Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 5760841, 16 pages
https://doi.org/10.1155/2018/5760841

Research Article

Improvement Analysis and Application of Real-Coded Genetic Algorithm for Solving Constrained Optimization Problems

Jiquan Wang[1], **Zhiwen Cheng**[1], **Okan K. Ersoy**[2], **Panli Zhang**[1], **Weiting Dai**[1] and **Zhigui Dong**[1]

[1]*College of Engineering, Northeast Agricultural University, Harbin Heilongjiang 150030, China*
[2]*Purdue University, School of Electrical and Computer Engineering West Lafayette, Indiana 47907-1285, USA*

Correspondence should be addressed to Jiquan Wang; wang-jiquan@163.com

Received 22 November 2017; Revised 23 March 2018; Accepted 24 April 2018; Published 6 June 2018

Academic Editor: Fazal M. Mahomed

An improved real-coded genetic algorithm (IRCGA) is proposed to solve constrained optimization problems. First, a sorting grouping selection method is given with the advantage of easy realization and not needing to calculate the fitness value. Secondly, a heuristic normal distribution crossover (HNDX) operator is proposed. It can guarantee the cross-generated offsprings to locate closer to the better one among the two parents and the crossover direction to be very close to the optimal crossover direction or to be consistent with the optimal crossover direction. In this way, HNDX can ensure that there is a great chance of generating better offsprings. Thirdly, since the GA in the existing literature has many iterations, the same individuals are likely to appear in the population, thereby making the diversity of the population worse. In IRCGA, substitution operation is added after the crossover operation so that the population does not have the same individuals, and the diversity of the population is rich, thereby helping avoid premature convergence. Finally, aiming at the shortcoming of a single mutation operator which cannot simultaneously take into account local search and global search, this paper proposes a combinational mutation method, which makes the mutation operation take into account both local search and global search. The computational results with nine examples show that the IRCGA has fast convergence speed. As an example application, the optimization model of the steering mechanism of vehicles is formulated and the IRCGA is used to optimize the parameters of the steering trapezoidal mechanism of three vehicle types, with better results than the other methods used.

1. Introduction

The genetic algorithm (abbreviated as GA or GAs) was proposed by Professor John H. Holland and his students at University of Michigan at the end of the 1960s and in the early 1970s [1–4]. In 1975, Professor Holland published the first monograph devoted to the elaboration of the basic theories and methods of genetic algorithms [1] and put forward the most important schema theorem of the genetic algorithm theory research and development. In the same year, De Hong KA proposed the evolutionary strategy of elitism preservation in his doctoral thesis. Several evolutionary strategies by elitism preservation and selective substitution and duplication were subsequently proposed [5–8]. At present, the GA is basically computed mostly by these evolutionary strategies. In 1989, Goldberg [9] made a comprehensive and systematic summary and discussion of the genetic algorithm and laid the foundation of the modern genetic algorithm.

The encoding schemes in GA are either binary coding or real coding. The shortcomings of binary coding of GA are [10] as follows: (1) for some high dimensional and high precision continuous function optimization problems, the randomness of the binary genetic algorithm makes the local search ability poor, and the binary coding of adjacent integers may have a large Hamming distance, thus reducing search efficiency and influencing the computational accuracy of the genetic operator; (2) binary coding needs to be encoded and decoded frequently, thus increasing the calculation time, with potential conversion errors; (3) with a finite discrete lattice, an individual approaching the extreme value may be omitted, causing the algorithm to reach premature convergence or optimization speed to be very slow; (4) algorithm

efficiency decreases sharply with the increase of variables and the improvement of computational accuracy; (5) as the accuracy of the solution is controlled by the encoding length, binary coding may need too long codes, resulting in excessive computing and memory space as well as reduced computational speed. In 1992, Michalewicz proposed a real-coded genetic algorithm (RCGA) which is more efficient in these regards [11]. RCGA has many attractive properties such as high precision, no coding and decoding required, effective large space search, simple computing, fast convergence, and resistance against falling into a local extreme value [6, 7]. It has been widely applied in areas such as automatic control, combinatorial optimization, machine learning, image processing, self-adaptation control, planning and design, industrial engineering, intelligent manufacturing systems, bioengineering, system engineering, artificial intelligence, intelligent machine system, and artificial life [6–8, 12–16]. It is especially suitable to deal with complex and nonlinear problems that cannot be solved by traditional search methods [6]. Thus, it is one of the key technologies of intelligent computing in the 21st century.

Research on RCGA algorithms can be grouped into the following categories: (1) the determination of an optimal population size [17–20]; (2) generation of the initial population [21–23]; (3) improvement of the existing crossover operator [24–27]; (4) improvement of the existing mutation operator [28–30]; (5) improvement of the evolutionary strategy [31–33]; (6) automatic adjustment of the operator parameters [34–37]. The progress with these algorithms has improved the computational speed and promoted the development of GA theory with new applications.

Crossover operators play a crucial role in expanding the solution space and obtaining the globally optimal solution. Mutations are also important to increase the population diversity, helping GA to expand the search scope and to avoid falling into local optima. As a result, many scholars have focused their attention on the improvement of crossover and mutation operators. In 2016, Chuang et al. [24] proposed a direction-based crossover operator (DBX), but the possible crossover directions of DBX are limited. Although it may generate a crossover direction capable of guiding the chromosomes to move towards the optimal solution, this is not highly probable. Meanwhile, when the dimensionalities of the population are few, the null vector solution is likely to be generated, resulting in no crossover direction. In addition, as some better chromosomes in the population are used to replace the worse chromosomes in the course of ranking selection, the population diversity may become worse after multiple iterations, eventually leading to lost diversity, and thus cannot converge to the global optimal solution. In 1996, D.W. Wang et al. proposed a mutation operator which mutated towards the gradient direction of the objective function [38]. When the objective function is not differentiable, mutation operation cannot be performed. In 2008, Peltokangas et al. summarized the uniform mutation operator, the nonuniform mutation operator, the power mutation operator, and the boundary mutation operator [39–44]. The computation times of the nonuniform mutation operator and the power mutation operator are larger. The boundary mutation is only suitable for the optimization problem when the optimal solution is on the boundary of the feasible region and the offsprings of uniform mutation operator have no connection with the individuals of other mutations. In 2016, Chuang et al. proposed a mutation operator for dynamic random mutation [24], whose step size computational formula had some conflicts with the interpretation of the formula. In addition, the mutation operator needs to give the maximum number of iterations in advance, which is difficult to estimate in advance.

In summary, a good crossover operator and mutation operator should fulfill four conditions: (1) the crossover generated offspring individuals should be in the vicinity of the better individual among the two-parent individuals of participation crossover, thus generating better offsprings; (2) the calculations should be as few as possible; (3) the mutation operator should take into account global as well as local search so that the algorithm can quickly converge to the global optimal solution rather than falling into local optima; (4) the loop statement should be avoided in programming to increase speed of computation. Aiming at these four conditions, this paper proposes a new IRCGA to solve constrained optimization problems. The results with nine test functions show that the IRCGA is effective and feasible in solving constrained optimization problems. As an example application, the optimization model of the steering mechanism of vehicles is formulated, and the IRCGA is used to optimize the parameters of the steering trapezoidal mechanism of three vehicle types, and the better results are obtained as compared to other methods used.

2. Penalty Function Method for Constrained Optimization Problems

The mathematic model of a constrained optimization problem can be generally expressed as follows:

$$\begin{aligned} \min \quad & f(X), \quad X = [X_1, X_2, \cdots, X_k, \cdots, X_n] \in R \\ \text{s.t.} \quad & h_i(X) = 0, \quad i = 1, 2, \cdots, p \\ & g_j(X) \geq 0, \quad j = 1, 2, \cdots, q \end{aligned} \tag{1}$$

where n is the population size, $h_i(X) = 0$ is the ith equation constraint, p is the number of equation constraints, $g_j(X) \geq 0$ is the jth inequality constraint, q is the number of inequality constraints, and X_k is a m-dimensional vector $X_k=(x_{k1},x_{k2},\ldots,x_{km})$.

Equation (1) can also be expressed as

$$\begin{aligned} \min \quad & f(X), \quad X = [X_1, X_2, \cdots, X_k, \cdots X_n] \in R \\ R = & \left\{ X \mid h_i(X) = 0,\ i = 1, 2, \cdots, p;\ g_j(X) \geq 0,\ j = 1, 2, \cdots, q \right\} \end{aligned} \tag{2}$$

Table 1: Sorting grouping selection.

Before Sorting		After Sorting		Grouping pairing				
				The first group		Pairing		The second group
X	$P(X_i,M)$	X'	$P(X_i', M)$	$P(X_i', M)$	X_A	$\leftrightarrow$	X_B	$P(X_i', M)$
X_1	$P(X_1,M)$	X_1'	$P(X_1', M)$	$P(X_1', M)$	X_1'	$\leftrightarrow$	X_6'	$P(X_6', M)$
X_2	$P(X_2,M)$	X_2'	$P(X_2', M)$	$P(X_2', M)$	X_2'	$\leftrightarrow$	X_7'	$P(X_7', M)$
X_3	$P(X_3,M)$	X_3'	$P(X_3', M)$	$P(X_3', M)$	X_3'	$\leftrightarrow$	X_8'	$P(X_8', M)$
X_4	$P(X_4,M)$	X_4'	$P(X_4', M)$	$P(X_4', M)$	X_4'	$\leftrightarrow$	X_9'	$P(X_9', M)$
X_5	$P(X_5,M)$	X_5'	$P(X_5', M)$	$P(X_5', M)$	X_5'	$\leftrightarrow$	X_{10}'	$P(X_{10}', M)$
X_6	$f(X_6,M)$	X_6'	$P(X_6', M)$					
X_7	$P(X_7,M)$	X_7'	$P(X_7', M)$					
X_8	$P(X_8,M)$	X_8'	$P(X_8', M)$					
X_9	$P(X_9,M)$	X_9'	$P(X_9', M)$					
X_{10}	$P(X_{10},M)$	X_{10}'	$P(X_{10}', M)$					

Letting X^* be the optimal solution to the constrained optimization problem means that $\forall\ X \in R$: $f(X^*) \leq f(X)$. In addition, if $g_j(X^*)=0$, the constraint is referred to as active constraint. Under this concept, all the equation constraints $h_i(X)=0$ (i=1,2,...p) are active at X^*.

The penalty function method can be converted to an unconstrained optimization problem by constructing two constrained functions; meanwhile, the penalty factors are introduced and added to the objective function, thus constructing the penalty function given by [45]

$$P(X, M) = f(X) + M_1 \sum_{i=1}^{p} \left[h_i(X)\right]^2 + M_2 \sum_{j=1}^{q} \left[\min\left(0, g_j(X)\right)\right]^2 \quad (3)$$

where M_1 and M_2 are the penalty factors, generally chosen as large enough positive constants; the second and third terms on the right are the penalty terms, and $P(X, M)$ is the penalty function.

In (3), when $X \in R$, there should be no penalty to the feasible points, thus $P(X, M)=f(X)$; when $X \notin R$, for the nonfeasible points, M_1 and M_2 should be very big; therefore, the values of the second and third terms in (3) are large, which is equivalent to the 'penalty' for the infeasible point. Moreover, when X gets farther away from the feasible region, the penalty should be larger. It is conceivable that when M_1 and M_2 become sufficiently large, the minimal point $X(M)$ of the unconstrained optimization problem of (3) is close enough to the minimum point of the original constrained optimization problem. And when $X(M) \in R$, it becomes the minimal point of the original constraint problem.

The minimum value of (3) is

$$\min P(X, M) \quad (4)$$

which is equivalent to the minimum value of (1).

3. New Real-Coded Genetic Algorithm

3.1. Sorting Grouping Selection (SGS). We assume the following: (1) the maximum of the objective function is sought, (2) the population size n is an even number, and (3) the individuals in the population are sorted with respect to their objective function value $P(X, M)$ in descending order. If the minimum of the objective function is sought, the objective function $P(X,M)$ is negated so that we still seek the maximum. The population before sorting is $X=(X_1,X_2,\cdots,X_n)$, and, after sorting, it becomes $X'=(X_1',X_2',\cdots,X_n')$ and satisfies $P(X_1',M) \geq P(X_2',M) \geq \cdots \geq P(X_n',M)$.

First, the chromosomes in the population are arranged into two groups. Group 1 includes the first $n/2$ chromosomes and Group 2 includes the second $n/2$ chromosomes. The 1st individual in Group 1 is matched with the 1st individual in Group 2, the 2nd individual in Group 1 is matched with the 2nd individual in Group 2, and so on. In this way, we can obtain $n/2$ pairs of individuals. See Table 1 for the specific operation of the sorting grouping selection.

In Table 1, we carry out crossover of X_1' in Group 1 with X_6' in Group 2, X_2' in Group 1 with X_7' in Group 2, and so on.

As compared with the roulette wheel selection and the tournament selection, the SGS method can enlarge the distance in every pair of the parent individuals and magnify the difference between the parent individuals under matching; in addition, the SGS is a direct operation based on the ranking of objective function values without computing the individual fitness values. The computation is simple and rapid in selecting the chromosomes for crossover.

3.2. Heuristic Normal Distribution Crossover (HNDX). If the objective function is to solve the optimization problem of maximum value, as the individual of bigger objective function value has higher probability of approaching the optimal solution, it is believed that the optimal solution is likely to be near the individual with a bigger objective function value [46]. Inspired by this, this paper proposes a heuristic normal distribution crossover (HNDX) operator

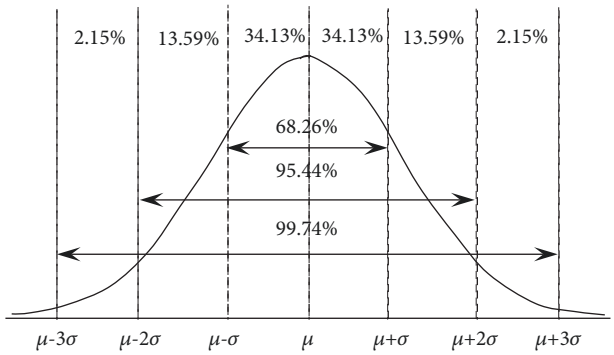


Figure 1: Density function of normal distribution.

which emphasizes the parent which has a bigger objective function value. With the resulting better crossover direction, two potential offsprings are generated, thus improving the convergence speed of IRCGA.

The key to HNDX is how to make the crossover generated offsprings in the vicinity of the superior parent or in the superior crossover direction. We note that a generated random number according to the normal distribution has a large probability near its mean μ. In the process of crossover, if the better one among the two parents is served as the mean μ of the normal distribution, then it is guaranteed that the generating offspring according to normal distribution is in the vicinity of the better parent. Therefore, an offspring is generated by normal distribution and the normal distribution is denoted as $N(\mu,\sigma^2)$. The density function of the normal distribution is shown in Figure 1.

It can be seen from Figure 1 that the probability of the random number X generated by $N(\mu, \sigma^2)$ in the interval $(\mu$-σ,μ+$\sigma)$ is 68.26%, the probability of X in the interval $(\mu$-$2\sigma,\mu$+$2\sigma)$ is 95.44%, and the probability of X in the interval $(\mu$-$3\sigma,\mu$+$3\sigma)$ is 99.74%. It can be seen that the probability that X falls outside $(\mu$-$3\sigma,\mu$+$3\sigma)$ is less than 0.3%, and it is often considered that the corresponding event does not occur in practice. Thus, the interval $(\mu$-$3\sigma,\mu$+$3\sigma)$ can be regarded as the actual possible interval of the random variable X, which is called the 6σ principle of the normal distribution. Based on the above analysis, as long as we control the size of σ value, we can basically guarantee that the random number X generated by $N(\mu, \sigma^2)$ is near the mean μ.

Let the two parents of participation crossover be X_i' (i=1,2,$\cdots$,n/2) and X_j' (j=n/2+1, n/2+2,$\cdots$,n). Assuming that X_i' is superior to X_j', the two offsprings Y_i (i=1,2,$\cdots$,n/2) and Y_j (j=n/2+1, n/2+2,$\cdots$,n) are generated by

$$Y_i = N\left(X_i', \varepsilon + \left[\frac{X_i' - X_j'}{6}\right]_{\bullet}\text{^}2\right) \quad i = 1, 2, \cdots, \frac{n}{2};\ j = \frac{n}{2} + 1,\ \frac{n}{2} + 2, \cdots, n$$

$$Y_j = \lambda\overrightarrow{D} \quad j = \frac{n}{2} + 1,\ \frac{n}{2} + 2, \cdots, n$$

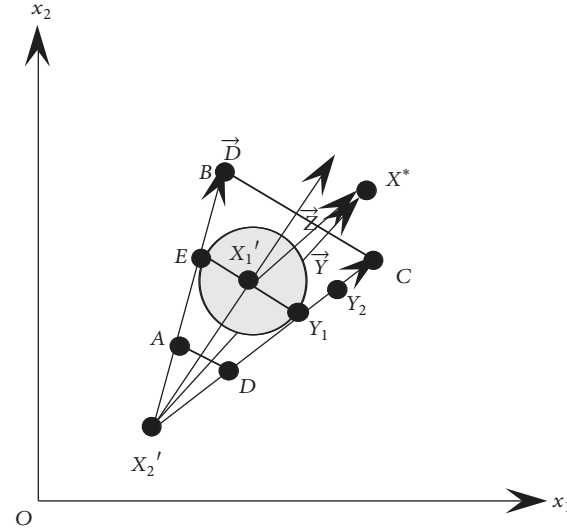


Figure 2: Schematic diagram of HNDX.

$$\overrightarrow{D} = Y_i - X_j' \quad i = 1, 2, \cdots, \frac{n}{2};\ j = \frac{n}{2} + 1,\ \frac{n}{2} + 2, \cdots, n \tag{5}$$

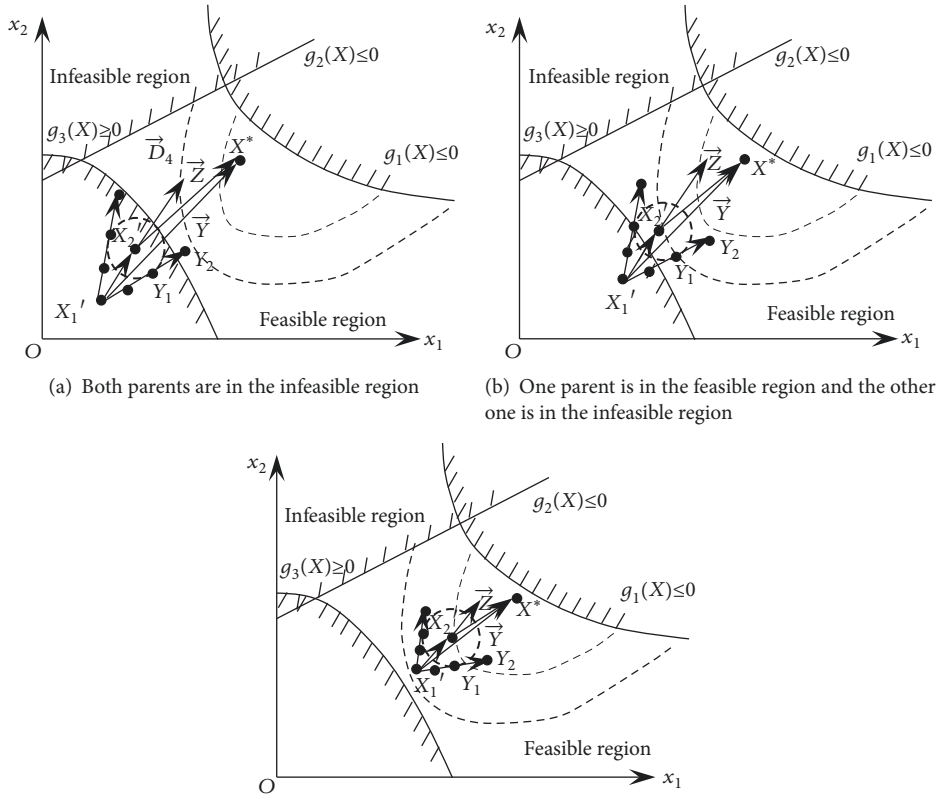
where X_i' is the mean of the normal distribution, $[(X_i' - X_j')/6]_{\bullet}\text{^}2$ is the variance of the normal distribution, $.\text{^}2$ is the square of each element in vector $(X_i' - X_j')/6$, ε is a small positive number to avoid that the variance of the normal distribution is 0, $\overrightarrow{D}$ is the crossover direction, and λ is a uniformly distributed random number in [0.5, 1.5].

Let the two-parent individuals of participation crossover be X_1' and X_2' and X_1' be superior to X_2'. To illustrate the principle of HNDX, letting the variable dimension m be 2, the principle of HNDX is shown in Figure 2.

In Figure 1, DY_1=Y_1C, AE=EB.

Equation (5) shows that Y1 is an offspring randomly generated according to $N\ (X_1', \varepsilon + ((X_1'$-$X_2')/6).\text{^}2)$, with its mean equal to the mean of X_1' and its variance equal to ε+$((X_1'$ - $X_2')/6).\text{^}2$. In addition, it can be seen from Figure 1 that the random number generated according to $N(X_1', \varepsilon + ((X_1' - X_2')/6).\text{^}2)$ has high probability of being in the vicinity of the mean X_1' of the normal distribution. According to the above analysis, an offspring Y_1 is generated by the crossover of the two parents X_1' and X_2' and Y_1 has a high probability of being within the shadow region of Figure 2; that is to say, Y_1 is within a circle whose center is X_1', which can basically guarantee that Y_1 is in the vicinity of the better parent X_1'.

The second offspring Y_2 is generated by the crossover of the two parents X_1' and X_2' as $Y_2 = \lambda\overrightarrow{D}$, where the crossover direction is given by $\overrightarrow{D} = Y_1 - X_1'$. Therefore, Y_2 might be at any point on the line CD. In addition, since Y_1 is likely to be at any point within the circle that takes the line segment $X_1'Y_1$ as radius and X_1' as center of the circle,



(a) Both parents are in the infeasible region

(b) One parent is in the feasible region and the other one is in the infeasible region

(c) Both parents are in the feasible region.

Figure 3: Schematic diagrams of HNDX operation in three cases in a two-dimensional space.

there are numerous possible cross directions $\vec{D}$. Let X^* be the optimal solution of the problem to be solved. For the parent X_1' of participation crossover, the optimal crossover direction is $\vec{Z} = X^* - X_1'$; and for X_2', the optimal crossover direction is $\vec{Y} = X^* - X_2'$. The crossover direction $\vec{D}$ generated by HNDX is likely to be completely consistent with the optimal crossover direction, even if it is not completely consistent with the optimal crossover direction $\vec{Y}$ and $\vec{Z}$. Hence, it has high probability of generating a direction that is very close to the optimal crossover direction $\vec{Y}$ and $\vec{Z}$. In summary, HNDX has high probability of generating better offsprings, thereby significantly improving the convergence speed of the algorithm.

Compared with the DBX in [24], HNDX can generate numerous crossover directions while DBX can only generate 2^m-1 crossover directions; moreover, the probability that the offsprings generated by HNDX are close to the optimal solution is much higher than that of DBX. In addition, HNDX can avoid use of loop statements in programming, and the program is relatively simple.

Let y be the individuals in the population after sorting according to the objective function values and x be the cross-generated offsprings. We adopt the evolutionary strategy of Section 3.5 and choose the crossover probability equal to 1. The corresponding Matlab program code of HNDX can be as follows:

```
x(1:n/2,:)=normrnd(y(1:n/2,:),0.001+((y(1:n/2,:)-
y(n/2+1:n,:))/6).^2);
x(n/2+1:n,:)=unifrnd(0.5,1.5)*(x(1:n/2,:)-
y(n/2+1:n,:));
```

It is observed that HNDX Matlab program has neither loop statement nor conditional statement, while DBX has both. As a result, HNDX program is simpler and takes less time to accomplish the corresponding statements.

As an example, in terms of 2D search space, for the spatial positions of two parents X_1' and X_2' participating in crossover, there are three possible cases: (*a*) both parents are in the infeasible region; (*b*) one parent is in the feasible region while the other is not; (*c*) both parent are in the feasible region. Of these three cases, the HNDX operation in a two-dimensional space is shown in Figure 3.

In Figure 3, the schematic of case (*a*) shows that the HNDX operator can effectively guide the movement of two parents from an infeasible region to a feasible region. In the case of (*b*), HNDX crossover enables the individuals in the infeasible region to move into the feasible region. When the individuals are already in the feasible region as in the case of (*c*), HNDX searches in the neighborhood of the two parents and generates a direction to guide them to move towards the optimal solution.

3.3. Substitution Operation. In the global optimization problem with many local optima, when the algorithm finds a region with an extreme value (whether it is a local extremum or a global extremum), individuals in the population constantly move closer to the region and there may be the same or similar individuals in the population. With the increase of the number of iterations, the same individuals in the population will gradually increase and may even make all the individuals in the population be the same, which will make the diversity of the population worse; thus the convergence speed of GA and the ability of exploring other extreme value regions are affected. In the case of the IRCGA proposed in the literature for the optimization problem with many local optima [24], it is possible that the vast majority of individuals in the population become the same or all individuals become the same such that the algorithm cannot converge to the global optimal solution. In order to avoid the occurrence of the above phenomena and to maintain the diversity of the population during the iteration process, substitution operation is added to the IRCGA proposed herein. Substitution operation is as follows.

If there are the same two or more individuals in a population after crossover, only one of them shall be reserved while eliminating the others. If the current population size is n_1, n-n_1 individuals are generated at random to maintain the population size unchanged as n. Since the n-n1 individuals in the substitution operation are randomly generated, the substitution operation has the function of jumping out of the local optimum.

3.4. Combinational Mutation. In the cases of the mutation operator given in the existing literature, some local search abilities are strong [47, 48] and some global search abilities are strong [48]. For the optimization problems with less extreme points, a mutation operator with stronger local search ability should be adopted. For the optimization problems with more extreme points, if a mutation operator with stronger local search ability is adopted, it is easy to converge to a local optimum; if a mutation operator with stronger global search ability is adopted and the accuracy requirement of the optimal solution for the problem to be optimized is higher, the convergence speed of the algorithm slows down. With some of the mutation operators given in the literature, there is strong global search capability at the beginning of the iterations, and with the increase of the number of iterations, local search capabilities are enhanced. However, this requires the maximum number of iterations to be given, which is difficult to do in advance. Although this method is theoretically feasible, it is usually worse in practice [24]. In summary, a single mutation operator is difficult to take into account both global and local search capabilities. For this reason, this paper proposes a method of combinatorial mutation which uses three-mutation operators. The first mutation operator enhances the local search ability with the increase of the number of iterations; the second mutation operator has strong global search capability; the third mutation operator has strong local search capability. The specific approach of combinational mutation is as follows.

We divide the number of iterations by 3. When the remainder is 1, the second mutation operator is used; when the remainder is 2, the first mutation operator is used; when the remainder is 0, the third mutation operator is used.

The advantage of combinatorial mutation is that the local search ability of mutation operator is taken into account in the iterative process and the global searching ability of mutation operator is also taken into account. The three-mutation operators are as follows:

$$\overline{X}_i' = \overline{X}_i + \lambda_1 * \frac{\left(b^T - a^T\right)}{gen} \tag{6}$$

where gen is the number of iteration, $\overline{X}_i$ is the offspring from crossover, $\overline{X}_i'$ is the offspring from mutation, vectors a and b are the lower and upper limits of variables, and $\lambda_1 * (b^T\text{-}a^T)$ is the product by multiplying λ_1 with the corresponding element of $b^T\text{-}a^T$ at the same position, $\lambda_1=(\lambda_{11},\lambda_{12},\ldots,\lambda_{1m})$, in which $\lambda_{11},\lambda_{12},\cdots,\lambda_{1m}$ are uniformly distributed random numbers in $[-1, 1]$.

$$\overline{X}_i' = \overline{X}_i + \overline{X}_i * Cauchy\,(0, 1) \tag{7}$$

where Cauchy $(0, 1)$ is standard Cauchy distribution, $\overline{X}_i$ is an individual to mutate, and $\overline{X}_i'$ is an individual after mutation.

$$\overline{X}_i' = N\left(\overline{X}_i, \delta^2\right) \tag{8}$$

where $\overline{X}_i$ is the mean of the normal distribution and δ^2 is the variance of the normal distribution.

δ^2 of (8) is given by

$$\delta^2 = \frac{\left|X_{best} - \overline{X}_i\right|}{6} \tag{9}$$

where X_{best} is the optimal individual in population.

The global search ability of the first mutation operator at the beginning of the iterations is strong, but when the number of iterations increases above a certain value, the first mutation operator almost loses mutation ability. The second mutation operator is Cauchy mutation, which can generate a larger mutation step compared to the normal mutation operator, giving the algorithm better global search ability. The third mutation operator is normal mutation, which focuses on searching for a local region near the mean with better local search ability, but the ability to guide the individual to jump out of the local optimal solution is weak, which is not conducive to global convergence. Therefore, the combination mutation operator takes into account both global exploration and local search, which makes GA converge quickly to the global optimal solution.

3.5. Evolutionary Strategy (ES). The evolutionary strategy of the IRCGA proposed in this paper is as follows:

(1) The initial population is randomly generated as parents, the population size is n, and the mutation probability is p_m.

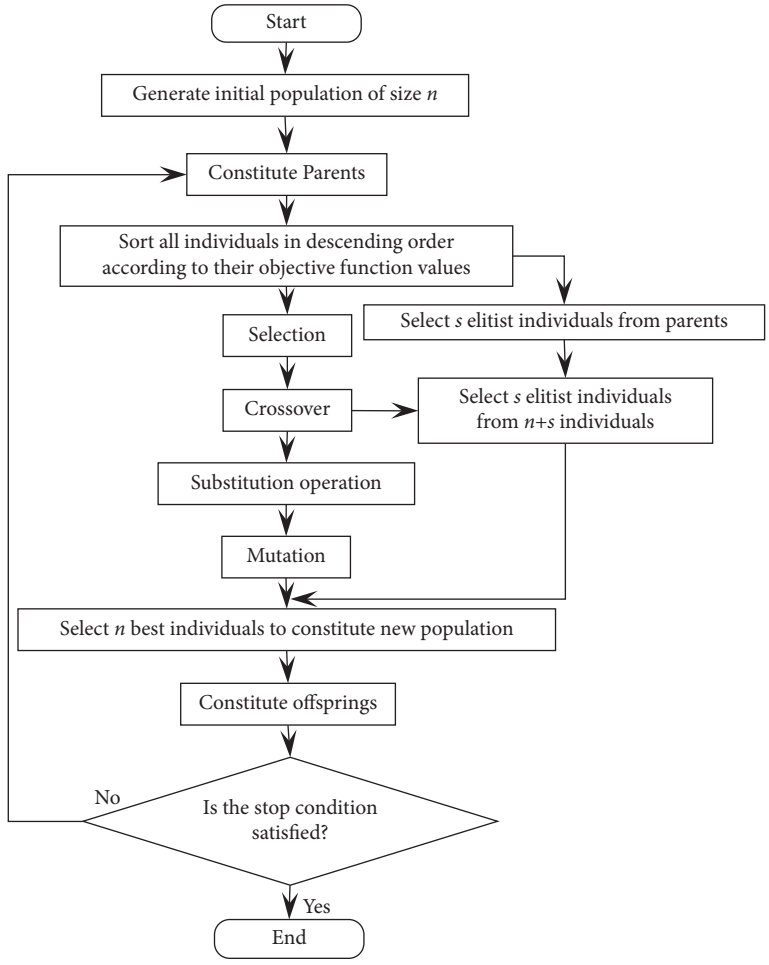


Figure 4: The evolutionary strategy flow diagram of IRCGA.

(2) Sorting operation is performed; that is to say, all individuals in the population are sorted in descending order according to their objective function values and s elitist individuals are selected with best ranking among n parents.

(3) Selection and crossover operations are carried out according to the sorting results with crossover probability equal to 1.

(4) Substitution operation is performed with cross-generated n offsprings and s elitist individuals.

(5) All individuals in the population are sorted in descending order according to their objective function values after substitution operation.

(6) If the population size is $n_1$less than $n+s$ now, randomly generate $n+s-n_1$ individuals.

(7) Repeat the above sorting operation, get new population, and select s elitist individuals and n individuals with best ranking in the new population.

(8) Using the mutation operator, modify the n individuals.

(9) Regenerate the new population consisting of the np_m mutated individuals, $n(1\text{-}p_m)$ unmutated individuals and s elitist individuals chosen in Step (7).

(10) Sort the individuals of the regenerated new population in descending order according to their objective function values.

(11) If the stop condition is met, output the optimal solution and optimal value;

(12) Otherwise select s elitist individuals and n individuals with best ranking within the last population. Go to Step (3) to start the next loop.

The evolutionary strategy flow diagram for IRCGA is shown in Figure 4.

4. Algorithmic Testing and Analysis

Below RCGA proposed in [24] is abbreviated as IRCGA-1, and IRCGA proposed in this paper is abbreviated as IRCGA-2; IRCGA-2 that removes the substitution operation is abbreviated as IRCGA-3; RCGA proposed in [49, 50] is abbreviated, respectively, as IRCGA-4 and IRCGA-5.

In order to verify the validity and feasibility of IRCGA-2, it is to be compared with the existing RCGA. Among the existing references of RCGA, [24] was published in 2016 with a great amount of comparative studies of the RCGA; therefore, IRCGA-2 will be compared with IRCGA-1. In addition, the IRCGA-2 will also be compared with IRCGA-4 and IRCGA-5.

4.1. Iteration Termination Condition. The iteration termination condition for the RCGA can be defined as

$$\left|f_i - f_i^*\right| \le \varepsilon_i, \quad i = 1, 2, \cdots, p \tag{10}$$

where f_i^* is the globally optimal value of the ith test function in theory, f_i is the optimal value of the ith test function obtained by RCGA, and ε_i is the precision for the ith test function.

4.2. Diversity of Population. If the individuals in the tth population $X(t)$ are $X_1(t),X_2(t),\cdots,X_n(t)$, respectively and the ith individual is $X_i(t)=(x_{i1}(t),x_{i2}(t),\ldots,x_{im}(t))$, the population center $\overline{X}(t)=(\overline{x}_1(t),\overline{x}_2(t),\cdots,\overline{x}_m(t))$ is calculated as follows:

$$\begin{aligned}\overline{X}(t) &= \left(\overline{x}_1(t), \overline{x}_2(t), \cdots, \overline{x}_m(t)\right) \\ &= \frac{X_1(t) + X_2(t) + \cdots + X_n(t)}{n} \\ &= \frac{\left(\sum_{i=1}^n x_{i1}, \sum_{i=1}^n x_{i2}, \cdots, \sum_{i=1}^n x_{im}\right)}{n}\end{aligned} \tag{11}$$

The average value of the sum of the variance of each individual to the population center in a population is $D(t)$ calculated by

$$D(t) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m}\left(x_{ij} - \overline{x}_j\right)^2} \tag{12}$$

The D value can reflect the degree of dispersion of the individual in the population. The larger the D value, the better the diversity of the population; the smaller the D value, the poorer the diversity of the population; when D value is equal to 0, the population diversity is completely lost. That is, all individuals in the population are the same.

4.3. Test Functions. In order to verify the validity and feasibility of IRCGA-2, nine frequently used test functions of certain complexity are selected. They are discussed below:

(1) Needle-in-a-haystack function to be maximized

$$f_1(x) = \left[\frac{3}{0.05 + x_1^2 + x_2^2}\right]^2 + \left(x_1^2 + x_2^2\right)^2 \quad -5.12 \le x_1, x_2 \le 5.12 \tag{13}$$

The optimal solution of f_1 is located at $x^* = (0,0)$ with $f_1(x^*) = 3600$.

(2) Schaffer function to be maximized

$$f_2(x) = 0.5 - \frac{\sin^2\sqrt{x_1^2 + x_2^2} - 0.5}{\left(1 + 0.001\left(x_1^2 + x_2^2\right)\right)^2} \quad -10 < x_1, x_2 < 10 \tag{14}$$

The optimal solution of f_2 is located at $x^* = (0,0)$ with $f_2(x^*) = 1$.

(3) Six-hump Camel Back function to be minimized

$$f_3(x) = \left(4 - 2.1x_1^2 + \frac{x_1^4}{3}\right)x_1^2 + x_1x_2 + \left(-4 + 4x_2^2\right)x_2^2 \quad -10 < x_1, x_2 < 10 \tag{15}$$

The optimal solution of f_3 is located at $x^* = (-0.0898, 0.7126)$ and $(0.0898, -0.7126)$ with $f_3(x^*) = -1.031628$.

(4) Shubert function to be minimized

$$f_4(x) = \left(\sum_{i=1}^{5} i\cos\left((i+1)x_1 + i\right)\right) \cdot \left(\sum_{i=1}^{5} i\cos\left((i+1)x_2 + i\right)\right) \quad -10 \le x_i \le 10 \tag{16}$$

The optimal solution of f_4 is located at $x^* = (-1.42513, 0.80032)$ with $f_4(x^*) = -186.7309$.

(5) Rosenbrock function to be minimized

$$f_5(x) = 100\left(x_2 - x_1^2\right)^2 + \left(x_1 - 1\right)^2 \quad -10 < x_1, x_2 < 10 \tag{17}$$

The optimal solution of f_5 is located at $x^* = (1,1)$ with $f_5(x^*) = 0$.

(6) Michalewiczs function to be minimized

$$f_6(x) = -\sin(x_1)\left[\sin\left(\frac{x_1^2}{\pi}\right)\right]^{20} - \sin(x_2)\left[\sin\left(\frac{2x_2^2}{\pi}\right)\right]^{20} \quad 0 \le x_1, x_2 \le \pi \tag{18}$$

The optimal solution of f_6 is located at $x^* = (2.0230, 2.0230)$ with $f_6(x^*) = -1.80130$.

(7) G_{08} function to be minimized

$$f_7(x) = -\frac{\sin^3(2\pi x_1)\sin(2\pi x_2)}{x_1^3(x_1 + x_2)}$$
$$s.t. \quad g_1(x) = x_1^2 - x_2 + 1 \le 0$$
$$g_2(x) = 1 - x_1 + \left(x_2 - 4\right)^2 \le 0 \quad 0 \le x_1, x_2 \le 10 \tag{19}$$

The optimal solution of f_7 is located at $x^* = (1.2279723, 4.2453733)$ with $f_7(x^*) = -0.095825$.

(8) Easom function to be minimized

$$f_8(x) = -\cos(x_1)\cos(x_2)\exp\left(-(x_1-\pi)^2-(x_2-\pi)^2\right) \quad (20)$$
$$-100 \le x_1, x_2 \le 100$$

The optimal solution of f_8 is located at $x^* = (\pi,\pi)$ with $f_8(x^*) = -1$.

(9) Generalized Rastrigin function to be minimized

$$f_9(x) = x_1^2 - 10\cos(2\pi x_1) + x_2^2 - 10\cos(2\pi x_2) + 20 \quad (21)$$
$$0 \le x_1, x_2 \le \pi$$

The optimal solution of f_8 is located at $x^* = (0,0)$ with $f_8(x^*) = 0$.

4.4. Parameter Settings. In order to obtain a fair performance comparison, the parameters related to IRCGA proposed in this paper are set as follows:

(1) The computing precision of various test functions $\varepsilon_1=\varepsilon_2=\varepsilon_3=\varepsilon_4=\varepsilon_5=\varepsilon_6=\varepsilon_7=\varepsilon_8=\varepsilon_9=10^{-4}$; (2) the population size n=100; (3) $M_1=10^7$ and $M_2=10^7$ in the penalty function; (4) in IRCGA-2, the mutation probability p_m=0.5 and the crossover probability p_c=1. In IRCGA-1, the mutation probability p_m=0.05 and the crossover probability p_c=0.9; (5) Reserve 50 elite individuals; (6) in IRCGA-4 and IRCGA-5, the population size n=100 and the remaining parameters are the same as in [50, 51].

4.5. Limitations of IRCGA-1

Definition 1. When all the individuals in the population are the same, the individuals in population are called the losing population diversity.

For the optimization problem with more local extreme points, ICRGA-1 has great possibility of falling into local extreme points. For example, consider that the first test function in Section 4.3. With IRCGA-1, the evolution of population as a function of number of iterations is shown in Figure 5. The corresponding results with IRCGA-2 are shown in Figure 6.

It can be seen from Figure 5 that the diversity of the population tends to become poor with the increase of the number of iterations. When the number of iterations is 160, all the individuals in the population are the same and the value of the objective function is 2978.2275. The population has lost its diversity, and the global optimal solution cannot be obtained. It can be seen from Figure 6 that the population diversity of IRCGA-2 tends to become poor with the increase of the number of iterations but the diversity of the population is significantly superior to that of IRCGA-1. In order to compare the population diversities of IRCGA-1 and IRCGA-2 further, consider the needle-in-a-haystack function discussed in Section 4.2. The IRCGA-1 and IRCGA-2 programs were run 1000 times, respectively. With IRCGA-1, the number of loss of population diversity was 697 times and the probability of losing population diversity was 69.7%; with IRCGA-2, the number of loss of population diversity was 0 times and the probability of losing population diversity was 0%. These results indicate that IRCGA-1 is more likely to lose population diversity for the problem of optimizing with many local optima.

In summary, IRCGA-1 has great limitations in solving the optimization problem with many local optima. In many cases, the global optimal solution cannot be obtained.

4.6. Test Results. In order to obtain a fair performance comparison, the 9 test functions were used as examples and each test function was run on the same computer for 1000 times. The computational results of IRCGA-1, IRCGA-2, IRCGA-3, IRCGA-4, and IRCGA-5 are shown in Table 2.

The average running time, the average number of iterations, and the average value of population diversity when converging to the optimal solution in Table 2 were calculated as follows: when the iteration termination condition is satisfied, the numbers of iterations, time, and population diversity of processing at the ith run are, respectively, $niter(i)$, $t(i)$, and $div(i)$, $i = 1, 2, \cdots, k$, where k is the number of runs used in each experiment. Then, the average running time, average number of iterations, and average value of population diversity are computed by

$$t_{av} = \frac{1}{k}\sum_{i=1}^{k} t(i) \quad (22)$$

$$niter_{av} = \frac{1}{k}\sum_{i=1}^{k} niter(i) \quad (23)$$

$$div_{av} = \frac{1}{k}\sum_{i=1}^{k} div(i) \quad (24)$$

where t_{av} is the average running time, $niter_{av}$ is the average number of iterations, and div_{av} is the average value of population diversity.

When the average running time of ICRGA-1 is t_1, the average running time of ICRGA-2 is t_2, IRCGA-2 reduces the average running time by x% for ith test function f_i in comparison to IRCGA-1, and, then, x% is computed by

$$t_1(1 - x\%) = t_2 \quad (25)$$

$$x = 100\left(1 - \frac{t_2}{t_1}\right) \quad (26)$$

The computational method of x% is the same as (25) and (26) for IRCGA-2, IRCGA-3, IRCGA-4, and IRCGA-5.

As can be observed in Table 2, the average running time, average number of iterations, and average value of population diversity of IRCGA-2 are significantly superior to those of IRCGA-1, IRCGA-3, IRCGA-4, and IRCGA-5.

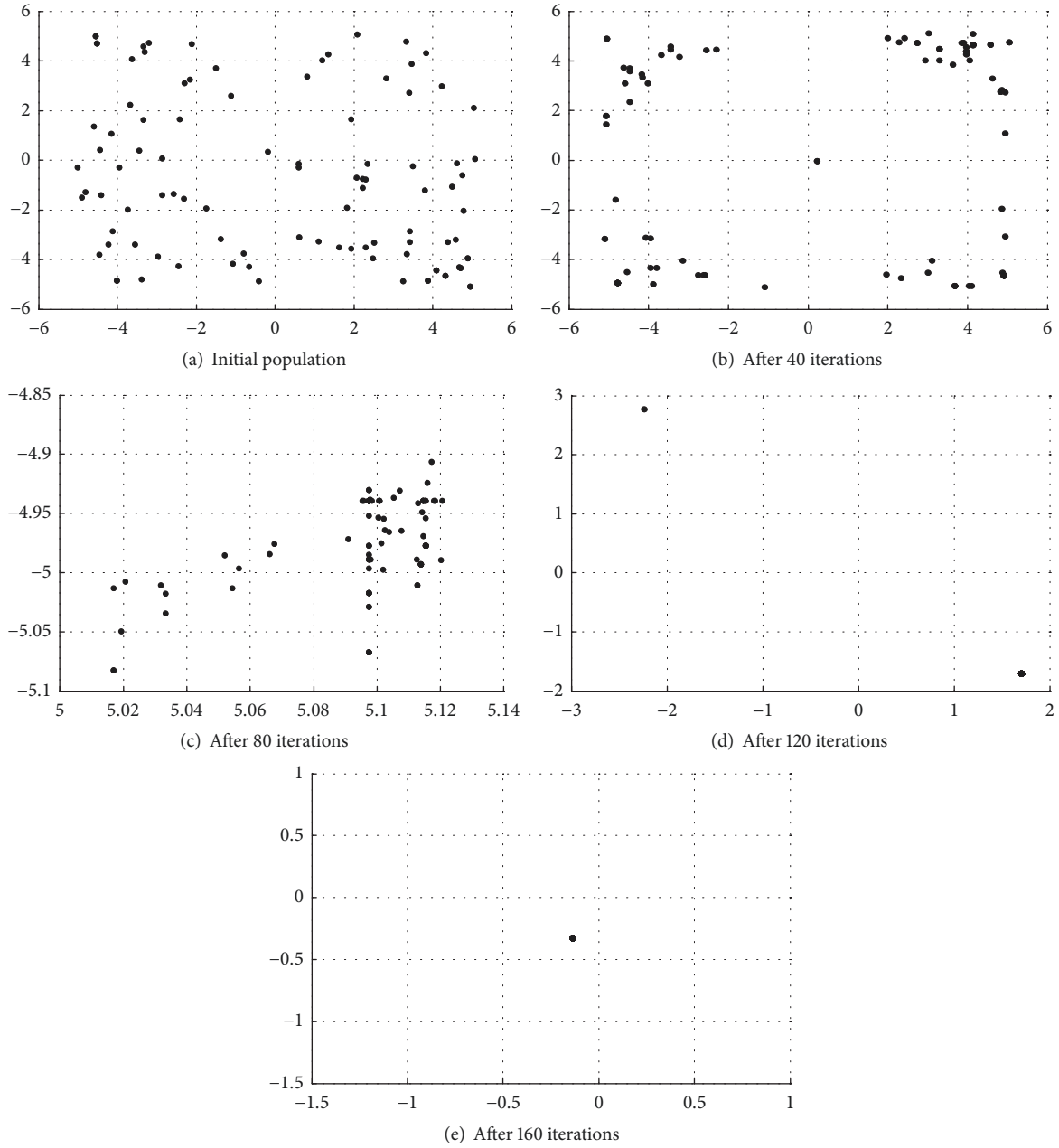


Figure 5: The evolution of population in IRCGA-1 as a function of the number of iterations.

IRCGA-2 reduces the average running time by 96.7328%, 17.3913%, 96.9867%, and 96.4353% for f_1, 96.8280%, 49.8113%, 97.8737%, and 73.0223% for f_2, 77.6050%, 47.2527%, 89.0327%, and 54.5731% for f_3, 98.5534%, 9.3625%, 98.2681%, and 72.8908% for f_4, 72.2076%, 21.9753%, 93.1557%, and 75.6923% f_5, 61.9744%, 33.3333%, 88.6401%, and 83.7500% for f_6, 65.1568%, 37.5000%, 58.2463%, and 45.6522% for f_7, 81.0282%, 20.1220%, 82.1038%, and 82.0320% for f_8, 91.9857%, and 50.9375%, 71.9643%, and 60.3535 for $f_9$in comparison to IRCGA-1, IRCGA-3, IRCGA-4, and IRCGA-5. Similarly, the numbers of iterations are reduced by 88.2454%, 20.2076%, 89.6488%, and 87.7177% for f_1, 88.6364%, 48.9817%, 93.6805%, and 84.6963% for f_2, 24.5985%, 11.7893%, 73.8572%, and 33.5144% for f_3, 95.2673%, 19.4441%, 95.0346%, and 82.6697% for f_4, 14.2070%, 4.7474%, 80.2807%, and 78.6939% for f_5, 11.0402%, 6.7012%, 93.6547%, and 92.0071% for f_6, 15.8179%, 6.1513%, 41.0115%, and 47.6137% for f_7, 34.9704%, 11.2156%, 40.9364%, and 42.8695% for f_8, and 73.1850%, 23.1377%, 38.9808%, and 30.6282% for f_9. Thus, IRCGA-2 is observed to be superior to IRCGA-1, IRCGA-3, IRCGA-4, and IRCGA-5 in terms of all the measures utilized.

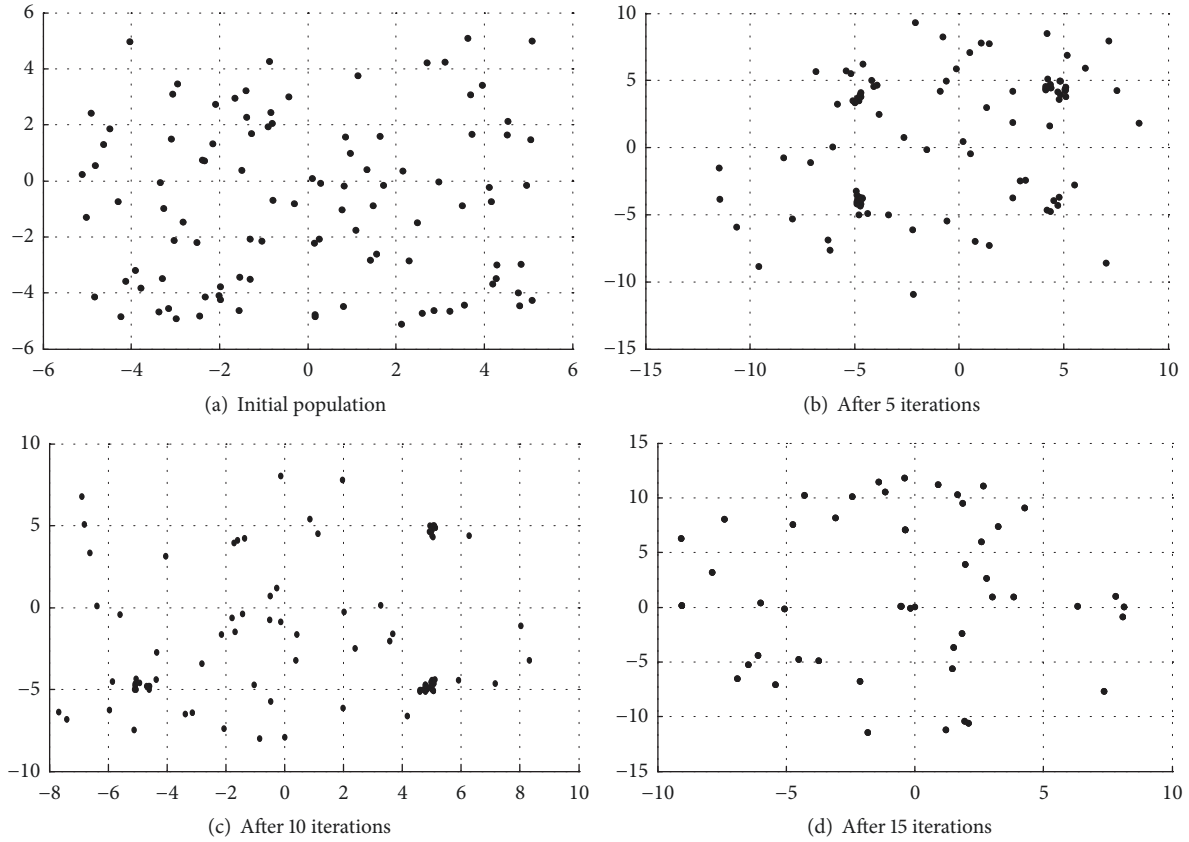


Figure 6: The evolution of population in IRCGA-2 as a function of the number of iterations.

5. Parameter Optimization of Vehicle Steering Trapezoid Mechanism

In the autosteering process, the difference between the actual movement trajectory and the theoretical movement trajectory of the vehicle steering mechanism results in the motion error, which increases the tire wear and destroys the safety and stability of steering. Through the optimization of the size and positioning parameters of the steering mechanism, the error can be effectively reduced, thereby improving the vehicle handling performance and improving the steering safety. Here the weighted sum of the relative error of the theoretical rotation angle and the actual rotation angle of the external front steering wheel is chosen as the objective function to be minimized. When the inside front wheel turning angle of the vehicle is α, the schematic diagram of the vehicle ideal steering process and the actual steering process are shown in Figures 7 and 8.

Note that α is actual rotational angle of inside front steering wheel, β is theoretical rotational angle of external front steering wheel, M is the distance between the left and right vertical shafts, L is the distance of rear and front axle of vehicle wheel, R is the steering radius of external front steering wheel, θ is the bottom angle of steering trapezoid mechanism, and a is the distance of wheel and steering pin.

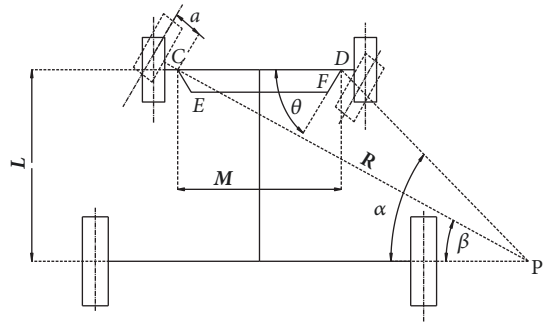


Figure 7: Theoretical steering process schematic.

Note that α is the actual rotational angle of inside front steering wheel, β is the theoretical rotational angle of external front steering wheel, M is the distance between the left and right vertical shafts, L is the distance of rear and front axles of the vehicle wheel, θ is the bottom angle of the steering trapezoid mechanism, β' is the ideal rotational angle of the external front steering wheel, m is the steering arm length of the steering trapezoid mechanism, N is the length of the auxiliary line, and δ_1 and δ_2 are the interior angles of auxiliary calculation.

Table 2: The computational results of five ICRGAs.

function	Algorithm	Average running time (s)	Average number of iterations (times)	The average value of population diversity when converging to the optimal solution
f_1	IRCGA-1	0.6397	205.3840	28.6652
	IRCGA-2	0.0209	24.1420	50.2294
	IRCGA-3	0.0253	30.2560	30.6849
	IRCGA-4	0.6936	233.2300	11.2693
	IRCGA-5	0.5863	196.5600	27.2.46
f_2	IRCGA-1	0.4193	115.0780	5.9865
	IRCGA-2	0.0133	13.0770	10.2328
	IRCGA-3	0.0265	25.6320	6.9731
	IRCGA-4	0.6255	206.9300	1.7962
	IRCGA-5	0.0493	85.4500	7.3493
f_3	IRCGA-1	0.0643	18.9930	3.4109
	IRCGA-2	0.0144	14.3210	5.8449
	IRCGA-3	0.0273	16.2350	4.7633
	IRCGA-4	0.1313	54.7800	1.4526
	IRCGA-5	0.0317	21.5400	2.8277
f_4	IRCGA-1	15.7263	4630.9300	3.2163
	IRCGA-2	0.2275	219.1660	11.0648
	IRCGA-3	0.2510	272.0670	3.4109
	IRCGA-4	13.1359	4413.9000	2.0108
	IRCGA-5	0.8392	1264.6400	4.4617
f_5	IRCGA-1	0.1137	36.1090	3.9642
	IRCGA-2	0.0316	30.9790	6.2602
	IRCGA-3	0.0405	32.5230	4.0683
	IRCGA-4	0.4617	157.1000	2.1785
	IRCGA-5	0.1300	145.4000	3.5308
f_6	IRCGA-1	0.0547	22.0830	3.6838
	IRCGA-2	0.0208	19.6450	5.2457
	IRCGA-3	0.0312	21.0560	4.0133
	IRCGA-4	0.1831	309.6000	2.1494
	IRCGA-5	0.1280	245.7800	3.4931
f_7	IRCGA-1	0.0574	18.3590	5.8063
	IRCGA-2	0.0200	15.4550	10.0381
	IRCGA-3	0.0320	16.4680	7.8151
	IRCGA-4	0.0479	26.2000	5.1667
	IRCGA-5	0.0368	29.5020	6.4071
f_8	IRCGA-1	0.1381	42.6790	78.8050
	IRCGA-2	0.0262	27.7540	127.9265
	IRCGA-3	0.0328	31.2600	86.3403
	IRCGA-4	0.1464	46.9900	75.9938
	IRCGA-5	0.1458	48.5800	74.1084
f_9	IRCGA-1	0.1959	60.9510	7.2904
	IRCGA-2	0.0157	16.3440	13.5928
	IRCGA-3	0.0320	21.2640	10.6016
	IRCGA-4	0.0560	26.7850	4.2582
	IRCGA-5	0.0396	23.5600	7.5200

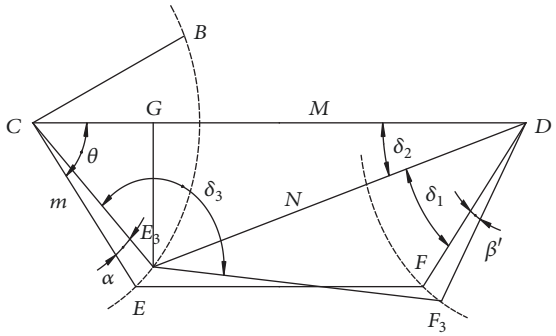


Figure 8: Schematic of vehicle actual steering process when the rotational angle of the inside steering wheel is α.

5.1. Optimization Model of the Vehicle Steering Trapezoid Mechanism. Using the geometry in Figure 6, we get

$$\beta = \operatorname{arccot}\left(\frac{M}{L} + \cot\alpha\right) \tag{27}$$

where α is the rotational angle of inside front steering wheel, β is the theoretical rotational angle of the external front steering wheel, M is the distance between the left and right vertical shafts, and L is the distance of rear and front axles of the vehicle wheel.

In the process of vehicle swerve, when the turning radius reaches the minimum, the rotational angle of inside front steering wheel reaches the maximum; that is,

$$\alpha_{\max} = \arctan\frac{L}{\sqrt{\left(R_{\min} - a\right)^2 - L^2} - M} \tag{28}$$

where R_{min} is the minimum turning radius and α_{max} is the maximum rotational angle of the inside front steering wheel.

Figure 7 shows that schematic of the vehicle actual steering process when the rotational angle of the inside steering wheel is α. The dotted line represents the positional relationship of the steering trapezium mechanism when the steering is not started. Using the geometry in Figure 7, we get

$$N^2 = M^2 + m^2 - 2mM\cos(\theta - \alpha) \tag{29}$$

where α is the rotational angle of the inside front steering wheel, M is the distance between the left and right vertical shafts, m is the steering arm length of the steering trapezoid mechanism, θ is the bottom angle of the steering trapezoid mechanism, and N is the length of the auxiliary line. We also have

$$S^2 = N^2 + m^2 - 2mN\cos\delta_1 \tag{30}$$

where S is the length of the tie rod and δ_1 and δ_2 are the interior angles of auxiliary calculation.

Using (29), (30), and Figure 5, we get

$$\beta' = \delta_1 - (\theta - \delta_2) = \delta_1 + \delta_2 - \theta \tag{31}$$

$$\delta_2 = \arcsin\frac{GE3}{N} = \arcsin\frac{m\sin(\theta - \alpha)}{N} \tag{32}$$

where β is the theoretical rotational angle of the external front steering wheel and β' is the ideal rotational angle of the external front steering wheel.

We find

$$\beta' = \arccos \cdot \frac{M^2 + 2m^2 - (M - 2m\cos\theta)^2 - 2Mm\cos(\theta - \alpha)}{2m\sqrt{m^2 + M^2 - 2Mm\cos(\theta - \alpha)}} + \arcsin\frac{m\sin(\theta - \alpha)}{\sqrt{m^2 + M^2 - 2Mm\cos(\theta - \alpha)}} - \theta \tag{33}$$

In order to ensure the steering performance of the vehicle, the actual rotational angle function of external front steering wheel should be as close as possible to the ideal rotational angle in the process of vehicle swerve. Hence, the optimization problem of vehicle steering trapezoidal structure can be written as

$$\min(F(X)) = \min\left(\sum_{\alpha_i=1}^{\alpha_{\max}} \left|\beta - \beta'\right| \omega(\alpha_i)\right) \tag{34}$$

where α is the rotational angle of the inside front steering wheel, α_{max} is the maximum rotational angle of the inside front steering wheel, α_I is the angle number corresponding to the rotational angle of the inside front steering wheel, and is a natural number between 1 and α_{max}, which is dimensionless; $\omega(\alpha_i)$ is a weighting function.

The computational method is as follows:

$$\omega(\alpha_i) = \begin{cases} 1.25, & 1 \le \alpha_i \le 10 \\ 0.90, & 10 < \alpha_i \le 20 \\ 0.45, & \alpha_i > 20 \end{cases} \tag{35}$$

The design variables selected by the objective function are the steering trapezoidal bottom angle θ and the trapezoidal arm length m, to be represented by X=[m,θ]. According to the design information in the literature [51, 52], the constraint condition of vehicle steering trapezium mechanism is given by

$$0.11M \le m \le 0.15M \tag{36}$$

$$\arctan\frac{1.2L}{M} \le \theta \le \frac{4\pi}{9} \tag{37}$$

$$\frac{(M - 2m\cdot\cos\theta)^2 - M^2 + 2Mm\cos\left(\theta + \arcsin\left(L/\left(R_{\min} - a\right)\right)\right)}{2m\cdot(M - 2m\cdot\cos\theta)} - \cos\left(\frac{7\pi}{9}\right) \le 0 \tag{38}$$

Table 3: Parameters of the vehicles.

Models	Distance of rear and front axle/m	Center distance of two main axles/m	Distance of wheel and steering pin/m	Min radius of vehicles steering/m
Patrol GRX	2.9	1.605	0.4	6.1
Patrol GL	2.9	1.555	0.4	6.1
Nissan Duke	2.9	1.500	0.5	5.5

Table 4: Optimization results with the three vehicles.

Models	Optimal value of Steering arm length (m)	Ideal value of Steering arm length (m)	Optimal value of trapezoid bottom angle (rad)	Ideal value of trapezoid bottom angle (rad)	Optimal value of objective function (rad)	Ideal value of objective function (rad)
Patrol GRX	0.1770	0.1771	1.2781	1.2780	0.1873	0.1872
Patrol GL	0.1712	0.1711	1.2911	1.2900	0.1866	0.1864
Nissan Duke	0.1711	0.1713	1.2903	1.2902	0.1868	0.1867

5.2. Optimization Results. We experimented with Patrol GRX, Patrol GL, and Nissan Duke data. The parameters of the vehicles are shown in Table 3. IRCGA-2 is used to optimize the parameters of the steering trapezium mechanism.

The population size and the maximum number of iterations were chosen as 100 and 5000, respectively. The optimization results are shown in Table 4.

It can be seen from Table 4 that the absolute error between the optimal value of the trapezoidal arm length and the ideal value is within 0.0001 m, the relative error being within 0.1%. The absolute error value of the steering trapezoidal bottom angle is within 0.0005 rad, the relative error being within 0.1%. The errors between the optimal and ideal values of the objective function are thus within 0.1%.

6. Conclusions

ICRGA-2 to solve constrained global optimization problems is proposed. The improvements of IRCGA-2 are mainly in four categories: the selection operator, the crossover operator, the substitution operator, and the mutation operator.

When methods such as roulette wheel selection, tournament selection, and local selection are used to compute the individual's fitness function, inferior population diversity is developed at later stages of iterations. ICRGA-2 improves performance by incorporating a number of novelties as summarized below.

ICRGA-2 uses the SGS method, which does not need compute the fitness value, achieves superior population diversity, and is easy to realize.

A HNDX operator is developed. Compared with the DBX operator in [24], the HNDX operator can guarantee that the cross-generated offsprings are located near the better one among the two parents and that crossover direction is very close to the optimal crossover direction or consistent with the optimal crossover direction. Thus, HNDX can ensure that there is a great chance of generating better offsprings and thereby improving the convergence speed of IRCGA-2.

In most previous GA algorithms, as iterations increase, the same individuals are likely to appear in the population, making the population diversity worse. To avoid this problem, the substitution operation is added after the crossover so that there is no duplication of the same individual in the population.

The local search ability and the global search ability of different mutation operators are different. Some mutation operators have strong local search ability, and some have strong global search ability. A single mutation operator is difficult to take into account both local search and global search. This paper proposes a combined mutation method which makes the mutation operation not only take into account the local search ability but also the global search ability.

In the optimization problems with many local optima, ICRGA-1 is likely to lose population diversity after many iterations and converges thereby to a local optimum instead of the global optimum. In such problems, the population diversity of IRCGA-2 remains superior to that of IRCGA-1.

The computational results with nine examples show that IRCGA-2 has better performance than the other IRCGA methods with respect to all the measures of performance considered.

As an example application, the optimization model of the steering mechanism of vehicles is formulated and IRCGA-2 is used to optimize the parameters of the steering trapezoidal mechanism of three kind of vehicle types. Better results are obtained with IRCGA-2 as compared to other methods.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Project of the Social Science Foundation of Heilongjiang Province, China (Grant no. 16JYB06), in part by the National Social Science Foundation (Grant no. 13JY098), and in part by the subproject of the

National Science and Technology Support Program (Grant no. 2014BAD06B01-23).

References

[1] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*, University of Michigan Press, Oxford, UK, 1975.

[2] S. Zheng and J. Lai, "Improved real-coded hybrid genetic algorithm," *Computer Applications*, vol. 26, no. 8, 2006, https://wenku.baidu.com/view/da6a390c844769eae009edbf.html?re=view.

[3] H. H. Liu, C. Cui, and J. Chen, "An improved genetic algorithm for solving travel salesman problem," *Transactions of Beijing Institute of Technology*, vol. 33, no. 4, pp. 390–393, 2013, http://journal.bit.edu.cn/zr/ch/reader/create_pdf.aspx?file_no=20130413.

[4] J. An and H. Jin, "Research on Application of Self-adaptive Strategy in Real-coding Genetic Algorithm," *Microelectronics & Computer*, vol. 28, no. 4, pp. 140–146, 2011, http://www.ixueshu.com/document/cfb21286006ff07b318947a18e7f9386.html.

[5] R. Qi, F. Qian, W. Du et al., "Multi objective genetic algorithm based on elitist selection and individual migration," *Control and Decision*, vol. 22, no. 2, pp. 164–168, 2007, http://www.ixueshu.com/document/8f0dee7b4fffcbdd.html.

[6] K. A. de Jong, *An analysis of the behavior of a class of genetic adaptive systems [Ph.D. thesis]*, University of Michigan, Ann Arbor, Mich, USA, 1975.

[7] Q. Liu, X. Wang, Q. Fu et al., "Double Elite Co-evolutionary Genetic Algorithm," *Journal of Software*, vol. 23, no. 4, pp. 765–775, 1894, http://www.ixueshu.com/document/a6a052536c12b09b37a18e7f9386.html.

[8] S. Yu and S. Kuang, "Convergence and convergence rate analysis of elitist genetic algorithm based on martingale approach," *Control Theory & Applications*, vol. 27, no. 7, pp. 843–848, 2010, http://www.ixueshu.com/document/fc8b215a622f956e318947-a18e7f9386.html.

[9] D. E. Goldberg, "Genetic algorithms in search, optimization, and machine learning," *Choice Reviews Online*, vol. 27, no. 02, pp. 27-0936–27-0936, 1989.

[10] C. Q. Zhang, J. G. Zheng, and J. Qian, "Comparison of coding schemes for genetic algorithms," *Application Research of Computers*, vol. 28, no. 3, 2011, http://www.ixueshu.com/document/efe7a7bbb0212ca5318947a18e7f9386.html819-822.

[11] Z. Michalewicz, C. Z. Janikow, and J. B. Krawczyk, "A modified genetic algorithm for optimal control problems," *Computers & Mathematics with Applications*, vol. 23, no. 12, pp. 83–94, 1992, http://www.cs.umsl.edu/~janikow/publications/1992/GAforOpt/text.pdf.

[12] C. T. Chen, C. K. Wu, and C. Hwang, "Optimal Design and Control of CPU Heat Sink Processes," *IEEE Transactions on Components and Packaging Technologies*, vol. 31, no. 1, pp. 184–195, 2008, http://ukacc.group.shef.ac.uk/proceedings/control2006/papers/f81.pdf.

[13] C. T. Chen and Y. C. Chuang, "An Intelligent Run-to-Run Control Strategy for Chemical–Mechanical Polishing Processes," *IEEE Transactions on Semiconductor Manufacturing*, vol. 23, no. 1, pp. 109–120, 2010.

[14] J. D. Dyer, R. J. Hartfield, G. V. Dozier, and J. E. Burkhalter, "Aerospace design optimization using a steady state real-coded genetic algorithm," *Applied Mathematics and Computation*, vol. 218, no. 9, pp. 4710–4730, 2012.

[15] C.-W. Tsai, C.-L. Lin, and C.-H. Huang, "Microbrushless DC motor control design based on real-coded structural genetic algorithm," *IEEE/ASME Transactions on Mechatronics*, vol. 16, no. 1, pp. 151–159, 2011.

[16] K. Valarmathi, D. Devaraj, and T. K. Radhakrishnan, "Real-coded genetic algorithm for system identification and controller tuning," *Applied Mathematical Modelling*, vol. 33, no. 8, pp. 3392–3401, 2009.

[17] D. E. Goldberg, K. Deb, and J. H. Clark, "Genetic algorithms, noise, and the sizing ofpopulation," *Complex Systems*, vol. 6, pp. 333–362, 1992, http://repository.ias.ac.in/82725/1/5-a.pdf.

[18] Á. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 124–141, 1999.

[19] V. K. Koumousis and C. P. Katsaras, "A saw-tooth genetic algorithm combining the effects of variable population size and reinitialization to enhance performance," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 19–28, 2006.

[20] M. Affenzeller, S. Wagner, and S. Winkler, "Self-adaptive population size adjustmentfor genetic algorithm," *Lecture Notes in Computer Science*, vol. 4739, pp. 820–828, 2007, https://doi.org/10.1007/978-3-540-75867-9_103.

[21] M. M. Meysenburg and J. A. Foster, "The quality of pseudo-random number generators and simple genetic algorithm performance in," in *Proceedings of Seventh International Conference on Genetic Algorithms (ICGA-7)*, pp. 276–281, East Lansing, MI, USA, 1997, https://scholar.google.com/scholar?q=The+quality+of+pseudo-random+number+generators+and+simple+genetic+algorithm+performance&btnG=&hl=zh-CN&as_sdt=0%2C34.

[22] E. Cantú-Paz, "On random numbers and the performance of genetic algorithms in," in *Proceedings of Genetic and Evolutionary Computation Conference*, pp. 311–318, New York, NY, USA, 2002, http://dl.acm.org/citation.cfm?id=2955546.

[23] M. M. Meysenburg, D. Hoelting, D. McElvain, and J. A. Foster, "How random generator quality impacts genetic algorithm performance in," in *Proceedings of Genetic and Evolutionary Computation Conference*, pp. 480–487, New York, NY, USA, 2002, http://dl.acm.org/citation.cfm?id=295556.

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

[25] K. Deep and K. N. Das, "Performance improvement of real coded genetic algorithm with Quadratic Approximation based hybridisation," *International Journal of Intelligent Defence Support Systems*, vol. 2, no. 4, p. 319, 2009.

[26] H. Nakanishi, H. Kinjo, N. Oshiro, and T. Yamamoto, "Searching performance of a real-coded genetic algorithm using biased probability distribution functions and mutation," *Artificial Life and Robotics*, vol. 11, no. 1, pp. 37–41, 2007.

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

[28] P.-H. Tang and M.-H. Tseng, "Adaptive directed mutation for real-coded genetic algorithms," *Applied Soft Computing*, vol. 13, no. 1, pp. 600–614, 2013.

[29] G. Zhang, J. Ma, and C. Zhou, "Three Real-Coded Genetic Algorithms With New Mutation Operators," *International Journal of Computational Intelligence Systems*, 2007, https://scholar.google.com/scholar?q=Three+Real-Coded+Genetic+

Algorithms+With+New+Mutation+Operators&btnG=&hl=zh-CN&as_sdt=0%2C34.

[30] A. Singh and K. Deep, "Real Coded Genetic Algorithm Operators Embedded in Gravitational Search Algorithm for Continuous Optimization," *International Journal of Intelligent Systems Technologies & Applications*, vol. 7, no. 12, pp. 1–12, 2015.

[31] A. G. Yushchenko, D. A. Pashko, and O. V. Gatilova, "Evolutionary Strategies of a Genetic Algorithm for a Traveling Salesman and New York Taxi Driver Problems," in *Research Gate*, pp. 163–173, 2014, https://www.researchgate.net/publication/262685900_EVOLUTIONARY_STRATEGIES_OF_A_GENETIC_ALGORITHM_FOR_A_TRAVELING_SALESMAN_AND_NEW_YORK_TAXI_DRIVER_PROBLEMS.

[32] H. Kita, "A comparison study of self-adaptation in evolution strategies and real-coded genetic algorithms.," *Evolutionary Computation*, vol. 9, no. 2, pp. 223–241, 2001.

[33] M. J. Mendes, "A comparative study of crossover operators for genetic algorithms to solve the job shop scheduling problem," *Wseas Transactions on Computers*, vol. 12, no. 4, pp. 164–173, 2013, https://pdfs.semanticscholar.org/3bda/cac99759ca1c71e241b815ea50226b05af70.pdf.

[34] P. Subbaraj, R. Rengaraj, and S. Salivahanan, "Enhancement of combined heat and power economic dispatch using self adaptive real-coded genetic algorithm," *Applied Energy*, vol. 86, no. 6, pp. 915–921, 2009.

[35] N. Amjady and H. Nasiri-Rad, "Solution of nonconvex and nonsmooth economic dispatch by a new Adaptive Real Coded Genetic Algorithm," *Expert Systems with Applications*, vol. 37, no. 7, pp. 5239–5245, 2010.

[36] P. Subbaraj, R. Rengaraj, and S. Salivahanan, "Enhancement of Self-adaptive real-coded genetic algorithm using Taguchi method for Economic dispatch problem," *Applied Soft Computing*, vol. 11, no. 1, pp. 83–92, 2011.

[37] Y.-C. Chuang, C.-T. Chen, and C. Hwang, "A real-coded genetic algorithm with a direction-based crossover operator," *Information Sciences*, vol. 305, pp. 320–348, 2015.

[38] D. Wang and S. C. Fang, "Just-In-Time production planning with Semi-infinite programming model and genetic algorithm," *Control and Decision*, vol. 11, no. 4, pp. 446–451, 1996, http://www.ixueshu.com/document/738ad652ae6307d7318947a18e7f9386.html.

[39] R. Peltokangas, A. Sorsa, and K. Leiviskä, "Real-coded genetic algorithms and nonlinear parameter identification," *University of OULU Control Engineering Laboratory Report A*, vol. 34, pp. 1–28, 2008, http://jultika.oulu.fi/files/isbn9789514287862.pdf.

[40] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, New York, NY, USA, 1996.

[41] R. A. Mäkinen, J. Periaux, and J. Toivanen, "Multidisciplinary shape optimization in aerodynamics and electromagnetics using genetic algorithms," *International Journal for Numerical Methods in Fluids*, vol. 30, no. 2, pp. 149–159, 1999, https://pdfs.semanticscholar.org/f108/0b7fc97bebc8829277d9f9a87d5082ac8bd4.pdf.

[42] K. Deep and M. Thakur, "A new mutation operator for real coded genetic algorithms," *Applied Mathematics Computation*, pp. 211–230, 2007, https://doi.org/10.1016/j.amc.2007.03.046.

[43] Z. Michalewicz, T. D. Logan, and S. Swaminathan, "Evolutionary operators for continuous convex parameter spaces," *Revista Española De Anestesiología Y Reanimación*, vol. 52, pp. 377-378, 1994, https://cs.adelaide.edu.au/~zbyszek/Papers/p1.pdf.

[44] K. S. Nawaz Ripon, S. Kwong, and K. F. Man, "A real-coding jumping gene genetic algorithm (RJGGA) for multiobjective optimization," *Information Sciences*, vol. 177, no. 2, pp. 632–654, 2007.

[45] Y. Q. Hu, *Operations Research Tutorial*, Tsinghua University Press, 4th edition, 2012.

[46] X. F. Xie, W. J. Zhang, and Z. L. Yang, "A parents selection strategy fighting premature convergence in floating genetic algorithms," *Control and Decision*, vol. 17, no. 5, pp. 625–628, 2002, http://www.wiomax.com/team/xie/paper/CCDC02.pdf.

[47] X. H. Wang, X. Y. Liu, and M. H. Bai, "Population migration algorithm with Gaussian mutation and the steepest descent operator," *Computer Engineering and Applications*, vol. 45, no. 20, pp. 57–60, 2009, http://www.ixueshu.com/document/18d8ba8fddd8106e.html.

[48] J. J. Li and Y. M. Dai, "Adaptive shuffled frog leaping algorithm adopting mixed mutation," *Computer Engineering and Applications*, vol. 49, no. 10, pp. 58–61, 2013, http://www.ixueshu.com/document/41e319b7957f8b4b318947a18e7f9386.html.

[49] Y. Zhao, Y. G. Cai, and D. F. Cheng, "A novel local exploitation scheme for conditionally breeding real-coded genetic algorithm," *Multimedia Tools & Applications*, vol. 76, pp. 17955–17969, 2017.

[50] Y. T. R. D. Zhang, D. Y. Xu et al., "Prediction and Analysis of Agricultural Machinery Total Power Based on Real - coded Genetic Algorithm and Neural Network," *Journal of Agricultural Mechanization Research*, vol. 7, pp. 13–18, 2017.

[51] Y. M. Zhou, *Tractor Ergonomics*, BeiJing:China Agriculture Press, 1998.

[52] X. X. Chen, H. M. Lv, and Q. Wang, "Optimum design of ackerman steering trapezium mechanism," *Mechinery*, vol. 34, no. 3, pp. 35–37, 2007, http://www.ixueshu.com/document/b2a017848ff82e68318947a18e7f9386.html.