Hindawi
Complexity
Volume 2020, Article ID 4609489, 18 pages
https://doi.org/10.1155/2020/4609489

Research Article

Analysis of Multitasking Evolutionary Algorithms under the Order of Solution Variables

Lei Wang,[1,2] Qian Sun,[1] Qingzheng Xu ,[3,4] Wei Li,[1] and Qiaoyong Jiang[1]

[1]*School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China*
[2]*Shaanxi Key Laboratory for Network Computing and Security Technology, Xi'an 710048, China*
[3]*College of Information and Communication, National University of Defense Technology, Xi'an 710106, China*
[4]*Youth Innovation Team of Shaanxi Universities, National University of Defense Technology, Xi'an 710106, China*

Correspondence should be addressed to Qingzheng Xu; xuqingzheng@hotmail.com

Received 21 July 2020; Revised 3 September 2020; Accepted 21 September 2020; Published 14 October 2020

Academic Editor: Kailong Liu

Recently, it was demonstrated that multitasking evolutionary algorithm (MTEA), a newly proposed algorithm, can solve multiple optimization problems simultaneously through a single run, breaking through the limitations of traditional evolutionary algorithms (EAs), with good convergence and exploration performance. As a novel algorithm, MTEA still has a lot of unexplored space. Generally speaking, the order of solution variables has no significant influence on the single-tasking EAs. To our knowledge, the effect of the order of variables in the multitasking scenario has not been explored. To fill in this research gap, three orders of variables in the multitasking scenario are proposed in this paper, including full reverse order, bisection reverse order, and trisection reverse order. An important feature of these orders of variables is that an individual can recover as himself after two times of changing the order of variables. In order to verify our idea, these orders of variables are embedded into MTEA. The experiment results revealed that the effect of the different orders of variables is universal but not significant enough in the practical application. Furthermore, tasks with high similarity and high degree of intersection are sensitive to the order of variables and get great impact between tasks.

1. Introduction

Optimization problems exist in all fields of science, engineering, and industry. In many cases, such optimization problems involve various decision variables, complex structured goals, and multiform constraints [1, 2]. In general, traditional mathematical optimization techniques will encounter difficulties in solving such practical optimization problems in their original form. Inspired by the natural selection mechanism of "survival of the fittest" and the laws of genetic information transmission in the process of biological evolution, evolutionary algorithms (EAs) were proposed to solve this kind of complex optimization problem. Evolutionary algorithms simulate the process of species reproduction through program iteration and consider the problem to be solved as the environment, seeking the optimal solution through natural evolution. EA has many advantages such as being powerful, efficient, flexible, and reliable, and the research work of evolutionary algorithm has been very rich [3–6]. Thus, evolutionary algorithms have been widely used in real-life applications, including battery health diagnosis and management [7–10].

The emergence of evolutionary algorithms has helped solve many tricky practical problems, but there is still room for improvement. In practice, real-world problems are rarely isolated, but some of them are similar or related. Therefore, we can handle some similar or complementary problems at the same time. As a new paradigm in evolutionary computation, multitasking evolutionary algorithm (MTEA) is very different from the traditional evolutionary algorithm, which utilizes experience (individual and global optimal guidance) and selective pressure intelligence to solve single-objective or multiobjective problems [11]. The biological basis of EA is single gene inheritance [12, 13]. In contrast, MTEA incorporates the concept of coordinated evolution between genes and cultures. Intuitively, multitasking is the equivalent of a multigene environment. The task can use the similarity of different gene evolution to learn information which is beneficial to its own evolution when multiple tasks are optimized at the same time. Obviously, knowledge drawn from past learning experiences can be constructively applied to more complex or invisible tasks. Multitasking optimization (MTO) significantly improves the efficiency of evolutionary optimization by solving multiple problems at once [11–13]. In the process of MTO, both gene migration and diversity play an important role when multiple tasks are optimized at the same time. In contrast, positive or negative gene migration demonstrates the nature of MTO [14]. In literature [13], MTEA is used to solve the practical application of multiobjective problems. At the same time, MTEA has a good universality and can be combined with operators with good search ability in evolutionary algorithms. For example, both the classical evolutionary algorithm and particle swarm optimization (PSO) can be combined with the MTEA [15].

In traditional single-tasking optimization, the order of variables has less significant influence on the solution process [16–18]. Undoubtedly, under the action of selective pressure, the solution will eventually approach the global optimum [12]. For single-objective optimization and multiobjective optimization problems, the order of variables does not obviously play a major role in EAs. In a sense, the implication of new order of variables for a special objective function can be thought of as a different objective function with the same optimal solution value. As a result, it has been intentionally or unintentionally ignored in the community of evolutionary algorithm and optimization. On the contrary, the situation is significantly different for MTO problems. In MTO, the optimization process of one task can affect the optimization process and the results of other tasks. At present, the influence of the order of variables on MTO has not been studied. Keeping this in mind, we demonstrate the effect of variable order on single-tasking and multitasking evolutionary algorithms in this paper.

In the experimental part, three kinds of transformation methods are designed to explore the influence of the order of variables on the quality of the solution in multitasking scenarios. The experiments show that the variable order affects the multitasking optimization process, and the degree of influence is related to the correlation degree between tasks, such as the task similarity and the same magnitude of the optimal solution between tasks. As a result, the actual impact is not significant. Furthermore, experiments have also shown that this effect is universal in the multitasking scenario.

To summarize, the core contributions of the current work are multifaceted, which are outlined as follows. (1) The influence of the order of solution variables on single-tasking optimization problems and multitasking optimization problems is analyzed in the view of evolutionary mechanism. The order of variables has no impact on the single-tasking evolutionary algorithm while has an impact on multitasking evolutionary algorithm. (2) Three orders of variables are proposed in this paper, including full reverse order, bisection reverse order, and trisection reverse order. An important feature of these orders of variables is that an individual can recover as himself after two times of changing the order of variables. (3) In order to verify our idea, these orders of variables are embedded into MTEA. The experiment results revealed that the effect of the different orders of variables is universal but not significant enough in the practical application. Furthermore, tasks with high similarity and high degree of intersection are sensitive to the order of variables and get great impact between tasks.

The rest of this paper is organized as follows. Section 2 gives a brief overview of the original multifactorial evolutionary algorithm (MFEA) and introduces MTEA based on the multipopulation evolution model. In addition, multifactorial differential evolution (MFDE) and the related works of MTEA are also explained in this part. In Section 3, the influence of the order of solution variables on single-tasking optimization and multitasking optimization algorithms is demonstrated. Besides, three orders of variable are also introduced here. Next, we carried out relevant experiments to verify our conjectures including the design ideas of the experiment and the discussion of the experimental results in Section 4. Finally, Section 5 summarizes this paper and looks forward to the future research field.

2. Background

2.1. Multifactorial Evolutionary Algorithm. In practice, many problems are related to each other to varying degrees. In other words, universal similarity between problems is the motivation for multitasking optimization. With the utilizability of similarity, the multitasking optimization algorithm makes solving multiple problems at the same time come true. Assume that there are K tasks: T_1, T_2, …, and T_K. A realistic task T_j can be represented by the function $f_j(x)$; here $j \in \{1, 2, 3, \ldots, |K|\}$. Mathematically, a multitasking optimization problem can be expressed as $G(X) = \min\{f_1(x), f_2(x), \ldots, f_K(x)\}$. MFEA maps multiple problems into a unified space by the uniform random-key scheme [13], and each individual in the search space has the following four characteristics.

Definition 1. Factorial fitness: the factorial cost φ_i^k denotes the objective fitness or value of an individual p_i on a particular task T_k. Every individual will calculate K factorial fitness based on K tasks.

Definition 2. Factorial rank: the factorial rank r_i^k simply denotes the index of individual p_i in the list of population members which is sorted in ascending order with respect to their factorial costs on task T_k. It should be noted that in the process of multiobjective multitasking optimization, the factor ordering is calculated according to the nondominant ordering and the crowding distance. This paper only focuses on single-objective multitasking optimization, and thus it will not be stated here.

Definition 3. Skill factor: the factorial rank τ_i of individual p_i represents the task corresponding to the most advanced index in the order of factorial rank. Skill factor is regarded as the computational equivalent of cultural characteristics. In the principle of meme calculation, the cultural characteristics of one individual can be transmitted to another.

Definition 4. Scalar fitness: the scalar fitness of individual p_i is calculated by $\gamma_i = 1/\tau_i$.

As shown in Algorithm 1, it is assumed that K optimization tasks have to be performed simultaneously. First, we initialize N individuals in the search space Y and then evaluate the initial population *current-pop* by calculating the factorial fitness φ_i^k of each individual p_i in the *current-pop*, where $i \in \{1, 2, 3, \ldots, |N|\}$. Then, we calculate the skill factor of p_i in the population according to φ_i^k. After the initialization, as shown in line 5 in Algorithm 1, the iteration begins to produce the *offspring-pop* which contains N children according to Algorithm 2. Algorithm 3 is used to assign skill factors to each individual. Next, the offspring and the parent are merged to form the *transitional-pop* which have *2N* individuals, and then the N best individuals are selected as the new population for the next generation.

The traditional genetic algorithm applies crossover and mutation operators, which are also used in the multitasking evolutionary algorithm. Furthermore, how the offspring are produced depends on the skill factor of the parents and random mating probability (*rmp*). The detailed description is given in Algorithm 2. The offspring will be produced using crossover directly when both parents have the same skill factors. Otherwise, either the offspring will be generated through crossover given the random mating probability or the offspring will produced by mutation when the parents have different skill factors. The parameter *rmp* permits cross-cultural mating among different tasks. A larger *rmp* means more knowledge exchanging between two tasks, while a smaller value indicates the opposite. An appropriate *rmp* can balance the thorough scanning of small areas in the search space with the exploration of the whole space. Unlike traditional evolutionary algorithms, it is not evaluated straightforward after offspring have been generated. After the offspring generation, skill factors are assigned to the offspring according to the mating mode of the offspring selection, and this procedure is detailed in Algorithm 3.

2.2. MTEA as Multipopulation Evolution Model.

Different from the classical MFEA, the multipopulation multitasking evolutionary algorithm no longer adopts one population but initializes K subpopulations according to the number of tasks. Individuals in a subpopulation evolve for a specific task throughout the optimization process [19].

Figure 1 illustrates a multipopulation optimization model with two optimization tasks. Clearly, task 1 and task 2 have their own populations. The dotted lines in Figure 1 represent possible scenarios. In reproductive selection, parents may come from the same task-specific subgroups or other groups. In this way, knowledge sharing and optimization efficiency can be improved during reproduction. A core feature of the multipopulation evolution model is that the crossover or mutation operators are deemed to help exchange information and assist to find the promising solutions.

Another important feature is that parental and progeny individuals must belong to the same subpopulation and evolve within the same subpopulation. One of the benefits of this is to keep the population as stable as possible. At the end of optimization, the optimal solution of subpopulation corresponding to the task is solved.

It should be noted here that interpopulation crossover probability (*icp*) in MTEA which controls the density of knowledge transfer between different tasks is different from *rmp* in MFEA. Parameter *icp* in MTEA directly controls knowledge transfer between tasks, whereas *rmp* is used to control knowledge sharing when parents have skill factors in MFEA.

2.3. Differential Evolution and Multifactorial Differential Evolution.

As a branch of stochastic EAs, differential evolution (DE) originally proposed by Price and Storn in 1995 [20] has been proven to be an effective, robust, and reliable global optimizer. DE distinguishes itself from other EAs with the individual difference-based mutation and crossover. DE produces a new candidate solution component based on the weighted difference between two randomly selected population individuals that is added to a third individual. For the original DE, mutation, crossover, and selection are the three key components in DE which are described as follows.

The mutation operation enables DE to explore the search space and maintain diversity. In [21], five mutation strategies have been commonly used, which are minutely given as follows:

$$\begin{aligned}
\text{DE/rand/1: } V_{i,g} &= x_{r1,g} + F \times \left(x_{r2,g} - x_{r3,g}\right), \\
\text{DE/best/1: } V_{i,g} &= x_{\text{best},g} + F \times \left(X_{r1,g} - X_{r2,g}\right), \\
\text{DE/current-best/1: } V_{i,g} &= X_{i,g} + F \\
&\quad \times \left(X_{\text{best},g} - X_{i,g} + X_{r1,g} - X_{r2,g}\right), \\
\text{DE/best/2: } V_{i,g} &= X_{\text{best},g} + F \\
&\quad \times \left(X_{r1,g} - X_{r2,g} + X_{r3,g} - X_{r4,g}\right), \\
\text{DE/rand/2: } V_{i,g} &= X_{r1,g} + F \\
&\quad \times \left(X_{r2,g} - X_{r3,g} + X_{r4,g} - X_{r5,g}\right),
\end{aligned} \tag{1}$$

where $V_{i,g}$ denotes the mutant vector with respect to each individual $X_{i,g}$ at generation g, D is the dimension of problem, $r1$, $r2$, $r3$, $r4$, and $r5$ are random and mutually exclusive integers chosen from the interval [1, D], F is the scaling factor which controls the amplitude of the difference vector, and $X_{\text{best},g}$ gives the best individual found so far at generation g.

The goal of crossover operator is to build trial vectors by recombining the current vector and the mutant one. The family of DE algorithms employs two crossover schemes: exponential crossover and binomial crossover. The binomial

```
(1) Generate $N$ individuals in $Y$ to form initial population $P_0$ as the current-pop ($C$).
(2) Calculate $\varphi_i^k$ and of each individual $p_i$ in the current-population and then get the factorial rank $r_i^k$ of each individual.
(3) Compute the skill factor $\tau_i$ for each $P_i$.
(4) Set $gen = 0$.
(5) while (stopping conditions are not satisfied) do
    Generate offspring-pop ($O$) according to Algorithm 2.
    Evaluate the individuals in offspring-pop according to Algorithm 3.
    Combine $C$ and $O$ into transitional-pop ($T$).
    For $P_i$ in $T$
      Update the scalar fitness ($\gamma_i$) and skill factor ($\tau_i$) of $P_i$.
    end
      Select $N$ fittest members from $T$ to form $C$.
    Set $gen = gen + 1$
(6) end while
```

Algorithm 1: Basic framework of the MFEA.

```
Consider candidate parents $c1$ and $c2$ in $C$
(1) Generate a random number rand between 0 and 1.
(2) if $\tau_1 == \tau_2$ or $rand < rmp$ then
        $(o1, o2) = \text{Crossover} + \text{Mutate}\ (c1, c2)$
(3) else
        $o1 = \text{Mutate}\ (c1)$
        $o2 = \text{Mutate}\ (c2)$
(4) end if
```

Algorithm 2: Assortative mating.

```
Consider offspring $o \in O$
(1) Generate a random number rand between 0 and 1.
(2) if $r = \text{Crossover} + \text{Mutate}\ (c1, c2)$ and $rand \leq 0.5$
        $o$ imitates skill factor of $c1$
(3) else if $o = \text{Crossover} + \text{Mutate}\ (c1, c2)$ and $rand > 0.5$
        $o$ imitates skill factor of $c2$
(4) else if $r = \text{Mutate}\ (c1)$
        $o$ imitates skill factor of $c1$
(5) else
        $o$ imitates skill factor of $c2$
(6) end if
```

Algorithm 3: Vertical cultural transmission via selective imitation.

crossover is utilized in this paper and is briefly discussed below. In binomial crossover, the trial vector $U_{i,g} = (u_{i,g}^1, u_{i,g}^2, \ldots, u_{i,g}^D)$ is defined as follows:

$$u_{i,j,g} = \begin{cases} v_{i,j,g}, & \text{if (rand} < \text{CR) or } (j = \text{rand}_j), \\ x_{i,j,g}, & \text{otherwise,} \end{cases} \tag{2}$$

where $\text{CR} \in [0, 1]$ is the predefined crossover rate, *rand* is a uniform random number within [0, 1], and $\text{rand}_j \in \{1, 2, \ldots, D\}$ is a randomly selected index which is used to ensure that at least one dimension of trial vector is changed.

After the crossover, a greedy selection mechanism is used to select the better one between the parent vector $X_{i,g}$ and the trial vector $U_{i,g}$ according to their fitness values as described below.

$$X_{i,g+1} = \begin{cases} U_{i,g}, & \text{if } f(U_{i,g}) \leq f(X_{i,g}), \\ X_{i,g}, & \text{otherwise.} \end{cases} \tag{3}$$

In recent research, MTO has been conducted with differential evolution, named multifactorial differential evolution (MFDE). MFDE is similar to the classic MFEA process, except for the operation of children production. The

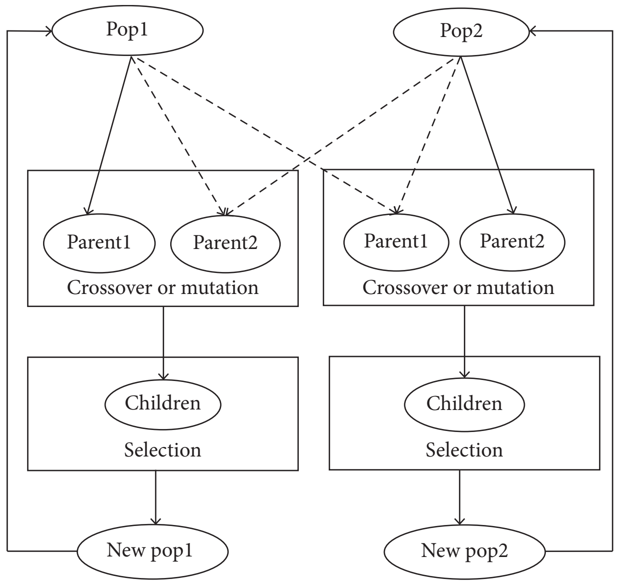


FIGURE 1: Model of multipopulation evolution for two tasks.

```
(1) **Generate** a random number *rand* between 0 and 1.
(2) **if** $rand < rmp$ **then**
        $V_i^g = V_{r_1}^g + F * (x_{r_2}^{\prime g} - x_{r_3}^{\prime g})$
(3) **else**
        $V_i^g = V_{r_1}^g + F * (x_{r_2}^g - x_{r_3}^g)$
```

ALGORITHM 4: Assortative mating in MFDE.

assortative mating of MFDE for producing children is summarized in Algorithm 4. Firstly, in the first generation, a random number is generated in the range of [0, 1]. Then, if this arbitrary number is less than *rmp*, two solutions $x_{r_2}^{\prime g}$ and $x_{r_3}^{\prime g}$ which have different skill factor with $V_{r_1}^g$ will be randomly selected to generate the new solution V_i^g. $V_{r_1}^g$ is a randomly selected solution that shares common skill factor with V_i^g. Otherwise, $V_{r_1}^g$ will be generated by the original *DE/rand/*1 strategy.

2.4. Related Works on MTEA. Since MFEA was first proposed in 2016, the research work of multitasking EA has gained wide attention. As a universal framework, multitasking EA can be implemented in a variety of ways [22]. At present, the main research directions of this community are algorithm framework, algorithm improvement, and typical application. At present, the algorithm framework is mainly divided into two frameworks, one is the multitasking optimization algorithm framework based on dynamic subgroups, and the directional optimization task of individuals in the population will continuously change during the whole optimization process [23]. The other is that individuals in the population will directly assign a task based on the number of optimized tasks and follow the assigned task throughout the optimization process. Experimental results showed that the multitasking evolutionary algorithm based on multipopulation is also very advantageous [22, 24–26].

Many improvements to the multitasking optimization algorithm have been proposed [27–30]. Generally speaking, the operation of knowledge transfer between different tasks has a crucial impact on the algorithm performance [31]. Migration mode and frequency are the directions of multitasking optimization research. Multitasking evolutionary algorithm can also be combined well with other evolutionary algorithms to absorb the advantages of other EAs. For example, differential evolutionary and particle swarm optimization can well combine with MTEA and perform better than MFEA [32–34]. In addition, the improved crossover operator and search mechanism can be combined with a MTEA prejudice to improve the algorithm performance [35]. At the same time, the idea of machine learning can be well combined with multitasking optimization [36, 37].

Multitasking optimization algorithm is not only proved to be feasible in theory [38] but also shows good efficiency in practical problems. Multitasking algorithm can be used to solve complex engineering design and expensive optimization problems. In the literature, there exist a lot of works to apply MTEA to tackle real-world problems, such as vehicle routing problem [39–42], optimization and control of photovoltaic systems [43], bilevel optimization problem [44], complex supply chain network management [45], double-pole balancing problem [46], and composite manufacturing problem [47].

3. Order of Variables on Optimization Problem

3.1. Single-Task Optimization Evolution Model. We take single-objective optimization problem as an example to investigate the effect of the order of variables. In general, an optimization problem can be formulated as

$$\text{minimize } f(x), \quad x = (x_1, x_2, \ldots, x_D) \in R^D, \tag{4}$$

subject to

$$\begin{aligned} h_i(x) &= 0, \quad (i = 1, 2, \ldots, I), \\ g_j(x) &\leq 0, \quad (j = 1, 2, \ldots, J), \end{aligned} \tag{5}$$

where $h_i(x)$ and $g_j(x)$ are the equality constraints and inequality constraints, respectively.

When the order of variables of candidate solution x is changed, the new solution x^{new} can be identified as an individual of the other function $f_{\text{new}}(x)$. It is noteworthy that two functions have the same search space and optimal solution (after coordinate transformation). For example, as shown in Figure 2, $x_A = (1, 2)$ and $x_B = (3, 4)$ are two individuals of function $f(x) = x_1 + x_2^2$. After changing their order of variables, $x_A^{\text{new}} = (2, 1)$ and $x_B^{\text{new}} = (4, 3)$ are two individuals of function $f_{\text{new}}(x) = x_1^2 + x_2$. Obviously, they have identical function values, respectively.

Even more important is that, as illustrated in Figure 2, no matter what genetic mechanism (crossover, mutation, etc.) is involved for a given evolutionary algorithm, their offspring are identical due to the same input of the given genetic mechanism. From this, we can discuss the effect of

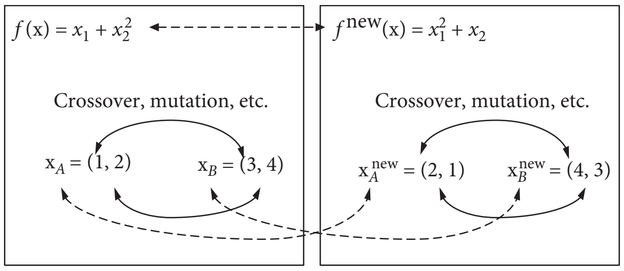

$f(x) = x_1 + x_2^2$
$f^{\text{new}}(x) = x_1^2 + x_2$
Crossover, mutation, etc.
Crossover, mutation, etc.
$x_A = (1, 2)$
$x_B = (3, 4)$
$x_A^{\text{new}} = (2, 1)$
$x_B^{\text{new}} = (4, 3)$

Figure 2: Corresponding relationship between two individuals before and after changing the order of variables for single-objective optimization problem.

order on single-task optimization. We have no reason to doubt that, although the change in the order of variables is quite a transformation of the objective function in a single-task optimization scenario, there is virtually no change in the optimization problem itself. Actually, the optimization result obtained by any evolutionary algorithm will not be influenced by changing the order of variables.

3.2. Multitasking Optimization Problem. Mathematically, a multitasking optimization problem can be defined as follows:

$$\{x_1, x_2, \ldots, x_K\} = \operatorname{argmin}\{f_1(x_1), f_2(x_2), \ldots, f_K(x_K)\}, \tag{6}$$

where $f_k(x_k)$: $X_k \longrightarrow R^D$ represents the k-th optimization task with search space X_k and $x_k = \{x_{k,1}, x_{k,2}, \ldots, x_{k,D_k}\}$ is a feasible solution in the solution space, in which D_k is the dimensionality of search space X_k.

In a multitasking optimization problem, the effect of changes in the order of variables is no longer insignificant. Before analyzing, we need to explain the concepts of order-dependent functions and order-independent functions. According to the role of variable order, the optimization functions can be divided into two classes: order-independent function and order-dependent function. For order-independent functions, the order of variables does not have an effect on the objective function of special forms. For instance, for Sphere function $f(\mathbf{x}) = \sum_{i=1}^{D} x_i^2$, no matter how the order of its variables changes, the objective function is kept unchanged. Thus, it cannot influence the performance of any multitasking optimization algorithm, including MFEA. Correspondingly, the order of variables has an effect on order-dependent functions, such as Rosenbrock function $f(x) = \sum_{i=1}^{D}(100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$.

Here, we take a simple example to further understand the variable order change on multitasking. As shown in Figure 3, we take 2-task optimization as an example to further investigate the effect of the order of variables. Two objective functions are $f_1(\mathbf{x}) = x_1 + x_2^2$ and $f_2(\mathbf{x}) = x_1^3 + x_2^4$. Expressly, two functions can change their order of variables in the same way. However, this is a trivial case that is not of interest. Thus, only the order of variables of the first function $f_1(\mathbf{x})$ is changed in the next discussion. Note that the order of variables has an effect on two functions and the first function is selected to change the order of its variables.

Generally speaking, for any multitasking optimization algorithm, two parent candidates will undergo intrapopulation and interpopulation reproduction processes, such as crossover in MFEA. Based on the analysis in Section 3.1, when the individuals in population undergo intragenetic mechanism, they will produce the same offspring even after changing the order of variables. At the same time, these individuals may undergo intergenetic mechanism. In this case, two parent candidates come from different populations. As shown in Figure 3, two individuals x_A and x_C undergo inter-crossover in the original situation, and two individuals x_A^{new} and x_C undergo inter-crossover after changing the order of variables. Due to the changed order of variables of the first objective function $f_1(\mathbf{x})$, they will produce different offspring, such as dotted lines ① vs. ⑤, ② vs. ⑥, ③ vs. ⑦, and ④ vs. ⑧ in Figure 3. Thus, it is obvious that ① and ⑤ have a wealth of opportunity to produce different offspring through cross mutation operator. The same one goes for the other three control groups. Therefore, changes in the order of variables in a multitasking scenario are analyzed to have a real impact on the optimization process.

3.3. Three Orders of Variables. In this paper, our purpose is to study the influence of the order of solution variables on the multitasking optimization. As we have analyzed before, the order of variables makes the original function become a new one, so the way to solve the order transformation has an impact on the optimization process. In our work, we design and study three orders of variables. They are defined as equations (7)–(9). For the first order named full reverse order, all variables of function are coded in reverse order. Similarly, for the bisection/trisection reverse order, all variables are divided into two/three parts evenly and then coded in reverse order in each part.

$$FuR(x) = (x_D, x_{D-1}, \ldots, x_2, x_1), \tag{7}$$

$$BiR(x) = \left(x_{(D/2)}, \ldots, x_1, x_D, \ldots, x_{(D/2-1)}\right), \tag{8}$$

$$TrR(x) = \left(x_{(D/3)}, \ldots, x_1, x_{(2D/3)}, \ldots x_{(D/3)-1}, x_D, \ldots, x_{(2D/3+1)}\right). \tag{9}$$

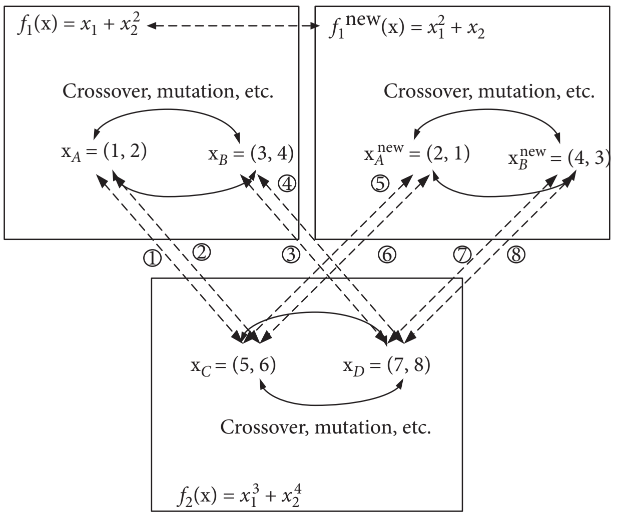


Figure 3: Corresponding relationship between two individuals before and after changing the order of variables for multitasking optimization problem.

There are two advantages of these variable order transformations. On the one hand, the important feature of these orders of variables is that an individual can recover as himself after changing the order of variables twice. Let $\mathbf{x} = (x_1, x_2, \ldots, x_D)$ be an arbitrary point in D-dimensional space; mathematically, $\mathbf{x} = FuR(FuR(\mathbf{x}))$, $\mathbf{x} = BiR(BiR(\mathbf{x}))$, and $\mathbf{x} = TrR(TrR(\mathbf{x}))$. On the other hand, the complexity of the transformation of the incremental design solution can lay a foundation for studying the influence of the complexity of the solution order transformation on the optimization process and facilitate analysis. Specifically, the corresponding output of the transformation of the order of solutions can be obtained by the above three equations.

Here, we need to make a small distinction between our designed transformation strategy for variable order and the strategy in [48]. First of all, the application scenarios are different. Literature [48] is about the multiobjective problem in single task, but our work mainly focuses on multiple tasks. Secondly, in [48], it studies the effect of principle selection and variable priority selection on convergence when different principles learned during innovization [49] involve different numbers of variables or different rules involve the same variables. The goal of the variable related strategy in [48] is to explore how dimensional repair order and repair rules combined can improve algorithm convergence to the greatest extent in a multiobjective environment. However, in this paper, three variable order change strategies are designed to explore the sensitivity of the whole multitasking optimization process to different variable orders, and the change of variables is not dynamic. Obviously, these three strategies are not intended to improve the convergence performance of MTEA.

4. Experiment Results and Discussion

4.1. Experiment Setup. In this paper, the basic algorithm adopted is multipopulation MFEA to highlight the direct effect of solution order in the multitasking scenario. In order to eliminate the randomness in individuals' generation, the initial seed of the random function is set to a fixed value so that it can always provide the same individual setting. In other words, the fixed result is obtained by using a given algorithm. In addition, MFDE is also used to verify the universality of the effect of the variable orders in the paper.

Seven commonly used optimization functions are used as components of 9 synthetic MTO problems. Among them, two functions (Rosenbrock and Griewank) are order-dependent functions that affect the four MTO problems. These problems can be divided into three categories: complete intersection (CI), partial intersection (PI), and no intersection (NI). Moreover, based on the similarity between the fitness landscapes, they can also be categorized into three groups: high similarity (HS), medium similarity (MS), and low similarity (LS). For more details about these benchmark problems, one can refer to the technical report [11]. For experimental convenience, task 1 and task 2 in the fifth task group are swapped so that the first task in all task groups is sequential dependent functions.

All parameter settings are listed as follows:

(1) Population size: $N_p = 100$

(2) Maximum number of function evaluations: $\text{MaxF} = 10^5$

(3) Parameter settings in MFDE

 (i) Differential amplification factor: $F = 0.5$
 (ii) Crossover probability: $\text{CR} = 0.9$
 (iii) Random mating probability: $rmp = 1.0$

(4) Parameter settings in MFEA

 (i) Index of simulated binary crossover: $\text{mu} = 2$
 (ii) Index of polynomial mutation: $\text{mum} = 5$
 (iii) Probability of mutation: $p_m = 1$
 (iv) Interpopulation crossover probability: $icp = 1.0$

It is noted that, in order to minimize the effect of stochastic nature on each measured metric, the reported result is the average over 50 trials. Lastly, the empirical studies presented in this paper are conducted under Windows10 using a computer with a 2.4 GHz Intel Corei5 processor and 4 GB RAM.

4.2. Parameter Sensitivity Analysis. In order to test the effectiveness of interpopulation crossover probability, experiments on a suite of single-objective multitasking benchmark problems are carried out in this part. First, the importance of parameter icp in the multitasking scenario is analyzed. After that, the analysis and discussion of the empirical results are also presented.

In the process of simultaneous optimization of multiple tasks, the similarity and complementarity between tasks play an important role [4]. High similarity between tasks tends to influence each other in a positive way, while complementarity helps tasks skip unnecessary searching. In the multifactorial optimization algorithm based on multipopulation, the crossover probability among the populations controls the influence of knowledge transfer between multiple tasks. The closer the crossover probability is to 1, the greater the mutual influence will be. Conversely, the smaller the probability of crossover between populations, the smaller the interaction between tasks.

When the similarity between task groups is low, the high probability of interpopulation crossover will make the optimization result counterproductive. Of course, high similarity and *icp* are not necessarily conducive to the optimization of the problem. For example, when an optimization problem falls into local optimization, other tasks may also fall into local optimization.

In this paper, we set the value of the *icp* as a linear increment to explore the influence of *icp* on MFEA. It should be noted that the intrapopulation crossover probability is set to 0.5 in the experiment. Table 1 shows the performance of MFEA with various probabilities between different populations. The best results are shown in bold. It can be seen that in the fully intersecting multitasking groups, the optimal results always occur simultaneously in the case of the same population crossover probability, indicating that a bigger *icp* will enhance the interaction between tasks. However, the results are not ideal to achieve good performance when the parameter *icp* reaches the maximum.

The reason may be that one problem in the multitasking group appears to stall, which affects the optimization process of other tasks. On the other hand, in the case that the optimization problem in multitasking group does not intersect at all, the effect between the tasks is opposite, and the algorithm performance is better in the case that the arbitrary mating rate between the populations is relatively low. Therefore, in order to enhance the influence of variable order on multitasking optimization, we set the crossover probability between populations as 1 in the following experiments.

4.3. Comparison of Three Orders of Variables. In this section, empirical studies are conducted to compare the solution quality of the MFEA with three orders of variables. And the experimental results are presented and discussed. Subsequently, we have made a proper cause analysis of the experimental results.

The optimal solutions obtained by MFEA on the 9 single-objective MFO benchmark problems are summarized in Table 2, the corresponding standard deviation is also given in the brackets, and the Wilcoxon rank sum test is adopted at a significance level of 5%. T1 and T2 denote the two tasks contained in MTO benchmark. Not surprisingly, as can be seen from Table 2, MFEA algorithm with different variable orders can obtain different optimal solutions, but the gap between these optimal solutions is very small. In a total of 18 optimization problems, the third order of variables performed better than the others seven times. The first and second ones do better three and five times, respectively. However, the difference between the results obtained by different variable orders is not regular. Better results can be obtained in all three orders, but the frequency in the third order is relatively high. For some reason, the complexity of the variables order has an impact on the optimization algorithm. And the multitasking algorithm is sensitive to the complexity of the order of variables.

Surprisingly, on the other hand, the effect on algorithm performance in practice is not significant for all cases. Some of the possible reasons include the following. (1) Essentially, the optimal solution is fixed after changing the order of variables. It means that the algorithm with different variable orders can evolve in the same direction. (2) At the operational level, when across-population crossover is executed in one generation, the offspring of different order strategies of variables are very similar because their parents come from the fixed-position individuals.

Figures 4–12 show the convergence plots of 9 groups of multitasking problems under different variable orders. Original order means there is no change in the order of variables, and order1, order2, and order3 indicate full reverse order, bisection reverse order, and trisection reverse order, respectively. It can be seen from Figures 4–12 that the convergence curves of the three variables orders are consistent. And the order of variables has no significant effect on the convergence of the algorithm. With the increase of generation, the convergence of the algorithm is almost the same as that without the change of the order of variables. After the evolution of a generation, the fitness value of the problem is the same in the order of different variables. Only in PI + LS, the convergence of the order of different variables has obvious difference in the later period. This may be because the PI + LS category has a low intertask similarity.

4.4. Universality of the Effect of Variable Order. In order to further verify the universal effect of variable order, we give the performance of MFDE algorithm with three different variable orders. Relevant experimental parameters about MFDE have been introduced in Section 4.1. Its performance on nine single-objective MTO benchmarks is summarized in Table 3, and the convergence of MFDE is illustrated in Figures 13–21.

It can be seen from the experimental results that the effect of variable order on MFDE is similar to MFEA. The relevant experimental data are shown in Table 3, which are the optimal value, the mean value in brackets, and the Wilcoxon rank sum test at a significance level of 5%. Notably, MFDE algorithm with different variable orders can also get different results but the difference between the different results is small. In the total 18 optimization tasks, the third variable order performs better than other orders four times. The first and second categories outperform others three times, respectively. In general, the third variable order has advantages over the other two, which is consistent with the performance of MFEA algorithm. Furthermore, the effect on the algorithm performance is also not significant. In the MFDE algorithm, there is no obvious difference in the convergence curve trend under different variable orders. Only in CI + LS and PI + LS, the convergence of different variable order shows a significant difference in the later stage, which is closely related to the similarity of the multitasking group itself. The effect of different variable orders is similar in MFDE and MFEA. The effect of variable order on

Table 1: Results obtained by MFEA under different interpopulation crossover probabilities.

Problem	Task	*icp* = 0.2	*icp* = 0.4	*icp* = 0.6	*icp* = 0.8	*icp* = 1.0
CI + HS	Griewank (T1)	**0.39991** (0.047121)	0.46988 (0.071905)	0.59349 (0.076058)	0.71105 (0.09389)	0.77662 (0.06994)
	Rastrigin (T2)	**194.2837** (45.6755)	214.4473 (45.8036)	209.6063 (38.6149)	235.6761 (30.4896)	280.2167 (28.5509)
CI + MS	Ackley (T1)	4.7949 (0.83182)	4.4248 (0.64978)	**3.8739** (0.43937)	3.9432 (0.46339)	3.9887 (0.39822)
	Rastrigin (T2)	233.0182 (53.0107)	221.7842 (38.1883)	**216.1091** (38.7009)	254.6503 (33.776)	285.4752 (34.862)
CI + LS	Ackley (T1)	**20.1832** (0.070467)	20.2581 (0.076936)	21.1314 (0.18667)	21.1755 (0.095171)	21.2042 (0.035706)
	Schwefel (T2)	**3824.8672** (496.1597)	4127.384 (543.3295)	4983.5594 (674.4369)	6483.7105 (651.6798)	9190.921 (809.3925)
PI + HS	Rastrigin (T1)	581.7632 (105.6933)	**523.3279** (105.1762)	487.0637 (75.8617)	551.2203 (52.6853)	703.3003 (53.7786)
	Sphere (T2)	**10.4851** (2.938)	23.4939 (5.4039)	77.4685 (18.854)	279.6843 (60.0515)	925.112 (136.8732)
PI + MS	Rosenbrock (T1)	**704.2483** (194.3793)	847.7707 (444.2602)	827.9882 (262.5255)	921.319 (274.8149)	994.1248 (360.4791)
	Ackley (T2)	3.536 (0.47139)	3.5078 (0.44899)	3.184 (0.52678)	3.0892 (0.35566)	**3.065** (0.44793)
PI + LS	Ackley (T1)	20.0635 (0.11943)	18.5992 (4.4149)	**8.1149** (5.0013)	12.1763 (4.2508)	17.6463 (1.985)
	Weierstrass (T2)	21.1636 (2.9837)	17.1749 (4.5975)	**7.1416** (3.7411)	9.5313 (2.9627)	12.7291 (2.4443)
NI + HS	Rosenbrock (T1)	**893.3438** (461.7737)	1270.6183 (464.4725)	1975.3255 (830.198)	3119.4286 (1320.8244)	3371.1812 (930.8882)
	Rastrigin (T2)	**256.4173** (72.8876)	270.6796 (56.3406)	258.0463 (42.4254)	289.2213 (37.8549)	314.6832 (25.0243)
NI + MS	Griewank (T1)	**0.43002** (0.055528)	0.62839 (0.069078)	0.91277 (0.067813)	1.1003 (0.055504)	1.1685 (0.048888)
	Weierstrass (T2)	26.604 (2.6594)	25.9726 (2.6735)	24.5367 (3.3814)	24.7251 (4.1937)	**24.0061** (1.8508)
NI + LS	Rastrigin (T1)	591.6303 (109.6072)	581.8746 (95.9773)	**575.5375** (77.6449)	812.6911 (94.0908)	3809.7344 (371.7017)
	Schwefel (T2	**3808.2556** (433.1423)	4299.1473 (452.9019)	5202.6947 (583.6562)	7050.294 (686.6561)	13849.5465 (430.4324)

Table 2: Results obtained by MFEA under different order of variables.

Problem	Task	MFEA	MFEA + FuR	MFEA + BiR	MFEA + TrR
CI + HS	Griewank (T1)	0.79685 (0.088892)	0.79888 (0.058188)	0.79729 (0.074131)	**0.79109** (0.05881)
	Wilcoxon rank sum test		5.19$E-01$	8.31$E-01$	1.84$E-01$
	Rastrigin (T2)	280.4036 (23.6354)	287.1282 (23.6655)	283.5717 (21.8295)	**277.5894** (23.7234)
	Wilcoxon rank sum test		1.61$E-01$	3.95$E-01$	5.74$E-01$
CI + MS	Ackley (T1)	4.1489 (0.45834)	**4.1138** (0.42517)	4.1355 (0.46903)	4.1242 (0.34747)
	Wilcoxon rank sum test		7.93$E-01$	9.42$E-01$	6.82$E-01$
	Rastrigin (T2)	282.8841 (38.8602)	279.624 (31.4245)	274.3095 (33.3069)	**269.5314** (31.1258)
	Wilcoxon rank sum test		5.65$E-01$	2.57$E-01$	7.25$E-02$
CI + LS	Ackley (T1)	21.2074 (0.031213)	21.2065 (0.036889)	**21.2037** (0.032431)	21.2064 (0.043875)
	Wilcoxon rank sum test		8.07$E-01$	4.59$E-01$	5.56$E-01$
	Schwefel (T2)	9011.6603 (966.4514)	9018.0075 (703.2951)	9071.5538 (742.7758)	**8916.1799** (743.8885)
	Wilcoxon rank sum test		9.92$E-01$	6.03$E-01$	7.59$E-01$
PI + HS	Rastrigin (T1)	700.4356 (35.1205)	707.0006 (49.4823)	701.6282 (44.2691)	**697.3025** (54.6428)
	Wilcoxon rank sum test		4.18$E-01$	9.48$E-01$	9.42$E-01$
	Sphere (T2)	**887.5937** (106.2772)	923.7637 (137.3392)	902.241 (141.8804)	900.9727 (135.6369)
	Wilcoxon rank sum test		2.16$E-01$	4.46$E-01$	9.31$E-01$
PI + MS	Rosenbrock (T1)	911.6519 (252.3363)	921.8783 (279.7278)	**900.0996** (304.4695)	966.8524 (354.607)
	Wilcoxon rank sum test		9.04$E-01$	5.15$E-01$	7.49$E-01$
	Ackley (T2)	**2.9614** (0.42019)	2.9664 (0.43933)	3.0137 (0.40994)	3.0059 (0.41482)
	Wilcoxon rank sum test		7.75$E-01$	5.06$E-01$	5.65$E-01$
PI + LS	Ackley (T1)	18.0366 (2.04)	18.1503 (2.1495)	**17.7703** (1.9321)	17.9673 (1.9811)
	Wilcoxon rank sum test		5.88$E-01$	3.65$E-01$	8.01$E-01$
	Weierstrass (T2)	13.3105 (2.5385)	13.0705 (2.797)	**12.4545** (3.0842)	13.1271 (2.5272)
	Wilcoxon rank sum test		7.54$E-01$	1.50$E-01$	6.72$E-01$

Table 2: Continued.

Problem	Task	MFEA	MFEA + FuR	MFEA + BiR	MFEA + TrR
NI + HS	Rosenbrock (T1)	3799.9063 (1200.8935)	**3728.7994** (1250.5653)	3830.1883 (1485.8082)	3814.4974 (1146.8015)
	Wilcoxon rank sum test		7.07*E* − 01	6.17*E* − 01	9.70*E* − 01
	Rastrigin (T2)	320.2278 (28.0176)	305.4074 (29.0316)	**304.0652** (30.8981)	312.4591 (28.8615)
	Wilcoxon rank sum test		2.71*E* − 02	2.86*E* − 02	3.19*E* − 01
NI + MS	Griewank (T1)	1.1668 (0.052264)	1.2348 (0.34088)	1.1743 (0.051695)	**1.1642** (0.046497)
	Wilcoxon rank sum test		3.34*E* − 01	2.95*E* − 01	8.23*E* − 01
	Weierstrass (T2)	23.8683 (2.1485)	25.0279 (3.4603)	23.5277 (2.1507)	**23.5026** (1.6825)
	Wilcoxon rank sum test		8.17*E* − 02	4.32*E* − 01	5.63*E* − 01
NI + LS	Rastrigin (T1)	3764.2782 (456.731)	**3742.6071** (494.4649)	3753.2077 (436.1078)	3790.9051 (425.3387)
	Wilcoxon rank sum test		9.97*E* − 01	8.23*E* − 01	6.37*E* − 01
	Schwefel (T2)	**13985.3097** (452.5025)	13987.3431 (621.4383)	14004.3855 (519.2037)	14004.714 (347.8701)
	Wilcoxon rank sum test		4.63*E* − 01	6.08*E* − 01	7.70*E* − 01

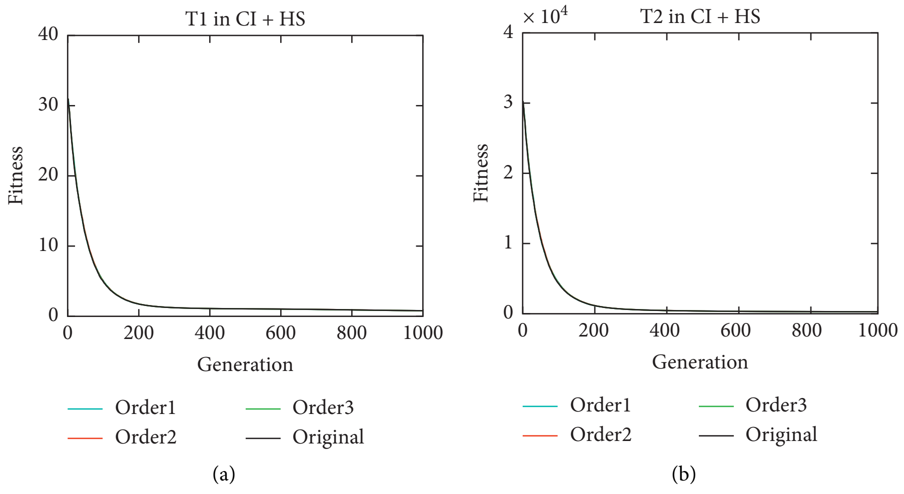


Figure 4: Convergence of MFEA for problem 1 (CI + HS).

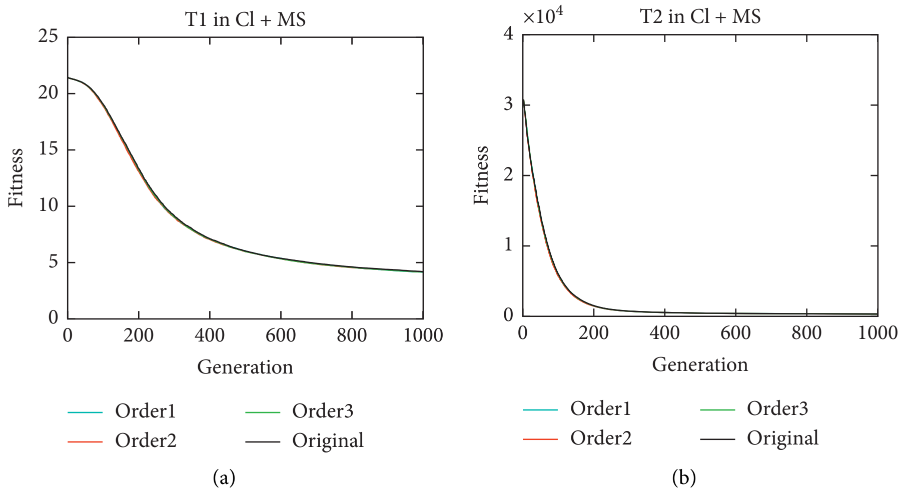


Figure 5: Convergence of MFEA for problem 2 (CI + MS).

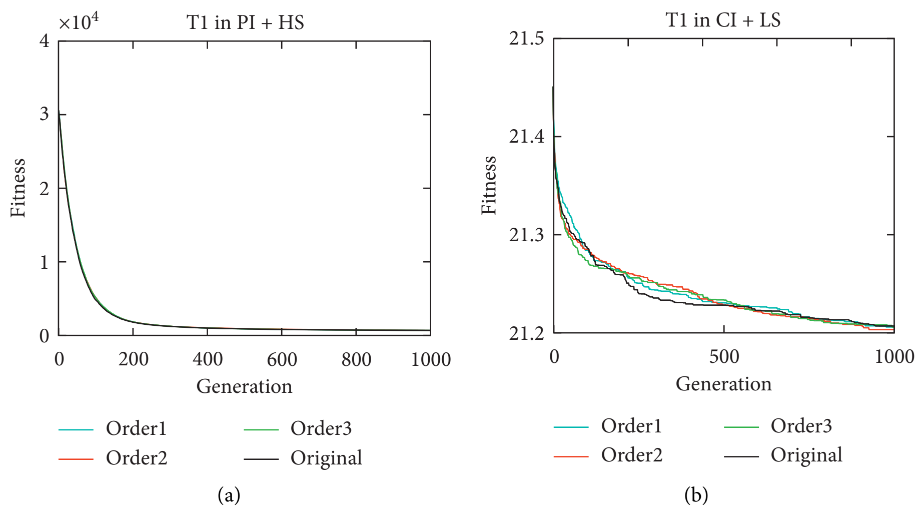


Figure 6: Convergence of MFEA for problem 3 (CI + LS).

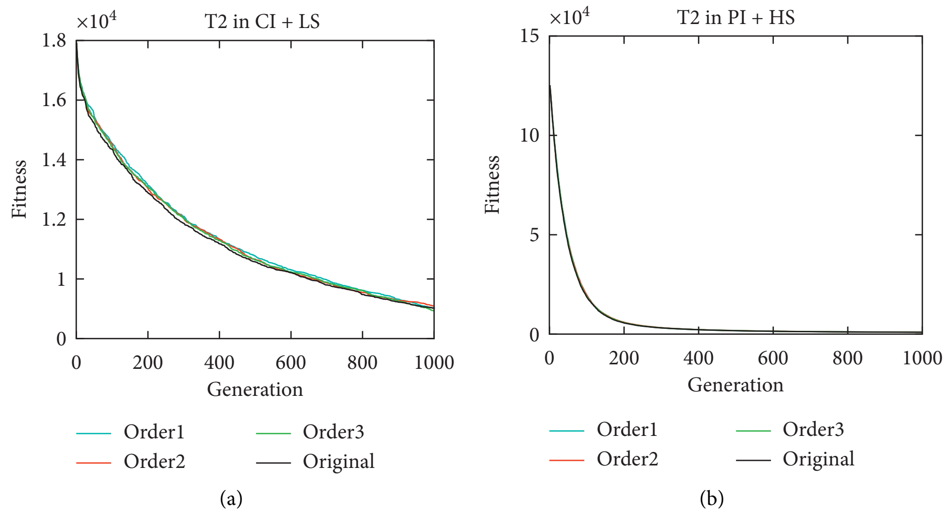


Figure 7: Convergence of MFEA for problem 4 (PI + HS).

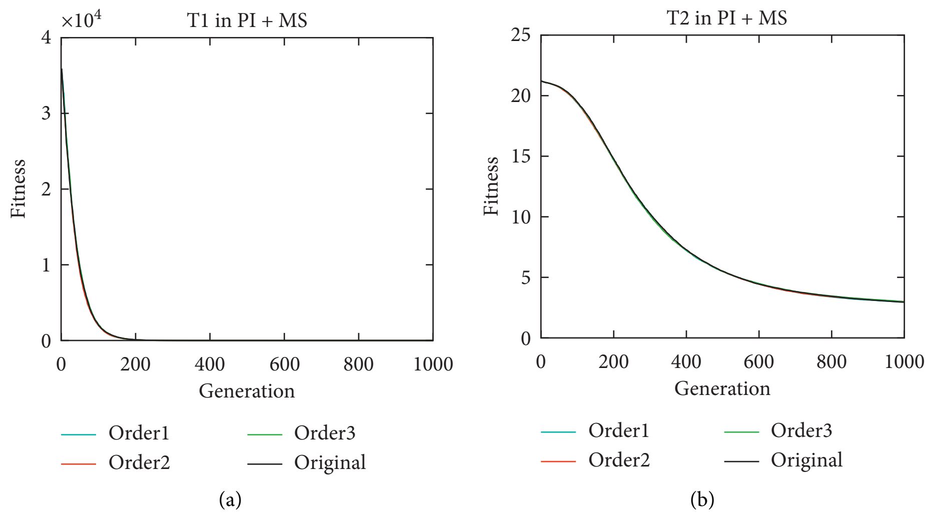


Figure 8: Convergence of MFEA for problem 5 (PI + MS).

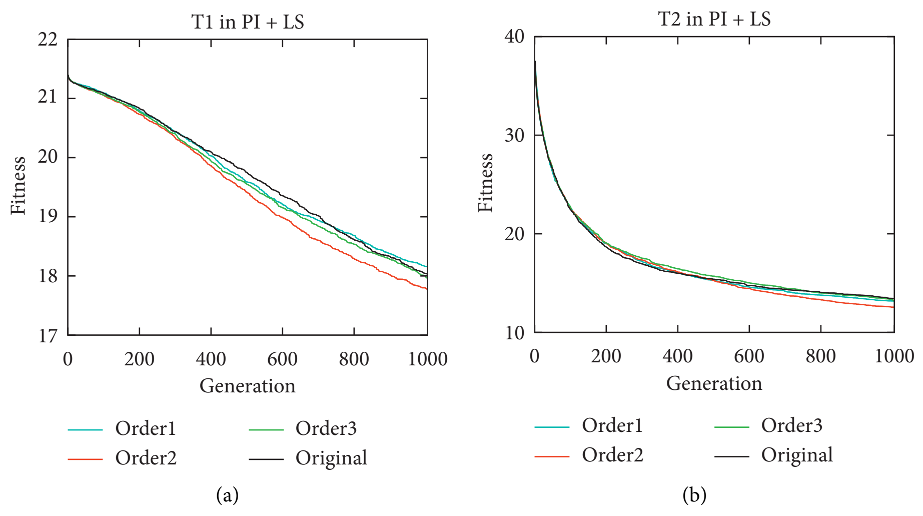


Figure 9: Convergence of MFEA for problem 6 (PI + LS).

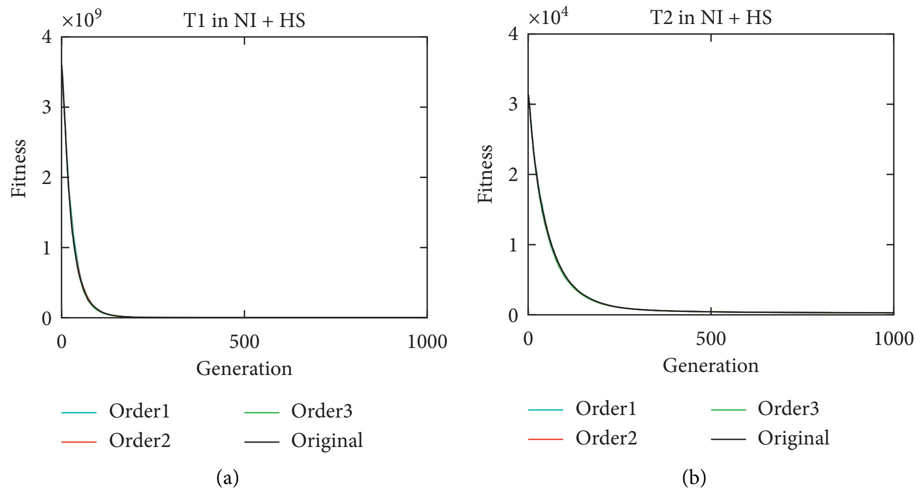


Figure 10: Convergence of MFEA for problem 7 (NI + HS).

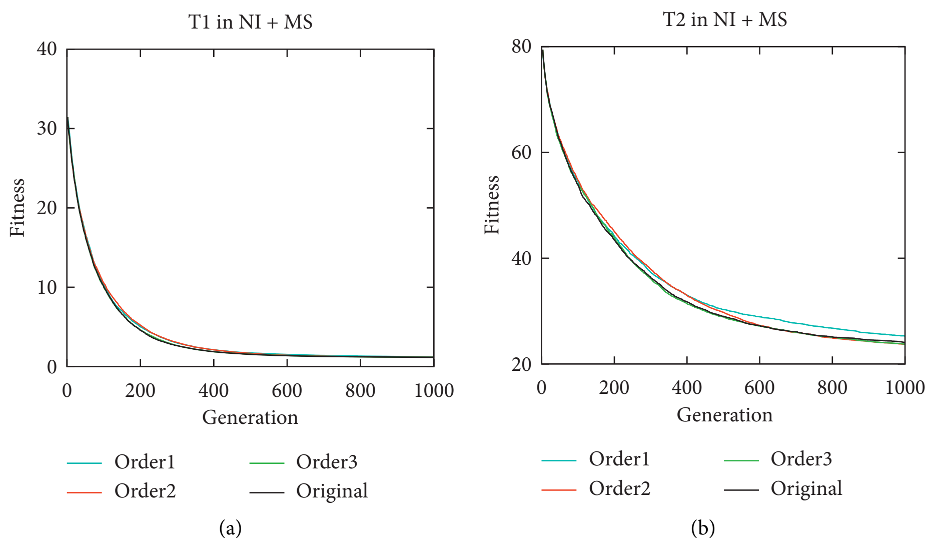


Figure 11: Convergence of MFEA for problem 8 (NI + MS).

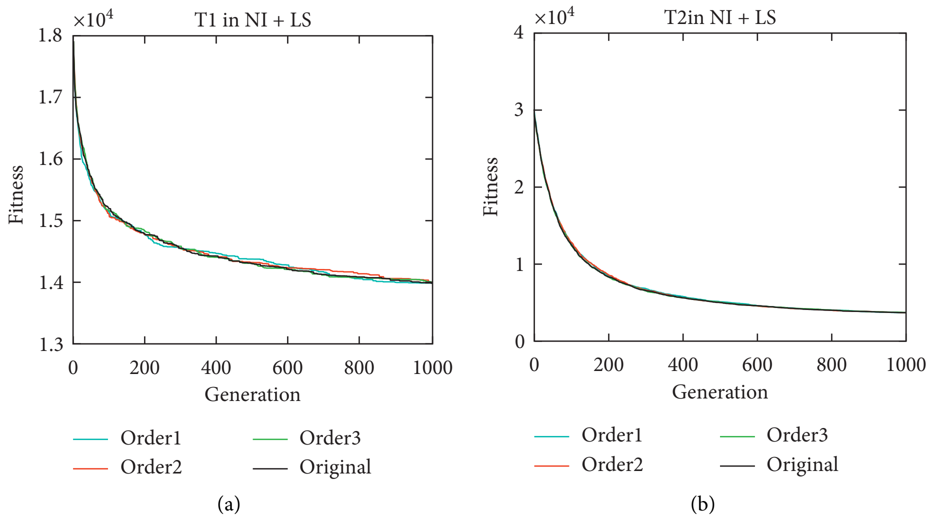


FIGURE 12: Convergence of MFEA for problem 9 (NI + LS).

TABLE 3: Results obtained by MFDE under different order of variables.

Problem	Task	MFDE	MFDE + FuR	MFDE + BiR	MFDE + TrR
CI + HS	Griewank (T1)	0.80067 (0.091715)	0.78732 (0.095784)	0.79683 (0.070848)	**0.78042** (0.069597)
	Wilcoxon rank sum test		5.60E − 01	6.42E − 01	9.53E − 02
	Rastrigin (T2)	383.6669 (17.341)	384.4004 (16.4345)	383.5204 (20.6672)	**378.5814** (18.5215)
	Wilcoxon rank sum test		7.75E − 01	8.44E − 01	2.32E − 01
CI + MS	Ackley (T1)	**3.7113** (0.31961)	3.7194 (0.27625)	3.7487 (0.39913)	3.8279 (0.52028)
	Wilcoxon rank sum test		4.40E − 01	4.14E − 01	1.51E − 01
	Rastrigin (T2)	**394.1759** (19.8874)	394.3457 (15.2135)	395.1253 (23.0533)	396.4384 (30.0564)
	Wilcoxon rank sum test		8.50E − 01	6.82E − 01	7.80E − 01
CI + LS	Ackley (T1)	21.2139 (0.039246)	**21.1924** (0.037366)	21.2083 (0.035688)	21.2038 (0.031897)
	Wilcoxon rank sum test		2.56E − 03	3.45E − 01	7.93E − 02
	Schwefel (T2)	14311.2877 (389.6504)	14320.5211 (437.0273)	**14287.7195** (372.0728)	14289.9965 (402.3736)
	Wilcoxon rank sum test		6.67E − 01	5.56E − 01	8.60E − 01
PI + HS	Rastrigin (T1)	375.0134 (22.5836)	**374.088** (34.9124)	378.3802 (21.5986)	377.1392 (19.3592)
			4.93E − 01	4.67E − 01	6.08E − 01
	Sphere (T2)	**38.3998** (11.9496)	39.5629 (11.8708)	39.3453 (9.7899)	43.3497 (11.5134)
	Wilcoxon rank sum test		4.88E − 01	5.98E − 01	4.03E − 02
PI + MS	Rosenbrock (T1)	3846.5413 (18670.6884)	618.9605 (988.0615)	**436.7277** (1004.12)	611.279 (1394.7849)
	Wilcoxon rank sum test		1.78E − 01	8.03E − 03	1.80E − 01
	Ackley (T2)	1.0478 (1.1419)	0.90033 (0.61353)	**0.89653** (0.67087)	1.0164 (0.73335)
	Wilcoxon rank sum test		9.20E − 01	5.79E − 01	3.72E − 01
PI + LS	Ackley (T1)	10.2738 (3.9853)	10.549 (3.7654)	9.8559 (3.405)	**9.8476** (3.2562)
	Wilcoxon rank sum test		4.34E − 01	9.53E − 01	7.80E − 01
	Weierstrass (T2)	7.8839 (4.8432)	**7.1112** (3.9246)	7.2987 (3.6848)	7.2383 (3.856)
	Wilcoxon rank sum test		6.82E − 01	1.00 E+00	9.53E − 01
NI + HS	Rosenbrock (T1)	4064.8109 (2340.4547)	3802.5141 (1764.8427)	4099.375 (1685.5901)	**3746.2983** (1596.6946)
	Wilcoxon rank sum test		9.09E − 01	3.50E − 01	6.92E − 01
	Rastrigin (T2)	**391.4913** (23.9269)	392.2342 (18.6947)	394.6522 (19.2541)	394.4849 (16.4735)
	Wilcoxon rank sum test		8.93E − 01	7.12E − 01	7.33E − 01
NI + MS	Griewank (T1)	**0.072548** (0.04229)	0.090939 (0.083529)	0.10188 (0.083764)	0.082862 (0.047972)
	Wilcoxon rank sum test		5.01E − 01	6.32E − 02	3.68E − 01
	Weierstrass (T2)	**8.2862** (2.948)	8.7113 (3.2989)	8.8365 (3.4151)	8.6162 (2.8182)
	Wilcoxon rank sum test		4.34E − 01	5.28E − 01	5.42E − 01
NI + LS	Rastrigin (T1)	**1675.7148** (1015.8628)	1970.881 (1243.1416)	1723.1033 (1029.3855)	1832.1993 (846.6522)
	Wilcoxon rank sum test		2.66E − 01	7.28E − 01	2.08E − 01
	Schwefel (T2)	**7007.7273** (1682.6573)	7469.9128 (1992.2856)	7259.8229 (1652.9156)	7492.6559 (1495.1018)
	Wilcoxon rank sum test		2.93E − 01	5.42E − 01	1.80E − 01

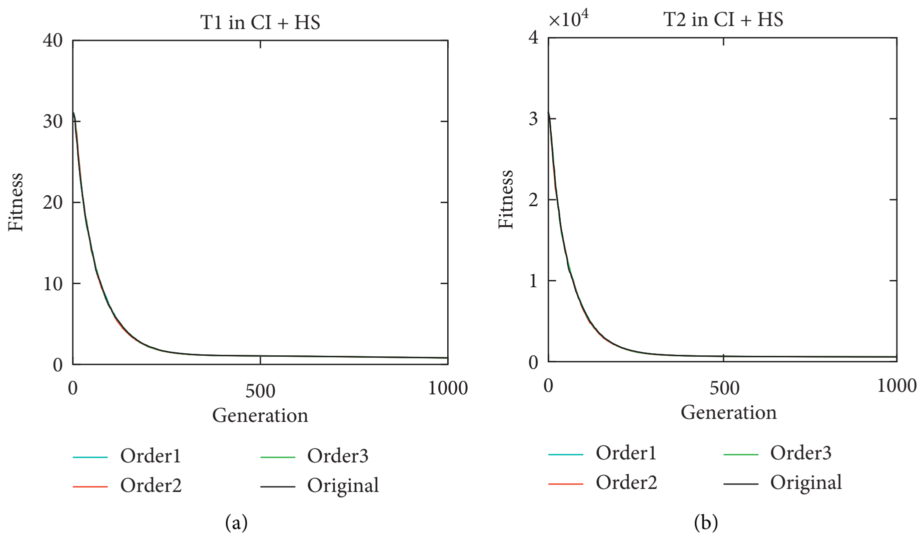


Figure 13: Convergence of MFDE for problem 1 (CI + HS).

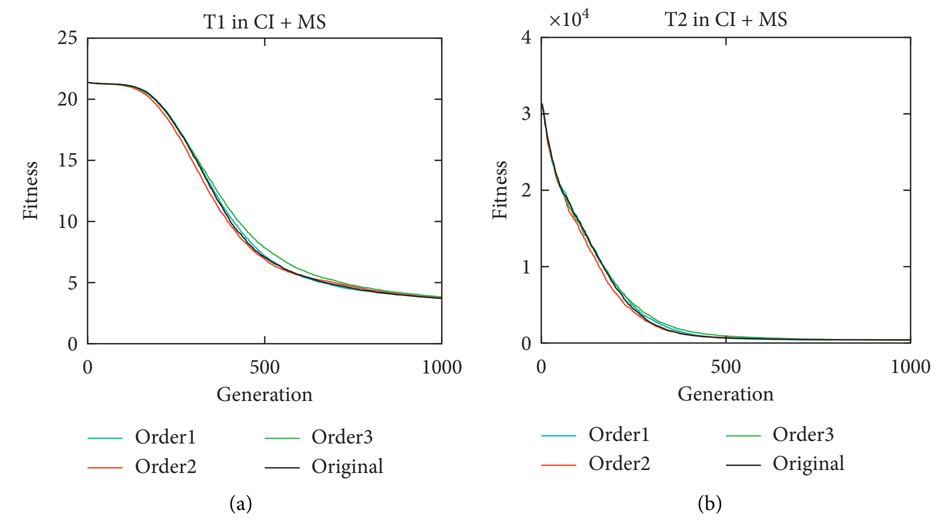


Figure 14: Convergence of MFDE for problem 2 (CI + MS).

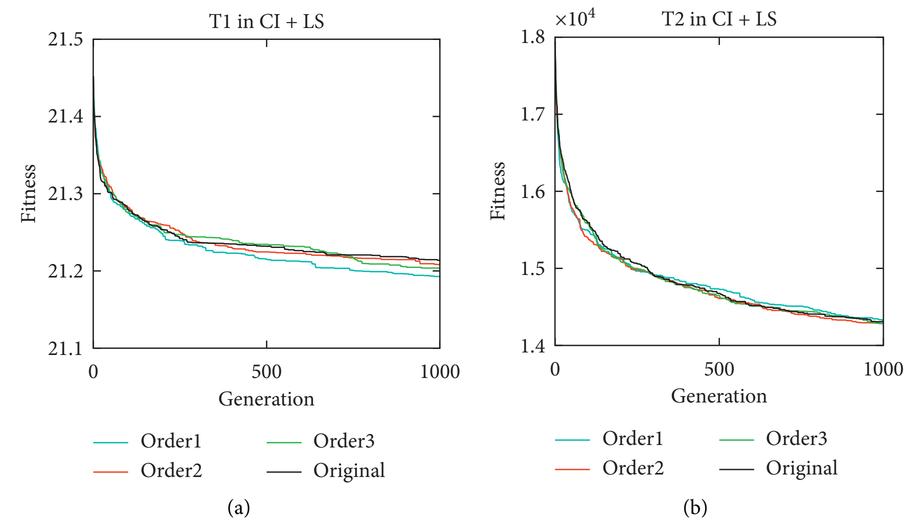
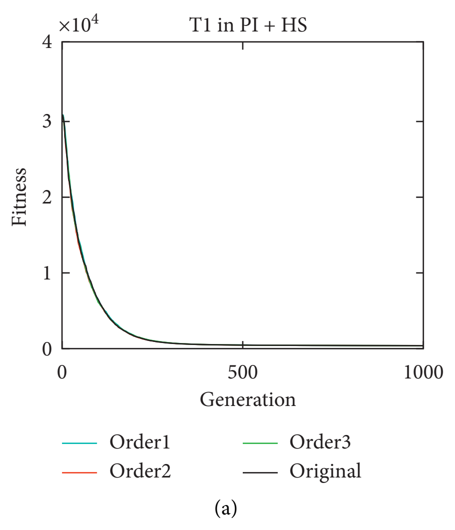


Figure 15: Convergence of MFDE for problem 3 (CI + LS).



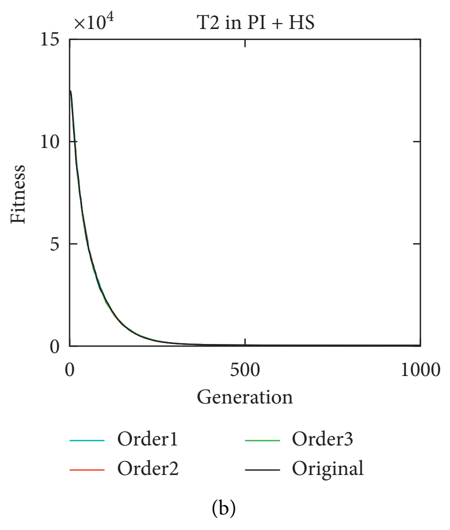
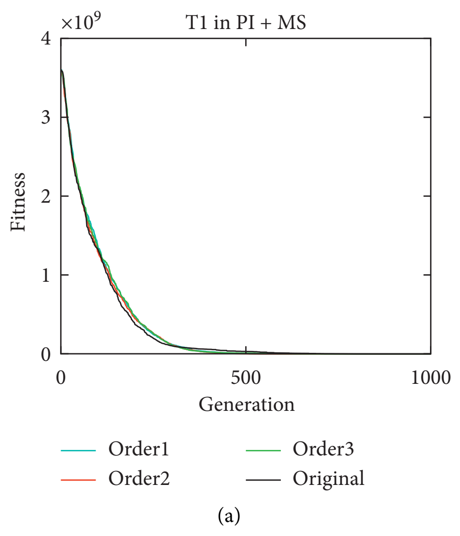


Figure 16: Convergence of MFDE for problem 4 (PI + HS).



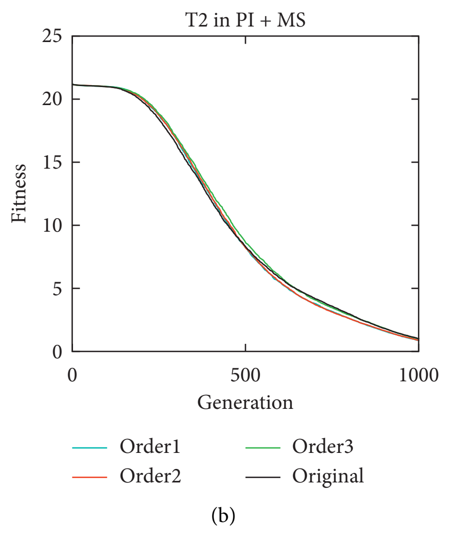
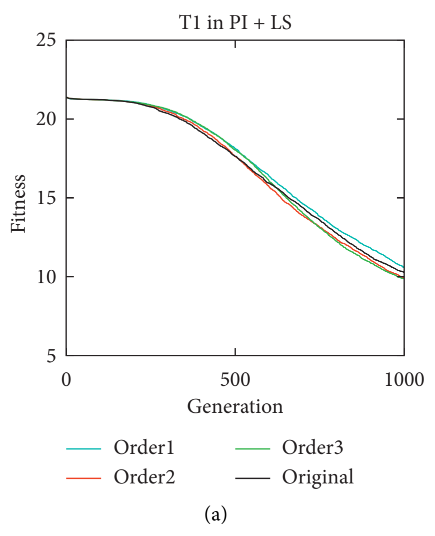


Figure 17: Convergence of MFDE for problem 5 (PI + MS).



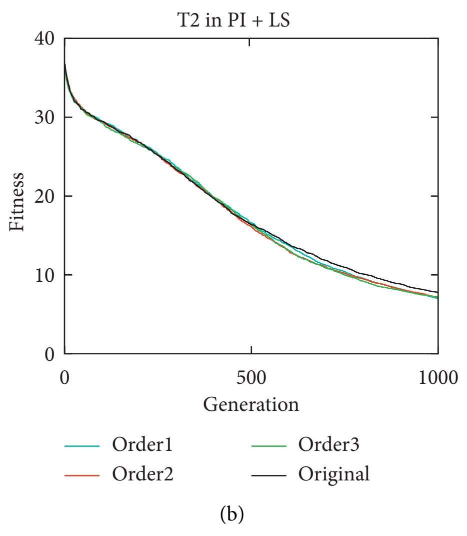
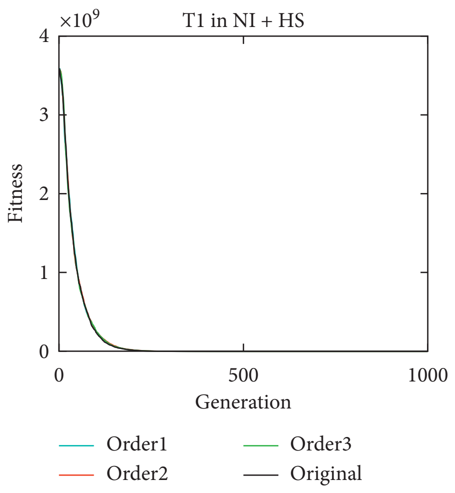
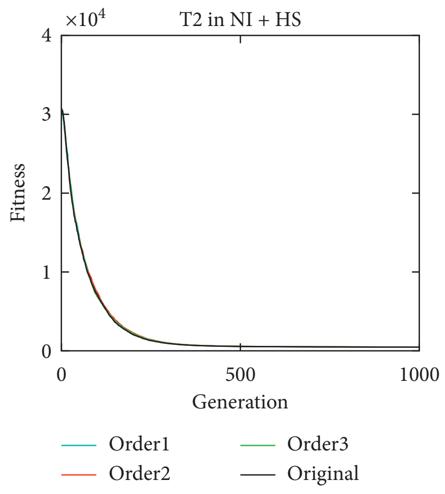


Figure 18: Convergence of MFDE for problem 6 (PI + LS).

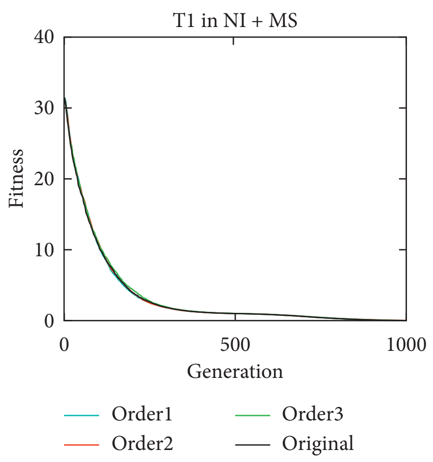


(a)

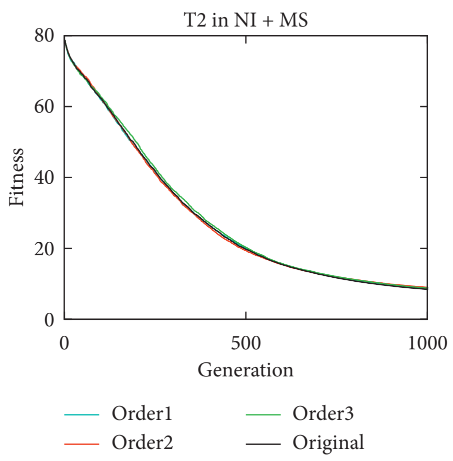


(b)

Figure 19: Convergence of MFDE for problem 7 (NI + HS).

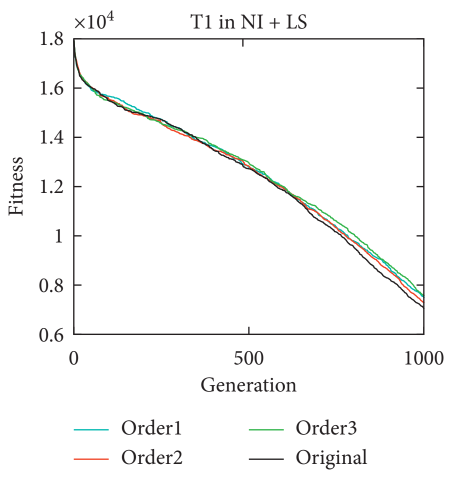


(a)

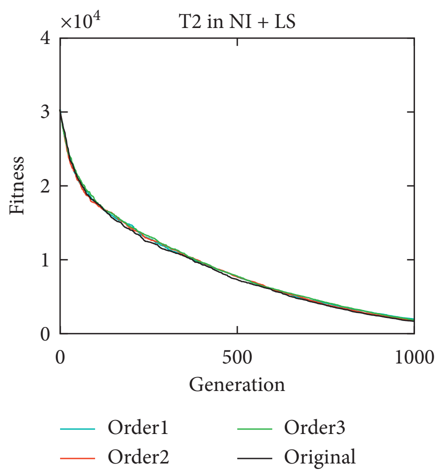


(b)

Figure 20: Convergence of MFDE for problem 8 (NI + MS).



(a)



(b)

Figure 21: Convergence of MFDE for problem 9 (NI + LS).

multitasking optimization does not change significantly due to the change of the framework of the multitasking algorithm.

5. Conclusion

In this paper, we investigate the effect of order of variables on the algorithm performance when solving multitasking optimization problems. The corresponding relationship between two individuals before and after changing the order of variables for single-task optimization problem and MTO problems is analyzed, respectively. When the order of variables of one optimization function is changed, different offspring will be generated for the MTO problem. Therefore, we design and study three orders of variables, namely, full reverse order, bisection reverse order, and trisection reverse order. An important feature of these orders of variables is that an individual can recover as himself after two times of changing the order of variables twice.

The experiment results showed that the effect of the order of variables on MFEA algorithm is not significant for all MTO problems in practice. Keeping this in mind, we analyze the difference of optimization results among different variable orders in different evolution stages. We find that the algorithm convergence does not change significantly because of the order of variables. In order to prove further, we also carried out the same order change on MFDE, and the results obtained are basically consistent with MFEA. However, MTO with high degree of similarity and intersection is more susceptible to be influenced by variable order and is more sensitive to complex variable order.

For future work, we would like to further study the effect of the order of variables under other situations, such as the position of optimal solution and multitasking optimization problem which contains more than two problems.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was partially supported by the National Natural Science Foundation of China under grant nos. 61773314 and 61803301, Natural Science Basic Research Program of Shaanxi under grant nos. 2019JZ-11 and 2020-JM-709, and Scientific Research Foundation of the National University of Defense Technology under grant no. ZK18-03-43.

References

[1] F. Feng, X. S. Hu, J. F. Liu, X. K. Lin, and B. Liu, "A review of equalization strategies for series battery packs: variables, objectives, and algorithms," *Renewable and Sustainable Energy Reviews*, vol. 116, Article ID 109464, 2019.

[2] K. Liu, K. Li, Q. Peng, and C. Zhang, "A brief review on key technologies in the battery management system of electric vehicles," *Frontiers of Mechanical Engineering*, vol. 14, no. 1, pp. 47–64, 2019.

[3] J. T. Tsai, J. H. Chou, and W. H. Ho, "Improved quantum-inspired evolutionary algorithm for engineering design optimization," *Mathematical Problems in Engineering*, vol. 2012, Article ID 836597, 27 pages, 2012.

[4] G. Q. Liu, W. Y. Chen, H. D. Chen, and J. H. Xie, "A quantum particle swarm optimization algorithm with teamwork evolutionary strategy," *Mathematical Problems in Engineering*, vol. 2019, Article ID 1805198, 12 pages, 2019.

[5] L. Cao, L. Xu, and E. D. Goodman, "A guiding evolutionary algorithm with greedy strategy for global optimization problems," *Computational Intelligence and Neuroscience*, vol. 2016, Article ID 2565809, 10 pages, 2016.

[6] J. F. Lin, K. W. Zhang, Y. L. Yao, Y. Xue, and T. Z. Guan, "A heuristic quasi-physical algorithm with coarse and fine adjustment for multi-objective weighted circles packing problem," *Computers & Industrial Engineering*, vol. 101, no. 1, pp. 416–426, 2016.

[7] K. Liu, C. Zou, K. Li, and T. Wik, "Charging pattern optimization for lithium-ion batteries with an electrothermal-aging model," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 12, pp. 5463–5474, 2018.

[8] F. Feng, S. Teng, K. Liu et al., "Co-estimation of lithium-ion battery state of charge and state of temperature based on a hybrid electrochemical-thermal-neural-network model," *Journal of Power Sources*, vol. 455, p. 227935, 2020.

[9] K. L. Liu, Y. L. Shang, Q. Ouyang, and W. D. Widanage, "A data-driven approach with uncertainty quantification for predicting future capacities and remaining useful life of lithium-ion battery," *IEEE Transactions on Industrial Electronics*, 2020.

[10] Q. Ouyang, Z. Wang, K. Liu, G. Xu, and Y. Li, "Optimal charging control for lithium-ion battery packs: a distributed average tracking approach," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 5, pp. 3430–3438, 2020.

[11] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: toward evolutionary multitasking," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 343–357, 2016.

[12] Q. Z. Xu, H. Yang, N. Wang, G. H. Wu, and Q. Y. Jiang, "Recent advances in multifactorial evolutionary algorithm," *Computer Engineering and Applications*, vol. 54, no. 11, pp. 15–20, 2018, in Chinese.

[13] Z. Wang, Q. Zhang, A. Zhou, M. Gong, and L. Jiao, "Adaptive replacement strategies for MOEA/D," *IEEE Transactions on Cybernetics*, vol. 46, no. 2, pp. 474–486, 2016.

[14] A. Gupta, Y.-S. Ong, and L. Feng, "Insights on transfer optimization: because experience is the best teacher," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 51–64, 2018.

[15] L. Feng, W. Zhou, L. Zhou et al., "An Empirical Study of Multifactorial PSO and Multifactorial DE," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 921–928, San Sebastian, Spain, June 2017.

[16] T. Back, U. Hammel, and H.-P. Schwefel, "Evolutionary computation: comments on the history and current state," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 3–17, 1997.

[17] Y. Jin and J. Branke, "Evolutionary optimization in uncertain environments—A survey," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 3, pp. 303–317, 2005.

[18] T. T. Nguyen, S. Yang, and J. Branke, "Evolutionary dynamic optimization: a survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 6, no. 1, pp. 1–24, 2012.

[19] N. Wang, Q. Xu, R. Fei, J. Yang, and L. Wang, "Rigorous analysis of multi-factorial evolutionary algorithm as multi-population evolution model," *International Journal of Computational Intelligence Systems*, vol. 12, no. 2, pp. 1121–1133, 2019.

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

[21] S. Das and P. N. Suganthan, "Differential evolution: a survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.

[22] R. Hashimoto, H. Ishibuchi, N. Masuyama, and Y. Nojima, "Analysis of evolutionary multi-tasking as an island model," in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1894–1897, Kyoto, Japan, July 2018.

[23] B. S. Da, Y. S. Ong, L. Feng et al., "Evolutionary multi-tasking for single-objective continuous optimization: benchmark problems, performance metric and baseline results," Technical Report, Nanyang Technological University, Singapore, 2017.

[24] G. H. Li, Q. F. Zhang, and W. F. Gao, "Multipopulation evolution framework for multifactorial optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 215-216, Kyoto, Japan, July 2018.

[25] H. Song, A. K. Qin, P. W. Tsai, and J. J. Liang, "Multitasking multi-Swarm optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1937–1944, Wellington, New Zealand, June 2019.

[26] G. Li, Q. Lin, and W. Gao, "Multifactorial optimization via explicit multipopulation evolutionary framework," *Information Sciences*, vol. 512, no. 1, pp. 1555–1570, 2020.

[27] Y. Chen, J. Zhong, L. Feng, and J. Zhang, "An adaptive archive-based evolutionary framework for many-task optimization," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 4, no. 3, pp. 369–384, 2020.

[28] Q. J. Chen, X. L. Ma, Y. W. Sun, and Z. X. Zhu, "Adaptive memetic algorithm based evolutionary multi-tasking single-objective optimization," in *Proceedings of the Asia-Pacific Conference on Simulated Evolution and Learning*, pp. 462–472, Shenzhen, China, November 2017.

[29] X. Zheng, Y. Lei, A. K. Qin, D. Y. Zhou, J. Shi, and M. G. Gong, "Differential evolutionary multi-task optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation Optimization*, pp. 1914–1921, Wellington, New Zealand, June 2019.

[30] X. X. Hao, R. Qu, and J. Liu, "A unified framework of graph-based evolutionary multitasking hyper-heuristic," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 8, pp. 1–1, 2020.

[31] A. Gupta and Y. S. Ong, "Genetic transfer or population diversification? Deciphering the secret ingredients of evolutionary multitask optimization," in *Proceedings of the IEEE Symposium Series on Computational Intelligence*, pp. 1–7, Athens, Greece, December 2016.

[32] Z. Tang and M. Gong, "Adaptive multifactorial particle swarm optimisation," *CAAI Transactions on Intelligence Technology*, vol. 4, no. 1, pp. 37–46, 2019.

[33] Y. Q. Cai, D. N. Peng, S. K. Fu, and H. Tian, "Multi-tasking differential evolution with difference vector sharing mechanism," in *Proceedings of the IEEE Symposium Series on Computational Intelligence*, pp. 3039–3046, Xiamen, China, December 2019.

[34] L. Zhou, L. Feng, K. Liu et al., "Towards effective mutation for knowledge transfer in multifactorial differential evolution," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1541–1547, Wellington, New Zealand, June 2019.

[35] L. Fen, L. Zhou, J. H. Zhong et al., "Evolutionary multi-tasking via explicit autoencoding," *IEEE Transactions on Cybernetics*, vol. 49, no. 9, pp. 3457–3470, 2019.

[36] J. B. Lin, H. L. Lin, B. Xue, M. J. Zhang, and F. Q. Gu, "Multi-objective multi-tasking optimization based on incremental learning," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 5, pp. 824–838, 2019.

[37] C. E. Yang, J. L. Ding, K. C. Tan, and Y. C. Jin, "Two-stage assortative mating for multi-objective multifactorial evolutionary optimization," in *Proceedings of the IEEE 56th Annual Conference on Decision and Control*, pp. 76–81, Melbourne, Australia, December 2017.

[38] Y. C. Lian, Z. X. Huang, Y. R. Zhou, and Z. F. Chen, "Improve theoretical upper bound of jumpk function by evolutionary multitasking," in *Proceedings of the High Performance Computing and Cluster Technologies Conference*, pp. 44–50, Guangzhou, China, June 2019.

[39] M.-Y. Cheng, A. Gupta, Y.-S. Ong, and Z.-W. Ni, "Coevolutionary multitasking for concurrent global optimization: with case studies in complex engineering design," *Engineering Applications of Artificial Intelligence*, vol. 64, no. 1, pp. 13–24, 2017.

[40] J. Ding, C. Yang, Y. Jin, and T. Chai, "Generalized multitasking for evolutionary optimization of expensive problems," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 1, pp. 44–58, 2019.

[41] J. L. Ding, C. E. Yang, Y. C. Jin, C. Z. Wang, and T. Y. Chai, "Multitasking multiobjective evolutionary operational indices optimization of beneficiation processes," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 3, pp. 1046–1057, 2019.

[42] L. Feng, L. Zhou, A. Gupta et al., "Solving generalized vehicle routing problem with occasional drivers via evolutionary multitasking," *IEEE Transactions on Cybernetics*, 2019.

[43] J. Liang, K. Qiao, M. Yuan et al., "Evolutionary multi-task optimization for parameters extraction of photovoltaic models," *Energy Conversion and Management*, vol. 207, no. 1, p. 112509, 2020.

[44] A. Gupta, J. Mańdziuk, and Y. S. Ong, "Evolutionary multitasking in bi-level optimization," *Complex Intelligent System*, vol. 1, no. 1–4, pp. 83–95, 2015.

[45] S. Jiang, C. Xu, A. Gupta et al., "Complex and intelligent systems in manufacturing," *IEEE Potentials*, vol. 35, no. 4, pp. 23–28, 2016.

[46] K. K. Bali, Y.-S. Ong, A. Gupta, and P. S. Tan, "Multifactorial evolutionary algorithm with online transfer parameter estimation: MFEA-II," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 69–83, 2020.

[47] A. Gupta, Y.-S. Ong, L. Feng, and K. C. Tan, "Multiobjective multifactorial optimization in evolutionary multitasking," *IEEE Transactions on Cybernetics*, vol. 47, no. 7, pp. 1652–1665, 2017.

[48] A. Gaur and K. Deb, "Effect of size and order of variables in rules for multi-objective repair-based innovization procedure," in *Proceedings of the IEEE Congress on evolutionary computation*, pp. 2177–2184, San Sebastian, Spain, June 2017.

[49] K. Deb and A. Srinivasan, "Innovization: innovating design principles through optimization," in *Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, pp. 1629–1636, Seattle, WA, USA, July 2006.