Received 1 December 2022, accepted 23 January 2023, date of publication 30 January 2023, date of current version 2 February 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3240467

RESEARCH ARTICLE

An Improved Genetic Algorithm for Constrained Optimization Problems

FULIN WANG, GANG XU, AND MO WANG
College of Engineering, Northeast Agricultural University, Harbin 150030, China

Corresponding author: Fulin Wang (wangfulin@neau.edu.cn)

This work was supported in part by the Natural Science Foundation of Heilongjiang Province under Grant LH2020C004.

ABSTRACT The mathematical form of many optimization problems in engineering is constrained optimization problems. In this paper, an improved genetic algorithm based on two-direction crossover and grouped mutation is proposed to solve constrained optimization problems. In addition to making full use of the direction information of the parent individual, the two-direction crossover adds an additional search direction and finally searches in the better direction of the two directions, which improves the search efficiency. The grouped mutation divides the population into two groups and uses mutation operators with different properties for each group to give full play to the characteristics of these mutation operators and improve the search efficiency. In experiments on the IEEE CEC 2017 competition on constrained real-parameter optimization and ten real-world constrained optimization problems, the proposed algorithm outperforms other state-of-the-art algorithms. Finally, the proposed algorithm is used to optimize a single-stage cylindrical gear reducer.

INDEX TERMS Genetic algorithm, constrained optimization problem, two-direction crossover, grouped mutation.

I. INTRODUCTION

Many engineering practices involve the solution of a constrained optimization problem [1], [2], [3], [4]. A constrained optimization problem (COP) aims at finding the optimal solution of a numerical function (called an objective function) with some equality or inequality constraints. COP could be classified into many categories, such as linear programming [5], convex quadratic programming [6], convex optimization [7], and non-convex optimization [8]. Both linear programming and convex quadratic programming are typical types of convex optimization. The convex optimization problem is relatively easy to solve because there is no local optimum, whereas non-convex optimization could have many local optima and it is impossible to assert that a solution is locally optimal unless a better one is found. Unfortunately, COPs in many engineering practices are non-convex optimization.

The associate editor coordinating the review of this manuscript and approving it for publication was Daniel Augusto Ribeiro Chaves.

Researchers have developed many types of evolutionary algorithms to solve COPs [9], [10], [11], [12]. The genetic algorithm (GA) is a powerful type of evolutionary algorithm that can solve the non-convex optimization problem because GA keeps a population of solutions and it not only utilizes information in the current population but also explores new areas in the search region. Therefore, individuals in GA can escape from local optima. Researchers have made many developments of GA, and the main efforts could be roughly classified into two categories: develop new crossover operators and new mutation operators.

The crossover operator generates offspring by recombining information of individuals in the current population. In the early period, crossover operators of the genetic algorithm are designed by directly imitating crossover operators of binary GA. Simulated binary crossover simulates the principle of the single-point crossover operator on binary strings [13]. Later researchers got rid of the shackles of binary coding and began to develop crossover operators on real numbers. Arithmetic crossover, local crossover, flat crossover, and blend crossover generate two offspring on the line segment between two parents [14], [15], [16], [17], [18]. Recently, researchers focus on the direction-based crossover operators, which search for new solutions using the information of direction from the worse parent to the better parent [19], [20], [21], [22].

The mutation operator randomly changes individuals to avoid getting stuck in local optima. To achieve this purpose, the mutation operators often add random perturbations to individuals [23], [24], [25], [26], [27], [28]. Therefore, the mutation is also searching for the optimal solution with certain rules, and different mutation operators have different characteristics. However, different evolution stages or different types of problems require different search strategies according to the evolution process or the nature of the problem [29]. Combinational mutation combines multiple mutation methods so that the mutation operator has the advantage of multiple mutation operators [21], [30].

In this paper, namely GA-TDX, an improved genetic algorithm is proposed, which consists of two-direction crossover and grouped mutation. The two-direction crossover uses the information from the worse parent to the better parent. However, due to COPs being nonlinear, this direction is usually not the optimal search direction. Therefore, the two-direction crossover finds a direction at an angle of 45 degrees to this direction as a new search direction, searches for an individual along each of the two search directions in the two parents, and retains the two optimal individuals. This additional search direction makes two-direction crossover a better search capability. In grouped mutation, individuals in different parts of the population are mutated by different single mutation operators. Individuals in different parts of the population suit different search tasks, to be precise, the best part of the population is good at finding the global optimum, whereas the worst part of the population is good at skipping the local optimum.

The remaining of this paper is organized as follows. Section II briefly introduces the mathematical model and constraint handling technique of COP. Section III introduces details of the proposed GA-TDX. Section IV shows the experimental results of the proposed algorithm compared with other algorithms on the IEEE CEC 2017 benchmark functions and ten COPs. In Section V, GA-TDX is applied to optimize the parameters of a single-stage cylindrical gear reducer. Finally, Section VI concludes this paper.

II. CONSTRAINT OPTIMIZATION PROBLEM

A. MATHEMATICAL MODEL OF COP

The objective function of real-valued COP is formulated in (1).

$$\min f(x),\ x = [x_1, x_2, \dots, x_n] \in \mathbb{R}^n \tag{1}$$

where $x \in \mathcal{F} \subseteq \mathcal{S}$. The objective function f is defined on the search space $\mathcal{S} \in \mathbb{R}^n$ and the set $\mathcal{F} \subseteq \mathcal{S}$ defines the feasible region. Usually, the search space $\mathcal{S}$ is defined as an n-dimensional rectangle in $\mathbb{R}^n$ (domains of variables defined by their lower and upper bounds) as shown in (2). The feasible region $\mathcal{F} \subseteq \mathcal{S}$ is defined by a set of $P + Q$ additional constraints as shown in (3). The infeasible region $\mathcal{I}$ is defined by $\mathcal{F} \cup \mathcal{I} = \mathcal{S}$ and $\mathcal{F} \cap \mathcal{I} = \emptyset$. At any point $x \in \mathcal{F}$, the constraints g_k that satisfy $g_k(x) = 0$ are called the active constraints at x [31].

$$a_i \le x_i \le b_i, \quad i = 1, 2, \dots, n \tag{2}$$

$$\begin{aligned} &g_i \le 0, \quad i = 1, 2, \dots, P \\ &h_i = 0, \quad i = 1, 2, \dots, Q \end{aligned} \tag{3}$$

COP is more challenging than an unconstrained optimization problem. The search region $\mathcal{S}$ of COP is divided into several small irregular regions by the constraints. Some of these small regions are feasible regions, whereas others are infeasible regions. Thus, COP is essentially an optimization problem to find the optimal solution in $\mathcal{F}$, whereas an unconstrained optimization problem is an optimization problem to find the optimal solution in $\mathcal{S}$. The first challenge of COP is that the optimal solution in $\mathcal{F}$ is always worse than the optimal solution in $\mathcal{S}$, as a result, the optimal solution is very likely located on the border between $\mathcal{F}$ and $\mathcal{I}$. However, it is difficult for Evolutionary algorithms to precisely search the boundary area [31]. Secondly, these small feasible regions may not be adjacent and evolutionary algorithms have to find new feasible regions which have not been found, because these undiscovered regions may contain the optimal solution.

B. CONSTRAINT HANDLING TECHNIQUE

Constraint handling techniques are of great significance in efficiently solving constrained optimization problems. Researchers have proposed many types of penalty functions, such as static penalty [32], dynamic penalty [33], [34], the superiority of feasible solutions [35], ranking-based method [36], and ensemble constraint handling technique [37]. In the evolutionary algorithm community, the most commonly used approach to handle constraints is the penalty function [38]. The idea of this method is to transform a COP into an unconstrained optimization problem by adding a penalty term (negative for maximization problems or positive for minimization problems) to the objective function based on the amount of constraint violation present in a certain solution. In this paper, the static penalty function is used because of its simplicity. The mathematical form of static penalty is shown in (4).

$$\phi(x) = f(x) + m \times (\sum_{i=1}^{P} max(0, g_i(x))^2 + \sum_{i=1}^{Q} h_i^2(x)) \tag{4}$$

where m is the predefined penalty parameter to define the tolerance of constraints violation. To ensure the feasibility of the obtained optimal solution, m should be a sufficiently large positive number.

III. THE PROPOSED ALGORITHMS

A. THE ALGORITHM STRUCTURE

Algorithm 1 shows the structure of the proposed algorithm, consisting of sorting grouping selection, two-direction crossover (TDX), and grouped mutation (GM). Sorting grouping selection cuts a population into two parts based on the fitness value and pairs individuals between these two parts to ensure that the algorithm searches in the direction of decreasing the value of the objective function [21]. Then, TDX searches along not only the direction from the worse parent to the better parent but also a random direction at 45 degrees to the former to avoid getting stuck in a local optimum. Finally, GM cut the population into two groups based on the fitness value. The best group is used for local search, and the worst group is used for global search.

Algorithm 1 Algorithm of GA-TDX

```
Require: Population size NP, Initialize population $x(0) = \{x_i(0) \in S, i = 1, 2, \ldots, NP\}$, Maximum number of iterations T.
Ensure: best found solutions $x_b$ and its function value $f(x_b)$
  k = 0
  while k < T do
    sort $x(k)$
    cut $x(k)$ into two parts, $x^1 = \{x_i(k), i = 1, 2, \ldots, \frac{NP}{2}\}$ and $x^2 = \{x_i(k) | i = \frac{NP}{2} + 1, \frac{NP}{2} + 2, \ldots, NP\}$ //Sorting Grouping Selection
    $x^c \leftarrow$ Two-direction Crossover($x^1, x^2$) //see Algorithm 2
    $x_i(k)$ = best of $x_i(k)$ and $x_i^c$, i=1,2,...,NP
    $x^m \leftarrow$ Grouped Mutation($x(k)$) //see Algorithm 3
    $x_i(k+1) \leftarrow$ best of $x_i(k)$ and $x_i^m$, i=1,2,...,NP
    $k = k + 1$
  end while
  return best found solution $x_b$ in $x(k)$ and its function value $f(x_b)$
```

B. TWO-DIRECTION CROSSOVER

Given that parents x_1, x_2, if the fitness value of x_1 is smaller than x_2, the first search direction is $d_1 = x_1 - x_2$. Then, using the method in (6) (details of this equation will be introduced later), a vector $d_p = (d_{p1}, d_{p2}, \ldots, d_{pn})^T$ perpendicular to $d_1 = (d_{11}, d_{12}, \ldots, d_{1n})^T$ is found and the length of d_p is scaled to $||d_1||$ using (5), where $||d|| = \sqrt{d^T \cdot d} = \sqrt{\sum d_i^2}$ represents the length of a vector d. Therefore, the length of d_p is $||d_p|| = \sqrt{\sum_{i=1}^n (\frac{d_{pi}||d_1||}{||d_p||})^2} = \sqrt{(\frac{||d_1||}{||d_p||})^2 \sum_{i=1}^n d_{pi}^2} = \frac{||d_1||}{||d_p||}\sqrt{\sum_{i=1}^n d_{pi}^2} = \frac{||d_1||}{||d_p||}||d_p|| = ||d_1||$. Next, the second search direction is $d_2 = \frac{d_1+d_p}{2}$ and its length is $||d_2|| = \sqrt{(\frac{d_1+d_p}{2})^T \cdot \frac{d_1+d_p}{2}} = \sqrt{\frac{d_1^T \cdot d_1 + d_1^T \cdot d_p + d_p^T \cdot d_1 + d_p^T \cdot d_p}{4}} = \sqrt{\frac{||d_1|| + ||d_p||}{4}} = \frac{\sqrt{2}}{2}||d_1||$. There is a 45° angle between d_1 and d_2 because $d_1 \cdot d_2 = d_1 \cdot \frac{(d_1+d_p)}{2} = \frac{1}{2}||d_1||^2 = \frac{\sqrt{2}}{2}||d_1|| \cdot ||d_2|| = cos(45°)||d_1|| \cdot ||d_2||$. Finally, TDX searches along each search direction at each parent. The search step is a uniformly distributed random number in the interval [0, 1]. To keep population size invariant, only the better one of two individuals searched from each parent is retained, which makes TDX finds offspring along the better direction of two at each parent. The whole algorithm of TDX is shown in Algorithm 2.

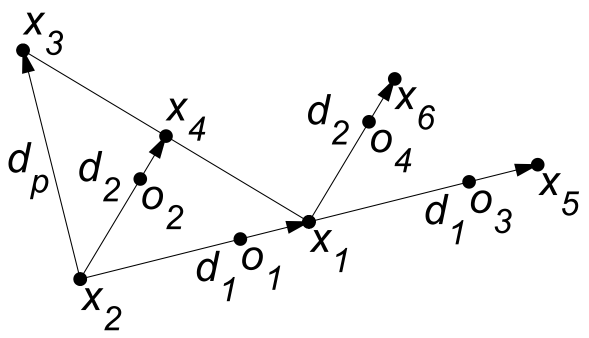


FIGURE 1. An example of TDX in a 2-dimensional plane. x_1 and x_2 are two parents and the function value of x_1 is better than x_2. $\overrightarrow{x_2x_1}, \overrightarrow{x_2x_4}, \overrightarrow{x_1x_5}$, and $\overrightarrow{x_1x_6}$ are four search directions, and o_1, o_2, o_3, and o_4 are four searched individuals. The better one between o_1, o_2 and the better one between o_3, o_4 are retained as offspring.

$$d_{pi} = \frac{d_{pi}||d_1||}{||d_p||}, \quad i = 1, 2, \ldots, n \tag{5}$$

Fig. 1 shows an example of TDX in a 2-dimensional plane. Given x_1 and x_2 as before. According to the method mentioned above, vector d_1, d_p, and d_2 is obtained, which are $\overrightarrow{x_2x_1}, \overrightarrow{x_2x_3}$, and $\overrightarrow{x_2x_4}$ respectively. Then, $\overrightarrow{x_1x_5}$ and $\overrightarrow{x_1x_6}$ are obtained by translating vectors $\overrightarrow{x_2x_1}$ and $\overrightarrow{x_2x_4}$ to start at x_1. Next, four offspring o_1, o_2, o_3, and o_4 are obtained by uniformly random finding a point in each segment of x_2x_1, x_2x_4, x_1x_5, and x_1x_6. Finally, the better one of o_1 and o_2 is retained as the first offspring, and the better one of o_3 and o_4 is retained as the second offspring.

TDX finds new points in the better direction of two in each iteration. The search region of TDX contains a line (d_1) and the surface of two cones ($\overrightarrow{x_2x_4}$ and $\overrightarrow{x_1x_6}$) whose apex angle is 90°, which makes it have a fast speed and avoid getting trapped in local optima. When the algorithm is iterated many times, TDX searches along the best of ∞ direction (Due to the bottom circle of a cone having infinite points, the number of vectors from the apex of a cone to a random point on the bottom circle is infinite).

1) FINDING A VECTOR PERPENDICULAR TO A GIVEN VECTOR

In linear algebra, if two vectors x and y are perpendicular, there must be a perpendicular equation $x \cdot y = \sum_{i=1}^n x_i y_i = 0$, which could be used to find a vector y perpendicular to a given vector x. Suppose $x, y \in \mathbb{R}^n$, an easy method to find y is to randomly generate y_1 to y_{n-1}. Then, use the perpendicular equation to calculate $y_n = -\frac{\sum_i^{n-1} x_i y_i}{x_n}$. However, if the denominator x_n is zero, the algorithm can not continue in an electronic computer. It is necessary to develop a robust method to calculate a vector perpendicular to a given vector.

It is obvious that if $x_n = 0$, $x \cdot y = \sum_{i=1}^n x_i y_i = \sum_{i=1}^{n-1} x_i y_i + x_n y_n = \sum_{i=1}^{n-1} x_i y_i + 0 = \sum_{i=1}^{n-1} x_i y_i$,

Algorithm 2 Algorithm of TDX

```
Require: Parents $x_1, x_2$ and fitness value of $x_1$ is smaller than $x_2$
Ensure: Offspring population $o$
    $d_1 = x_1 - x_2$
    $d_{pj} = \mathcal{U}(0,1), j = 1, 2, \ldots, n-1$
    $d_{pn} = \begin{cases} \frac{\sum_{j=1}^{n-1} d_{1j}d_{pj}}{d_{1n}}, & d_{1n} \neq 0 \\ \frac{\sum_{j=1}^{n-1} d_{1j}d_{pj}}{\mathcal{U}(-mean(d_{1n}), mean(d_{1n}))}, & else \end{cases}$
    $d_p = \frac{||d_1||}{||d_p||} d_p$
    $d_2 = (d_1 + d_p)/2$
    $\alpha = \mathcal{U}(0,1)$
    $o_1 = x_1 + \alpha d_1$
    $o_2 = x_1 + \alpha d_2$
    $o_3 = x_2 + \alpha d_1$
    $o_4 = x_2 + \alpha d_2$
    $o_5$ = the better of $o_1$ and $o_2$
    $o_6$ = the better of $o_3$ and $o_4$
    return offspring $o_5$ and $o_6$
```

the perpendicular equation becomes $\sum_i^{n-1} x_i y_i = 0$. Then, the objective is transformed into calculating vector $y^1 = (y_1, y_2, \ldots, y_{n-1})$ perpendicular to a given vector $x^1 = (x_1, x_2, \ldots, x_{n-1})$. Thus, we could use the above method to randomly generate y_1 to y_{n-2} and calculate y_{n-1}. However, if x_{n-1} to $x_{n-k}(n > k > 1)$ are zeros too, we have to transform the objective into calculating vector $y^{k+1} = (y_1, y_2, \ldots, y_{n-(k+1)})$ perpendicular to a given vector $x^{k+1} = (x_1, x_2, \ldots, x_{n-(k+1)})$. This recursive procedure has uncontrollable recursion times and computational time.

Although any $x_n = 0$ will stop the algorithm, the probability of $x_n = 0$ is very small. Therefore, We develop a fast algorithm to find a perpendicular vector, which has a certain computational time. Recall the perpendicular equation, if $x_n = 0$, the algorithm can not continue. Our purpose is to remove the risk of the appearance of ∞ and make the algorithm could continue. Therefore, we just replace the denominator x_n with a non-zero uniform distribution number in $[-mean, mean]$, where $mean$ is the mean value of x_n in the population. The mathematical form of this method is shown in (6). This method ensures that x_n and y_n are perpendicular and void recursive calculations.

$$y_j = \mathcal{U}(0,1), \quad j = 1, 2, \ldots, n-1$$
$$y_n = \begin{cases} \frac{\sum_{j=1}^{n-1} x_j y_j}{x_n}, & x_n \neq 0 \\ \frac{\sum_{j=1}^{n-1} x_j y_j)}{\mathcal{U}(-mean, mean)}, & else \end{cases} \tag{6}$$

C. GROUPED MUTATION

The sorted population is cut into two groups and two single mutation operators are absorbed into GM. The first group consists of the best group of individuals, which are mutated by a local search mutation operator. The second group

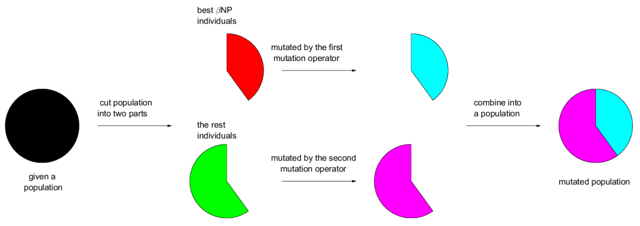

best βNP individuals
mutated by the first mutation operator
cut population into two parts
combine into a population
the rest individuals
mutated by the second mutation operator
given a population
mutated population

FIGURE 2. Flow chart of GM. In this paper, GM uses two single mutation operators. Each mutation operator mutates one group of the population and the two offspring populations are combined to the offspring population of GM.

consists of the rest individuals, which are mutated by a global mutation operator. Fig. 2 shows the process of GM and an algorithm of GM is shown in Algorithm 3.

Algorithm 3 Algorithm of GM

```
Require: populations $x$
Ensure: population $o$
    $x^s \leftarrow$ sort $x$, where the first individual is the best one
    cut $x_s$ into two groups, $x^1 = \{x_i^s | i = 1, 2, \ldots, \beta NP\}$, $x^2 = \{x_i^s | i = \beta NP + 1, \beta NP + 2, \ldots, NP\}$
    $o^1 \leftarrow$ mutate $x^1$ using (7)
    $o^2 \leftarrow$ mutate $x^2$ using (8)
    $o \leftarrow$ combine $o^1, o^2$ into one population
    return offspring population o
```

The first mutation operator is a normal mutation [21], whereas its variance is calculated by two individuals as shown in (7). The calculated variance ensures this normal mutation operator could adaptively search a small region.

$$o_{ij} = x_{ij} + \mathcal{N}(0,1) \times \frac{|x_{1j} - x_{(\beta NP)j}|}{6} \tag{7}$$

where o_{ij} and x_{ij} are $j - th$ variable of i_{th} individual in offspring population and parent population respectively, β is the proportion of the best group in the population and $X_{\beta NP}$ is the last individual in this group.

The second mutation operator is the non-uniform mutation, which is shown in (8), where $rand$ is a uniformly distributed random number generated within the interval $[0, 1]$, t is the current generation, T is the maximum number of generations and γ is a shape parameter that controls the speed at which the step length decreases. The search range of this mutation operator decreases adaptively with the increase of iteration times, so this operator can have a larger search range in the early stage and better convergence in the later stage.

$$o_{ij} = \begin{cases} (b_j - x_{ij}) \times \delta_i(t), if\ rand \leq 0.5 \\ (a_j - x_{ij}) \times \delta_i(t), else \end{cases}$$
$$\delta_i(t) = (1 - rand^{(1-\frac{t}{T})^\gamma}) \tag{8}$$

Individuals in the first group are the better group of the population and are considered to be the closer to the global optimum. Therefore, mutating them using the normal mutation operator to search the local region has the higher probability to find the global optimum. Individuals in the second group are the worse group of the population and are considered to be the farther from the global optimum. Therefore, mutating them using the non-uniform mutation operator to search in a large region has the higher probability to find the global optimum.

IV. EXPERIMENTAL RESULTS AND ANALYSIS

A. SYSTEM DETAILS

The proposed algorithm was implemented in MATLAB R2022a under Windows 10 with AMD 5700X CPU @ 2.3GHz with 32 GB of RAM.

B. BENCHMARK SUIT FUNCTIONS

Performance of GA-TDX and other methods are compared on solving the IEEE CEC 2017 competition on constrained real-parameter optimization [39] and ten real COPs from [31], [40]. The ten COPs are mathematical models extracted from real-world optimization problems. Appendix A shows details of these COPs.

C. PARAMETER SETTINGS AND ALGORITHMS COMPARISONS

Penalty parameter is set as $m = 10^{10}$. Population size is set as $NP = 100$. When the number of iterations of the algorithms reaches the maximum number of iterations T, the algorithm stops. For the IEEE CEC 2017 benchmark functions, T is $100D$, where D is the number of decision variables. As for ten COPs, T is 1000.

Eight state-of-the-art algorithms are adopted for comparison: GA-MPC [41], GA-LX [19], GA-DBX [20], GA-HNDX [21], GA-DEX [22], BOA [42], CS [43], OSCPSO [44]. The parameters of the eight state-of-the-art algorithms are the same as the values in the original paper where they are proposed. The initial population of all algorithms is the same, but the initial population of different runs is different.

D. PERFORMANCE METRICS

Each COP is solved 25 times by each algorithm. Since the theoretical optimal solution of COP is unknown, a non-negative difference between the optimal solution obtained by a specific method and the optimal solution obtained by all methods is used as the error of this method. To assess the performance of all contestant algorithms, the mean (denoted as $\overline{F}$) and standard deviation of the error are used. The mean indicates the accuracy of the search process of the algorithm, while the standard deviation is an indicator of how consistent the algorithm is when solving a certain optimization function.

To comprehensively compare the results of different algorithms for solving multiple functions, a rank value is used and the best algorithm will obtain the lowest rank value. For each problem, algorithm ranks are determined in terms of the mean values and median solutions at the maximum allowed number of evaluations, respectively. The total rank value of each algorithm is calculated as in (9), where $meanrank_i$ and $medianrank_i$ are the rank of mean and median of the ith functions respectively.

For a rigorous performance comparison between GA-TDX and the other algorithms, the Wilcoxon signed rank test [45] is used for pairwise comparison between the contestant algorithms at the significance level of $\alpha = 0.05$, and the results are reported as W^+, W^-, p, and l. The W^+, W^- symbols are used to indicate the sum of ranks where the performance of GA-TDX is superior or inferior to the other peers, respectively. The p-value is used as the minimum level of significance to detect a difference in performance between the algorithms. If the p-value is less than α, then this means that the result of the well-performing algorithm is statistically significant. From the statistical point of view, and using α and the p-value, the l value can be used to indicate if GA-TDX is significantly worse (l = "$-$"), insignificant (l = "$=$"), or significantly better (l = "$+$") than the contestant methods.

E. SENSITIVITY ANALYSIS

It is anticipated that some parameters like β and γ have an important impact on search efficiency. In this subsection, a set of parameter sensitivity tests are carried out to investigate the impact of these parameters on the performance of GA-TDX.

1) EFFECTS OF β

The value of β determines the size of two groups in GM, which may affect the effect of GM. The search performance of GA-TDX was investigated when β is set from 0 to 1 with an interval of 0.1 using the IEEE CEC 2017 benchmark functions at $D = 10, 30, 50$. In Fig. 3, different values of β behave differently in different functions, and it is difficult to conclude that the value of a certain value of β is superior to other values. The value of β has little impact on the performance of the algorithm. As shown in Tab. 1, although the gap is small, the β of 0.2 has the smallest rank value at all dimensions. This value is used in the following parts of this article.

$$rank\ value = \sum_{i=1}^{28}(meanrank_i + medianrank_i) \qquad (9)$$

2) EFFECTS OF γ

The value of γ affects the reduction rate of the search area of the non-uniform mutation operator as the number of iterations increases, which may affect the effect of GM. The search performance of GA-TDX was investigated when γ is set from 0 to 10 with an interval of 1 using the IEEE CEC 2017 benchmark functions at $D = 10, 30, 50$. In Fig. 4, different values of γ behave differently in different functions, and it is difficult to conclude that the value of a certain value of γ is superior to other values. The value of γ has

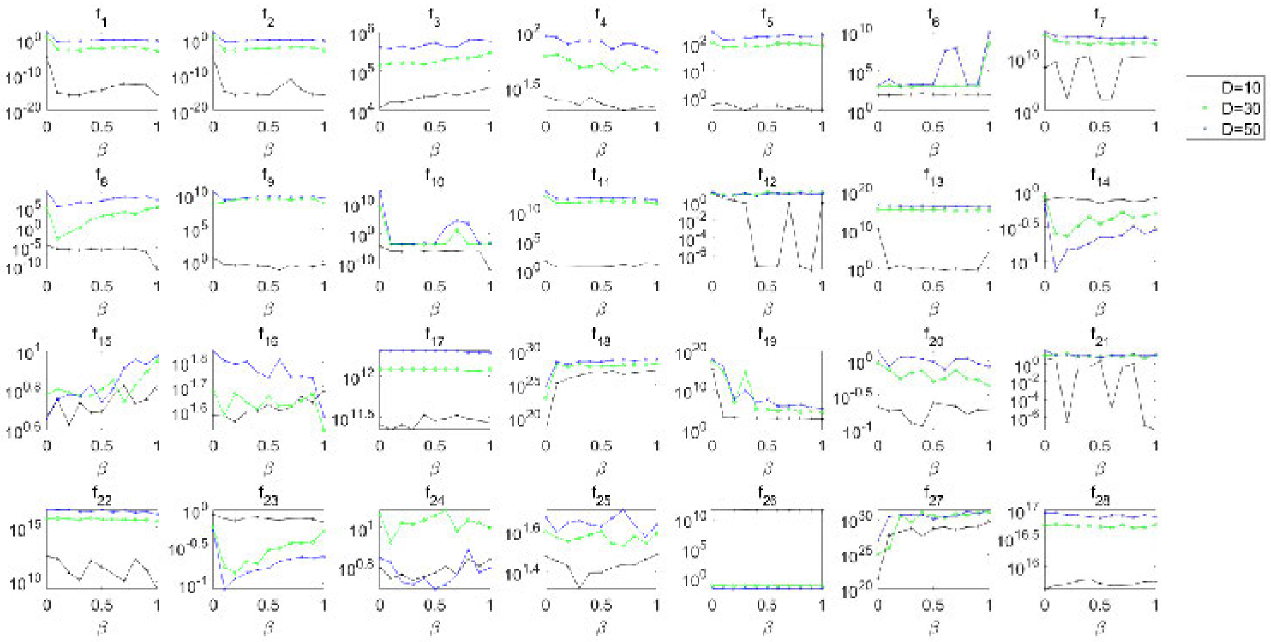


FIGURE 3. Performance of different values of β on CEC 2017 test functions at D = 10,30,50.

TABLE 1. Rank value of different values of β.

	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
D=10	422	318	**254**	282	335	306	345	382	358	340	354
D=30	451	314	**264**	276	311	343	337	363	388	351	298
D=50	473	262	**258**	300	312	326	343	403	392	320	307
sum	1346	894	**776**	858	958	975	1025	1148	1138	1011	959

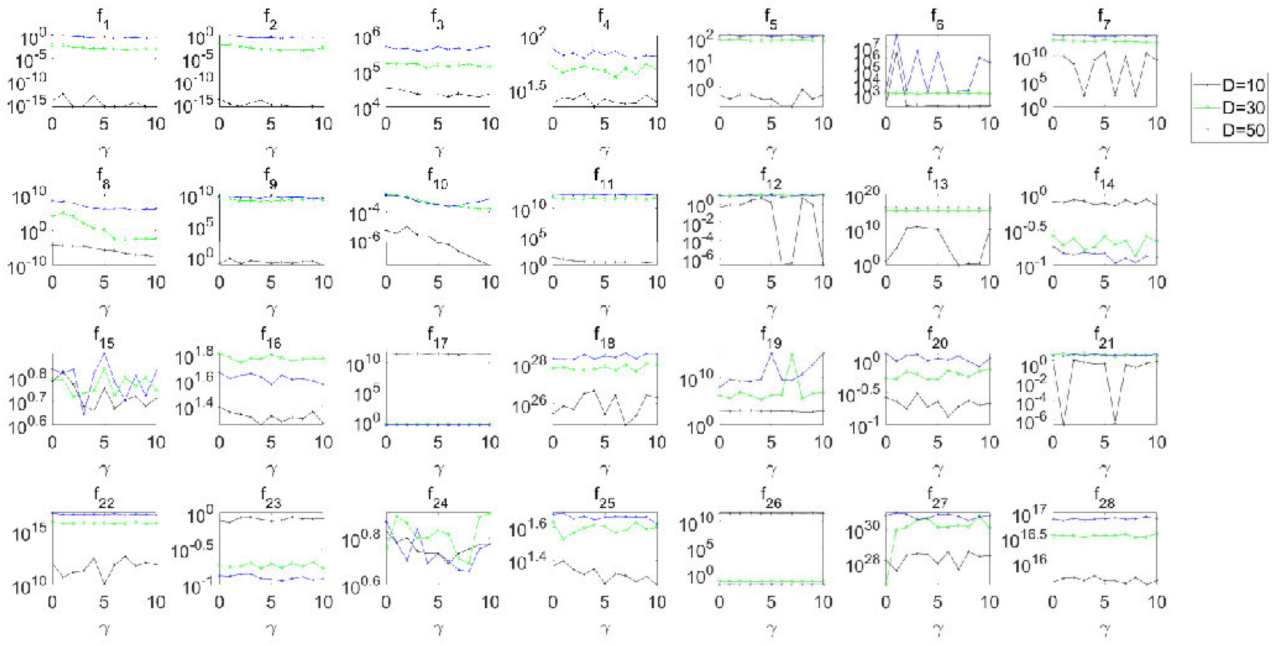


FIGURE 4. Performance of different values of γ on CEC 2017 test functions at D = 10, 30, 50.

little impact on the performance of the algorithm. As shown in Tab. 2, although the gap is small, the γ of 6 has the smallest rank value at all dimensions. This value is used in the following parts of this article.

TABLE 2. Rank value of different values of γ.

	0	1	2	3	4	5	6	7	8	9	10
D=10	428	426	404	414	305	311	**269**	275	284	290	290
D=30	460	386	382	310	307	385	**243**	286	264	338	335
D=50	446	391	392	343	292	322	**285**	301	287	294	343
sum	1334	1203	1178	1067	904	1018	**797**	862	835	922	968

TABLE 3. Performance comparison between GA-TDX and other selected algorithms on the IEEE CEC 2017 benchmark functions at D = 10.

	GA-TDX	GA-MPC	GA-LX	GA-DBX	GA-HNDX	GA-DEX	BOA	CS	OSCPSO
f_1	**2.00e-17**(5.23e-17)	3.23e-01(2.03e-01)	1.96e+03(8.05e+02)	5.98e+01(1.80e+01)	3.62e+03(5.40e+03)	4.34e-08(1.97e-08)	7.29e+04(3.09e+04)	8.07e+02(3.88e+02)	3.07e-04(4.97e-04)
f_2	**2.31e-16**(1.10e-15)	3.62e-01(2.16e-01)	2.08e+03(8.45e+02)	5.45e+01(1.72e+01)	8.11e+03(3.84e+04)	4.37e-08(2.49e-08)	8.25e+04(1.68e+04)	7.41e+02(4.34e+02)	4.99e-04(1.05e-03)
f_3	**1.67e+04**(1.20e+04)	9.33e+04(1.54e+05)	5.42e+05(4.13e+05)	7.60e+04(7.97e+04)	4.78e+09(2.39e+10)	6.98e+05(1.21e+06)	4.16e+11(3.29e+11)	1.08e+06(2.39e+06)	9.39e+04(1.29e+05)
f_4	2.81e+01(5.84e+00)	7.88e+01(7.18e+00)	1.02e+02(1.57e+01)	6.10e+01(7.71e+00)	3.83e+12(1.92e+13)	6.25e+01(1.32e+01)	1.05e+02(9.89e+00)	6.73e+01(1.01e+01)	**1.82e+01**(9.62e+00)
f_5	4.80e-01(1.32e+00)	6.96e+00(1.20e+00)	7.25e+08(3.62e+09)	2.88e+01(1.18e+01)	7.31e+18(3.66e+19)	**7.54e-08**(3.32e-08)	5.13e+13(1.44e+14)	5.12e+02(3.36e+02)	9.60e+00(1.62e+01)
f_6	**9.06e+01**(4.94e+01)	3.81e+11(1.50e+11)	1.68e+11(1.30e+11)	2.84e+10(1.12e+10)	7.03e+12(3.52e+13)	5.89e+10(3.10e+10)	1.96e+11(1.12e+11)	2.65e+11(1.98e+11)	4.24e+10(5.28e+10)
f_7	**9.77e+01**(5.91e+01)	3.98e+13(1.26e+13)	3.98e+10(1.21e+11)	6.99e+13(2.12e+13)	9.47e+14(4.85e+14)	1.14e+12(7.50e+11)	9.72e+14(5.06e+14)	1.17e+13(1.47e+13)	1.27e+04(5.26e+04)
f_8	**1.07e+02**(2.42e-06)	5.55e+07(6.76e+07)	2.60e+16(1.80e+16)	1.12e+11(2.03e+11)	4.51e+17(5.00e+17)	1.07e+02(7.67e-05)	2.14e+14(8.88e+13)	4.05e+15(4.35e+15)	1.07e+02(3.20e-03)
f_9	1.86e+01(3.19e-01)	9.13e+03(4.56e+04)	2.09e+10(2.52e+10)	3.29e+12(1.64e+13)	1.95e+01(4.67e+00)	**1.84e+01**(2.01e-06)	3.79e+14(5.92e+14)	1.28e+08(2.74e+08)	2.22e+01(3.79e+00)
f_{10}	**4.34e-07**(6.98e-07)	1.36e+09(1.56e+09)	1.48e+18(1.08e+18)	1.01e+13(1.55e+13)	8.22e+17(4.11e+18)	2.30e-04(5.35e-05)	2.13e+19(4.15e+18)	1.85e+16(1.71e+16)	2.79e-04(1.93e-04)
f_{11}	3.07e+00(2.27e+00)	5.79e+11(2.90e+12)	2.96e+15(1.26e+16)	3.05e+11(1.24e+12)	9.60e+16(4.80e+17)	**4.30e-05**(2.90e-05)	8.93e+22(3.86e+22)	8.81e+13(3.76e+14)	5.68e+09(7.25e+09)
f_{12}	**2.58e-07**(2.17e-07)	3.70e+01(6.30e+00)	2.29e+16(2.20e+16)	9.93e+09(4.16e+10)	9.27e+16(4.64e+17)	2.33e+01(8.22e+00)	3.48e+17(5.64e+16)	8.37e+14(8.29e+14)	2.05e-05(1.18e-05)
f_{13}	7.99e-01(1.63e+00)	2.23e+03(1.19e+03)	1.57e+16(1.09e+16)	2.13e+08(1.07e+09)	5.96e+16(2.98e+17)	**4.78e-01**(1.32e+00)	3.51e+17(4.68e+16)	4.92e+14(5.32e+14)	1.86e+01(3.21e+01)
f_{14}	**6.37e-01**(3.31e-01)	7.37e+06(1.54e+07)	5.15e+16(4.51e+16)	4.99e+11(1.15e+12)	1.43e+17(7.15e+17)	1.08e+01(4.30e+01)	6.11e+17(9.64e+16)	1.82e+15(1.93e+15)	1.01e+00(2.22e-01)
f_{15}	**4.67e+00**(2.57e+00)	2.25e+03(4.98e+03)	5.66e+15(7.51e+15)	3.47e+01(8.84e+01)	2.02e+07(1.01e+08)	1.00e+02(2.31e+02)	2.42e+17(4.54e+16)	7.17e+07(2.38e+08)	6.28e+00(3.27e+00)
f_{16}	**1.28e+01**(6.87e+00)	8.19e+02(1.19e+03)	3.70e+15(7.57e+15)	2.09e+01(9.91e+00)	4.97e+16(2.48e+17)	1.29e+02(4.10e+02)	2.41e+17(4.96e+16)	7.59e+06(8.70e+06)	2.66e+01(1.11e+01)
f_{17}	2.72e+11(1.90e+11)	3.60e+11(1.02e+11)	1.53e+16(1.16e+16)	2.54e+11(1.71e+11)	6.71e+16(3.35e+17)	**2.14e+11**(1.78e+11)	3.45e+17(5.65e+16)	6.38e+14(7.23e+14)	3.84e+11(8.00e+10)
f_{18}	1.24e+26(4.07e+26)	3.65e+27(1.79e+28)	1.15e+27(1.01e+27)	4.10e+26(8.06e+26)	1.07e+30(5.35e+30)	2.95e+27(7.02e+27)	4.44e+27(9.21e+26)	1.93e+25(4.45e+25)	**6.41e+23**(8.24e+23)
f_{19}	**1.28e+03**(9.02e+02)	8.30e+15(3.25e+15)	1.53e+16(9.13e+14)	1.00e+16(1.03e+15)	2.04e+16(1.02e+15)	1.54e+11(3.14e+10)	1.89e+16(9.07e+14)	1.12e+16(2.42e+15)	4.82e+15(3.52e+15)
f_{20}	**2.35e-01**(1.07e-01)	1.31e+00(1.98e-01)	1.39e+00(2.45e-01)	1.32e+00(2.43e-01)	2.78e-01(6.71e-01)	1.64e+00(1.78e-01)	3.19e+00(4.39e-01)	1.48e+00(2.29e-01)	1.21e+00(2.39e-01)
f_{21}	**4.25e-01**(2.12e+00)	4.10e+01(1.00e+01)	3.28e+17(2.30e+17)	5.76e+12(1.85e+13)	1.16e+18(5.80e+18)	2.34e+01(7.28e+00)	2.93e+18(5.61e+17)	1.12e+16(1.08e+16)	3.23e+00(6.19e+00)
f_{22}	2.29e+10(9.31e+10)	2.85e+03(1.99e+03)	2.42e+17(1.74e+17)	8.34e+09(3.37e+10)	4.52e+17(2.26e+18)	**6.39e-01**(1.49e+00)	2.96e+18(4.82e+17)	1.05e+16(1.04e+16)	8.14e+10(2.11e+11)
f_{23}	**7.19e-01**(2.57e-01)	5.85e+08(1.11e+09)	5.38e+17(4.64e+17)	3.04e+12(5.89e+12)	1.60e+18(8.02e+18)	2.04e+02(8.41e+02)	5.72e+18(9.36e+17)	1.94e+16(1.92e+16)	8.28e-01(2.20e-01)
f_{24}	**8.04e+00**(2.04e+00)	2.32e+04(3.79e+04)	2.40e+17(2.79e+17)	5.50e+01(1.81e+02)	4.62e+17(2.31e+18)	1.92e+02(3.68e+02)	2.61e+18(4.58e+17)	1.50e+15(3.85e+15)	1.21e+01(3.10e+00)
f_{25}	**1.68e+01**(8.28e+00)	3.63e+03(4.96e+03)	1.87e+17(1.82e+17)	3.52e+01(2.91e+01)	3.84e+17(1.92e+18)	5.54e+01(8.49e+01)	2.61e+18(5.11e+17)	2.15e+15(5.68e+15)	3.34e+01(9.87e+00)
f_{26}	3.52e+11(1.33e+11)	4.00e+11(6.95e+06)	3.16e+17(2.24e+17)	3.07e+11(1.74e+11)	4.57e+17(2.29e+18)	**2.33e+11**(1.79e+11)	2.94e+18(4.54e+17)	9.91e+15(1.26e+16)	3.68e+11(1.11e+11)
f_{27}	2.58e+28(9.61e+28)	3.75e+30(1.87e+31)	1.02e+29(7.87e+28)	2.68e+28(5.12e+28)	1.13e+32(5.65e+32)	4.44e+29(1.20e+30)	8.72e+29(1.84e+29)	3.37e+27(1.62e+28)	**5.06e+23**(5.33e+23)
f_{28}	4.46e+15(2.51e+15)	1.80e+16(2.14e+15)	1.74e+16(1.11e+15)	1.39e+16(1.46e+15)	2.19e+16(8.27e+14)	**3.22e+11**(8.17e+10)	1.99e+16(7.11e+14)	1.78e+16(1.26e+15)	1.37e+16(2.22e+15)
rank value	81	282	393	253	325	186	484	346	170

TABLE 4. Performance comparison between GA-TDX and other selected algorithms on the IEEE CEC 2017 benchmark functions at D = 30.

	GA-TDX	GA-MPC	GA-LX	GA-DBX	GA-HNDX	GA-DEX	BOA	CS	OSCPSO
f_1	**1.02e-03**(1.27e-03)	2.05e+03(5.92e+02)	1.51e+04(2.80e+03)	4.37e+03(1.11e+03)	7.42e+04(4.22e+04)	8.02e-02(1.71e-01)	3.67e+05(7.17e+03)	4.11e+04(1.47e+04)	3.68e+02(2.53e+02)
f_2	**5.47e-04**(7.07e-04)	1.93e+03(7.39e+02)	1.64e+04(2.92e+03)	4.54e+03(7.25e+02)	8.46e+04(8.46e+04)	8.50e-02(1.17e-01)	3.67e+05(6.61e+03)	3.17e+04(1.40e+04)	3.29e+02(1.65e+02)
f_3	**1.28e+05**(6.03e+04)	8.14e+05(4.44e+05)	1.74e+07(2.44e+07)	2.36e+05(1.87e+05)	1.25e+09(4.39e+09)	5.28e+06(1.78e+07)	8.93e+13(1.43e+13)	3.99e+06(3.96e+06)	6.50e+05(5.81e+05)
f_4	1.70e+02(2.06e+01)	4.06e+02(2.23e+01)	3.66e+02(5.47e+01)	3.24e+02(1.86e+01)	**1.20e+01**(1.40e+01)	1.90e+02(6.29e+01)	4.25e+02(1.38e+01)	7.09e+02(4.17e+01)	1.73e+02(2.39e+01)
f_5	7.65e+01(3.29e+01)	1.72e+03(5.77e+02)	1.97e+15(3.12e+15)	3.64e+03(1.77e+03)	4.67e+01(3.05e+01)	**2.39e+01**(1.07e+00)	2.93e+05(4.98e+04)	1.84e+16(6.56e+15)	1.93e+02(9.20e+01)
f_6	**9.68e+02**(5.63e+02)	6.00e+12(1.92e+12)	2.00e+11(1.79e+11)	5.53e+11(2.44e+11)	1.16e+13(6.30e+12)	9.66e+11(3.80e+11)	6.84e+11(3.38e+11)	1.08e+12(4.66e+11)	3.38e+11(2.09e+11)
f_7	7.86e+12(1.15e+13)	7.36e+14(1.14e+14)	**6.09e+12**(3.04e+13)	2.00e+15(6.22e+14)	2.09e+16(8.34e+15)	3.52e+15(2.74e+15)	5.40e+16(1.10e+16)	3.41e+15(7.26e+14)	2.11e+15(4.77e+15)
f_8	**1.01e+02**(1.52e-03)	1.37e+15(1.84e+15)	2.19e+18(1.67e+18)	3.17e+17(1.93e+17)	3.90e+19(3.53e+19)	1.35e+04(3.29e+04)	7.95e+15(3.84e+15)	8.94e+18(2.11e+18)	1.61e+16(1.97e+16)
f_9	1.47e+09(2.31e+09)	7.39e+12(7.12e+12)	7.32e+14(1.60e+15)	2.15e+13(1.31e+13)	**8.04e-01**(1.08e+00)	4.98e+00(2.16e+00)	9.22e+17(5.39e+17)	2.86e+16(2.23e+16)	5.03e+09(7.48e+09)
f_{10}	**1.47e+02**(9.70e-05)	6.92e+16(3.26e+16)	3.34e+19(1.80e+19)	3.31e+18(2.01e+18)	4.45e+18(2.22e+19)	1.63e+07(4.10e+07)	2.49e+22(3.76e+20)	1.75e+20(2.68e+19)	9.09e+13(2.85e+14)
f_{11}	4.76e+11(2.80e+11)	1.80e+14(1.25e+14)	9.00e+15(3.20e+15)	3.99e+14(1.41e+14)	1.68e+16(1.53e+16)	**8.19e+04**(4.06e+05)	3.67e+16(1.90e+15)	1.18e+16(8.10e+15)	1.77e+11(1.98e+11)
f_{12}	8.95e+00(9.36e+00)	1.54e+15(7.38e+14)	1.07e+17(5.35e+16)	1.42e+16(1.07e+16)	1.69e+18(1.34e+18)	1.33e+01(1.11e+01)	3.43e+18(3.60e+16)	2.23e+18(2.84e+17)	**2.69e+00**(4.27e+00)
f_{13}	1.80e+15(1.30e+15)	3.40e+15(1.65e+15)	1.49e+17(9.11e+16)	2.34e+16(1.29e+16)	1.48e+18(1.37e+18)	**2.93e+13**(4.47e+13)	3.52e+18(2.48e+16)	2.31e+18(3.35e+17)	5.30e+13(5.67e+13)
f_{14}	**1.91e-01**(1.82e-01)	3.80e+15(2.33e+15)	2.62e+17(1.10e+17)	2.87e+16(2.64e+16)	3.15e+18(2.63e+18)	7.00e-01(6.50e-02)	6.57e+18(5.16e+16)	3.95e+18(5.63e+17)	6.78e-01(1.07e-01)
f_{15}	5.28e+00(2.97e+00)	2.54e+05(4.40e+05)	9.92e+15(1.84e+16)	**2.99e+00**(2.25e+00)	2.31e+01(8.11e+00)	2.50e+02(7.10e+02)	2.42e+18(1.75e+16)	1.47e+18(2.69e+17)	1.08e+01(4.72e+00)
f_{16}	7.10e+01(1.22e+01)	7.97e+04(2.07e+05)	1.21e+16(1.70e+16)	**6.31e+01**(1.31e+01)	6.69e+17(8.90e+17)	1.06e+02(1.69e+02)	2.42e+18(2.34e+16)	1.47e+18(1.94e+17)	7.33e+01(1.59e+01)
f_{17}	**1.56e-01**(1.67e-02)	9.47e+14(6.07e+14)	1.47e+17(7.12e+16)	1.00e+16(1.06e+16)	1.30e+18(1.27e+18)	1.97e+05(4.46e+05)	3.39e+18(2.72e+16)	2.20e+18(3.36e+17)	1.60e-01(4.57e-02)
f_{18}	4.94e+27(9.71e+27)	1.13e+28(3.77e+27)	2.10e+28(4.24e+27)	2.11e+28(5.01e+27)	2.34e+28(6.29e+27)	2.50e+28(7.00e+27)	3.18e+28(6.81e+26)	1.65e+28(4.81e+27)	**5.60e+25**(2.71e+25)
f_{19}	**2.24e+06**(8.75e+06)	1.98e+17(1.29e+16)	1.69e+17(9.92e+15)	1.91e+17(5.94e+15)	2.31e+17(4.14e+15)	3.31e+16(1.30e+16)	2.23e+17(3.77e+15)	2.14e+17(3.77e+15)	1.41e+17(1.71e+16)
f_{20}	**7.67e-01**(4.47e-01)	7.41e+00(4.51e-01)	5.97e+00(7.39e-01)	7.25e+00(4.49e-01)	3.75e+00(4.03e+00)	8.21e+00(5.10e-01)	1.21e+01(7.56e-01)	7.74e+00(3.79e-01)	7.20e+00(4.89e-01)
f_{21}	**5.96e+00**(5.39e+00)	3.07e+16(1.82e+16)	2.22e+18(1.56e+18)	2.57e+17(1.73e+17)	5.33e+19(2.34e+19)	1.40e+01(1.07e+01)	9.21e+19(6.95e+17)	4.65e+19(1.05e+19)	2.72e+10(8.60e+10)
f_{22}	5.90e+15(4.65e+15)	3.20e+16(1.61e+16)	1.97e+18(9.69e+17)	2.74e+17(1.81e+17)	5.96e+19(2.26e+19)	**7.00e+13**(5.34e+13)	9.25e+19(7.70e+17)	4.54e+19(8.23e+18)	1.08e+16(9.96e+15)
f_{23}	**1.20e-01**(6.76e-02)	6.27e+16(3.85e+16)	4.09e+18(2.77e+18)	4.26e+17(3.05e+17)	8.94e+19(4.73e+19)	6.08e-01(8.31e-02)	1.73e+20(1.25e+18)	8.51e+19(1.50e+19)	4.12e+11(1.09e+12)
f_{24}	7.43e+00(3.00e+00)	3.92e+07(1.33e+08)	1.16e+18(1.19e+18)	8.81e+15(2.24e+16)	4.83e+19(2.54e+19)	4.71e+02(1.14e+03)	8.67e+19(6.22e+17)	4.11e+19(9.97e+18)	**7.30e+00**(3.45e+00)
f_{25}	**4.48e+01**(1.25e+01)	4.10e+07(1.21e+08)	9.99e+17(6.44e+17)	2.13e+16(3.85e+16)	4.41e+19(2.54e+19)	6.41e+01(7.85e+01)	8.65e+19(8.23e+17)	4.21e+19(6.89e+18)	6.76e+01(1.11e+01)
f_{26}	**4.49e-02**(1.02e-02)	2.91e+16(2.67e+16)	1.97e+18(1.27e+18)	2.59e+17(1.64e+17)	5.47e+19(2.51e+19)	1.10e+05(1.69e+05)	9.20e+19(7.94e+17)	4.42e+19(7.96e+18)	4.88e-02(4.92e-03)
f_{27}	1.40e+30(6.98e+30)	1.05e+31(4.89e+30)	2.36e+31(1.53e+31)	3.68e+31(2.09e+31)	4.09e+31(1.56e+31)	3.64e+31(1.45e+31)	6.51e+31(1.25e+30)	1.94e+31(1.01e+31)	**4.64e+29**(2.10e+30)
f_{28}	1.17e+17(1.22e+16)	1.80e+17(2.45e+15)	1.68e+17(3.05e+15)	1.63e+17(5.54e+15)	1.85e+17(3.15e+15)	**3.55e+16**(1.79e+16)	1.77e+17(2.68e+15)	1.73e+17(2.45e+15)	1.57e+17(7.45e+15)
rank value	96	261	319	259	371	183	474	400	157

F. ANALYSIS OF RESULTS USING THE IEEE CEC 2017 BENCHMARK FUNCTIONS

The simulation results from solving the IEEE CEC 2017 benchmark problems at $D = 10, 30, 50$, obtained using the proposed GA-TDX algorithm and other selected algorithms are presented in Tab. 3, 4, 5 respectively. The best results are marked in boldface. The standard deviation is shown in parentheses. It can also be noted that performance comparisons between GA-TDX and its different peers are summarized in the last row of the tables using rank value.

Tab. 3, 4, 5 demonstrated the superiority of the proposed GA-TDX algorithm among the other algorithms with its ability to achieve the best $\overline{F}$ over 17, 14, 16 functions out of the 28 selected benchmark functions. The proposed GA-TDX algorithm has the lowest rank value, suggesting this algorithm outperforms other algorithms in solving these benchmark functions at D = 10, 30, 50.

TABLE 5. Performance comparison between GA-TDX and other selected algorithms on the IEEE CEC 2017 benchmark functions at D = 50.

	GA-TDX	GA-MPC	GA-LX	GA-DBX	GA-HNDX	GA-DEX	BOA	CS	OSCPSO
f_1	**2.62e-01**(1.44e-01)	2.26e+04(6.70e+03)	4.56e+04(1.03e+04)	2.13e+04(3.55e+03)	3.26e+05(6.15e+04)	6.80e+01(4.26e+01)	5.04e+05(5.67e+03)	1.17e+05(2.19e+04)	5.28e+03(1.91e+03)
f_2	**2.08e-01**(1.45e-01)	1.90e+04(6.59e+03)	4.69e+04(1.17e+04)	1.42e+04(1.51e+03)	3.33e+05(4.31e+04)	6.03e+01(2.71e+01)	5.05e+05(4.47e+03)	1.88e+05(5.28e+04)	4.15e+03(9.66e+02)
f_3	**2.31e+05**(1.47e+05)	1.65e+06(9.66e+05)	2.13e+07(2.62e+07)	1.34e+06(6.79e+05)	8.88e+07(3.71e+08)	4.45e+06(3.29e+06)	1.13e+07(1.03e+07)	6.58e+06(7.99e+06)	1.18e+06(7.18e+05)
f_4	3.18e+02(4.35e+01)	9.51e+02(2.75e+02)	5.48e+02(6.80e+01)	6.39e+02(2.96e+01)	**1.58e+01**(1.06e+01)	2.68e+02(3.74e+01)	7.66e+02(1.73e+01)	1.41e+03(6.41e+01)	3.53e+02(5.76e+01)
f_5	1.31e+02(4.58e+01)	4.87e+04(7.30e+04)	8.31e+15(9.45e+15)	3.13e+04(1.30e+04)	9.45e+01(5.54e+01)	**7.05e+01**(3.87e+01)	9.86e+11(4.93e+12)	1.02e+17(2.48e+16)	2.60e+03(2.44e+03)
f_6	**1.30e+03**(8.46e+02)	1.06e+13(3.47e+12)	1.28e+11(9.61e+10)	2.06e+12(8.23e+11)	2.32e+13(1.15e+13)	1.96e+12(6.38e+11)	7.59e+11(4.46e+11)	1.31e+12(4.33e+11)	7.68e+11(2.73e+11)
f_7	8.27e+13(4.93e+13)	1.47e+15(2.80e+14)	**6.61e+12**(2.96e+13)	6.73e+15(1.63e+15)	7.11e+16(2.50e+16)	4.67e+16(1.25e+16)	2.04e+17(2.79e+16)	1.30e+16(2.31e+15)	5.62e+16(5.41e+16)
f_8	**3.53e+05**(6.00e+05)	1.41e+18(3.84e+18)	9.18e+18(5.12e+18)	1.48e+19(5.60e+18)	2.28e+20(3.04e+20)	7.52e+08(1.42e+09)	6.28e+16(4.38e+16)	6.15e+19(1.32e+19)	4.76e+17(2.21e+17)
f_9	2.50e+09(1.97e+09)	8.08e+13(5.23e+13)	1.04e+15(1.67e+15)	2.16e+14(1.45e+14)	**1.04e+00**(3.10e-01)	5.45e+01(1.97e+02)	7.72e+18(2.65e+18)	2.36e+18(9.60e+17)	7.62e+12(1.33e+13)
f_{10}	**1.45e+02**(1.30e-04)	5.78e+18(2.16e+18)	1.17e+20(4.15e+19)	1.20e+20(3.37e+19)	6.22e+19(3.00e+20)	5.96e+13(7.87e+13)	1.04e+23(1.52e+21)	9.47e+20(1.83e+20)	3.81e+16(4.33e+16)
f_{11}	2.11e+12(1.29e+12)	1.36e+15(6.78e+14)	2.84e+16(1.12e+16)	3.86e+15(8.24e+14)	8.88e+16(6.80e+15)	**1.59e+10**(3.91e+10)	1.02e+17(3.60e+15)	7.79e+16(1.76e+16)	5.17e+12(5.48e+12)
f_{12}	**6.03e+00**(8.02e+00)	1.79e+17(5.24e+16)	3.93e+17(1.64e+17)	3.08e+17(1.43e+17)	9.13e+18(4.05e+17)	7.89e+00(8.70e+00)	1.02e+19(5.84e+16)	8.07e+18(7.21e+17)	1.30e+01(1.07e+01)
f_{13}	1.56e+16(7.06e+15)	2.22e+17(9.11e+16)	4.08e+17(1.50e+17)	3.43e+17(1.33e+17)	9.26e+18(9.07e+17)	**2.00e+14**(3.13e+14)	1.05e+19(6.89e+16)	8.31e+18(6.15e+17)	3.54e+15(2.35e+15)
f_{14}	**9.90e-02**(6.38e-02)	5.09e+17(3.66e+17)	9.50e+17(4.40e+17)	5.37e+17(2.39e+17)	1.80e+19(4.99e+17)	5.97e-01(2.22e-02)	1.99e+19(1.01e+17)	1.56e+19(1.38e+18)	5.37e-01(9.89e-02)
f_{15}	**5.28e+00**(3.48e+00)	3.13e+15(1.14e+16)	4.94e+16(6.31e+16)	2.41e+15(1.20e+16)	2.90e+01(3.43e+00)	5.18e+01(1.15e+02)	7.27e+18(3.64e+16)	5.49e+18(5.80e+17)	1.21e+01(5.71e+00)
f_{16}	5.14e+01(1.33e+01)	1.60e+16(5.96e+16)	3.60e+16(5.20e+16)	1.43e+15(7.16e+15)	6.42e+18(3.15e+17)	1.09e+02(2.81e+02)	7.27e+18(3.58e+16)	5.58e+18(5.51e+17)	**3.59e+01**(2.32e+01)
f_{17}	1.80e-01(4.58e-03)	1.95e+17(9.86e+16)	4.10e+17(1.95e+17)	2.39e+17(1.21e+17)	9.07e+18(5.67e+17)	6.61e+04(1.11e+05)	1.01e+19(6.44e+16)	7.95e+18(5.90e+17)	**1.76e-01**(4.29e-02)
f_{18}	1.80e+28(3.42e+28)	5.88e+28(1.16e+28)	8.76e+28(1.00e+28)	8.21e+28(1.72e+28)	9.39e+28(8.12e+27)	9.35e+28(8.68e+27)	9.93e+28(1.16e+27)	7.15e+28(1.28e+28)	**1.09e+27**(2.75e+27)
f_{19}	**3.29e+09**(9.69e+09)	6.41e+17(1.98e+16)	4.80e+17(4.17e+16)	5.95e+17(1.42e+16)	6.73e+17(6.32e+15)	2.47e+17(3.66e+16)	6.57e+17(5.82e+15)	6.37e+17(7.63e+15)	4.79e+17(2.98e+16)
f_{20}	**1.12e+00**(6.22e-01)	1.47e+01(6.49e-01)	9.91e+00(1.18e+00)	1.46e+01(4.89e-01)	1.26e+01(4.87e+00)	1.57e+01(7.27e-01)	2.02e+01(5.92e-01)	1.48e+01(5.81e-01)	1.40e+01(1.08e+00)
f_{21}	**5.94e+00**(8.05e+00)	3.39e+18(1.83e+18)	6.49e+18(3.87e+18)	3.20e+18(1.43e+18)	1.24e+20(1.44e+19)	5.71e+01(4.37e+01)	1.44e+20(9.10e+17)	1.04e+20(1.32e+19)	5.24e+12(7.29e+12)
f_{22}	3.47e+16(1.86e+16)	3.81e+18(2.69e+18)	6.00e+18(3.05e+18)	3.33e+18(1.40e+18)	1.22e+20(1.67e+19)	**4.62e+14**(3.80e+14)	1.45e+20(7.30e+17)	1.05e+20(1.05e+19)	8.09e+16(3.31e+16)
f_{23}	**7.56e-02**(5.19e-02)	6.13e+18(3.17e+18)	1.12e+19(4.68e+18)	5.56e+18(1.83e+18)	2.46e+20(1.26e+19)	1.97e+10(5.40e+10)	2.79e+20(1.72e+18)	1.96e+20(2.66e+19)	1.01e+13(1.30e+13)
f_{24}	**5.15e+00**(2.85e+00)	1.76e+18(1.34e+18)	3.49e+18(1.93e+18)	1.22e+18(1.02e+18)	1.10e+20(2.03e+19)	3.67e+01(1.58e+02)	1.32e+20(9.63e+17)	9.35e+19(9.76e+18)	6.28e+00(3.27e+00)
f_{25}	**6.80e+01**(1.31e+01)	1.99e+18(1.68e+18)	4.24e+18(2.19e+18)	1.48e+18(1.35e+18)	1.14e+20(5.81e+18)	1.57e+02(2.69e+02)	1.32e+20(9.07e+17)	9.53e+19(1.15e+19)	7.73e+01(2.13e+01)
f_{26}	**7.81e-03**(2.87e-03)	3.17e+18(1.82e+18)	5.51e+18(2.69e+18)	3.44e+18(1.61e+18)	1.23e+20(1.13e+19)	1.84e+04(4.46e+04)	1.44e+20(8.27e+17)	1.01e+20(1.06e+19)	7.81e-03(2.39e-03)
f_{27}	1.19e+30(5.95e+30)	2.81e+31(9.47e+30)	3.74e+31(1.88e+31)	6.92e+31(1.77e+31)	7.12e+31(1.89e+31)	6.00e+31(1.95e+31)	8.63e+31(1.67e+30)	3.55e+31(9.74e+30)	**5.12e+28**(1.63e+29)
f_{28}	2.49e+17(2.31e+16)	3.64e+17(5.43e+15)	3.28e+17(1.14e+16)	3.36e+17(7.90e+15)	3.74e+17(4.68e+15)	**9.16e+16**(5.71e+16)	3.54e+17(5.53e+15)	3.49e+17(4.43e+15)	3.26e+17(1.39e+16)
rank value	93	277	309	276	386	166	463	384	166

TABLE 6. Wilcoxon signed rank test for the pairwise comparison between GA-TDX and other selected algorithms on the IEEE CEC 2017 benchmark functions at D = 10.

	W+	W-	p-value	l-value
GA-MPC	387.00	19.00	1.47e-05	+
GA-LX	406.00	0.00	2.00e-06	+
GA-DBX	363.00	43.00	1.41e-04	+
GA-HNDX	406.00	0.00	2.00e-06	+
GA-DEX	289.00	117.00	2.58e-02	+
BOA	406.00	0.00	2.00e-06	+
CS	351.00	55.00	3.91e-04	+
OSCPSO	337.00	69.00	1.18e-03	+

TABLE 7. Wilcoxon signed rank test for the pairwise comparison between GA-TDX and other selected algorithms on the IEEE CEC 2017 benchmark functions at D = 30.

	W+	W-	p-value	l-value
GA-MPC	406.00	0.00	2.00e-06	+
GA-LX	399.00	7.00	4.26e-06	+
GA-DBX	402.00	4.00	3.09e-06	+
GA-HNDX	391.00	15.00	9.79e-06	+
GA-DEX	284.00	122.00	3.34e-02	+
BOA	406.00	0.00	2.00e-06	+
CS	406.00	0.00	2.00e-06	+
OSCPSO	259.00	92.00	1.75e-02	+

The pairwise comparison between the proposed algorithm and its peer algorithms is conducted using Wilcoxon signed rank test and is summarized in Tab. 6, 7, 8 in terms of $W+$, $W-$, p, and l values, as was stated in Section IV-D. The l-values show significant improvement of GA-TDX over the other eight competent algorithms, shown as a "+" sign.

G. ANALYSIS OF RESULTS USING TEN COPs

Tab. 9 shows the simulation results from solving ten COPs. The proposed GA-TDX achieve the best $\overline{F}$ over 6 COPs out of 10 and has the lowest rank value, suggesting this algorithm outperforms other algorithms in solving these COPs.

TABLE 8. Wilcoxon signed rank test for the pairwise comparison between GA-TDX and other selected algorithms on the IEEE CEC 2017 benchmark functions at D = 50.

	W+	W-	p-value	l-value
GA-MPC	406.00	0.00	2.00e-06	+
GA-LX	399.00	7.00	4.26e-06	+
GA-DBX	406.00	0.00	2.00e-06	+
GA-HNDX	391.00	15.00	9.79e-06	+
GA-DEX	283.00	123.00	3.51e-02	+
BOA	406.00	0.00	2.00e-06	+
CS	406.00	0.00	2.00e-06	+
OSCPSO	275.00	76.00	5.96e-03	+

Tab. 10 shows the pairwise comparison between the proposed algorithm and its peer algorithms, conducted using Wilcoxon signed rank test. The l-values show a significant improvement of GA-TDX over seven competent algorithms, shown as a "+" sign, and an insignificant improvement of GA-TDX over GA-TEX, shown as a "=" sign.

V. OPTIMIZATION DESIGN OF SINGLE-STAGE CYLINDRICAL GEAR REDUCER

A gear reducer is a widely used transmission device, used to reduce speed and increase torque. Constrained by the strength of the material, the size of the gears and shafts of the reducer must be large enough to carry the corresponding loads. However, the larger the size of the gears and shafts, the higher the cost. Therefore, the design of a reducer is to find the smallest shaft and gear size while satisfying the constraints.

In this paper, we optimize a single-stage cylindrical gear reducer. The structural sketch of the reducer is shown in Fig. 5. The gear ratio is $u = 5$, input power is $P = 58$ kW, speed of driving gear is $n_1 = 1000$ r/min, allowable stress of gear is $[\sigma]_H = 550$ MPa, allowable bending stress is $[\sigma]_f = 400$ MPa. The purpose is to find the design parameters that minimize the volume of the reducer under the condition

TABLE 9. Performance comparison between GA-TDX and other selected algorithms on 10 COPs.

	GA-TDX	GA-MPC	GA-LX	GA-DBX	GA-HNDX	GA-DEX	BOA	CS	OSCPSO
f_{29}	**8.02e-06**(1.64e-05)	2.89e+00(7.06e-01)	7.90e+08(3.43e+09)	6.54e+09(5.54e+09)	8.67e-01(1.16e+00)	3.93e-01(1.26e-01)	1.54e+01(2.59e-01)	2.79e+09(4.65e+09)	9.68e+00(1.50e+00)
f_{30}	**0.00e+00**(7.43e-13)	1.30e-04(1.20e-04)	3.09e+02(9.65e+01)	1.66e+02(4.23e+01)	1.73e+01(2.10e+01)	1.08e-04(1.51e-04)	8.14e+02(2.18e+02)	4.33e+01(2.63e+01)	5.32e+01(1.52e+01)
f_{31}	2.73e-02(3.20e-02)	5.30e+00(1.30e+00)	8.46e+12(1.10e+13)	6.76e+09(2.04e+10)	1.44e+00(1.01e+00)	**2.24e-03**(7.99e-04)	7.52e+13(4.86e+13)	2.29e+02(2.12e+02)	7.33e+00(3.11e+00)
f_{32}	**4.53e-06**(2.19e-05)	6.97e-03(3.36e-03)	3.90e+02(1.83e+02)	1.82e+00(6.00e-01)	2.83e-02(1.26e-02)	6.31e-06(2.33e-06)	1.65e+02(4.70e+01)	5.90e+00(4.24e+00)	1.55e-01(7.78e-02)
f_{33}	2.78e+01(4.80e+01)	8.58e+02(2.87e+02)	2.84e+06(8.40e+06)	5.24e+03(1.61e+03)	1.94e+10(9.70e+09)	**1.25e+00**(2.44e+00)	2.99e+09(1.80e+09)	3.37e+03(1.28e+03)	9.59e+02(5.53e+02)
f_{34}	**3.73e-09**(2.52e-09)	1.38e+02(2.65e+02)	3.61e+05(1.20e+05)	7.88e+04(3.99e+04)	1.56e+19(7.76e+19)	2.40e+01(7.51e-01)	5.42e+09(1.80e+10)	2.49e+04(2.10e+04)	8.53e+04(3.20e+04)
f_{35}	**1.35e-04**(2.35e-04)	8.67e+00(3.85e+00)	2.09e+09(2.96e+09)	5.19e+02(1.37e+03)	1.25e+02(5.63e+01)	1.60e-03(6.77e-04)	5.26e+10(3.87e+10)	1.05e+08(4.24e+08)	1.60e+07(8.01e+07)
f_{36}	5.66e-05(1.89e-04)	4.19e-02(1.08e-02)	3.19e+08(3.48e+08)	5.83e+06(3.37e+06)	1.99e+05(4.12e+05)	**6.53e-06**(1.95e-06)	1.51e+10(8.93e+09)	8.54e+06(6.41e+06)	1.46e+06(1.75e+06)
f_{37}	1.68e-02(4.67e-02)	5.74e-02(3.47e-02)	1.97e+12(1.55e+12)	5.96e-02(1.84e-02)	9.64e-02(1.28e-01)	**1.78e-04**(1.24e-04)	5.26e-01(1.14e-01)	8.48e+09(1.53e+10)	2.02e-01(1.31e-01)
f_{38}	**2.03e+01**(1.41e+01)	6.87e+01(1.47e+01)	1.84e+10(3.56e+10)	1.70e+11(2.02e+11)	1.38e+13(5.90e+12)	2.22e+01(7.98e+00)	5.78e+12(2.31e+12)	6.16e+11(4.96e+11)	7.76e+01(2.17e+01)
rank value	27	65	151	122	104	34	164	129	104

TABLE 10. Wilcoxon signed rank test for the pairwise comparison between GA-TDX and other selected algorithms on ten COPs.

	W+	W-	p-value	l-value
GA-MPC	55.00	0.00	9.77e-04	+
GA-LX	55.00	0.00	9.77e-04	+
GA-DBX	55.00	0.00	9.77e-04	+
GA-HNDX	55.00	0.00	9.77e-04	+
GA-DEX	32.00	23.00	3.48e-01	=
BOA	55.00	0.00	9.77e-04	+
CS	55.00	0.00	9.77e-04	+
OSCPSO	55.00	0.00	9.77e-04	+

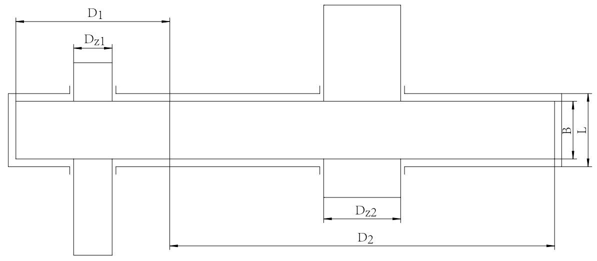


FIGURE 5. Schematic of a single-stage cylindrical gear reducer. D_1 is the diameter of the small gear, D_{z1} is the diameter of the output axle, D_2 is the diameter of the big gear, D_{z2} is the diameter of the input axle, B is the depth of gears, L is the thickness of the shell.

of strength and stiffness. Since the parts inside the shell, i.e. gears and axles, are the basis for determining the volume of the reducer, the objective function (shown in (10)) can be established according to the principle of minimizing the sum of their volumes, where the first two terms are the volumes of the two gears, and the latter term is the sum of the volumes of the two shafts.

$$\begin{aligned} V &= 0.25\pi B(D_1^2 - D_{z1}^2) + 0.25\pi B(D_2^2 - D_{z2}^2) \\ &\quad + 0.25\pi L(D_{z1}^2 + D_{z2}^2) \\ &= 0.25\pi[M^2Z_1^2B - D_{z1}^2B + M^2Z_1^2u^2B - D_{z2}^2B \\ &\quad + D_{z1}^2L + D_{z2}^2L] \\ &= 0.25\pi[26M^2Z_1^2B + (L-B)(D_{z1}^2 + D_{z2}^2)] \qquad (10) \end{aligned}$$

The volume of the reducer depends on six parameters: modulus M, number of teeth of the first gear Z_1, the thickness of gears B, the thickness of the gearbox L, and the diameter of the first shaft D_{z1} and the second shaft D_{z2}. Let design variable $x = [x_1, x_2, x_3, x_4, x_5, x_6]^T = [M, Z_1, B, L, D_{z1}, D_{z2}]^T$. The objective function is $minf_{39}(x) = V$, and constraints are shown in (11), where g_1 is the lower limit constraint on the number of teeth, g_2 and g_3 are the lower and upper limit constraints on the tooth width factor, g_4 is the lower limit constraint on the modulus, g_5 is the upper limit constraint on the diameter of the first gear, g_6 and g_7 are the lower and upper limit constraints on the diameter of the first axis, g_8 and g_9 are the lower and upper limits of the diameter of the second shaft, g_{10} is the lower limit constraint of the box thickness, g_{11} is the contact stress constraint of the first gear, g_{12} and g_{13} are the bending stress constraints of the two gears, g_{14} and g_{15} are the bending stress constraints of the two shafts.

$$\begin{aligned} g_1(x) &= z_{min} - Z_1 = 17 - x_2 \leq 0 \\ g_2(x) &= \psi_{min} - B/Z_1M = 0.9 - x_3/(x_1x_2) \leq 0 \\ g_3(x) &= B/Z_1M - \psi max = x_3/(x_1x_2) - 1.4 \leq 0 \\ g_4(x) &= 2{-}M = 2 - x_1 \leq 0 \\ g_5(x) &= Z_1M - D_{1max} = x_1x_2 - 300 \leq 0 \\ g_6(x) &= D_{z1min} - D_{z1} = 100 - x_5 \leq 0 \\ g_7(x) &= D_{z1} - D_{z1max} = x_5 - 150 \leq 0 \\ g_8(x) &= D_{z2min} - D_{z2} = 130 - x_6 \leq 0 \\ g_9(x) &= D_{z2} - D_{z2max} = x_6 - 200 \leq 0 \\ g_{10}(x) &= B + 0.5D_{z2} + 40{-}L = x_3 + 0.5x_6 - x_4 + 40 \leq 0 \\ g_{11}(x) &= \sigma_H - [\sigma]_H = 1486250/(x_1x_2\sqrt{x_3}) - 550 \leq 0 \\ g_{12}(x) &= \sigma_{F1} - [\sigma]_F = 9064860y_{11}y_{12}/(x_1^2x_2^2x_3) - 400 \leq 0 \\ g_{13}(x) &= \sigma_{F2} - [\sigma]_F = 9064860y_{21}y_{22}/(x_1^2x_2^2x_3) - 400 \leq 0 \\ g_{14}(x) &= \sigma_{w1} - [\sigma]_w = \frac{1}{x_5^3}\sqrt{(\frac{2.85\times 10^6x_4}{x_1x_2})^2 + 2.4\times 10^{12}} \\ &\quad - 5.5 \leq 0 \\ g_{15}(x) &= \sigma_{w2} - [\sigma]_w = \frac{1}{x_6^3}\sqrt{(\frac{2.85\times 10^6x_4}{x_1x_2})^2 + 6\times 10^{13}} \\ &\quad - 5.5 \leq 0 \qquad (11) \end{aligned}$$

When solving this problem, we increase the value of m to 10^{200} to avoid the phenomenon that the previous value will result in an unreasonable negative objective function value. Other parameters are the same as before. Tab. 11 shows the simulation results from optimizing this reducer. The proposed GA-TDX achieve the best $\overline{F}$, suggesting this algorithm outperforms other algorithms in optimizing this reducer. Tab. 12 shows the pairwise comparison between the proposed algorithm and its peer algorithms, conducted using Wilcoxon signed rank test. The l-values show significant improvement

TABLE 11. Performance comparison between GA-TDX and other selected algorithms on a single-stage cylindrical gear reducer.

	GA-TDX	GA-MPC	GA-LX	GA-DBX	GA-HNDX	GA-DEX	BOA	CS	OSCPSO
f_{39}	**3.37e+04**(2.15e+04)	2.12e+05(1.04e+05)	1.27e+07(1.15e+07)	9.99e+204(Inf)	6.37e+212(Inf)	4.25e+04(8.82e+03)	4.30e+206(Inf)	6.20e+05(2.73e+05)	2.06e+06(1.36e+06)

TABLE 12. Wilcoxon signed rank test for the pairwise comparison between GA-TDX and other selected algorithms a single-stage cylindrical gear reducer.

	W+	W-	p-value	l-value
GA-MPC	325.00	0.00	6.54e-06	+
GA-LX	325.00	0.00	6.54e-06	+
GA-DBX	325.00	0.00	6.54e-06	+
GA-HNDX	325.00	0.00	6.54e-06	+
GA-DEX	246.00	79.00	1.28e-02	+
BOA	325.00	0.00	6.54e-06	+
CS	325.00	0.00	6.54e-06	+
OSCPSO	325.00	0.00	6.54e-06	+

of GA-TDX over other competent algorithms, shown as a "+" sign. The GA-TDX algorithm obtained an optimal function value of 4.3815×10^6, with the parameters set as 0.6003, 77.2057, 41.7126, 146.7126, 118.5073, 130.0000.

VI. CONCLUSION

In this paper, a novel class of improved genetic algorithms is introduced to solve constrained optimization problems. The two-direction crossover utilizes the orientation information of paired individuals in the current population and adds an additional search direction. Searching for an individual in both directions and keeping the better individual improves the search efficiency. The grouped mutation introduces two mutation operators with different properties and performs different mutation operations on different groups of the population. Grouped mutation takes full advantage of the characteristics of each mutation operator and improves the search efficiency.

The performance of the proposed algorithms was tested on a challenging set of 28 benchmark problems taken from the IEEE CEC 2017 competition on constrained real-parameter optimization and ten real constrained optimization problems. The simulation results affirm the fact that the proposed algorithm significantly outperforms other improved genetic algorithms as well as state-of-the-art evolutionary algorithms. Finally, the optimization results for a single-stage cylindrical gear reducer show that the proposed algorithm can significantly reduce the size and cost of the gearbox.

APPENDIX A
MATHEMATICAL MODEL OF TEN COPs

It is difficult to find the precise optimal solution of COPs used in this paper, due to the objective and constraints are very complicated. The optimal solution shown below is not the exact optimal solution, but a reference value. Details of ten COPs are shown as follow:

COP1. There are 13 variables and 9 constraints in this COP. The best solution is $x^* = (1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3, 1)$ and its function value is $f(x^*) = -15$. 6 constraints are active at x^*.

$$\begin{aligned}
min\, f_{29}(x) &= 5\sum_{i=1}^{4}(x_i - x_i^2) - \sum_{i=5}^{13} x_i \\
s.t.\ & 2x_1 + 2x_2 + x_{10} + x_{11} \le 10, \\
& 2x_1 + 2x_3 + x_{10} + x_{12} \le 10, \\
& 2x_2 + 2x_3 + x_{11} + x_{12} \le 10, \\
& -8x_1 + x_{10} \le 0, \\
& -8x_2 + x_{11} \le 0, \\
& -8x_3 + x_{12} \le 0, \\
& -2x_4 - x_5 + x_{10} \le 0, \\
& -2x_6 - x_7 + x_{11} \le 0, \\
& -2x_8 - x_9 + x_{12} \le 0, \\
& 0 \le x_i \le 1, \quad i = 1, 2, \ldots, 9, \\
& 0 \le x_i \le 100, \quad i = 10, 11, 12, \\
& 0 \le x_{13} \le 1
\end{aligned}$$

COP2. There are 5 variables and 6 constraints in this COP. The best solution is $x^* = (78, 33, 27.071, 45, 44.9692)$ and its function value is $f(x^*) = -31025.37961619$. 2 constraints are active at x^*.

$$\begin{aligned}
min\, f_{30}(x) &= 5.3578547x_3^2 + 0.8356891x_1x_5 \\
&\quad + 37.293239x_1 - 40792.141 \\
s.t.\ & 0 \le 85.334407 + 0.0056858x_2x_5 + 0.00026x_1x_4 \\
&\quad + 0.0022053x_3x_5 \le 92, \\
& 90 \le 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 \\
&\quad + 0.0021813x_3^2 \le 110, \\
& 20 \le 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 \\
&\quad + 0.0019085x_3x_4 \le 25, \\
& 78 \le x_1 \le 102, \\
& 33 \le x_2 \le 45, \\
& 27 \le x_i \le 45, \quad i = 3, 4, 5,
\end{aligned}$$

COP3. There are ten variables and eight constraints in this COP. The best solution is $x^* = (2.171996, 2.363683, 8.773926, 5.095984, 0.9906548, 1.430574, 1.321644, 9.828726, 8.280092, 8.375927)$ and its function value is $f(x^*) = 24.3062091$. 6 constraints are active at x^*.

$$\begin{aligned}
min\, f_{31}(x) &= x_1^2 + x_2^2 + x_1x_2 - 14x_1 - 16x_2 + (x_3 - 10)^2 \\
&\quad + 4(x_4 - 5)^2 + (x_5 - 3)^2 + 2(x_6 - 1)^2 + 5x_7^2 \\
&\quad + 7(x_8 - 11)^2 + 2(x_9 - 10)^2 + (x_{10} - 7)^2 + 45
\end{aligned}$$

$$
\begin{aligned}
s.t.\ & 105 - 4x_1 - 5x_2 + 3x_7 - 9x_8 \geq 0,\\
& -3(x_1-2)^2 - 4(x_2-3)^2 - 2x_3^2 + 7x_4 + 120 \geq 0,\\
& -10x_1 + 8x_2 + 17x_7 - 2x_8 \geq 0,\\
& -x_1^2 - 2(x_2-2)^2 + 2x_1x_2 - 14x_5 + 6x_6 \geq 0,\\
& 8x_1 - 2x_2 - 5x_9 + 2x_{10} + 12 \geq 0,\\
& -5x_1^2 - 8x_2 - (x_3-6)^2 + 2x_4 + 40 \geq 0,\\
& 3x_1 - 6x_2 - 12(x_9-8)^2 + 7x_{10} \geq 0,\\
& -0.5(x_1-8)^2 - 2(x_2-4)^2 - 3x_5^2 + x_6 + 30 \geq 0,\\
& -10 \leq x_i \leq 10, \quad i = 1, \ldots, 10
\end{aligned}
$$

COP4. There are 7 variables and 4 constraints in this COP. The best solution is $x^* = (2.330499, 1.951372, -0.4775414, 4.365726, -0.6244870, 1.038131, 1.594227)$ and its function value is $f(x^*) = 680.6300573$. 2 constraints are active at x^*.

$$
\begin{aligned}
min\, f_{32}(x) = & (x_1-10)^2 + 5(x_2-12)^2 + x_3^4 + 3(x_4-11)^2\\
& + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - 10x_6 - 8x_7\\
s.t.\ & 127 - 2x_1^2 - 3x_2^4 - x_3 - 4x_4^2 - 5x_5 \geq 0,\\
& 196 - 23x_1 - x_2^2 - 6x_6^2 + 8x_7 \geq 0,\\
& 282 - 7x_1 - 3x_2 - 10x_3^2 - x_4 + x_5 \geq 0,\\
& -4x_1^2 - x_2^2 + 3x_1x_2 - 2x_3^2 - 5x_6 + 11x_7 \geq 0,\\
& -10 \leq x_i \leq 10, \quad i = 1, \ldots, 7
\end{aligned}
$$

COP5. There are 8 variables and 6 constraints in this COP. The best solution is $x^* = (579.3167, 1359.943, 5110.071, 182.0174, 295.5985, 217.9799, 286.4162, 395.5979)$ and its function value is $f(x^*) = 7049.3307$. 3 constraints are active at x^*.

$$
\begin{aligned}
min\, f_{33}(x) = & x_1 + x_2 + x_3\\
s.t.\ & 1 - 0.0025(x_4 + x_6) \geq 0,\\
& 1 - 0.0025(x_5 + x_7 - x_4) \geq 0,\\
& 1 - 0.01(x_8 - x_5) \geq 0,\\
& x_1x_6 - 833.33252x_4 - 100x_1 + 83333.333 \geq 0,\\
& x_2x_7 - 1250x_5 - x_2x_4 + 1250x_4 \geq 0,\\
& x_3x_8 - 1250000 - x_3x_5 + 2500x_5 \geq 0,\\
& 100 \leq x_1 \leq 10000,\\
& 1000 \leq x_i \leq 10000, i = 2, 3,\\
& 10 \leq x_i \leq 1000, i = 4, \ldots, 8
\end{aligned}
$$

COP6. There are 5 variables and 6 constraints in this COP. The best solution is $x^* = (4.53743097, 2.4, 60, 9.3, 7)$ and its function value is $f(x^*) = -5280335.12777216$. 1 constraint is active at x^*.

$$
\begin{aligned}
min\, f_{34}(x) = & -24345 + 8720288.849x_1\\
& - 150512.5253x_1x_2 + 156.6950325x_1x_3\\
& - 476470.3222x_1x_4 - 729482.8271x_1x_5,\\
s.t.\ & 294000 \geq -145421.402x_1 + 2931.1506x_1x_2\\
& - 40.427932x_1x_3 + 5106.192x_1x_4\\
& + 15711.36x_1x_5 \geq 0,\\
& 294000 \geq -155011.1084x_1 + 4360.53352x_1x_2\\
& + 12.9492344x_1x_3 + 10236.884x_1x_4\\
& + 13176.786x_1x_5 \geq 0,\\
& 277200 \geq -326669.5104x_1 + 7390.68412x_1x_2\\
& - 27.8986976x_1x_3 + 16643.076x_1x_4\\
& + 30988.146x_1x_5 \geq 0,\\
& 0 \leq x_1 \leq 1000,\\
& 1.2 \leq x_2 \leq 2.4,\\
& 20 \leq x_3 \leq 60,\\
& 9 \leq x_4 \leq 9.3,\\
& 6.5 \leq x_5 \leq 7
\end{aligned}
$$

COP7. There are 7 variables and 6 constraints in this COP. The best solution is $x^* = (5.17123, 0.786669, 2.636062, 3.78776, 0.766223, 1.001394, 0.021073)$ and its function value is $f(x^*) = 559.29739702$. 3 constraints are active at x^*.

$$
\begin{aligned}
min\, f_{35}(x) = & 10x_1x_2^{-1}x_4^2x_6^{-3}x_7^{0.5} + 15x_1^{-1}x_2^{-2}x_3x_4x_5^{-1}x_7^{0.5}\\
& + 20x_1^{-2}x_2x_4^{-1}x_5^{-2}x_6 + 25x_1^2x_2^2x_3^{-1}x_5^{0.5}x_6^{-2}x_7\\
s.t.\ & 1 - 0.5x_1^{0.5}x_3^{-1}x_6^{-2}x_7 - 0.7x_1^3x_2x_3^{-2}x_6x_7^{0.5}\\
& - 0.2x_2^{-1}x_3x_4^{-0.5}x_6^{2/3}x_7^{1/4} \geq 0,\\
& 1 - 1.3x_1^{-0.5}x_2x_3^{-1}x_5^{-1}x_6 - 0.8x_3x_4^{-1}x_5^{-1}x_6^2\\
& - 3.1x_1^{-1}x_2^{0.5}x_4^{-2}x_5^{-1}x_6^{1/3} \geq 0,\\
& 1 - 2x_1x_3^{-1.5}x_5x_6^{-1}x_7^{1/3} - 0.1x_2x_3^{-0.5}x_5x_6^{-1}x_7^{-0.5}\\
& - x_1^{-1}x_2x_3^{0.5}x_5 - 0.65x_2^{-2}x_3x_5x_6^{-1}x_7 \geq 0,\\
& 1 - 0.2x_1^{-2}x_2x_4^{-1}x_5^{0.5}x_7^{1/3} - 0.3x_1^{0.5}x_2^2x_3x_4^{1/3}x_7^{1/4}x_5^{-2/3}\\
& - 0.4x_1^{-3}x_2^{-2}x_3x_5x_7^{3/4} - 0.5x_3^{-2}x_4x_7^{0.5} \geq 0,\\
& 100 \leq f(x) \leq 3000,\\
& 0.1 \leq x_i \leq 10, \quad i = 1, \ldots, 6,\\
& 0.01 \leq x_7 \leq 10
\end{aligned}
$$

COP8. There are 8 variables and 6 constraints in this COP. The best solution is $x^* = (6.465114, 2.232709, 0.6673975, 0.5957564, 5.932676, 5.527235, 1.013322, 0.4006682)$ and its function value is $f(x^*) = 3.95117606$. 4 constraints are active at x^*.

$$
\begin{aligned}
min\, f_{36}(x) = & 0.4x_1^{0.67}x_7^{-0.67} + 0.4x_2^{0.67}x_8^{-0.67} + 10 - x_1 - x_2\\
s.t.\ & 1 - 0.0588x_5x_7 - 0.1x_1 \geq 0,\\
& 1 - 0.0588x_6x_8 - 0.1x_1 - 0.1x_2 \geq 0,\\
& 1 - 4x_3x_5^{-1} - 2x_3^{-0.71}x_5^{-1} - 0.0588x_3^{-1.3}x_7 \geq 0,\\
& 1 - 4x_4x_6^{-1} - 2x_4^{-0.71}x_6^{-1} - 0.0588x_4^{-1.3}x_8 \geq 0,\\
& 1 \leq f(x) \leq 4.2,\\
& 0.1 \leq x_i \leq 10, \quad i = 1, \ldots, 8
\end{aligned}
$$

COP9. There are 9 variables and 13 constraints in this COP. The best solution is $x^* = (0.8841292, 0.4672425, 0.03742076, 0.9992996, 0.8841292, 0.4672424, 0.03742076,$

0.9992996, 26e-19) and its function value is $f(x^*) = -0.86600353$. 8 constraints are active at x^*.

$$\begin{aligned}
&min\, f_{37}(x) = -0.5(x_1x_4 - x_2x_3 + x_3x_9 + x_5x_8 - x_6x_7)\\
&s.t.\ 1 - x_9^2 \geq 0,\\
&\quad 1 - x_3^2 - x_4^2 \geq 0,\\
&\quad 1 - x_5^2 - x_6^2 \geq 0,\\
&\quad 1 - (x_1 - x_5)^2 - (x_2 - x_6)^2 \geq 0,\\
&\quad 1 - (x_1 - x_7)^2 - (x_2 - x_8)^2 \geq 0,\\
&\quad 1 - (x_3 - x_5)^2 - (x_4 - x_6)^2 \geq 0,\\
&\quad 1 - (x_3 - x_7)^2 - (x_4 - x_8)^2 \geq 0,\\
&\quad 1 - x_1^2 - (x_2 - x_9)^2 \geq 0,\\
&\quad 1 - x_7^2 - (x_8 - x_9)^2 \geq 0,\\
&\quad x_3x_9 \geq 0,\\
&\quad - x_5x_9 \geq 0,\\
&\quad x_5x_8 - x_6x_7 \geq 0,\\
&\quad x_1x_4 - x_2x_3 \geq 0,\\
&\quad x_9 \geq 0
\end{aligned}$$

COP10. There are 15 variables and 29 constraints in this COP. The best solution is $x^* = (8, 49, 3, 1, 49, 0, 0, 63, 7, 0, 77, 8, 0, 91, 9)$ and its function value is $f(x^*) = 641.73505000$. 9 constraints are active at x^*.

$$\begin{aligned}
&min\, f_{38}(x) = \sum_{k=0}^{4}(2.3x_{3k+1} + 0.0001x_{3k+1}^2 + 1.7x_{3k+2}\\
&\quad + 0.0001x_{3k+2}^2 + 2.2x_{3k+3} + 0.00015x_{3k+3}^2)\\
&s.t.\ 0 \leq x_{3j+1} - x_{3j-2} + 7 \leq 13, \quad j = 1, \ldots, 4,\\
&\quad 0 \leq x_{3j+2} - x_{3j-1} + 7 \leq 14, \quad j = 1, \ldots, 4,\\
&\quad 0 \leq x_{3j+3} - x_{3j} + 7 \leq 13, \quad j = 1, \ldots, 4,\\
&\quad x_1 + x_2 + x_3 - 60 \geq 0,\\
&\quad x_4 + x_5 + x_6 - 50 \geq 0,\\
&\quad x_7 + x_8 + x_9 - 70 \geq 0,\\
&\quad x_{10} + x_{11} + x_{12} - 85 \geq 0,\\
&\quad x_{13} + x_{14} + x_{15} - 100 \geq 0,\\
&\quad 8 \leq x_1 \leq 21,\\
&\quad 43 \leq x_2 \leq 57,\\
&\quad 3 \leq x_3 \leq 16,\\
&\quad 0 \leq x_{3k+1} \leq 90, \quad k = 1, \ldots, 4,\\
&\quad 0 \leq x_{3k+2} \leq 120, \quad k = 1, \ldots, 4,\\
&\quad 0 \leq x_{3k+3} \leq 60, \quad k = 1, \ldots, 4
\end{aligned}$$

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their valuable and constructive comments that greatly helped improve the quality and completeness of this article.

REFERENCES

[1] M. Kalikatzarakis, A. Coraddu, L. Oneto, and D. Anguita, "Optimizing fuel consumption in thrust allocation for marine dynamic positioning systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 1, pp. 122–142, Jan. 2022.

[2] W. Sun, G. Tang, and K. Hauser, "Fast UAV trajectory optimization using bilevel optimization with analytical gradients," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 2010–2024, Dec. 2021.

[3] J. Wang, X. Song, Y. Le, W. Wu, B. Zhou, and X. Ning, "Design of self-shielded uniform magnetic field coil via modified pigeon-inspired optimization in miniature atomic sensors," *IEEE Sensors J.*, vol. 21, no. 1, pp. 315–324, Jan. 2021.

[4] G. M. Cappello, G. Colajanni, P. Daniele, and D. Sciacca, "A constrained optimization model for the provision of services in a 5G network with multi-level cybersecurity investments," *Soft Comput.*, pp. 1–18, May 2022.

[5] G. B. Dantzig, *Linear Programming and Extensions*. Princeton, NJ, USA: Princeton Univ. Press, 1963.

[6] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York, NY, USA: Springer, 1999.

[7] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.

[8] P. Vasant, T. Ganesan, and I. Elamvazuthi, "An improved PSO approach for solving non-convex optimization problems," in *Proc. 9th Int. Conf. ICT Knowl. Eng.*, Jan. 2012, pp. 80–87.

[9] G. D'Angelo and F. Palmieri, "GGA: A modified genetic algorithm with gradient-based local search for solving constrained optimization problems," *Inf. Sci.*, vol. 547, pp. 136–162, Feb. 2021.

[10] R. S. Al-Gharaibeh, M. Z. Ali, M. I. Daoud, R. Alazrai, H. Abdel-Nabi, S. Hriez, and P. N. Suganthan, "Real-parameter constrained optimization using enhanced quality-based cultural algorithm with novel influence and selection schemes," *Inf. Sci.*, vol. 576, pp. 242–273, Oct. 2021.

[11] H.-Y. Tseng, P.-H. Chu, H.-C. Lu, and M.-J. Tsai, "Easy particle swarm optimization for nonlinear constrained optimization problems," *IEEE Access*, vol. 9, pp. 124757–124767, 2021.

[12] A. Kumar, S. Das, and R. Mallipeddi, "A reference vector-based simplified covariance matrix adaptation evolution strategy for constrained global optimization," *IEEE Trans. Cybern.*, vol. 52, no. 5, pp. 3696–3709, May 2022.

[13] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Syst.*, vol. 9, p. 115–148, Apr. 1995.

[14] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*. Berlin, Germany: Springer, 2015.

[15] D. Dumitrescu, B. Lazzerini, L. C. Jain, and A. Dumitrescu, *Evolutionary Computation*. Boca Raton, FL, USA: CRC Press, 2000.

[16] D. Ortiz-Boyer, C. Hervás-Martínez, and N. García-Pedrajas, "Improving crossover operator for real-coded genetic algorithms using virtual parents," *J. Heuristics*, vol. 13, no. 3, pp. 265–314, Jun. 2007.

[17] L. J. Eshelman and J. D. Schaffer, *Real-Coded Genetic Algorithms and Interval-Schemata*, vol. 2. Amsterdam, The Netherlands: Elsevier, 1993, pp. 187–202.

[18] M. Takahashi and H. Kita, "A crossover operator using independent component analysis for real-coded genetic algorithms," in *Proc. Congr. Evol. Comput.*, vol. 1, 2001, pp. 643–649.

[19] K. Deep and M. Thakur, "A new crossover operator for real coded genetic algorithms," *Appl. Math. Comput.*, vol. 188, no. 1, pp. 895–911, 2007.

[20] Y.-C. Chuang, C.-T. Chen, and C. Hwang, "A simple and efficient real-coded genetic algorithm for constrained optimization," *Appl. Soft Comput.*, vol. 38, pp. 87–105, Jan. 2016.

[21] J. Wang, Z. Cheng, O. K. Ersoy, P. Zhang, W. Dai, and Z. Dong, "Improvement analysis and application of real-coded genetic algorithm for solving constrained optimization problems," *Math. Problems Eng.*, vol. 2018, pp. 1–16, Jun. 2018.

[22] M. Z. Ali, N. H. Awad, P. N. Suganthan, A. M. Shatnawi, and R. G. Reynolds, "An improved class of real-coded genetic algorithms for numerical optimization," *Neurocomputing*, vol. 275, pp. 155–166, Jan. 2018.

[23] H. Mühlenbein and D. Schlierkamp-Voosen, "Predictive models for the breeder genetic algorithm I. continuous parameter optimization," *Evol. Comput.*, vol. 1, no. 1, pp. 25–49, 1993.

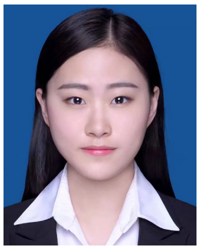
[24] R. Hinterding, "Gaussian mutation and self-adaption for numeric genetic algorithms," in *Proc. IEEE Int. Conf. Evol. Comput.*, vol. 1, Nov./Dec. 1995, p. 384.

[25] K. Deep and M. Thakur, "A new mutation operator for real coded genetic algorithms," *Appl. Math. Comput.*, vol. 193, no. 1, pp. 211–230, Oct. 2007.

[26] M. V. Pathan, S. Patsias, and V. L. Tagarielli, "A real-coded genetic algorithm for optimizing the damping response of composite laminates," *Comput. Struct.*, vol. 198, pp. 51–60, Mar. 2018.
[27] R. A. E. Mäkinen, J. Périaux, and J. Toivanen, "Multidisciplinary shape optimization in aerodynamics and electromagnetics using genetic algorithms," *Int. J. Numer. Methods Fluids*, vol. 30, no. 2, pp. 149–159, May 1999.
[28] A. K. Das and D. K. Pratihar, "A direction-based exponential mutation operator for real-coded genetic algorithm," in *Proc. 5th Int. Conf. Emerg. Appl. Inf. Technol. (EAIT)*, Jan. 2018, pp. 1–4.
[29] W. Gao, Z. Wei, Y. Luo, and J. Cao, "Artificial bee colony algorithm based on Parzen window method," *Appl. Soft Comput.*, vol. 74, pp. 679–692, Jan. 2019.
[30] Y. Song and F. Yan, "An improved mutation operator which can improve the performance of genetic algorithm," in *Proc. 2nd Int. Conf. Artif. Intell. Comput. Eng. (ICAICE)*, Nov. 2021, pp. 85–88.
[31] Z. Michalewicz and M. Schoenauer, "Evolutionary algorithms for constrained parameter optimization problems," *Evol. Comput.*, vol. 4, no. 1, pp. 1–32, 1996.
[32] K. Jong-Hwan and M. Hyun, "Evolutionary programming techniques for constrained optimization problems," *IEEE Trans. Evol. Comput.*, vol. 1, no. 2, pp. 129–140, Jul. 1997.
[33] B. Tessema and G. G. Yen, "A self adaptive penalty function based algorithm for constrained optimization," in *Proc. IEEE Int. Conf. Evol. Comput.*, Jul. 2006, pp. 246–253.
[34] A. Kumar, S. Das, A. K. Misra, and D. Singh, "A υ-constrained matrix adaptation evolution strategy with Broyden-based mutation for constrained optimization," *IEEE Trans. Cybern.*, vol. 52, no. 6, pp. 4784–4796, Jun. 2022.
[35] K. Zielinski and R. Laur, "Constrained single-objective optimization using differential evolution," in *Proc. IEEE Int. Conf. Evol. Comput.*, Jul. 2006, pp. 223–230.
[36] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 4, no. 3, pp. 284–294, Sep. 2000.
[37] X. Wen, G. Wu, M. Fan, R. Wang, and P. N. Suganthan, "Voting-mechanism based ensemble constraint handling technique for real-world single-objective constrained optimization," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2020, pp. 1–8.
[38] C. A. C. Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art," *Comput. Methods Appl. Mech. Eng.*, vol. 191, no. 11, pp. 1245–1287, Jan. 2002.
[39] G. Wu, R. Mallipeddi, and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2017 competition on constrained real-parameter optimization," Nanyang Technol. Univ., Singapore, Tech. Rep., 2017.
[40] W. Hock and K. Schittkowski, *The Test Problems*. Berlin, Germany: Springer, 1981, pp. 23–127.
[41] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "GA with a new multi-parent crossover for solving IEEE-CEC2011 competition problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jun. 2011, pp. 1034–1040.
[42] S. Arora and S. Singh, "Butterfly optimization algorithm: A novel approach for global optimization," *Soft Comput.*, vol. 23, no. 3, pp. 715–734, 2019.
[43] X. S. Yang and D. Suash, "Cuckoo search via Lévy flights," in *Proc. World Congr. Nature Biol. Inspired Comput. (NaBIC)*, 2009, pp. 210–214.
[44] H. Shi, S. Liu, H. Wu, R. Li, S. Liu, N. Kwok, and Y. Peng, "Oscillatory particle swarm optimizer," *Appl. Soft Comput.*, vol. 73, pp. 316–327, Dec. 2018.
[45] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.

FULIN WANG received the B.E. and M.E. degrees in agricultural mechanization and automation from Northeast Agricultural University, China, in 1981 and 1988, respectively, and the Ph.D. degree in agricultural equipment engineering technology from Shenyang Agricultural University, China, in 1993. He is currently a Professor at the Engineering College, Northeast Agricultural University. He has published over 100 papers in domestic and international academic journals and conference proceedings. His current research interests include genetic algorithm theory and its application, neural network theory and its application, parallel computing, and image recognition.

GANG XU received the B.E. and M.E. degrees in industry engineering from Northeast Agricultural University, China, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D. degree with the Engineering College. His current research interest includes genetic algorithm theory and its application.



MO WANG received the B.E. degree in industry engineering and the M.M. degree in management science and engineering from Northeast Agricultural University, China, in 2018 and 2022, respectively. Her current research interest includes genetic algorithm theory and its application.