, the time complexity of this algorithm mainly depends on the algorithm (E) and the total number of problems (d). Through analysis, it can be concluded that the overall time complexity of this algorithm is O (MOLGOA) = O (initialization of the population position) + O (calculation of the individual comfort) + O (choice of the best comfort level) + $E \times (O$ (update of the individual location) + O (choice of the variation) + O (orthogonal conduction and Choice of the best comfort level) + O (calculation of the individual comfort and choice of the best comfort level).

The time complexity of initializing population location is $O(n \times d)$, the time complexity of the calculation of individual comfort is $O(n)$, the time complexity of the choice of the optimal comfort is $O(n)$, the time complexity of the update of the individual position to $O(n \times n \times d)$, the time complexity of the choice of the variation is $O(n)$, the time complexity of conduction of orthogonal learning and choice of the best comfort level is $O(n \times (d + n \times d \times d))$, and the time complexity of calculating individual comfort and the choice of the best comfort level is $O(n)$.

Therefore, the time complexity of the algorithm proposed in this paper is O (MOLGOA) = O $(n \times d) + O$ $(n) + O$ $(n) + E \times (O(n \times n \times d) + O(n) + O(n \times (d + n \times d \times d)) + O$ $n)) = O(n \times d) + 2O(n) + E \times (O(n \times n \times d) + O(n \times (d + n \times d \times d)) + 2O(n))$.

4. Experimental Results and Discussions

In this section, various experiments on optimization benchmark problems are implemented, and the results are discussed in detail.

4.1. Comparative Results on CEC2017 Problems.

To check the optimizer's efficiency, 30 well-known benchmark functions

(1) Initialize the population GrasshopperPositions$_i$ ($i = 1, 2, \ldots, N$)
(2) Calculate the fitness of initial Grasshoppers
(3) Evaluate the fitness of each grasshopper and copy the best grasshopper's fitness to FoodFitness and copy the best grasshopper's
     position to FoodPosition.
(4)   while ($t <$ maximum number of iterations) do
(5)   Update c1 by the equation (7)
(6)   for (each Grasshopper (GrasshopperPositions$_i$)) do
(7)   Update the position of the Grasshopper by equation (6)
(8)   if (nowiter%5! = 0)
(9)   Generate mut_GrasshopperPositions_C using Cauchy mutation by equation (11)
(10)   Evaluate the fitness of V_i and copy it to Fitness$V$
(11)   if (Fitness$V <$ FoodFitness)
(12)   Update the best position and copy it to GrasshopperPosition
(13)   Randomize n
(14)   else
(15)   Perform nonlinear mutation using equation (9) to generate mut_GrasshopperPositions_G
(16)   Perform OL strategy to generate GrasshopperPosition
(17)   Return FoodPosition and FoodFitness

ALGORITHM 1: The pseudocode of MOLGOA.
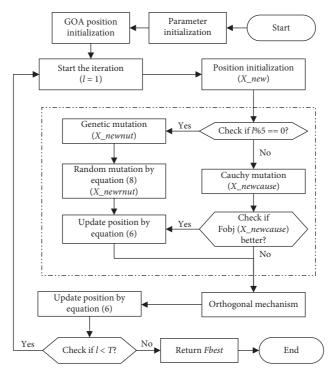
FIGURE 1: The flowchart of MOLGOA.

are chosen. Functions are listed in Table 3, which can be split into three classifications: $F1$–$F3$ are single combined transports, $F4$–$F10$ are multimodal transports, $F11$–$F20$ are blended functions, and $F21$–$F30$ are combined functions. Since they have only one worldwide best and no local optimum (LO), monomodal features can be implemented to benchmark development capacities. Compared with single-mode functions, multimode functions have a large amount of LO and can boost with volume, which can be used to test the algorithm's capacity of exploration and that of jumping out of LO. Furthermore, the multimodal fixed-dimensional

transport functions only have one ideal global solution but a large amount of LOs, as well as multimodal transport functions. However, since the former's solution space is tiny, it is necessary to reduce the search agent's adaptive step size. Therefore, in order to comprehensively assess the algorithm's efficiency and tackle with various kinds of issues, thirty benchmark features are used to evaluate the algorithms.

As mentioned above, the solution space is tiny; therefore, the search agent's adaptive step size is low. Thus, to investigate the algorithm's efficiency and for solving various

TABLE 3: CEC2017 benchmark tests.

Function	Name of the function	Class	Optimum
Shifted and rotated (SRF) functions			
*F*01	SRF Bent cigar function	Unimodal	100
*F*02	SRF sum of different power function	Unimodal	200
*F*03	SRF Zakharov function	Unimodal	300
*F*04	SRF Rosenbrock's function	Multimodal	400
*F*05	SRF Rastrigin's function	Multimodal	500
*F*06	SRF expanded Scaffer's F6 function	Multimodal	600
*F*07	SRF Lunacek Bi-Rastrigin function	Multimodal	700
*F*08	SRF noncontinuous Rastrigin's function	Multimodal	800
*F*09	SRF Lévy function	Multimodal	900
*F*10	SRF Schwefel's function	Multimodal	1000
Hybrid function (HF)			
*F*11	HF 1 ($N = 3$)	Hybrid	1100
*F*12	HF 2 ($N = 3$)	Hybrid	1200
*F*13	HF 3 ($N = 3$)	Hybrid	1300
*F*14	HF 4 ($N = 4$)	Hybrid	1400
*F*15	HF 5 ($N = 4$)	Hybrid	1500
*F*16	HF 6 ($N = 4$)	Hybrid	1600
*F*17	HF 6 ($N = 5$)	Hybrid	1700
*F*18	HF 6 ($N = 5$)	Hybrid	1800
*F*19	HF 6 ($N = 5$)	Hybrid	1900
*F*20	HF 6 ($N = 6$)	Hybrid	2000
Composition function (CF)			
*F*21	CF 1 ($N = 3$)	Composition	2100
*F*22	CF 2 ($N = 3$)	Composition	2200
*F*23	CF 3 ($N = 4$)	Composition	2300
*F*24	CF 4 ($N = 4$)	Composition	2400
*F*25	CF 5 ($N = 5$)	Composition	2500
*F*26	CF 6 ($N = 5$)	Composition	2600
*F*27	CF 7 ($N = 6$)	Composition	2700
*F*28	CF 8 ($N = 6$)	Composition	2800
*F*29	CF 9 ($N = 3$)	Composition	2900
*F*30	CF 10 ($N = 3$)	Composition	3000

types of problems, and evaluate the optimization algorithm's performance comprehensively, thirty functions are adopted for the experimental tests.

For the fairness of the trial, all tests were carried out by 30 search agents for 1,000 iterations under the same conditions. Furthermore, it should be observed that to mitigate the impacts of randomness, each benchmark feature is performed 30 times separately.

4.2. The Impact of Orthogonal Learning and Combined Mutation. As can be seen from the above sections of this paper, two strategies, namely, combinatorial mutation and orthogonal mechanism, have been implemented into the initial GOA. We created three distinct GOA algorithms to study the impact of each mechanism and mixture, as shown in Table 4. "*M*" and "OL" mean "combinatorial mutation" and "orthogonal mechanism," respectively. In Table 4, "1" implies that this operation is integrated into GOA, and "0" implies that there is no corresponding operation. For instance, OLGOA implies that GOA combines the orthogonal mechanism, excluding combinatorial mutations.

The efficiency of various enhanced GOA variants was evaluated based on these functions, as shown in Table 4. Because nonparametric statistical tests were usually adopted in optimization algorithms comparison, the Wilcoxon sign rank test [59] and the Friedman test [60] have been conducted. Table 5 shows the statistical outcomes of different enhanced GOA variants. First, to study the statistical significance distinction between pairing algorithms, the nonparametric Wilcoxon sign rank test with a meaning rate of 5 percent was used. It is shown in Table 5, the *p* values of MOLGOA and other competitors in 30 functions (*F*1–*F*30) are given in rows 2 through 5. The symbols of "+," "−," and "=" show that MOLGOA is higher than, lower than, and equal to other algorithms, respectively. MOLGOA showed statistically significant distinctions in the situation of "+/−/=" compared to these techniques of comparison. MOLGOA is usually better than OLGOA, MGOA, and GOA among the 30 functions, as you can see from the figure, so MOLGOA is the best performing variant. In addition, to further explore the distinctions between different techniques, in order to evaluate the average classification value of all these algorithms, the Friedman test is used in the experiments. Obviously, the combination of the two

TABLE 4: Various GOAs with two strategies.

Function	OL	M
GOA	0	0
MGOA	0	1
OLGOA	1	0
MOLGOL	1	1

TABLE 5: Comparison results of GOA variants.

Function	MOLGOL	OLGOA	MGOA	GOA
$F1$	N/A	$1.3059E - 01$	$1.7344E - 06$	$1.7344E - 06$
$F2$	N/A	$8.4508E - 01$	$1.7344E - 06$	$1.7344E - 06$
$F3$	N/A	$1.7344E - 06$	$1.7344E - 06$	$1.7344E - 06$
$F4$	N/A	$3.3269E - 02$	$3.0650E - 04$	$1.8519E - 02$
$F5$	N/A	$7.3433E - 01$	$1.1499E - 04$	$4.7292E - 06$
$F6$	N/A	$3.1123E - 05$	$1.7344E - 06$	$1.7344E - 06$
$F7$	N/A	$4.9498E - 02$	$9.3157E - 06$	$5.7517E - 06$
$F8$	N/A	$2.4519E - 01$	$1.9646E - 03$	$1.9729E - 05$
$F9$	N/A	$2.3038E - 02$	$2.8434E - 05$	$2.8786E - 06$
$F10$	N/A	$1.0639E - 01$	$6.3391E - 06$	$1.9209E - 06$
$F11$	N/A	$7.7122E - 04$	$9.8421E - 03$	$1.9209E - 06$
$F12$	N/A	$8.9718E - 02$	$2.1266E - 06$	$9.3157E - 06$
$F13$	N/A	$4.2767E - 02$	$2.3038E - 02$	$1.7088E - 03$
$F14$	N/A	$1.6503E - 01$	$2.5967E - 05$	$4.7292E - 06$
$F15$	N/A	$1.4795E - 02$	$1.4704E - 01$	$1.8519E - 02$
$F16$	N/A	$8.9364E - 01$	$4.8603E - 05$	$1.7344E - 06$
$F17$	N/A	$2.9894E - 01$	$1.5658E - 02$	$4.4493E - 05$
$F18$	N/A	$4.4052E - 01$	$5.2165E - 06$	$2.6033E - 06$
$F19$	N/A	$6.7328E - 01$	$2.8786E - 06$	$1.7344E - 06$
$F20$	N/A	$1.9569E - 02$	$3.1618E - 03$	$4.7292E - 06$
$F21$	N/A	$4.9498E - 02$	$2.8434E - 05$	$1.7344E - 06$
$F22$	N/A	$3.0010E - 02$	$9.2710E - 03$	$1.0570E - 04$
$F23$	N/A	$6.5833E - 01$	$1.0570E - 04$	$3.1817E - 06$
$F24$	N/A	$9.4261E - 01$	$2.1266E - 06$	$1.3601E - 05$
$F25$	N/A	$5.4463E - 02$	$6.3391E - 06$	$1.7344E - 06$
$F26$	N/A	$2.9894E - 01$	$4.0702E - 02$	$7.7122E - 04$
$F27$	N/A	$3.2857E - 01$	$1.1499E - 04$	$3.1817E - 06$
$F28$	N/A	$3.1849E - 01$	$3.8822E - 06$	$1.9209E - 06$
$F29$	N/A	$8.3071E - 04$	$1.7344E - 06$	$1.7344E - 06$
$F30$	N/A	$1.1748E - 02$	$5.7517E - 06$	$1.7344E - 06$
+/−/=	N/A	10/3/17	29/0/1	30/0/0
ARV	1.3	1.8	2.97	3.93

approaches has the smallest value according to these sorting outcomes, which verifies that MOLGOA performs better than other combinations in solving these test functions. Lastly, based on the above assessment, MOLGOA is chosen as GOA's best technique of enhancement.

4.3. Comparison with Other Well-Established Methods. In this section, the performance of the MOLGOA will be validated by comparing with the original GOA and some more successful heuristic methods such as gray wolf optimizer (GWO), whale optimization algorithm (WOA) [61], sine cosine algorithm (SCA) [62], differential evolution algorithm (DE), dragonfly algorithm (DA), bat algorithm (BA), moth-flame optimization algorithm (MFO), particle swarm optimization (PSO), Harris hawks optimization (HHO) [23], and GOA [25]. The dimension of each algorithm to be tested is set to 30. Table 6 shows the detailed comparison results.

Table 6 shows that MOLGOA considerably increases the initial GOA's performance. Furthermore, in Table 6, all values indicate that MOLGOA has the lowest deviation compared to other techniques. MOLGOA has better stability compared to other techniques and is able to locate the ideal solution in a narrower spectrum. As shown in Table 7, it can be seen from the "+/−/=" line that MOLGOA accomplished the outstanding outcomes. The findings indicate that MOLGOA has evident benefits and statistical significance in different kinds of functions compared to other rivals. We performed the Friedman test on the median ranking value shown in Table 8 further to investigate the distinctions between MOLGOA and the other rivals. From the outcomes, it can be evidently seen that MOLGOA has reached the

TABLE 6: Comparison results MOLGOA and other well-known methods.

	F1		F2		F3		F4	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**6.8785E + 03**	**6.3610E + 03**	**2.6835E + 08**	1.2785E + 09	**3.0065E + 02**	**4.1742E − 01**	4.9352E + 02	1.4376E + 01
DE	9.0567E + 03	5.2012E+03	1.7456E + 29	4.4348E + 29	1.4660E + 05	2.4566E + 04	4.9965E + 02	**1.1162E + 01**
GWO	2.2702E + 09	1.8219E + 09	1.4316E + 33	4.8426E + 33	5.0649E + 04	1.0914E + 04	6.0239E + 02	9.1661E + 01
DA	5.1284E + 09	2.3880E + 09	7.0190E + 36	3.2232E + 37	1.2365E + 05	3.5724E + 04	1.4329E + 03	6.2275E + 02
WOA	1.6899E + 09	7.8173E + 08	5.3321E + 36	2.9203E + 37	2.3882E + 05	7.9468E + 04	8.1847E + 02	1.1993E + 02
SCA	1.9012E + 10	3.5747E + 09	1.7430E + 36	3.2618E + 36	6.8241E + 04	1.4682E + 04	2.3190E + 03	6.7213E + 02
BA	1.9898E + 07	3.0051E + 06	3.5710E + 08	**9.0056E + 08**	1.2347E + 04	6.4939E + 03	4.9846E + 02	2.7500E + 01
MFO	1.2236E + 10	8.7649E + 09	1.2107E + 39	5.9361E + 39	1.4100E + 05	5.2032E + 04	1.3630E + 03	1.0277E + 03
PSO	1.7553E + 08	2.5798E + 07	5.6217E + 15	6.9198E + 15	1.6620E + 04	8.8850E + 03	**4.8015E + 02**	3.5053E + 01
HHO	2.7626E + 07	6.4787E + 06	2.7118E + 26	1.0102E + 27	3.8113E + 04	6.9825E + 03	5.5474E + 02	2.9373E + 01
GOA	1.7369E + 07	1.6204E + 07	2.5875E + 21	1.3255E + 22	2.6210E + 04	1.1461E + 04	5.2418E + 02	2.4824E + 01
	F5		F6		F7		F8	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**5.8866E + 02**	2.3469E + 01	6.0371E + 02	3.5678E + 00	**8.1072E + 02**	**2.4888E + 01**	**8.8593E + 02**	2.3219E + 01
DE	6.6875E + 02	**1.0068E + 01**	**6.0000E + 02**	1.1010E − 04	9.0265E + 02	8.3966E + 00	9.7254E + 02	**1.3214E + 01**
GWO	6.0924E + 02	3.4569E + 01	6.1183E + 02	5.6765E + 00	8.7902E + 02	4.4260E + 01	9.0420E + 02	2.1550E + 01
DA	8.5382E + 02	6.7076E + 01	6.8714E + 02	1.3539E + 01	1.1318E + 03	8.4007E + 01	1.1384E + 03	6.4021E + 01
WOA	8.2349E + 02	5.8324E + 01	6.8026E + 02	1.1446E + 01	1.2937E + 03	9.3922E + 01	1.0393E + 03	4.1462E + 01
SCA	8.1665E + 02	2.4336E + 01	6.6282E + 02	7.1392E + 00	1.2075E+03	4.5594E + 01	1.0802E + 03	2.6579E + 01
BA	8.2772E + 02	8.0239E + 01	6.7713E + 02	1.0801E + 01	1.7294E + 03	2.1279E + 02	1.0913E + 03	6.3244E + 01
MFO	7.1626E + 02	4.5073E + 01	6.4020E + 02	1.0940E + 01	1.1629E + 03	2.2327E + 02	1.0153E + 03	5.2213E + 01
PSO	7.6886E + 02	3.2325E + 01	6.6555E + 02	9.2340E + 00	9.6075E + 02	2.3826E + 01	1.0206E + 03	2.2099E+01
HHO	7.5232E + 02	4.4666E + 01	6.6559E + 02	6.1016E + 00	1.2589E + 03	8.4221E + 01	9.7559E + 02	2.7083E + 01
GOA	6.6605E + 02	3.8163E + 01	6.5273E + 02	1.7572E + 01	9.3110E + 02	5.4827E + 01	9.5217E + 02	3.4711E + 01
	F9		F10		F11		F12	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	1.3708E + 03	9.7237E + 02	**3.7815E + 03**	5.5991E + 02	**1.2388E + 03**	**4.7459E + 01**	**3.9214E + 06**	**3.8428E + 06**
DE	**1.0456E + 03**	**9.8940E + 01**	7.2296E + 03	2.7918E + 02	1.4003E + 03	1.3995E + 02	1.4227E + 07	8.5370E + 06
GWO	2.4267E + 03	7.8503E + 02	4.8711E + 03	1.4054E + 03	2.2069E+03	9.0045E + 02	9.9085E + 07	1.0767E + 08
DA	1.3037E + 04	4.4699E + 03	7.8752E + 03	6.2199E + 02	3.6855E + 03	1.8238E + 03	6.0624E + 08	4.2386E + 08
WOA	9.7036E + 03	3.2164E + 03	7.0657E + 03	7.7430E + 02	5.7085E + 03	2.4789E + 03	2.6189E + 08	2.1268E + 08
SCA	7.5039E + 03	1.7203E + 03	8.7405E + 03	**4.1764E + 02**	3.5013E + 03	1.0544E + 03	2.2612E + 09	7.9687E + 08
BA	1.6608E + 04	5.1974E + 03	5.9617E + 03	6.6635E + 02	1.3602E + 03	9.6779E + 01	1.3973E + 07	8.3005E + 06
MFO	7.4708E + 03	2.5417E + 03	5.3896E + 03	6.1893E + 02	6.1682E + 03	5.4470E + 03	3.5229E + 08	6.5045E + 08
PSO	6.4653E + 03	2.4988E + 03	6.7756E + 03	5.5226E + 02	1.3255E + 03	5.5894E + 01	4.0794E + 07	1.5687E + 07
HHO	7.7947E + 03	9.4905E + 02	5.5621E + 03	7.8170E + 02	1.2892E + 03	5.3377E + 01	2.7049E + 07	1.8006E + 07
GOA	5.7352E + 03	1.9417E + 03	5.5113E + 03	5.4704E + 02	1.4320E + 03	8.0622E + 01	2.8476E + 07	2.7133E + 07
	F13		F14		F15		F16	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**8.8012E + 04**	**4.8236E + 04**	**4.7170E + 03**	**2.4535E + 03**	6.7941E + 04	3.6697E + 04	**2.2121E + 03**	2.6483E + 02
DE	1.0782E + 06	8.3281E + 05	2.2348E + 05	1.3873E + 05	2.1715E + 05	**1.1654E + 05**	2.7547E + 03	**1.5014E + 02**
GWO	1.8566E + 07	6.8531E + 07	2.6218E + 05	3.1280E + 05	2.0874E + 06	6.9914E + 06	2.5079E + 03	2.5790E + 02
DA	6.7619E + 07	9.8730E + 07	1.5905E + 06	1.9490E + 06	6.1567E + 05	1.0168E + 06	4.3643E + 03	5.1779E + 02
WOA	2.4102E + 06	2.4197E + 06	1.9252E + 06	2.4760E + 06	1.2140E + 06	1.8709E + 06	4.2105E + 03	5.4265E + 02
SCA	8.7284E + 08	4.5145E + 08	5.1732E + 05	4.1373E + 05	4.3864E + 07	3.4322E + 07	3.9220E + 03	2.7916E + 02
BA	1.0037E + 06	2.5816E + 05	4.4126E + 04	3.3310E + 04	2.1101E + 05	9.8990E + 04	3.5400E + 03	4.9182E + 02
MFO	8.1962E+06	2.1609E + 07	5.5092E + 05	6.7908E + 05	**6.5669E + 04**	4.7395E + 04	3.2158E + 03	3.7499E + 02
PSO	7.6704E + 06	2.3685E + 06	3.6865E + 04	3.8892E + 04	1.0945E + 06	4.8844E + 05	3.0467E + 03	3.6433E + 02
HHO	6.9064E + 05	4.0041E + 05	4.5370E + 05	6.2852E + 05	7.5166E + 04	4.4876E + 04	3.4095E + 03	3.7443E + 02
GOA	1.8159E + 05	1.2639E + 05	4.3035E + 04	3.4894E + 04	9.6997E + 04	8.8586E + 04	2.8657E + 03	3.6207E + 02

TABLE 6: Continued.

	F17		F18		F19		F20	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**1.9198E + 03**	1.0433E + 02	**8.6754E + 04**	**3.8427E + 04**	**4.5180E + 04**	**3.7196E + 04**	**2.3339E + 03**	1.2299E + 02
DE	2.0303E + 03	**9.4862E + 01**	1.7685E + 06	9.1768E + 05	1.4777E + 05	1.5762E + 05	2.3994E + 03	**1.0963E + 02**
GWO	2.1066E + 03	1.4615E + 02	1.3570E + 06	1.5751E + 06	5.4538E + 06	2.5508E + 07	2.4336E + 03	1.4427E + 02
DA	2.8292E + 03	2.9644E + 02	1.0748E + 07	1.8987E + 07	3.0307E + 07	2.3372E + 07	2.9254E + 03	2.1330E + 02
WOA	2.7372E + 03	2.5466E + 02	1.1409E + 07	1.2360E + 07	1.3673E + 07	8.5519E + 06	2.8431E + 03	2.1237E + 02
SCA	2.7143E + 03	2.1781E + 02	1.2650E + 07	8.5650E + 06	6.9007E + 07	5.1141E + 07	2.8492E + 03	1.7887E + 02
BA	2.8920E + 03	3.8498E + 02	3.7591E + 05	2.1487E + 05	1.7780E + 06	1.0212E + 06	3.0447E + 03	2.2712E + 02
MFO	2.5765E + 03	2.2697E + 02	3.8026E + 06	5.2500E + 06	3.1321E + 06	1.3454E + 07	2.7098E + 03	2.7041E + 02
PSO	2.4123E + 03	1.8891E + 02	6.5743E + 05	5.7542E + 05	3.0303E + 06	1.4035E + 06	2.7058E + 03	1.3995E + 02
HHO	2.5923E + 03	3.2030E+02	1.3397E + 06	1.3335E + 06	9.3660E + 05	7.6851E + 05	2.7567E + 03	1.9598E + 02
GOA	2.3307E + 03	2.7500E + 02	8.8030E + 05	6.6240E + 05	4.5654E + 06	2.7410E + 06	2.7325E + 03	1.8687E + 02

	F21		F22		F23		F24	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**2.3740E + 03**	2.6533E + 01	**3.5600E + 03**	1.5117E + 03	**2.7397E + 03**	2.8675E + 01	**2.9062E + 03**	3.1565E + 01
DE	2.4713E + 03	**1.1571E + 01**	5.1459E + 03	1.6437E + 03	2.8183E + 03	**1.1471E + 01**	3.0177E + 03	1.4101E + 01
GWO	2.4014E + 03	2.6247E + 01	5.0238E + 03	1.8950E + 03	2.7817E + 03	4.6975E + 01	2.9381E + 03	5.4834E + 01
DA	2.6725E + 03	7.1727E + 01	8.1600E + 03	2.1230E + 03	3.2995E + 03	1.6421E + 02	3.4738E + 03	1.1949E+02
WOA	2.6016E + 03	5.9648E + 01	8.1303E + 03	1.4118E + 03	3.1516E + 03	9.5556E + 01	3.2084E+03	8.0744E + 01
SCA	2.5944E + 03	2.3551E + 01	9.7025E + 03	1.3430E + 03	3.0704E + 03	4.8957E + 01	3.2058E+03	3.6092E + 01
BA	2.6357E + 03	8.1644E + 01	7.5551E + 03	**1.2178E + 03**	3.3272E + 03	1.2821E + 02	3.3649E + 03	1.6896E + 02
MFO	2.4913E + 03	3.9669E + 01	6.4657E + 03	1.5119E + 03	2.8259E + 03	3.1308E + 01	2.9837E + 03	3.2450E + 01
PSO	2.5541E + 03	3.6961E + 01	4.9742E + 03	3.0512E + 03	3.1852E + 03	1.3434E + 02	3.2347E + 03	9.1128E + 01
HHO	2.5840E + 03	5.6751E + 01	6.3843E + 03	2.0342E + 03	3.2842E + 03	1.5766E + 02	3.4834E + 03	1.5396E + 02
GOA	2.4583E + 03	3.2771E + 01	6.1809E + 03	1.7168E + 03	2.8193E + 03	5.7805E + 01	2.9787E + 03	4.4575E + 01

	F25		F26		F27		F28	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**2.8871E + 03**	2.5792E + 00	**4.1858E + 03**	7.1754E + 02	**3.2167E + 03**	1.1662E + 01	**3.2092E + 03**	3.8257E + 01
DE	2.8881E + 03	**1.1670E + 00**	5.2726E + 03	**1.2789E + 02**	3.2196E + 03	**3.7225E + 00**	3.2663E + 03	2.7068E + 01
GWO	3.0056E + 03	7.7183E + 01	5.0001E + 03	4.4650E + 02	3.2595E + 03	3.5859E + 01	3.4408E + 03	9.6281E + 01
DA	3.3304E + 03	1.9212E + 02	8.7636E + 03	1.4840E + 03	3.5445E + 03	1.9141E + 02	4.0431E + 03	4.5301E + 02
WOA	3.1015E + 03	5.0027E + 01	8.4656E + 03	1.0357E + 03	3.4391E + 03	1.5744E + 02	3.5536E + 03	1.1202E + 02
SCA	3.4314E + 03	1.9051E + 02	7.4593E + 03	3.1113E + 02	3.4907E + 03	5.3065E + 01	4.2627E + 03	3.2075E + 02
BA	2.9080E + 03	2.3028E + 01	9.0881E + 03	1.6519E + 03	3.4473E + 03	1.6793E + 02	3.2311E + 03	**1.9964E + 01**
MFO	3.2870E + 03	4.4945E + 02	5.7998E + 03	5.6183E + 02	3.2532E + 03	2.9025E + 01	4.4574E + 03	1.0534E + 03
PSO	2.9255E + 03	2.3735E + 01	6.2024E + 03	2.3170E + 03	3.2207E + 03	1.0893E + 02	3.2644E + 03	2.2966E + 01
HHO	2.9459E + 03	2.5387E + 01	7.3756E + 03	1.8787E + 03	3.5099E + 03	1.6868E + 02	3.3282E + 03	3.5523E + 01
GOA	2.9359E + 03	2.4641E + 01	4.9204E + 03	1.0633E + 03	3.2884E + 03	6.6286E + 01	3.2906E + 03	3.9440E + 01

	F29		F30		—	—	—	—
	Mean	Std	Mean	Std	—	—	—	—
MOLGOA	**3.5505E + 03**	1.2973E + 02	4.3114E + 05	2.1451E + 05	—	—	—	—
DE	3.9493E + 03	1.3912E + 02	**1.4915E + 05**	**1.1293E + 05**	—	—	—	—
GWO	3.9150E + 03	1.9545E + 02	1.0448E + 07	7.9080E + 06	—	—	—	—
DA	5.4063E + 03	8.1628E + 02	6.6973E + 07	6.1309E + 07	—	—	—	—
WOA	5.1786E + 03	6.0381E + 02	4.1071E + 07	3.9088E + 07	—	—	—	—
SCA	5.1118E + 03	2.2918E + 02	1.3931E + 08	6.8121E + 07	—	—	—	—
BA	5.0931E + 03	6.0314E + 02	3.4564E + 06	2.0434E + 06	—	—	—	—
MFO	4.2726E + 03	2.5783E + 02	5.3831E + 05	7.1366E + 05	—	—	—	—
PSO	4.5573E + 03	2.9536E + 02	6.7913E + 06	3.7255E + 06	—	—	—	—
HHO	4.7033E + 03	4.7771E + 02	5.2143E + 06	4.0168E + 06	—	—	—	—
GOA	4.2931E + 03	2.9965E + 02	9.1051E + 06	6.9892E + 06	—	—	—	—

highest quality of the solutions with the average ranking value of 1.16667.

Figure 2 demonstrates the convergence characteristics of MOLGOA and other counterparts on various 30-dimensional benchmark issues. As shown, the MOLGOA has the greatest convergence precision and efficiency when it comes to $F3$, $F14$, $F18$, $F22$, and $F26$ issues, while other algorithms fall prematurely into local optimization. However, they all have comparatively decent convergence precision in $F1$, $F5$, and $F10$. In sum, MOLGOA can achieve better convergence curve and shows a quicker convergence rate and can find the best solution to the issue compared to other rivals. From the above assessment, it can be seen that MOLGOA displays great global search capability and high search precision for low-dimensional tasks as well as high-dimensional tasks, so we can

TABLE 7: Statistical comparison results of different dimensions by Wilcoxon test.

	DE	GWO	DA	WOA	SCA	BA	MFO	PSO	HHO	GOA
F1	$1.9861E-01$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F2	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$3.5009E-02$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F3	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F4	$4.9498E-02$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.0639E-01$	$2.3534E-06$	$7.5213E-02$	$1.7344E-06$	$3.7243E-05$
F5	$1.7344E-06$	$1.1748E-02$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.9209E-06$
F6	$\mathbf{1.7344E-06}$	$9.3157E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F7	$1.7344E-06$	$3.5152E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F8	$1.7344E-06$	$1.1079E-02$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$2.6033E-06$
F9	$5.9994E-01$	$8.9187E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F10	$1.7344E-06$	$1.6394E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$2.3534E-06$	$1.7344E-06$	$2.1266E-06$	$1.7344E-06$
F11	$2.1266E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$2.3704E-05$	$1.7344E-06$	$6.3391E-06$	$6.0350E-03$	$1.7344E-06$
F12	$1.7988E-05$	$2.3534E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$3.4053E-05$	$4.2857E-06$	$1.7344E-06$	$3.1817E-06$	$6.3391E-06$
F13	$1.7344E-06$	$1.1748E-02$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$5.2872E-04$	$1.7344E-06$	$1.7344E-06$	$7.2710E-03$
F14	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$2.3534E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$2.1266E-06$
F15	$7.6909E-06$	$1.1748E-02$	$7.6909E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$9.5899E-01$	$1.7344E-06$	$6.7328E-01$	$2.1336E-01$
F16	$1.7344E-06$	$1.7423E-04$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$2.1266E-06$	$1.9209E-06$	$5.2165E-06$
F17	$7.7122E-04$	$1.6394E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$2.3534E-06$
F18	$1.7344E-06$	$3.1817E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$4.2857E-06$	$2.1266E-06$	$2.3534E-06$	$2.3534E-06$	$2.3534E-06$
F19	$4.5336E-04$	$6.3391E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$6.1564E-04$	$1.7344E-06$	$2.3534E-06$	$1.7344E-06$
F20	$3.6826E-02$	$1.1748E-02$	$1.7344E-06$	$2.6033E-06$	$1.7344E-06$	$1.7344E-06$	$6.3391E-06$	$2.1266E-06$	$2.1266E-06$	$1.9209E-06$
F21	$1.7344E-06$	$7.7122E-04$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.9209E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F22	$1.1973E-03$	$1.9646E-03$	$4.7292E-06$	$1.7344E-06$	$2.3534E-06$	$1.7344E-06$	$1.0246E-05$	$2.0671E-02$	$8.9187E-05$	$1.2381E-05$
F23	$1.9209E-06$	$7.1570E-04$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$2.3534E-06$	$1.7344E-06$	$1.7344E-06$	$6.9838E-06$
F24	$1.7344E-06$	$2.4308E-02$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.0246E-05$	$2.3534E-06$	$1.7344E-06$	$1.7344E-06$	$3.1817E-06$
F25	$3.1123E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F26	$1.9209E-06$	$2.5967E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.0246E-05$	$1.9209E-06$	$2.0515E-04$	$3.5152E-06$	$1.0444E-02$
F27	$2.1336E-01$	$2.3534E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$8.2206E-02$	$1.7344E-06$	$4.7292E-06$
F28	$7.6909E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$6.4242E-03$	$1.7344E-06$	$1.3601E-05$	$1.7344E-06$	$2.6033E-06$
F29	$1.7344E-06$	$2.8786E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F30	$\mathbf{3.5152E-06}$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$9.4261E-01$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
+/−/=	25/2/3	30/0/0	30/0/0	30/0/0	30/0/0	29/0/1	28/0/2	28/0/2	29/0/1	29/0/1

TABLE 8: Average ranking values of the Friedman test.

Algorithm	AV-ranking	Ranking
MOLGOA	1.16667	1
DE	3.86667	2
GWO	4.8	4
DA	9.8	11
WOA	8.73333	9
SCA	9.06667	10
BA	6.56667	8
MFO	6.53333	7
PSO	5.03333	5
HHO	6.13333	6
GOA	4.3	3

conclude that MOLGOA is superior to those of algorithms in this domain.

4.4. Comparison with the State-of-The-Art Algorithms. To further evaluate the performance of MOLGOA, this impressive method was compared with other sophisticated algorithms in this chapter, including improved GOA (IGOA) [37], CSSA [63], ESSA [64], CGSCA [65], m_SCA [66], LWOA [67], BWOA [4], CCMWOA [5], CLPSO [68], and GL25 [69]. The detailed illustration of the comparison functions and the parameters involved in the experiment are listed in Table 3.

In this experiment, Table 9 gives the mean and standard deviation values achieved by these algorithms, while the Wilcoxon test demonstrates the final outcomes of the comparison of distinct algorithms. Table 10 demonstrates the outcomes of the Friedman test, which can demonstrate the benefits of each algorithm more obviously.

According to the information shown in Table 9, it can be found that MOLGOA is superior to other sophisticated algorithms in most cases. As shown in Table 11, we can find that MOLGOA has strong competitiveness among the selected functions in the Wilcoxon test, and the last row of Table 10 shows that MOLGOA performs better in most problems than other advanced methods. Table 11 lists the final ranking outcomes acquired through the Friedman test by distinct algorithms. The convergence rate of MOLGOA is presented in Figure 3; it can be seen that the convergence speed of the proposed algorithm is the fastest among all the methods.

According to the convergence curves in Figure 3, it can be seen that MOLGOA has the greatest convergence precision in all these functions. In $F1$, although MOLGOA's convergence rate is slow at an early point, in the later phase, MOLGOA can jump from the location optimum and have high convergence precision. As shown, it is evident that MOLGOA has better convergence precision and quicker convergence rate on $F2$, $F6$, $F14$, $F18$, $F22$, and $F26$ functions.

MOLGOA has improved efficiency in the single modal functions, multimodal functions, and fixed-dimensional multimodal functions compared to other advanced methods. This is primarily because MOLGOA adopts the orthogonal mechanism and the joint mutation mechanism, which significantly enhances the algorithm's global

search capability, while the orthogonal mechanism offers efficient variables with the improvement feature. Overall, the benchmark issues can be efficiently solved by MOLGOA.

5. MOLGOA for the Engineering Benchmarks

MOLGOA is implemented in this chapter to deal with two engineering benchmark issues, namely, pressure vessel (PV) design problem and tension-compression string (TCS) design problem. The flowchart of MOLGOA for constraint optimization is shown in Figure 4.

5.1. PV Problem. This mathematical modeling is aimed at minimizing the total cost of the cylindrical pressure vessel, which is closely related to material, forming, and welding. In this optimization task, the thickness of the shell, the thickness of the head, the inner radius, and the length of the cylindrical section, without considering the head, are the variables to be optimized. The formulation can be described as bellow:

Consider $\overrightarrow{x} = [x_1 x_2 x_3 x_4]$,

minimize $f(\overrightarrow{x})_{\min} = 0.624x_1x_3x_4 + 1.7781x_3x_1^2$

$$+3.1661x_4x_1^2 + 19.84x_3x_1^2$$

$$g_1(\overrightarrow{x}) = -x_1 + 0.0193x_3 \leq 0$$

$$g_2(\overrightarrow{x}) = -x_3 + 0.00954x_3 \leq 0$$

subject to

$$g_3(\overrightarrow{x}) = -\pi x_4 x_3^2 - \frac{4}{3}\pi x_3^3 + 1296000 \leq 0$$

$$g_4(\overrightarrow{x}) = x_4 - 240 \leq 0,$$

(15)

where $0 \leq x_1 \leq 99, 0 \leq x_2 \leq 99, 10 \leq x_1 \leq 200, 10 \leq x_1 \leq 200$.

MOLGOA solves this issue and is compared to initial algorithms such as GOA, IGOA, and others. Table 12 shows
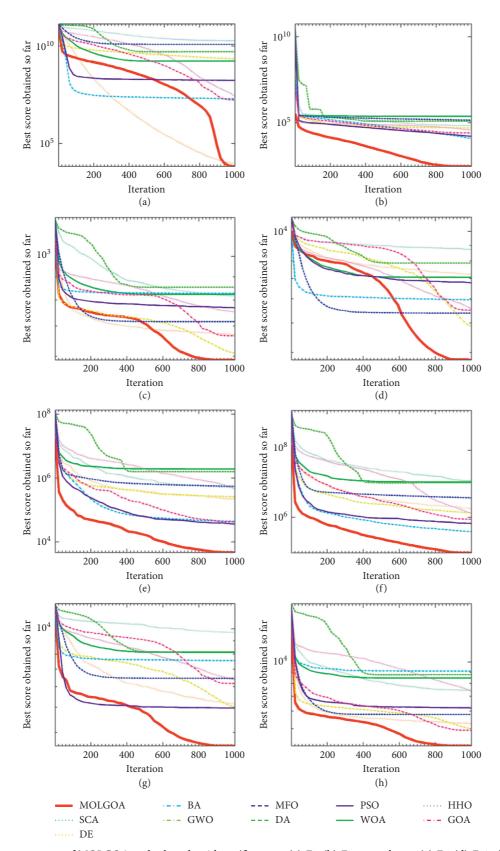
FIGURE 2: Convergence curve of MOLGOA and other algorithms (first row: (a) $F1$, (b) $F3$; second row: (c) $F5$, (d) $F10$; third row: (e) $F14$, (f) $F18$; fourth row: (g) $F22$, (h) $F26$).

TABLE 9: Comparison results of MOLGOA and other advanced algorithms.

	F1		F2		F3		F4	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**6.7328E + 03**	**6.1596E + 03**	**1.8467E + 08**	**7.9891E + 08**	**3.0055E + 02**	**3.4377E − 01**	**4.9615E + 02**	**1.5654E + 01**
IGOA	2.0899E + 06	1.4556E + 06	2.4526E + 10	6.1096E + 10	3.3077E + 02	1.5069E + 01	4.9878E + 02	2.5031E + 01
CSSA	7.2644E + 10	6.5063E + 09	2.7956E + 53	6.3686E + 53	9.2496E + 04	3.8314E + 03	2.0304E + 04	3.1193E + 03
ESSA	4.1420E + 10	6.6746E + 09	7.8257E + 49	4.2786E + 50	7.9405E + 04	8.4696E + 03	1.1115E + 04	4.0274E + 03
CGSCA	2.1714E + 10	3.7609E + 09	1.6842E + 38	7.6687E + 38	6.4712E + 04	9.2693E + 03	2.9373E + 03	8.4434E + 02
m_SCA	8.5794E + 09	2.5749E + 09	4.0787E + 33	1.2703E + 34	4.4609E + 04	1.1223E + 04	9.3861E + 02	3.8885E + 02
LWOA	2.8101E + 06	6.5480E + 05	5.9138E + 12	1.1120E + 13	8.0022E + 04	3.9169E + 04	5.2070E + 02	2.9682E + 01
BWOA	1.7211E + 09	7.0510E + 08	2.8457E + 33	9.4060E + 33	7.0680E + 04	1.0914E + 04	8.8866E + 02	1.3894E + 02
CCMWOA	1.0203E + 10	4.0708E + 09	8.8346E + 33	4.5584E + 34	6.9744E + 04	6.0701E + 03	1.4306E + 03	5.7468E + 02
CLPSO	1.5935E + 09	3.7433E + 08	2.3684E + 35	8.8553E + 35	1.2316E + 05	2.5476E + 04	9.0820E + 02	8.2676E + 01
GL25	2.8802E + 09	1.2037E + 09	2.4277E + 34	8.2761E + 34	1.5943E + 05	4.2158E + 04	1.0493E + 03	2.8172E + 02

	F5		F6		F7		F8	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**5.9111E + 02**	2.8280E + 01	**6.0371E + 02**	5.5283E + 00	**8.2088E + +02**	2.9479E + 01	**8.9038E + 02**	2.6264E + 01
IGOA	6.3358E + 02	3.8001E + 01	6.3528E + 02	1.9273E + 01	8.6357E + 02	3.9628E + 01	9.5211E + 02	4.5873E + 01
CSSA	9.8212E + 02	2.7371E + 01	7.0417E + 02	5.6504E + 00	1.5229E + 03	4.8062E + 01	1.1864E + 03	2.7369E + 01
ESSA	9.1763E + 02	2.9505E + 01	6.8814E + 02	6.0369E + 00	1.3829E + 03	7.1599E + 01	1.1245E + 03	2.7082E + 01
CGSCA	8.3468E + 02	2.5538E + 01	6.6577E + 02	6.8581E + 00	1.2319E + 03	4.9971E + 01	1.0945E + 03	2.3190E + 01
m_SCA	6.9373E + 02	3.1822E + 01	6.3431E + 02	8.8961E + 00	1.0609E + 03	6.2420E + 01	9.6575E + 02	3.1185E + 01
LWOA	7.8341E + 02	6.8366E + 01	6.6370E + 02	1.0326E + 01	1.1564E + 03	7.9927E + 01	1.0172E + 03	5.4661E + 01
BWOA	7.9954E + 02	2.3356E + 01	6.6802E + 02	7.2014E + 00	1.3014E + 03	5.5530E + 01	1.0027E + 03	2.3076E + 01
CCMWOA	7.9587E + 02	2.9608E + 01	6.6539E + 02	6.8358E + 00	1.2407E + 03	6.1179E + 01	1.0113E + 03	2.6642E + 01
CLPSO	7.4057E + 02	**1.5733E + 01**	6.2693E + 02	**4.1236E + 00**	1.0333E + 03	**2.6872E + 01**	1.0371E + 03	**1.6455E + 01**
GL25	7.8064E + 02	4.6588E + 01	6.4251E + 02	7.6518E + 00	1.1361E + 03	5.7137E + 01	1.0724E + 03	4.9084E + 01

	F9		F10		F11		F12	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**1.2142E + 03**	**4.8245E + 02**	**3.7324E + 03**	5.2366E + 02	**1.2530E + 03**	**4.8004E + 01**	**4.0771E + 06**	**3.1753E + 06**
IGOA	6.6657E + 03	2.7799E + 03	4.4992E + 03	7.1668E + 02	1.3384E + 03	6.7085E + 01	5.6518E + 06	4.3395E + 06
CSSA	1.3165E + 04	1.5762E + 03	9.5465E + 03	3.9152E + 02	1.2095E+04	3.1256E + 03	1.9013E + 10	4.1468E + 09
ESSA	1.0127E + 04	1.4704E + 03	8.5785E + 03	5.0397E + 02	7.7090E + 03	2.5164E + 03	6.7025E + 09	3.5830E + 09
CGSCA	8.6421E + 03	1.4404E + 03	8.6174E + 03	3.5726E + 02	3.5352E + 03	9.7992E + 02	2.5987E + 09	7.7470E + 08
m_SCA	4.8636E + 03	1.4924E + 03	5.8179E + 03	8.2617E + 02	2.1383E + 03	7.6068E + 02	3.2589E + 08	2.5248E + 08
LWOA	7.5846E + 03	2.4348E + 03	5.4110E + 03	7.2713E + 02	1.2834E + 03	5.5900E + 01	1.0015E + 07	7.4156E + 06
BWOA	7.3536E + 03	1.0410E + 03	6.6473E + 03	7.9762E + 02	2.5144E + 03	5.2729E + 02	2.5829E + 08	2.4089E + 08
CCMWOA	6.4485E + 03	7.7270E + 02	6.2442E + 03	4.7639E + 02	2.1921E+03	4.2309E + 02	3.6229E + 08	3.0243E + 08
CLPSO	6.6316E + 03	1.2673E + 03	7.4105E + 03	**3.0723E + 02**	2.8799E + 03	5.2196E + 02	2.0663E + 08	8.9674E + 07
GL25	6.0738E + 03	1.9717E + 03	8.9794E + 03	1.0983E + 03	5.5828E + 03	2.6634E + 03	1.8150E + 08	1.2236E + 08

	F13		F14		F15		F16	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**1.3263E + 05**	**7.2781E + 04**	**4.7128E + 03**	**2.7990E + 03**	6.8102E + 04	4.3610E + 04	**2.2152E + 03**	**1.7050E + 02**
IGOA	2.2844E + 05	1.6074E + 05	2.9959E + 04	3.0492E + 04	9.8914E + 04	7.2951E + 04	2.6002E + 03	3.0357E + 02
CSSA	2.1963E+10	6.3682E + 09	4.9661E + 07	5.8530E + 07	2.1966E+09	1.0891E + 09	8.3033E + 03	1.7467E + 03
ESSA	2.2251E + 09	2.2861E + 09	4.8027E + 06	5.5255E + 06	9.2135E + 07	9.8802E + 07	4.9481E + 03	8.8456E + 02
CGSCA	1.0349E + 09	3.7756E + 08	6.2189E + 05	3.2399E + 05	1.9454E+07	2.2072E+07	4.1619E + 03	2.6271E + 02
m_SCA	1.0863E + 08	1.3243E + 08	2.1101E + 05	2.6696E + 05	3.5230E + 06	9.3627E + 06	2.7422E + 03	2.5587E + 02
LWOA	2.1140E + 05	1.0651E + 05	7.6422E + 04	5.6086E + 04	8.9787E + 04	4.9156E + 04	3.2454E + 03	4.9019E + 02
BWOA	2.2466E + 06	2.5013E + 06	1.7074E + 06	1.9855E + 06	1.4516E + 06	1.7890E + 06	4.1838E + 03	5.5413E + 02
CCMWOA	2.4857E + 06	6.5263E + 06	5.3562E + 05	4.6913E + 05	2.0405E + 05	2.1359E + 05	3.6800E + 03	4.3685E + 02
CLPSO	9.2756E + 07	5.6692E + 07	2.6959E + 05	1.9619E + 05	6.8876E + 06	7.2401E + 06	3.3257E + 03	2.8019E + 02
GL25	5.4209E + 06	7.1349E + 06	9.4812E + 05	7.2618E + 05	**3.2410E + 04**	**3.2244E + 04**	4.0311E + 03	4.0588E + 02

Table 9: Continued.

	F17		F18		F19		F20	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**2.0058E + 03**	1.4359E + 02	**8.1696E + 04**	**3.4710E + 04**	**4.3063E + 04**	**2.1971E + 04**	**2.3443E + 03**	1.1342E + 02
IGOA	2.2905E + 03	1.9635E + 02	3.9817E + 05	2.6756E + 05	1.1624E + 05	8.3622E + 04	2.5808E + 03	1.7355E + 02
CSSA	2.1638E + 04	2.0522E + 04	4.3601E + 08	5.1185E + 08	1.9497E + 09	9.9345E + 08	3.3868E + 03	1.8637E + 02
ESSA	3.2355E + 03	3.9901E + 02	6.3925E + 07	6.0758E + 07	1.6916E + 08	2.2194E + 08	2.9636E + 03	2.7356E + 02
CGSCA	2.7794E + 03	1.8960E + 02	8.4686E + 06	6.4608E + 06	7.0038E + 07	4.2043E+07	2.8672E + 03	1.3562E + 02
m_SCA	2.2217E + 03	1.9829E + 02	1.7406E + 06	1.6621E + 06	4.5372E + 06	7.6197E + 06	2.5396E + 03	1.6072E + 02
LWOA	2.4291E + 03	2.6588E + 02	1.2197E + 06	1.3510E + 06	6.4081E + 05	5.3403E + 05	2.8633E + 03	1.7853E + 02
BWOA	2.7223E + 03	3.1328E + 02	7.2965E + 06	5.5916E + 06	6.1547E + 06	5.2712E + 06	2.7357E + 03	2.1146E + 02
CCMWOA	2.5230E + 03	2.2598E + 02	2.9667E + 06	3.0034E + 06	1.2589E + 06	1.2380E + 06	2.5837E + 03	1.4830E + 02
CLPSO	2.3563E + 03	**1.3333E + 02**	2.1650E + 06	1.1089E + 06	5.7492E + 06	5.2759E + 06	2.5676E + 03	**1.0899E + 02**
GL25	2.5517E + 03	2.7872E + 02	7.7654E + 06	6.8176E + 06	1.5433E + 05	2.8315E + 05	3.0314E + 03	2.5054E + 02

	F21		F22		F23		F24	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**2.3880E + 03**	3.0059E + 01	**3.5290E + 03**	1.4856E + 03	**2.7408E + 03**	**2.6524E + 01**	**2.9078E + 03**	3.2716E + 01
IGOA	2.4434E + 03	4.6783E + 01	5.1541E + 03	1.4171E + 03	2.7795E + 03	4.6761E + 01	2.9530E + 03	3.7102E + 01
CSSA	2.8629E + 03	5.2796E + 01	1.0557E + 04	**5.3265E + 02**	3.9929E + 03	3.1839E + 02	4.3025E + 03	3.2162E + 02
ESSA	2.7214E + 03	6.1695E + 01	9.1581E + 03	7.8162E + 02	3.3724E + 03	1.4135E + 02	3.5507E + 03	2.0359E + 02
CGSCA	2.6107E + 03	2.7294E + 01	5.2274E + 03	1.5793E + 03	3.0618E + 03	3.0965E + 01	3.2090E+03	4.0796E + 01
m_SCA	2.4556E + 03	4.2844E + 01	6.6792E + 03	1.3854E + 03	2.8622E + 03	3.9986E + 01	3.0210E + 03	4.3890E + 01
LWOA	2.5745E + 03	5.2720E + 01	6.4052E + 03	1.7099E + 03	3.0466E + 03	1.1436E + 02	3.2331E + 03	1.0017E + 02
BWOA	2.6212E + 03	8.8068E + 01	7.1370E + 03	2.1782E + 03	3.1217E + 03	8.9793E + 01	3.2405E + 03	1.0309E + 02
CCMWOA	2.5789E + 03	4.3092E + 01	6.9118E + 03	1.5541E + 03	3.1068E + 03	8.5832E + 01	3.2274E + 03	9.8426E + 01
CLPSO	2.5289E + 03	**2.1106E + 01**	4.6376E + 03	1.7143E + 03	2.9307E + 03	2.7137E + 01	3.1071E + 03	**1.8214E + 01**
GL25	2.5460E + 03	4.8547E + 01	4.0841E + 03	2.4719E + 03	2.9893E + 03	4.9368E + 01	3.1232E + 03	3.7053E + 01

	F25		F26		F27		F28	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std
MOLGOA	**2.8877E + 03**	3.6014E + 00	**4.3212E + 03**	5.7822E + 02	**3.2180E + 03**	**1.1652E + 01**	**3.2172E + 03**	2.8531E + 01
IGOA	2.8903E + 03	1.2086E + 01	4.8091E + 03	6.0692E + 02	3.2205E + 03	1.4622E + 01	3.2480E + 03	3.8060E + 01
CSSA	6.3007E + 03	6.2341E + 02	1.3271E + 04	1.4889E + 03	5.5025E + 03	6.2239E + 02	8.8506E + 03	7.4526E + 02
ESSA	4.4058E + 03	5.6440E + 02	1.0297E + 04	1.4185E + 03	3.9912E + 03	2.8736E + 02	6.3268E + 03	7.2100E + 02
CGSCA	3.5462E + 03	1.9047E + 02	7.5790E + 03	6.0604E + 02	3.5145E + 03	7.4987E + 01	4.4469E + 03	3.6735E + 02
m_SCA	3.1134E + 03	1.2538E + 02	5.7110E + 03	3.9380E + 02	3.3002E + 03	4.2353E + 01	3.6509E + 03	2.1463E + 02
LWOA	2.9039E + 03	1.8606E + 01	6.5608E + 03	1.5620E + 03	3.3174E + 03	6.8617E + 01	3.2414E + 03	**2.5825E + 01**
BWOA	3.1173E + 03	5.8024E + 01	7.7199E + 03	1.3785E + 03	3.4160E + 03	9.7984E + 01	3.5552E + 03	9.3524E + 01
CCMWOA	3.1742E + 03	9.5655E + 01	8.0934E + 03	1.1276E + 03	3.4845E + 03	1.2121E + 02	3.8187E + 03	2.4092E + 02
CLPSO	3.1095E + 03	4.6081E + 01	6.1376E + 03	**5.1783E + 02**	3.3459E + 03	2.4700E + 01	3.7989E + 03	1.7508E + 02
GL25	3.1794E + 03	8.3430E + 01	7.0119E + 03	9.0554E + 02	3.4215E + 03	4.5569E + 01	3.7540E + 03	1.9547E + 02

	F29		F30		—	—	—	—
	Mean	Std	Mean	Std				
MOLGOA	**3.6066E + 03**	**1.5376E + 02**	**5.0116E + 05**	**2.6827E + 05**	—	—	—	—
IGOA	3.9534E + 03	2.5490E + 02	6.0695E + 05	4.0479E + 05	—	—	—	—
CSSA	1.6881E + 04	2.4528E + 04	3.5401E + 09	2.0507E + 09	—	—	—	—
ESSA	6.9401E + 03	1.3441E + 03	5.3096E + 08	5.4642E + 08	—	—	—	—
CGSCA	5.1594E + 03	2.5750E + 02	1.6927E + 08	4.9229E + 07	—	—	—	—
m_SCA	4.1292E + 03	2.5298E + 02	2.2928E + 07	1.7622E + 07	—	—	—	—
LWOA	4.2958E + 03	2.6323E + 02	1.8136E + 06	9.2331E + 05	—	—	—	—
BWOA	5.1411E + 03	5.1418E + 02	4.1038E + 07	3.7057E + 07	—	—	—	—
CCMWOA	4.8515E + 03	4.1658E + 02	3.1906E + 07	3.1706E + 07	—	—	—	—
CLPSO	4.3744E + 03	2.2238E + 02	1.2836E + 07	7.5208E + 06	—	—	—	—
GL25	4.9366E + 03	3.2893E + 02	5.9287E + 06	5.3498E + 06	—	—	—	—

Algorithm	AV-ranking	Ranking
MOLGOA	1.408888889	1
IGOA	2.593333333	2
CSSA	10.85444444	11
ESSA	9.572222222	10
CGSCA	8.035555556	9
m_SCA	4.478888889	3
LWOA	4.556666667	4
BWOA	6.617777778	8
CCMWOA	6.231111111	7
CLPSO	5.52	5
GL25	6.131111111	6

the comparison outcomes, which indicates that the searching capability of MOLGOA is superior to other techniques. The findings of the comparison indicate that MOLGOA suggested in this article can also fix this issue efficiently.

5.2. TCS Problem. TCS problem requires minimizing the weight of a tension-compression spring that is subject to minimum deflection, shear stress, surge frequency, outer diameter limits, and design factors. The TCS issue solving model is defined as follows:

$$\text{consider } \overrightarrow{x} = [x_1 x_2 x_3],$$

$$\text{minimize } \quad f(\overrightarrow{x}) = x_1^2 x_2 (x_3 + 2)$$

$$y_1(x) = 1 - \frac{x_2^3 x_3}{71785 x_1^4} \le 0$$

$$y_2(x) = \frac{4x_2^2 - x_1 x_3}{12566(x_2 x_1^3 - x^4)} + \frac{1}{5108 x_1^2} - 1 \le 0$$

subject to

$$y_3(x) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0$$

$$y_4(x) = \frac{x_1 + x_2}{1.5} - 1 \le 0. \tag{16}$$

$$0.05 \le x_1 \le 2,$$

$$\text{variable range } 0.25 \le x_2 \le 1.3, \tag{17}$$

$$2 \le x_3 \le 15.$$

We adopted MOLGOA to solve the TCS problem and compared the results with other studies in the literature [70–77]. Mathematical approaches play an important role in engineering problems. However, there are still some problems in the mathematical approaches, such as the difficulty in solving the generalized geometric programming, the nondifferentiability of the dual objective function, and the divergence of the solutions of the sequential programming formed by the union terms. The proposed algorithm has the advantages of strong randomness and fewer parameters, which can avoid the above problems. The results recorded in Table 13 indicated that OLGOA found the minimum optimal cost. Compared with other existing methods, the effectiveness of this method is verified.

To conclude, our studies on these two classical engineering design issues demonstrate that the suggested MOLGOA can well optimize the practical issues. The algorithm's efficacy in addressing constraint issues is further confirmed. The reason why MOLGOA is superior to other techniques in constraint issues is that the powerful exploration capability of MOLGOA can efficiently help the GOA in coordinating the inclination for exploration.

5.3. Discussion. The main reason why the proposed MOLGOA can be applied to engineering optimization problems is the solution strategy formed by the mutual promotion of GOA, orthogonal mechanism, and joint mutation mechanism. Firstly, the basic equation of GOA is used to construct the population to maintain the diversity of the population. Secondly, Cauchy mutation has a significant ability to break through new areas in the exploration space, which means that the exploration ability of the population may be more powerful. A genetic algorithm is another mechanism for exploring new regions in solving nonlinear programming. The joint mutation mechanism, combined with these two different exploration methods, can help GOA to achieve the best solution faster and improve the convergence performance of the proposed algorithm. The orthogonal mechanism in the algorithm can reduce the search scale and make GOA find the global optimal value faster. Although the proposed MOLGOA has advantages in solving optimization problems, it may require considerable computational cost in the addition of orthogonal steps in MOLGOA. Therefore, although the orthogonal mechanism is very effective in some applications, a new mechanism should be studied in future research to dynamically select the most reasonable Q and F, so that MOLGOA can take into account the computational

Table 11: Statistical comparison results by Wilcoxon test.

	IGOA	CSSA	ESSA	CGSCA	m_SCA	LWOA	BWOA	CCMWOA	CLPSO	GL25
F1	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F2	$4.0715E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$2.3534E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7333E-06$
F3	$1.7344E-06$	$1.7333E-06$	$1.7333E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F4	$3.0861E-01$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$2.2551E-03$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F5	$1.7423E-04$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F6	$2.3534E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.9209E-06$	$1.7344E-06$
F7	$3.5888E-04$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F8	$1.3601E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$2.1266E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F9	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F10	$1.7423E-04$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$3.1817E-06$	$2.1266E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F11	$9.7110E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.0444E-02$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F12	$2.8021E-01$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7423E-04$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F13	$1.1079E-02$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$5.6672E-03$	$1.7344E-06$	$2.3534E-06$	$1.7344E-06$	$2.8786E-06$
F14	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.9209E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F15	$7.1903E-02$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$7.5137E-05$	$8.5896E-02$	$1.9209E-06$	$1.8326E-03$	$1.7344E-06$	$2.0515E-04$
F16	$6.3198E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F17	$1.0246E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.4773E-04$	$4.7292E-06$	$1.7344E-06$	$1.7344E-06$	$2.3534E-06$	$2.1266E-06$
F18	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.9209E-06$	$1.7344E-06$	$1.7344E-06$
F19	$5.7924E-05$	$1.7333E-06$	$1.7333E-06$	$1.7344E-06$	$1.7344E-06$	$2.3534E-06$	$1.7344E-06$	$1.9209E-06$	$1.7344E-06$	$8.5896E-02$
F20	$1.6394E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$4.0715E-05$	$1.7344E-06$	$2.8786E-06$	$6.3391E-06$	$9.3157E-06$	$1.9209E-06$
F21	$6.9838E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$3.4053E-05$	$1.7344E-06$	$1.9209E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F22	$2.8308E-04$	$1.7344E-06$	$1.7344E-06$	0.00089443	$4.2857E-06$	$7.5137E-05$	$6.3391E-06$	$5.2165E-06$	$2.1827E-02$	$5.3044E-01$
F23	$2.2551E-03$	$1.7333E-06$	$1.7333E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F24	$2.0515E-04$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.9209E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F25	$2.8486E-02$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.6046E-04$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F26	$4.3896E-03$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.9209E-06$	$1.0246E-05$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.9209E-06$
F27	$9.9179E-01$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F28	$6.8923E-05$	$1.7333E-06$	$1.7333E-06$	$1.7344E-06$	$1.7344E-06$	$2.4147E-03$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F29	$8.1878E-05$	$1.7333E-06$	$1.7333E-06$	$1.7344E-06$	$2.1266E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$
F30	$3.1849E-01$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$6.3391E-06$	$1.7344E-06$	$1.7344E-06$	$1.7344E-06$	$3.8822E-06$
+/−/=	25/0/5	30/0/0	30/0/0	30/0/0	30/0/0	29/0/1	30/0/0	30/0/0	30/0/0	27/1/2
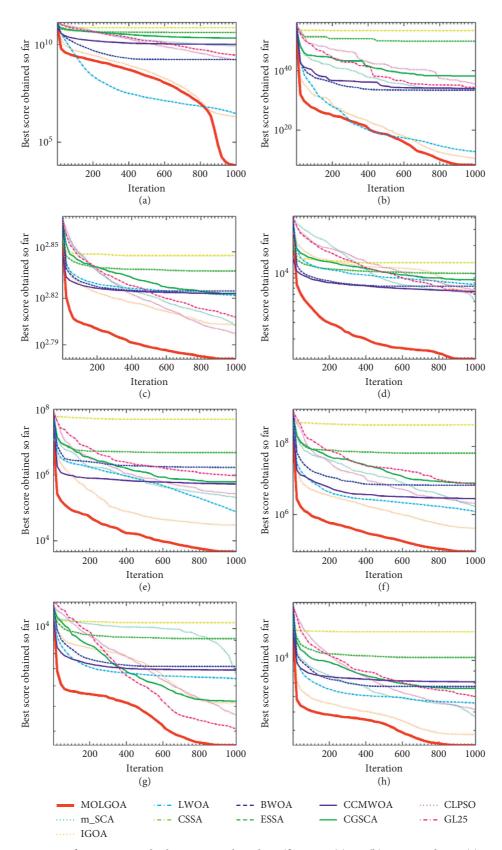
Figure 3: Convergence curve of MOLGOA and other variants algorithms (first row: (a) $F1$, (b) $F2$; second row: (c) $F6$, (d) $F9$; third row: (e) $F14$, (f) $F18$; fourth row: (g) $F22$, (h) $F26$).
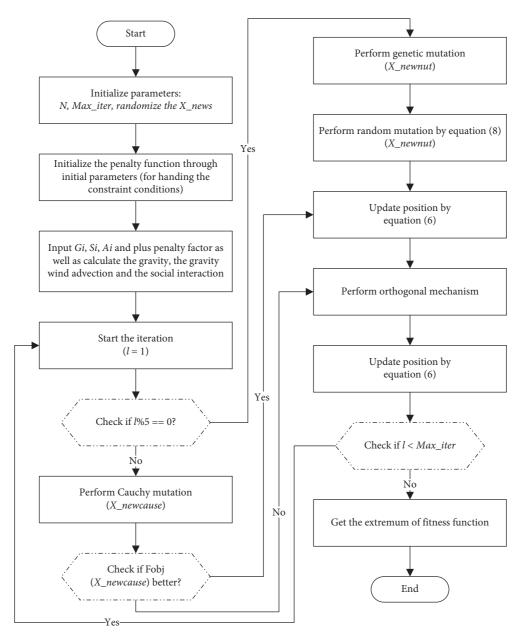
FIGURE 4: The flowchart of MOLGOA for constraint optimization. X_new is the position of the grasshopper, l is the current iteration, N is the population and the X_newcause is the position after the mutation operation.

TABLE 12: Comparison results of the PV problem.

Algorithm	Optimal values for variables				Optimum cost
	X1	X2	X3	X4	
MOLGOA	1.384396614	0.68093793000	66.09756461	10	6011.9457
GOA	1.666276393	0.722918903	65.01641084	11.27921945	6410.2862
IGOA	1.746050294	0.647991579	67.29749345	10	6129.4126
WOA	0.812500	0.437500	42.098209	176.638998	6059.7410
HIS	1.125000	0.625000	58.29015	43.6927	7197.7300
PSO (He et al.)	0.812500	0.437500	42.091266	176.746500	6061.0777

TABLE 13: Comparison results of the TCS problem.

Algorithm	Optimal values for variables			Optimum cost
	X1	X2	X3	
MOLGOA	0.051699009	0.35695648	11.27564335	0.012665877
GOA	0.052113	0.367013	10.71071	0.012669273
IGOA	0.051961	0.363295	10.91954	0.012672589
CPSO	0.051728	0.357644	11.244543	0.0126747
Canayaz and Karci	0.052036	0.365113	10.815329	0.012670
Coello and Montes	0.051989	0.351661	11.632201	0.0126810
Belegundu	0.05000	0.315900	14.250000	0.0128334
Arora	0.053396	0.399180	9.185400	0.0127303
Coello	0.051480	0.351661	11.632201	0.0127048
Coello and Becerra	0.050000	0.317395	14.031795	0.0127210

cost and get the best solution, to reduce the problem of high computational cost. At this stage, we mainly conducted a large number of experiments, continuously adjusted to select reasonable values for Q and F. However, in the existing experiments and engineering applications, we can see that the introduction of optimization mechanism in MOLGOA can greatly enhance the performance of GOA. We hope this method can be applied to other methods in the optimization field.

6. Conclusions and Future Works

In the presented study, MOLGOA is developed to alleviate the limits of the original GOA, such as insufficient exploration ability and slow convergence speed. This approach mainly introduces the orthogonal learning mechanism to make MOLGOA converge faster. Moreover, a combinatorial mutation strategy is proposed to realize the complementary advantages of Cauchy mutation and genetic mutation. The numerical results of typical benchmark functions show that the proposed MOLGOA can effectively tackle different types of optimization tasks. For the constrained optimization issues, the simulation results also illustrate that MOLGOA can enhance the calculation results and have practical value when solving two engineering design problems.

Of course, as a relatively new algorithm, there are still many problems worthy of further study because of the tremendous potential of GOA. Firstly, classical metaheuristic algorithms can be integrated into GOA to achieve a better balance between exploration and exploitation and improve the global search ability of GOA. Then, how to use MOLGOA to solve multiobjective tasks and dynamic optimization is scheduled as the next work. Besides, considering the time and space complexity of the algorithm when solving relatively complex problems, for further improving computational capacity and efficiency of the algorithm, GPU parallel algorithm and multithread parallel processing technology can be adopted to explore more possibilities. Finally, the exploration of the proposed method to other promising problems will be also interesting such as the parameter tuning for deep learning models [78–80], feature selection problems [81–84], social evolution modelling [85], and video coding optimization [86].

Data Availability

The data involved in this study are all public data, which can be downloaded through public channels.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of the article.

Authors' Contributions

Guoxi Liang and Huiling Chen conceptualized the study. Hanfeng Zhou, Hongxin Peng, and Zitao Tang performed data curation. Guoxi Liang and Huiling Chen were responsible for funding acquisition. Hanfeng Zhou, Hongxin Peng, and Huiling Chen performed investigation. Hanfeng Zhou, Hongxin Peng, Zitao Tang, Chao Ma, Huiling Chen, Mingjing Wang were responsible for methodology. Hanfeng Zhou, Zewei Ding, Huiling Chen wrote the original draft. Guoxi Liang and Huiling Chen reviewed and edited the manuscript. Guoxi Liang and Huiling Chen contributed equally to this work.

References

[1] Y. Sun, G. G. Yen, and Z. Yi, "IGD indicator-based evolutionary algorithm for many-objective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 2, pp. 173–187, 2019.
[2] Y. Sun, B. Xue, M. Zhang, and G. G. Yen, "Evolving deep convolutional neural networks for image classification," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 2, pp. 394–407, 2020.

[3] Y. Sun, B. Xue, M. Zhang, G. G. Yen, and J. Lv, "Automatically designing CNN architectures using the genetic algorithm for image classification," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3840–3854, 2020.

[4] H. Chen, Y. Xu, M. Wang, and X. Zhao, "A balanced whale optimization algorithm for constrained engineering design problems," *Applied Mathematical Modelling*, vol. 71, pp. 45–59, 2019.

[5] J. Luo, H. Chen, A. A. Heidari, Y. Xu, Q. Zhang, and C. Li, "Multi-strategy boosted mutative whale-inspired optimization approaches," *Applied Mathematical Modelling*, vol. 73, pp. 109–123, 2019.

[6] H. Yu, N. Zhao, P. Wang, H. Chen, and C. Li, "Chaos-enhanced synchronized bat optimizer," *Applied Mathematical Modelling*, vol. 77, pp. 1201–1215, 2020.

[7] H. Chen, M. Wang, and X. Zhao, "A multi-strategy enhanced sine cosine algorithm for global optimization and constrained practical engineering problems," *Applied Mathematics and Computation*, vol. 369, Article ID 124872, 2020.

[8] X. Zhang, Y. Xu, C. Yu et al., "Gaussian mutational chaotic fruit fly-built optimization and feature selection," *Expert Systems with Applications*, vol. 141, Article ID 112976, 2020.

[9] X. Zhang, D. Wang, Z. Zhou, and Y. Ma, "Robust low-rank tensor recovery with rectification and alignment," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 1, 2019.

[10] X. Zhao, D. Li, B. Yang, C. Ma, Y. Zhu, and H. Chen, "Feature selection based on improved ant colony optimization for online detection of foreign fiber in cotton," *Applied Soft Computing*, vol. 24, pp. 585–596, 2014.

[11] M. Wang and H. Chen, "Chaotic multi-swarm whale optimizer boosted support vector machine for medical diagnosis," *Applied Soft Computing*, vol. 88, p. 105946, 2020.

[12] X. Zhao, X. Zhang, Z. Cai et al., "Chaos enhanced grey wolf optimization wrapped ELM for diagnosis of paraquat-poisoned patients," *Computational Biology and Chemistry*, vol. 78, pp. 481–490, 2019.

[13] X. Xu and H.-L. Chen, "Adaptive computational chemotaxis based on field in bacterial foraging optimization," *Soft Computing*, vol. 18, no. 4, pp. 797–807, 2014.

[14] L. Shen, H. Chen, Z. Yu et al., "Evolving support vector machines using fruit fly optimization for medical data classification," *Knowledge-Based Systems*, vol. 96, pp. 61–75, 2016.

[15] M. Wang, H. Chen, B. Yang et al., "Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses," *Neurocomputing*, vol. 267, pp. 69–84, 2017.

[16] Y. Xu, H. Chen, J. Luo, Q. Zhang, S. Jiao, and X. Zhang, "Enhanced Moth-flame optimizer with mutation strategy for global optimization," *Information Sciences*, vol. 492, pp. 181–203, 2019.

[17] H. Chen, Q. Zhang, J. Luo, Y. Xu, and X. Zhang, "An enhanced bacterial foraging optimization and its application for training kernel extreme learning machine," *Applied Soft Computing*, vol. 86, p. 105884, 2020.

[18] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the ICNN'95—International Conference on Neural Networks*, Perth, Australia, November 1995.

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

[20] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007.

[21] X. S. Yang, "Firefly algorithms for multimodal optimization," in *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pp. 169–178, Springer, Berlin, Germany, 2009.

[22] W.-T. Pan, "A new fruit fly optimization algorithm: taking the financial distress model as an example," *Knowledge-Based Systems*, vol. 26, pp. 69–74, 2012.

[23] A. A. Heidari, S. Mirjalili, H. Faris, I. Aljarah, M. Mafarja, and H. Chen, "Harris hawks optimization: algorithm and applications," *Future Generation Computer Systems*, vol. 97, pp. 849–872, 2019.

[24] S. Li, H. Chen, M. Wang, A. A. Heidari, and S. Mirjalili, "Slime mould algorithm: a new method for stochastic optimization," *Future Generation Computer Systems*, vol. 111, pp. 300–323, 2020.

[25] S. Saremi, S. Mirjalili, and A. Lewis, "Grasshopper optimisation algorithm: theory and application," *Advances in Engineering Software*, vol. 105, pp. 30–47, 2017.

[26] I. Aljarah, A. M. Al-Zoubi, H. Faris, M. A. Hassonah, S. Mirjalili, and H. Saadeh, "Simultaneous feature selection and support vector machine optimization using the grasshopper optimization algorithm," *Cognitive Computation*, vol. 10, no. 3, pp. 478–495, 2018.

[27] M. Barman, N. B. Dev Choudhury, and S. Sutradhar, "A regional hybrid GOA-SVM model based on similar day approach for short-term load forecasting in Assam, India," *Energy*, vol. 145, pp. 710–720, 2018.

[28] A. Hamad, E. H. Houssein, A. E. Hassanien, and A. A. Fahmy, "Hybrid grasshopper optimization algorithm and support vector machines for automatic seizure detection in EEG signals," in *The International Conference on Advanced Machine Learning Technologies and Applications (AMLTA2018)*, pp. 82–91, Springer, Berlin, Germany, 2018.

[29] H. T. Ibrahim, W. J. Mazher, O. N. Ucan, and O. Bayat, "A grasshopper optimizer approach for feature selection and optimizing SVM parameters utilizing real biomedical data sets," *Neural Computing and Applications*, vol. 31, no. 10, pp. 5965–5974, 2019.

[30] S. Arora and P. Anand, "Chaotic grasshopper optimization algorithm for global optimization," *Neural Computing and Applications*, vol. 31, no. 8, pp. 4385–4405, 2019.

[31] A. Saxena, S. Shekhawat, and R. Kumar, "Application and development of enhanced chaotic grasshopper optimization algorithms," *Modelling and Simulation in Engineering*, vol. 2018, Article ID 4945157, 14 pages, 2018.

[32] A. A. El-Fergany, "Electrical characterisation of proton exchange membrane fuel cells stack using grasshopper optimiser," *IET Renewable Power Generation*, vol. 12, no. 1, pp. 9–17, 2018.

[33] A. Fathy, "Recent meta-heuristic grasshopper optimization algorithm for optimal reconfiguration of partially shaded PV array," *Solar Energy*, vol. 171, pp. 638–651, 2018.

[34] A. A. Ewees, M. Abd Elaziz, and E. H. Houssein, "Improved grasshopper optimization algorithm using opposition-based learning," *Expert Systems with Applications*, vol. 112, pp. 156–172, 2018.

[35] A. A. Heidari, H. Faris, I. Aljarah, and S. Mirjalili, "An efficient hybrid multilayer perceptron neural network with grasshopper optimization," *Soft Computing*, vol. 23, no. 17, pp. 7941–7958, 2019.

[36] J. Liu, A. Wang, Y. Qu, and W. Wang, "Coordinated operation of multi-integrated energy system based on linear weighted sum and grasshopper optimization algorithm," *IEEE Access*, vol. 6, pp. 42186–42195, 2018.

[37] J. Luo, H. Chen, Q. Zhang, Y. Xu, H. Huang, and X. Zhao, "An improved grasshopper optimization algorithm with application to financial stress prediction," *Applied Mathematical Modelling*, vol. 64, pp. 654–668, 2018.

[38] M. Mafarja, I. Aljarah, H. Faris, A. I. Hammouri, A. M. Al-Zoubi, and S. Mirjalili, "Binary grasshopper optimisation algorithm approaches for feature selection problems," *Expert Systems with Applications*, vol. 117, pp. 267–286, 2019.

[39] S. Z. Mirjalili, S. Mirjalili, S. Saremi, H. Faris, and I. Aljarah, "Grasshopper optimization algorithm for multi-objective optimization problems," *Applied Intelligence*, vol. 48, no. 4, pp. 805–820, 2018.

[40] A. Tharwat, E. H. Houssein, M. M. Ahmed, A. E. Hassanien, and T. Gabel, "MOGOA algorithm for constrained and unconstrained multi-objective optimization problems," *Applied Intelligence*, vol. 48, no. 8, pp. 2268–2283, 2018.

[41] J. Wu, Z. Cui, Y. Chen, D. Kong, and Y.-G. Wang, "A new hybrid model to predict the electrical load in five states of Australia," *Energy*, vol. 166, pp. 598–609, 2019.

[42] J. Wu, H. Wang, N. Li et al., "Distributed trajectory optimization for multiple solar-powered UAVs target tracking in urban environment by adaptive grasshopper optimization algorithm," *Aerospace Science and Technology*, vol. 70, pp. 497–510, 2017.

[43] A. Zakeri and A. Hokmabadi, "Efficient feature selection method using real-valued grasshopper optimization algorithm," *Expert Systems with Applications*, vol. 119, pp. 61–72, 2019.

[44] X. Zhang, Q. Miao, H. Zhang, and L. Wang, "A parameter-adaptive VMD method based on grasshopper optimization algorithm to analyze vibration signals from rotating machinery," *Mechanical Systems and Signal Processing*, vol. 108, pp. 58–72, 2018.

[45] H. Zhao, H. Zhao, and S. Guo, "Short-term wind electric power forecasting using a novel multi-stage intelligent algorithm," *Sustainability (Switzerland)*, vol. 10, no. 3, p. 881, 2018.

[46] H. Zhang, A. A. Heidari, M. Wang, L. Zhang, H. Chen, and C. Li, "Orthogonal nelder-mead moth flame method for parameters identification of photovoltaic modules," *Energy Conversion and Management*, vol. 211, Article ID 112764, 2020.

[47] H. Zhang, R. Li, Z. Cai et al., "Advanced orthogonal moth flame optimization with Broyden-Fletcher-Goldfarb-Shanno algorithm: framework and real-world problems," *Expert Systems with Applications*, vol. 159, Article ID 113617, 2020.

[48] Y. Yang, H. Chen, S. Li, A. A. Heidari, and M. Wang, "Orthogonal learning harmonizing mutation-based fruit fly-inspired optimizers," *Applied Mathematical Modelling*, vol. 86, pp. 368–383, 2020.

[49] W. Zhu, C. Ma, X. Zhao et al., "Evaluation of sino foreign cooperative education project using orthogonal sine cosine optimized kernel extreme learning machine," *IEEE Access*, vol. 8, pp. 61107–61123, 2020.

[50] H. Chen, A. A. Heidari, X. Zhao, L. Zhang, and H. Chen, "Advanced orthogonal learning-driven multi-swarm sine cosine optimization: framework and case studies," *Expert Systems with Applications*, vol. 144, Article ID 113113, 2020.

[51] A. F. Ba, H. Huang, M. Wang et al., "Levy-based antlion-inspired optimizers with orthogonal learning scheme," *Engineering with Computers*, 2020.

[52] S. Jiao, G. Chong, C. Huang et al., "Orthogonally adapted Harris hawks optimization for parameter estimation of photovoltaic models," *Energy*, vol. 203, Article ID 117804, 2020.

[53] Z. Xu, Z. Hu, A. A. Heidari et al., "Orthogonally-designed adapted grasshopper optimization: a comprehensive analysis," *Expert Systems with Applications*, vol. 150, Article ID 113282, 2020.

[54] G. Xiong and D. Shi, "Orthogonal learning competitive swarm optimizer for economic dispatch problems," *Applied Soft Computing*, vol. 66, pp. 134–148, 2018.

[55] X. Dan and B. Yuan, "Partially adaptive array design based on orthogonal projection algorithm," in *Proceedings of Third International Conference on Signal Processing (ICSP'96)*, Beijing, China, October 1996.

[56] T. Yokota and M. Gen, "Solving for nonlinear integer programming problem using genetic algorithm and its application," in *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pp. 1602–1609, San Antonio, TX, USA, October 1994.

[57] A. Hu, D. M. York, and T. K. Woo, "Time-dependent density functional theory calculations of molecular static and dynamic polarizabilities, cauchy coefficients and their anisotropies with atomic numerical basis functions," *Journal of Molecular Structure: THEOCHEM*, vol. 591, no. 1–3, pp. 255–266, 2002.

[58] G.-G. Wang, S. Deb, A. H. Gandomi, and A. H. Alavi, "Opposition-based krill herd algorithm with Cauchy mutation and position clamping," *Neurocomputing*, vol. 177, pp. 147–157, 2016.

[59] G. Salvador, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining," *Experimental Analysis of Power Information Sciences*, vol. 180, no. 10, pp. 2044–2064, 2010.

[60] J. Alcalá-Fdez, L. Sánchez, S. García et al., "KEEL: a software tool to assess evolutionary algorithms for data mining problems," *Soft Computing*, vol. 13, no. 3, pp. 307–318, 2008.

[61] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[62] S. Mirjalili, "SCA: a sine cosine algorithm for solving optimization problems," *Knowledge-Based Systems*, vol. 96, pp. 120–133, 2016.

[63] Z. Yuhang, L. Qi, and W. Yang, "Materialized view selection algorithm—CSSA_VSP," in *Proceedings of the 2010 Second International Conference on Computational Intelligence and Natural Computing*, Wuhan, China, September 2010.

[64] T.-C. Lu and J.-C. Juang, "An evolutionary space search algorithm (ESSA) for global numerical optimization, decision and control," in *Proceedings of the 48h ieee conference on decision and control (cdc) held jointly with 2009 28th chinese control conference*, Shanghai, China, December 2009.

[65] W. Zhu, P. Xu, T. Bui, G. Wu, and Y. Yang, "Energy-efficient cell-association bias adjustment algorithm for ultra-dense networks," *Science China Information Sciences*, vol. 61, no. 2, 2017.

[66] S. Gupta and K. Deep, "A hybrid self-adaptive sine cosine algorithm with opposition based learning," *Expert Systems with Applications*, vol. 119, pp. 210–230, 2019.

[67] Y. Ling, Y. Zhou, and Q. Luo, "Lévy flight trajectory-based whale optimization algorithm for global optimization," *IEEE Access*, vol. 5, pp. 6168–6186, 2017.

[68] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 3, pp. 281–295, 2006.

[69] C. García-Martínez, M. Lozano, F. Herrera, D. Molina, and A. M. Sánchez, "Global and local real-coded genetic algorithms based on parent-centric crossover operators," *European Journal of Operational Research*, vol. 185, no. 3, pp. 1088–1113, 2008.

[70] C. A. Coello Coello, "Constraint-handling using an evolutionary multiobjective optimization technique," *Civil Engineering and Environmental Systems*, vol. 17, no. 4, pp. 319–346, 2000.

[71] C. A. C. Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art," *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 11-12, pp. 1245–1287, 2002.

[72] C. A. Coello Coello and E. M. Montes, "Constraint-handling in genetic algorithms through the use of dominance-based tournament selection," *Advanced Engineering Informatics*, vol. 16, no. 3, pp. 193–203, 2002.

[73] K. Deb, "An effcient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2–4, pp. 311–338, 2000.

[74] Q. He and L. Wang, "An effective co-evolutionary particle swarm optimization for constrained engineering design problems," *Engineering Applications of Artificial Intelligence*, vol. 20, no. 1, pp. 89–99, 2007.

[75] K. M. Ragsdell and D. T. Phillips, "Optimal design of a class of welded structures using geometric programming," *Journal of Manufacturing Science and Engineering*, vol. 98, no. 3, pp. 1021–1025, 1976.

[76] J. N. Siddall, *Analytical Decision-Making in Engineering Design*, Prentice-Hall, Englewood Cliffs, NJ, USA, 1972.

[77] M. Canayaz and A. Karci, "Cricket behaviour-based evolutionary computation technique in solving engineering optimization problems," *Applied Intelligence*, vol. 44, no. 2, pp. 362–376, 2016.

[78] X. Zhang, T. Wang, J. Wang, G. Tang, and L. Zhao, "Pyramid channel-based feature attention network for image dehazing," *Computer Vision and Image Understanding*, vol. 197-198, Article ID 103003, 2020.

[79] Y. Li, W.-G. Cui, H. Huang, Y.-Z. Guo, K. Li, and T. Tan, "Epileptic seizure detection in EEG signals using sparse multiscale radial basis function networks and the Fisher vector approach," *Knowledge-Based Systems*, vol. 164, pp. 96–106, 2019.

[80] Y. Li, J. Liu, Z. Tang, and B. Lei, "Deep spatial-temporal feature fusion from adaptive dynamic functional connectivity for MCI identification," *IEEE Transactions on Medical Imaging*, vol. 39, no. 9, pp. 2818–2830, 2020.

[81] X. Zhang, M. Fan, D. Wang, P. Zhou, and D. Tao, "Top-k feature selection framework using robust 0-1 integer programming," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–15, 2020.

[82] Y. Zhang, R. Liu, X. Wang, H. Chen, and C. Li, "Boosted binary Harris hawks optimizer and feature selection," *Engineering with Computers*, 2020.

[83] Q. Li, H. Chen, H. Huang et al., "An enhanced grey wolf optimization based feature selection wrapped kernel extreme learning machine for medical diagnosis," *Computational and Mathematical Methods in Medicine*, vol. 2017, Article ID 9512741, 15 pages, 2017.

[84] H. Huang, X. Feng, S. Zhou et al., "A new fruit fly optimization algorithm enhanced support vector machine for diagnosis of breast cancer based on high-level features," *BMC Bioinformatics*, vol. 84, no. S8, 2019.

[85] X. Xue, S. Wang, L. Zhang, Z. Feng, and Y. Guo, "Social learning evolution (SLE): computational experiment-based modeling framework of social manufacturing," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3343–3355, 2019.

[86] Y. Zhou, L. Tian, C. Zhu, X. Jin, and Y. Sun, "Video coding optimization for virtual reality 360-degree source," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 1, pp. 118–129, 2020.