Research Article

An Effective Variable Transformation Strategy in Multitasking Evolutionary Algorithms

Qingzheng Xu,1,2 Lei Wang,3 Jungang Yang,1 Na Wang,1 Rong Fei,3 and Qian Sun3

1College of Information and Communication, National University of Defense Technology, Xi’an 710106, China
2Youth Innovation Team of Shaanxi Universities, National University of Defense Technology, Xi’an 710106, China
3School of Computer Science and Engineering, Xi’an University of Technology, Xi’an 710048, China

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

Received 8 August 2020; Revised 15 September 2020; Accepted 24 September 2020; Published 8 October 2020

Academic Editor: Zhile Yang

Copyright © 2020 Qingzheng Xu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Multitasking evolutionary algorithm (MTEA), which solves multiple optimization tasks simultaneously in a single run, has received considerable attention in the community of evolutionary computation, and several algorithms have been proposed in the literature. Unfortunately, knowledge transfer between constituent tasks may cause negative effect on algorithm performance, especially when the optimal solutions of all tasks are in different locations of the unified search space. To address this issue, an effective variable transformation strategy and the corresponding inverse transformation are proposed in multitasking optimization scenario. After using variable transformation strategy, the estimated optimal solutions of all tasks are both near the center point of the unified search space. More importantly, this strategy can enhance the task similarity, and then the effectiveness of knowledge transfer will probably be positive in this case, which can help us to improve the algorithm performance. Keeping this in mind, a multitasking evolutionalary algorithm (named MTDE-VT) is realized as an instance by embedding the proposed variable transformation strategy into multitasking differential evolution. In MTDE-VT, the individuals in the original population are first transformed into new locations by the variable transformation strategy. Once the offspring is generated in the transformed unified search space, it must be transformed back to the original unified search space. The statistical analysis of experimental results on some multitasking optimization benchmark problems illustrates the superiority of the proposed MTDE-VT algorithm in terms of solution accuracy and robustness. Furthermore, the basic principle and the good parameter combination are also provided based on massive simulated data.

1. Introduction

Optimization problems in science and engineering solved by the evolutionary algorithm (EA) can generally be classified into two groups: single-objective optimization and multi-objective optimization [1–4]. As a third category of optimization problems, multitasking optimization (MTO) was recently introduced, for solving multiple optimization tasks concurrently [5]. Within this scenario, multiple search spaces corresponding to different tasks, each possessing a unique function landscape, are used.

It is well known that real-world problems seldom exist in isolation and are usually mixed with each other. The knowledge extracted from the past learning experience can be constructively applied to solve more complex or unseen tasks. In contrast to traditional EA, multifactorial evolutionary algorithm (MFEA) has been proposed for MTO problems [5]. As an emerging paradigm in evolutionary computation, MFEA is capable to deal with multiple self-contained tasks simultaneously by using a single run. Due to knowledge transfer and sharing between tasks, it has demonstrated the capability to outperform its single-task counterpart in terms of both convergence speed and solution quality.

It should be noted here that the good performance of MFEA is greatly affected by the similarity between tasks [6–8]. In the literature, there are some similarity measures between multitasks, such as maximum mean discrepancy (MMD), Sperman rank correlation coefficient (SRCC), and crosstask fitness distance correlation (CTFDC) [9].
Intuitively, when the task similarity is high, positive transfer is most likely to occur and helps improve the algorithm performance. On the contrary, when the task similarity is low, negative transfer between tasks may happen and then may cause the population to evolve in the wrong direction or to fall into local optimum.

For MTO problems, different constituent tasks have their own characteristics. Generally speaking, the optimal solutions of all tasks tend to be in different locations of the unified search space. Within the range between those optimal solutions of different tasks, the trend of those objective functions may be in different directions. As a result, the effectiveness of knowledge transfer and sharing in evolutionary multitasking may degrade or even negative in this case. In this paper, our motivation is to map the optimal solution to the center point of the unified search space so that the growth trends of all tasks are similar. Few works have been conducted in the literature for variable transformation in evolutionary multitasking. To the best of our knowledge, Ding et al. have considered the variable transformation in [10], which will be reviewed and distinguished in Section 3.3. This paper thus presents another attempt to fill this gap.

The main contributions of this work can be summarized as follows: (1) An effective variable transformation strategy and the corresponding inverse transformation are proposed in multitasking optimization scenario. (2) This paper proposes a new multitasking evolutionary algorithm, namely, multitasking differential evolution with variable transformation strategy (MTDE-VT). In MTDE-VT, the individuals in the original population are first transformed into new locations by the variable transformation strategy. Once the offspring is generated in the transformed unified search space, it must be transformed back to the original unified search space. (3) Comprehensive experimental study on some MTO problems is carried out to verify the efficacy of the proposed MTDE-VT, and the basic principle and the good parameter combination are also provided based on massive simulated data.

The remainder of this paper is organized as follows: in Section 2, a brief description of MFEA, differential evolution (DE), and related works on MTEA is given. Section 3 begins with a detailed introduction of multitasking differential evolution. The motivation and mathematical definition of variable transformation strategy and the corresponding inverse transformation are also presented in this section. The proposed MTDE-VT algorithm for MTO problems is then described in detail. Section 4 discusses the experimental studies conducted to verify the efficacy of our proposed MTDE-VT in the multitasking optimization scenario. Finally, Section 5 concludes this article with several remarks and a discussion on potential future works.

2. Preliminaries

2.1. Multifactorial Evolutionary Algorithm. As the first realization of MTEA, MFEA shrewdly combines the genetic algorithm (GA) into a MTEA paradigm [5]. To seamlessly transfer the valuable knowledge in multitasking, all candidate solutions are encoded into a unified search space by a random key scheme. Accordingly, while tackling one of the optimization tasks, a subset of variables are extracted from the corresponding chromosome and decoded into a task-specific solution space.

Then, during population evolution of MFEA, assortative mating and vertical cultural transmission are both utilized to generate offspring, thereby leading to population diversification and implicit knowledge in multitasking optimization scenario. The notion of assortative mating guarantees that the individuals from the same task have high probability to mate and generate an offspring. Following vertical cultural transmission in MFEA, the phenotype of offspring is directly inherited by its parents. We again recommend [5] for more details on these procedures.

2.2. Related Works on Multitasking Evolutionary Algorithms. Since the first invention of MFEA in 2016, due to its simple implementation and strong search capability, intensive research efforts have been devoted to this area. For the sake of clarity, these works can be divided into three categories: algorithm framework, algorithm improvement, and typical application [11].

A general framework, evolution of bioecenosis through symbiosis (EBS), was proposed for EAs to deal with many-tasking problems [12]. This framework has two main features, i.e., the adaptive control of knowledge transfer among different tasks and the selection of candidates for evaluation from concatenate offspring. Another multipopulation evolution framework (MPEF) was first established for MTO, wherein each optimization task possesses an independent population and has its random mating probability for exploiting the useful knowledge of other tasks [13]. Recently, MTO with dynamic resource allocation (MTO-DRA) was proposed [14]. It can allocate and control computational resources to each task adaptively according to the requirements. Hou et al. [15] proposed an evolutionary transfer reinforcement learning framework for multiagent intelligent system, which can adapt to the dynamic environment.

In multitasking optimization scenario, the major function of across-population reproduction is knowledge transfer between different subpopulations in order to accelerate the search process. In addition to routine operations [13, 16–18], an interesting reproduction approach is called explicit genetic transfer across tasks [19]. By configuring the input and output layers to represent two task domains, the hidden representation provides a possibility for conducting knowledge transfer across task domains. However, this approach will fail on combinatorial optimization problems. To handle this issue, a weighted $L_1$-norm-regularized formulation was derived to capture the sparse mapping across vehicle routing problems (VRPs), and a solution-based knowledge transfer was also proposed by Feng et al. [20]. A novel genetic transform strategy has been proposed very recently [21]. Based on the two mapping vectors, two parent individuals can be mapped to the vicinity of the other solutions. More recently, a new MFEA with adaptive
knowledge transfer called MFEA-AKT was presented to achieve an efficient and robust performance [22]. It is able to adaptively configure the appropriate crossover operator for knowledge transfer across tasks based on the information collected along the evolutionary search process. A novel multiobjective MTEA algorithm (called MOMFEA-SADE), the winner of the Competition on Evolutionary Multitask Optimization within IEEE 2019 Congress on Evolutionary Computation, was proposed in [23]. Particularly, two transforming matrices are used to transform the search space of the population and reduce the probability of negative knowledge transfer between tasks.

Another potential research direction is parameter control or adaptation, especially for random mating probability (rmp). In classic MFEA, it is used to balance the diversity and convergence capability of the obtained solutions and is set as a constant of 0.3 [5]. If a task can be improved more times by the offspring from other tasks, the probability of knowledge transfer should be increased; otherwise, this rate will be decreased [12]. A simple random searching method was introduced to adjust this parameter [14]. The current rmp is stored in the candidate list when at least one of the K best solutions is updated by a better solution. Otherwise, the parameter is adaptively determined. It is very recently reported that an online rmp estimation technique was proposed in order to theoretically minimize the negative interactions between distinct optimization tasks [24]. Specifically, the extent of transfer parameter matrix is learned and adapted online based on the optimal blending of probabilistic models in a purely data-driven manner. We note that this approach can be applied to the domain of multiobjective optimization [25] and permutation-based discrete optimization [26] as well.

In the literature, there exist a lot of works to apply multitasking evolutionary algorithms to tackle real-world problems, such as complex supply chain network management [27], composites manufacturing problem [28], pollution-routing problem [29], operational indices optimization of beneficiation process [30], HIV protease cleavage sites prediction [31], hyperspectral image unmixing [32, 33], and time series prediction problem [34].

2.3. Differential Evolution. As a branch of stochastic EA, differential evolution, originally proposed by Price and Storn in 1995 [35], has been proven to be a simple yet effective, robust, and reliable global optimizer. Like other EAs, it starts with a population of \( N_p \) vectors representing the candidate solutions, where \( N_p \) indicates the population size. Let us assume that \( X_{i,g} = [x_{i,1,g}, x_{i,2,g}, \ldots, x_{i,D,g}] \) is the \( i \)th candidate solution vector in generation \( g \), where \( i = 1, 2, \ldots, N_p \), \( D \) is the problem’s dimension, and \( g \) is the generation index. For the original DE, its working depends on the manipulation and efficiency of three operators: mutation, crossover, and selection, which are described as follows.

The mutation operation enables DE to explore the search space and maintain diversity. In the literature, five mutation strategies have been commonly used [36], detail description of which are given as follows:

\[
\begin{align*}
\text{DE/rand/1:} & \quad V_{i,g} = X_{r_1,g} + F \times (X_{r_2,g} - X_{r_3,g}), \\
\text{DE/best/1:} & \quad V_{i,g} = X_{\text{best},g} + F \times (X_{r_1,g} - X_{r_2,g}), \\
\text{DE/current - best/2:} & \quad V_{i,g} = X_{\text{best},g} + F \times (X_{r_1,g} - X_{r_2,g} + X_{r_3,g} - X_{r_4,g}), \\
\text{DE/best/2:} & \quad V_{i,g} = X_{\text{best},g} + F \times (X_{r_1,g} - X_{r_2,g} + X_{r_3,g} - X_{r_4,g}), \\
\text{DE/rand/2:} & \quad V_{i,g} = X_{r_1,g} + F \times (X_{r_2,g} - X_{r_3,g} + X_{r_4,g} - X_{r_5,g}),
\end{align*}
\]

where \( V_{i,g} = [v_{i,1,g}, v_{i,2,g}, \ldots, v_{i,D,g}] \) denotes the mutant vector generated for each individual \( X_{i,g}, r_1, r_2, r_3, r_4, \) and \( r_5 \) are the random and mutually exclusive integers chosen from the set \( \{1, 2, \ldots, N_p\} \), \( F \) is the scaling factor which controls the amplitude of the difference vector, and \( X_{\text{best},g} \) gives the solution with the best fitness found so far.

The aim of the crossover operation is to build trial vectors by recombining the current vector and the mutant one. The family of DE algorithms employs two crossover schemes: exponential and binomial crossovers. The binomial crossover is utilized in this paper and is briefly discussed below.

In the binomial crossover, the trial vector \( U_{i,g} = (u_{i,1,g}, u_{i,2,g}, \ldots, u_{i,D,g}) \) is defined as follows:

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

where \( \text{CR} \in [0, 1] \) is the predefined crossover rate, \( \text{rand} \) is a uniform random number within \([0, 1]\), and \( \text{rand}, \epsilon \in \{1, 2, \ldots, D\} \) is a randomly selected index which is used to ensure that at least one dimension of the 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 \( f() \). If, and only if, the trial vector is better or equal than the parent vector, then it replaces the current vector, otherwise the current vector survives while the trial one is eliminated, as described below:

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

3. Proposed Algorithms: MTDE and MTDE-VT

3.1. MTDE. As mentioned above, MTEA is a new and general framework for solving MTO problems where diverse population-based search mechanisms are employed. In particular, MTEA is conducted with the DE algorithm, namely, MTDE in this paper. Its basic structure is depicted in Algorithm 1.
In the previous work [11], from theoretical analysis and simulation experiment, MFEA and its variations were reviewed carefully in view of multipopulation evolution model. Therefore, it should be especially explained that MTDE is also conducted as the multipopulation evolution model, wherein each population is represented independently and evolved for the selected task only.

In the original MFEA, the skill factor of each individual should be computed and updated every generation. On the contrary, the obvious feature of multipopulation evolution strategy is that the new candidate solution generated by mutation and crossover is assigned to a fixed subpopulation. One of the advantages of this approach is to analyze the contribution of each population.

The main difference between DE and MTDE are mainly in mutation strategy. In the original DE/rand/1 explained in Section 2.3, three parents are chosen randomly from the same population (task) and then a mutant vector \(V_{i,g}\) is generated by using Equation (1). On the contrary, in MTDE, a random value \(\text{rand}\) is first generated in the range of \([0, 1]\]. If \(\text{rand}\) is less than the predefined mating probability value rmp, across-population mutation is executed, where three parents are chosen randomly from different populations (tasks), as shown in lines 2–7 of Algorithm 2. Otherwise, the original mutation is done as shown in lines 9–10 of Algorithm 2.

Note that, in order to avoid population drift, we use a new mutation approach across subpopulations \(P_k\) and \(P_r\) \((k \neq r)\) in MTDE as follows:

\[
V_{i,g}^k = \begin{cases} 
X_{r_1,g}^k + F \times (X_{r_2,g}^r - X_{r_3,g}^r), & \text{rand} < 0.5, \\
X_{r_1,g}^r + F \times (X_{r_2,g}^k - X_{r_3,g}^k), & \text{rand} \geq 0.5,
\end{cases}
\]

(8)

where \(V_{i,g}^k, X_{r_1,g}^k, X_{r_2,g}^k,\) and \(X_{r_3,g}^k\) are the different vectors in subpopulations \(P_k\) and \(X_{r_1,g}^r, X_{r_2,g}^r, \) and \(X_{r_3,g}^r\) are the vectors in subpopulations \(P_r\).

3.2. Motivation. In order to illustrate the motivation of variable transformation visually, some problem instances and conceptual map are depicted in Figure 1 in multitasking optimization scenario. Specifically, in the top of Figure 1, two multitask problems are shown, in which the blue solid and the red dotted line represent task 1 and task 2, respectively. Next, in the middle of Figure 1, only the exact information of optimal solution and boundary value are reserved, and the other details of each task are ignored deliberately. In this way, a simple conceptual map of multitasking optimization problems can be built easily. Note that different tasks with the same optimum will map into the similar conceptual map, as shown in Figure 1. Lastly, for each task, the key transformations (the variable transformation strategy and the corresponding inverse
transformation) in the bottom of Figure 1 are described in Section 3.3.

For MTO problems, different constituent tasks have their own characteristics. Generally speaking, the optimal solutions of all tasks tend to be in different locations of the unified search space, as shown in the left column in Figure 1. In this case, the offspring can be generated across the entire search space by assortative mating (e.g., across-population mutation). In this way, the population diversity remains at high level during the entire optimization process. Note, however, that excessive diversity may seriously slow down the algorithm convergence.

These MTO problem instances can be further abstracted into a simple conceptual map as shown in the middle row of Figure 1. It has been found that within the range between these optimal solutions of multitasks, the trends of those objective functions are in different direction. As illustrated in Figure 1, the optimal solutions of two tasks, indicated by blue solid and red dashed lines, are located at 0.2 and 0.8, respectively. It is observed that from 0.2 to 0.8, it is getting larger and larger for task 1, while the observed trend of task 2 is in the opposite direction. In this particular case, the similarity between two tasks is considered to be the lowest. Therefore, the effectiveness of knowledge transfer in evolutionary multitasking may degrade or even negative in this case. As a result, the multitasking evolutionary algorithm (e.g., MFEA) does not work well for these MTO problems due to inefficient knowledge transfer.

3.3. Variable Transformation Strategy. To address the above limitation of all MTEA algorithms for MTO problems whose optimums are located in different positions in the unified search space, this paper proposes an effective variable transformation strategy in the multitasking optimization algorithm. The detail of variable transformation strategy is summarized in Algorithm 3.

**Algorithm 2: Reproduction and selection.**

| Line | Code |
|------|------|
| 1-2 | Input: target vector in the current generation \(X_{i,g}^k\) Parameter: population size \((N_p)\), crossover rate \((CR)\), scaling factor \((F)\), and random mating probability \((rmp)\) Output: target vector in the next generation \(X_{i,g+1}^k\) |
| 3 | Step 1: mutation |
| 4 | \(/ */ Generate a mutant vector \(V_{i,g}^k\) for each target vector \(X_{i,g}^k\) */ |
| 5 | \(/ */ across-population mutation */ |
| 6 | (1) If rand < rmp then |
| 7 | (2) Select six vectors \(X_{i,1,g}^k, X_{i,2,g}^k, X_{i,3,g}^k, X_{r,1,g}^k, X_{r,2,g}^k,\) and \(X_{r,3,g}^k\) randomly from two subpopulations \(P_i\) and \(P_r\), where \(i \neq r \neq r \neq r \neq r\) and \(k \neq r\) |
| 8 | (3) If rand < 0.5 then |
| 9 | (4) \(V_{i,g}^k = X_{i,1,g}^k + F \times (X_{r,2,g}^k - X_{r,3,g}^k)\) |
| 10 | (5) Else |
| 11 | (6) \(V_{i,g}^k = X_{i,1,g}^k + F \times (X_{r,2,g}^k - X_{r,3,g}^k)\) |
| 12 | (7) End if |
| 13 | (8) Else |
| 14 | (9) Select three vectors \(X_{i,1,g}^k, X_{i,2,g}^k,\) and \(X_{i,3,g}^k\) randomly from the current subpopulation \(P_i\) where \(i \neq r \neq r \neq r \neq r\) |
| 15 | (10) \(V_{i,g}^k = X_{i,1,g}^k + F \times (X_{i,2,g}^k - X_{i,3,g}^k)\) |
| 16 | (11) End if |
| 17 | Step 2: crossover |
| 18 | \(/ */ Generate a trial vector \(U_{i,g}^k\) for each target vector \(X_{i,g}^k\) */ |
| 19 | (12) \(rand = \text{rand} \times D\) |
| 20 | (13) For \(j = 1\) to \(D\) do |
| 21 | (14) If \((\text{rand} < \text{CR})\) or \((j = \text{rand})\) then |
| 22 | (15) \(U_{i,g,k} = V_{i,g,k}\) |
| 23 | (16) Else |
| 24 | (17) \(U_{i,g,k} = X_{i,g,k}\) |
| 25 | (18) End if |
| 26 | (19) End for |
| 27 | Step 3: selection |
| 28 | \(/ */ Select a better target vector in generation \(g + 1\) \((X_{i,g+1})\) */ |
| 29 | (20) Evaluation the trial vector \(U_{i,g}\) |
| 30 | (21) If \(f(U_{i,g}) \leq f(X_{i,g})\) then |
| 31 | (22) \(X_{i,g+1} = U_{i,g}\) |
| 32 | (23) Else |
| 33 | (24) \(X_{i,g+1} = X_{i,g}\) |
| 34 | (25) End if |
Figure 1: Problem instances and conceptual map of multitasking optimization problems.
The variable transformation strategy (see in the bottom of Figure 1) and the corresponding inverse transformation (see in the bottom of Figure 1) are defined as equations (9) and (10), respectively:

$$
op = \begin{cases} 
\frac{\mu \times \rho}{m}, & p \leq m, \\
\frac{\mu - 1}{m-1} \times \rho + \frac{m - \mu}{m-1}, & p > m,
\end{cases}
$$

(9)

$$p = \begin{cases} 
\frac{m}{\mu} \times \op, & \op \leq \mu p, \\
\frac{m-1}{\mu-1} \times \op + \frac{\mu - m}{\mu-1}, & \op > \mu p,
\end{cases}
$$

(10)

where $\mu = (0.5, 0.5, \ldots, 0.5)$ is the center point of the unified search space, $p$ is the solution in the original unified search space, and $\op$ is the corresponding solution in the transformed unified search space. Furthermore, $m$ is the estimated optimal solution, which can be calculated as the mean value of the top $\mu \times N_p$ best solutions in the current generation according to Equation (11). Note that four key variables ($p$, $\op$, $m$, and $\mu$ in Equations (9) and (10)) are marked in the bottom of Figure 1:

$$m = \frac{\sum_{i=1}^{\mu \times N_p} \op_i}{\mu \times N_p}.
$$

(11)

For example, the population size $N_p = 10$, and the percent of top best solutions $\mu = 40\%$ in one case. Thus, the top 4 ($=40\% \times 10$) best solutions in the current generation can be used to determine the estimated optimal solution $m$. Let us suppose $\op_1^\text{best} = (0.10, 0.15, 0.10, 0.15), \op_2^\text{best} = (0.13, 0.08, 0.20, 0.15), \op_3^\text{best} = (0.06, 0.25, 0.11, 0.14), \op_4^\text{best} = (0.31, 0.12, 0.07, 0.16)$. Therefore, the estimated optimal solution $m = \frac{\sum_{i=1}^{4} \op_i^\text{best}}{4} = (0.15, 0.15, 0.12, 0.15)$ according to Equation (11).

Next, we can calculate the corresponding solution in the transformed unified search space based on the proposed variable transformation strategy. According to Equation (9), we get the following results: $\op_1^\text{best} = (0.3333, 0.5, 0.417, 0.5), \op_2^\text{best} = (0.4333, 0.2667, 0.5455, 0.5), \op_3^\text{best} = (0.2, 0.5588, 0.4583, 0.4667)$, and $\op_4^\text{best} = (0.5941, 0.4, 0.2917, 0.5059)$.

As illustrated in the right column of Figure 1, after using variable transformation strategy in multitasking scenario, the estimated optimal solutions of all tasks are both near the center point of the unified search space, and the individuals in the left/right region of each task will be mapped (or scaled) into the left/right region. More importantly, according to the right conceptual map in Figure 1, in both the left and right regions of the unified search space, it is getting smaller and smaller for all tasks. Therefore, the task similarity is very high, and then the effectiveness of knowledge transfer will probably be positive and can help us to improve the convergence rate in this case.

More recently, another attempt for variable transformation was conducted by Ding et al. in [10]. In particular, each individual in the population is translated to a new location according to the following equation:

$$\op_i = p_i + d_k, \quad \text{(12)}$$

where $p_i$ and $\op_i$ ($i = 1, 2, \ldots, N_p$) are the $i$th solution and the corresponding transformed solution, respectively, in the unified search space, $N_p$ is the population size, and the translated value $d_k$ is estimated based on the promising solutions of the $k$th task.

The main purpose of this approach proposed by Ding et al. is that the transformed optimums of all tasks are located at the same position in the unified search space. Note that the translated direction and distance are both fixed for all individuals. Unfortunately, it is easy for the individuals to jump out of the legal range, and then manual efforts (e.g., random selection within a predefined range) are required to ensure the legality of candidate solutions. Thus, this will destroy the original population distribution characteristics.

On the contrary, our approach in this paper is capable of dealing with these issues. To be specific, the variable transformation strategy and the corresponding inverse transformation are both linear transformation as shown in equations (9) and (10). Their output is limited strictly in the whole unified search space $[0, 1]$. As a result, the relative quality of all individuals will remain unchanged after using variable transformation strategy. It should be noted that the
idea of calculating the estimated optimal solutions, defined by equation (11) in this paper, is borrowed from [10].

3.4. MTDE-VT. In order to show the effectiveness of variable transformation strategy, we propose herein a new multitasking evolutionary algorithm, namely, MTDE-VT in this paper.

The difference between the original MTDE and MTDE-VT is mainly in the process of offspring reproduction. At each generation, the individuals in the original population are first transformed into new locations by the proposed variable transformation strategy according to equation (9) and then undergo across-population mutation as a conventional genetic operator. Once the mutant vector is generated in the transformed unified search space, the offspring must be transformed back to the original unified search space according to equation (10). The detail of across-population mutation with variable transformation strategy is displayed in Algorithm 4.

Note that, the variable transformation strategy and the corresponding inverse transformation can only be applied to across-population mutation. When the original mutation is selected for execution based on random mating probability, the variable transformation strategy does not need to be executed in MTDE.

4. Simulation Results and Discussion

4.1. Experimental Setup. Similar to the pioneering work by Gupta et al. [7], the same benchmark problem set is employed to assess the performance of MTDE-VT in this paper. The utilized test suite includes nine MTO problems, and each problem contains two distinct minimization tasks. Based on the degree of intersection of the global optima, 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). More details about these benchmark problems can refer to the technical report [37] and the homepage of Prof. Yew-Song Ong (http://www.ntu.edu.sg/home/asssong/).

As mentioned above, the idea and definition of the estimated optimal solutions are borrowed from [10]. Therefore, three key parameters of variable transformation strategy (φ, θ, and μ in Algorithm 3) are set as the recommended values in [10]. For fairness of comparison, all parameter settings for the proposed MTDE-VT and other algorithms are kept identical to [37] and listed as follows:

(i) Population size, \( N_p = 100 \)
(ii) Differential amplification factor, \( F = 0.5 \)
(iii) Crossover probability, \( CR = 0.9 \)
(iv) Maximum number of function evaluations, \( \text{Max}_{\text{FEE}} = 10^8 \)
(v) Random mating probability, \( \text{rmp} = 0.3 \)
(vi) Start-stop generation, \( \varphi = 0.1_{\text{start}} \)

(vii) Generation interval, \( \theta = 0.01 \)
(viii) Percent of top best solutions in the current population, \( \mu = 40\% \)

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 Windows XP OS using the computer with a 2.2 GHz Intel Corei3 processor and 4 GB RAM.

4.2. Experimental Series 1: Comparison of MTDE and MTDE-VT. The experimental results of MTDE and MTDE-VT are provided in Table 1, and the best entry for each task is highlighted in boldface. Furthermore, the Wilcoxon rank sum test is also adopted at a significance level of 0.05 [38].

From the results in Table 1, we can see that MTDE-VT performs significantly better than the original MTDE on 8 out of 18 tasks and has equivalent performance on other 10 tasks. This result indicates that the variable transformation strategy is able to maintain the high-quality solution during the population evolution and knowledge transformation between tasks.

Interestingly, the best optimal solution obtained by MTDE-VT is better than the original MFEA on 16 out of 18 tasks, while the mean optimal solution by MTDE-VT is better than the MFEA on 14 tasks. Although the average value is a good index of the overall performance of stochastic algorithms, the best value obtained is a more valuable indicator. The reason is that MTDE-VT can always get a satisfactory solution by trying more times. In our opinion, the best optimal solution obtained by MTDE-VT benefits from the fast convergence speed is compared with MFEA, regardless of the final performance level. In fact, the proposed variable transformation strategy has great exploitation ability. When the estimated optimal solution is near the global optimal solution, MTDE-VT leads to fast convergence to the global best solution. On the contrary, when it is near the local optimal solution, MTDE-VT may easily trap into local optimal solution or even cannot solve successfully the problem.

4.3. Experimental Series 2: Universality for Evolutionary Computing. One of the most prominent features of variable transformation strategy is its universality for all evolutionary algorithms. The single prerequisite to variable transformation strategy is the differential optimal solutions of all tasks to be solved simultaneously. Obviously, this hypothesis of the proposed strategy and its corresponding computing process has nothing to do with the benchmark evolutionary algorithm. In other words, the variable transformation strategy works well for any swarm and evolutionary algorithms.

For simplicity, in order to reveal its universality of variable transformation strategy, several common variants of DE are tested in this section. For example, MTDE (\( F = 0.7, CR = 0.3 \)) represents the basic MTDE/rand/1 algorithm with new parameters (\( F = 0.7, CR = 0.3 \)), MTDE(best/1) is the new
MTDE/best/1 algorithm under default parameters ($F = 0.5$ and $CR = 0.9$), and so on. Comparatively speaking, two different vector strategies, such as DE/rand/2 algorithm, have the ability to improve the diversity by producing more trial vectors, and the DE/best/1 algorithm speeds up convergence due to its powerful exploitation ability. Their details are given in Section 2.3. The experimental results of several variants of MTDE and MTDE-VT algorithms are summarized in Table 2, and the Wilcoxon rank sum test is also adopted at a significance level of 0.05.

First of all, we observe that, in most cases, the performance of the MTDE (or MTDE-VT) algorithm with diverse parameters or mutation strategies is distinctly different. For example, when solving task 1 of problem 1 (CI + HS), the obtained solution by using the MTDE/rand/1 algorithm under $F = 0.5$, $CR = 0.9$ (the first row in Table 1) is near the true optimal solution, while the mean solution obtained by using the MTDE/rand/2 algorithm (the first row in Table 2) is really bad. This fact from a side indicates that these variants of MTDE (or MTDE-VT) can be considered as independent evolutionary algorithms.

By comparing different MTDE and the corresponding MTDE-VT algorithms, the effect of variable transformation strategy on different evolutionary algorithms can be observed and measured. Impressively, the MTDE/best/1 algorithm is obviously improved by variable transformation strategy. Except for the task 1 of problem 3 (CI + HS), the MTDE-VT/best/1 algorithm performs significantly better than the corresponding MTDE on the remaining 17 tasks.

However, the outcome of the MTDE-VT/rand/2 algorithm is not very ideal, and compared with MTDE/rand/2, it even goes bad for task 1 of problem 4 (PI + HS). We hold that the variable transformation strategy reveals a very positive effect on evolutionary algorithm with powerful exploitation. The possible reason may be that, for these MTO problems, MTDE/rand/2 under the given parameters can improve the population diversity but speed down the convergence to some extent.

Fortunately, the result of the MTDE-VT ($F = 0.7$, $CR = 0.3$) algorithm can further examine the rationality of the above explanation. Generally, a large value of CR speeds up convergence, and a larger $F$ increases the probability of
| Problem | Task | MTDE ($F=0.7$, $CR=0.3$) | MTDE-VT ($F=0.7$, $CR=0.3$) | MTDE (best/1) | MTDE-VT (best/1) | MTDE (rand/2) | MTDE-VT (rand/2) |
|---------|------|---------------------------|-----------------------------|---------------|-----------------|---------------|-----------------|
|         |      | Best | Mean | Sig | Best | Mean | Sig | Best | Mean | Sig | Best | Mean | Sig | Best | Mean | Sig | Best | Mean | Sig | Best | Mean | Sig | Best | Mean | Sig | Best | Mean | Sig |
| CI+ HS  | T1   | 0.9910 | 1.0287 | = | 0.9869 | 1.0272 | = | 1.2828 | 2.2215 | = | 0 | 0 | + | 2.5122 | 3.5455 | = | 2.4837 | 3.7895 | = | 0 | 0 | + | 1976.20 | 3087.01 | = | 2140.84 | 3193.10 | = |
|         | T2   | 486.803 | 540.931 | = | 495.718 | 532.786 | + | 568.791 | 1558.47 | = | 0 | 0 | + | 17.1592 | 20.3226 | = | 16.783 | 20.1552 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
| CI+ MS  | T1   | 21.0594 | 21.1618 | = | 20.9116 | 21.1648 | = | 13.4906 | 15.1868 | = | 8.8818E-16 | 8.8818E-16 | = | 17.1592 | 20.3226 | = | 16.783 | 20.1552 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
|         | T2   | 617.472 | 716.903 | = | 617.472 | 720.803 | = | 1455.94 | 2308.90 | = | 0 | 0 | + | 1931.12 | 2724.51 | = | 1566.39 | 2690.02 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
| CI+ LS  | T1   | 21.0932 | 21.2141 | = | 21.0409 | 21.1983 | = | 20.37 | 20.7181 | = | 20.3359 | 20.6890 | = | 21.1238 | 21.2084 | = | 21.1245 | 21.2104 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
|         | T2   | 9445.55 | 10155.5 | = | 9574.31 | 10206.9 | = | 1328.5 | 1435.7 | = | 1320.12 | 1437.8 | = | 1320.12 | 1437.8 | = | 1320.12 | 1437.8 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
| PI+ HS  | T1   | 539.825 | 586.451 | = | 479.513 | 573.891 | = | 1531.08 | 2806.08 | = | 1659.74 | 2857.91 | = | 2152.06 | 3153.18 | = | 2152.06 | 3153.18 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
|         | T2   | 148.619 | 214.091 | = | 157.479 | 194.669 | = | 4025.66 | 9437.99 | = | 1567.39 | 3059.43 | = | 1567.39 | 3059.43 | = | 1567.39 | 3059.43 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
| PI+ MS  | T1   | 21.1037 | 21.1725 | = | 21.0945 | 21.1716 | = | 11.3778 | 14.0907 | = | 11.3778 | 14.0907 | = | 11.3778 | 14.0907 | = | 11.3778 | 14.0907 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
|         | T2   | 1975854 | 3908437 | = | 1921341 | 3908437 | = | 1921341 | 3908437 | = | 1921341 | 3908437 | = | 1921341 | 3908437 | = | 1921341 | 3908437 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
| NI+ HS  | T1   | 77312.4 | 145253 | = | 62299.4 | 131280 | = | 1274640 | 7243283 | = | 1274640 | 7243283 | = | 1274640 | 7243283 | = | 1274640 | 7243283 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
|         | T2   | 554.873 | 631.959 | = | 532.431 | 631.201 | = | 996.083 | 1607.79 | = | 996.083 | 1607.79 | = | 996.083 | 1607.79 | = | 996.083 | 1607.79 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
| NI+ MS  | T1   | 1.0673 | 1.1268 | = | 1.0714 | 1.1263 | = | 1.2374 | 1.5161 | = | 1.2374 | 1.5161 | = | 1.2374 | 1.5161 | = | 1.2374 | 1.5161 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
|         | T2   | 60.9857 | 66.5399 | = | 56.7094 | 66.8431 | = | 28.8785 | 36.9356 | = | 28.8785 | 36.9356 | = | 28.8785 | 36.9356 | = | 28.8785 | 36.9356 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
| NI+ LS  | T1   | 640.299 | 736.840 | = | 636.713 | 723.386 | = | 2210.24 | 3746.83 | = | 2210.24 | 3746.83 | = | 2210.24 | 3746.83 | = | 2210.24 | 3746.83 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
|         | T2   | 8311.30 | 9011.46 | = | 7333.44 | 8784.23 | = | 8309.90 | 10851.1 | = | 8309.90 | 10851.1 | = | 8309.90 | 10851.1 | = | 8309.90 | 10851.1 | = | 0 | 0 | + | 1566.39 | 2690.02 | = | 0 | 0 | + |
escaping from a local optimum. Therefore, the MTDE (\(F=0.7, CR=0.3\)) algorithm has a good exploration ability, and the corresponding MTDE-VT (\(F=0.7, CR=0.3\)) algorithm does not reveal much more significance affection on the performance improvement. Specifically, from Table 2, it just performs significantly better than MTDE on 5 out of 18 tasks.

4.4. Experimental Series 3: Parameter Sensitivity Analysis.
As illustrated in Algorithm 3, the variable transformation strategy consists of three control parameters (\(\varphi, \theta, \mu\)). Here, we analyze their effects on the performance of MTDE-VT. To this end, we test MTDE-VT with various parameter settings on nine MTO problems. In this paper, we set \(\mu=2\%\), 10\%, 20\%, 40\%, and 50\%; \(\varphi=0.2\_start, 0.1\_start, 0, 0.8\_stop,\) and \(0.9\_stop\); and \(\theta=0.05, 0.02, 0.01,\) and \(0.001\) as the potential values. The MTDE-VT algorithm runs under all of these parameter combinations. In total, that is exactly 100 parameter combinations (or MTDE-VT algorithms) in this paper. The other parameters and differential evolution operators remain the same as those in Section 4.1.

It should be noted that, in this paper, the variable transformation strategy may not be conducted at the beginning or the end of population evolution. Specifically, \(g > \varphi \times \text{Max}_\text{gen}\) and \(g < \varphi \times \text{Max}_\text{gen}\) represent two conditions, respectively, as described by Line 1 in Algorithm 3. In this paper, the suffix “start” and “stop” are used to distinguish the two cases clearly. For example, \(\varphi=0.2\_start\) means that the variable transformation strategy is conducted from the \(0.2\times\text{Max}_\text{gen}\) generation to the end of population evolution. Similarly, \(\varphi=0.8\_stop\) means that it is conducted from the very beginning to the \(0.8\times\text{Max}_\text{gen}\) generation.

In this paper, 100 MTDE-VT algorithms with different key parameters (\(\theta, \varphi, \mu\)) are analyzed. In order to distinguish them, the serial number of each MTDE-VT algorithm is used and calculated based on their parameter sequences as follows. Firstly, let \(N\theta=1, 2, 3,\) and \(4\) represent \(\theta=0.05,\) 0.02, 0.01, and 0.001, respectively. In the similar way, \(N\varphi=1, 2, 3, 4,\) and \(5\) represent \(\varphi=0.2\_start, 0.1\_start, 0, 0.8\_stop,\) and \(0.9\_stop\); and \(N\mu=1, 2, 3, 4,\) and \(5\) represent \(\mu=2\%\), 10\%, 20\%, 40\%, and 50\%, respectively. Finally, the serial number of each MTDE-VT algorithm is equal to \((N\theta-1) \times 25 + (N\varphi-1) \times 5 + N\mu\). For example, the “96” of MTDE-VT algorithms means that \(\theta=0.001\) (\(N\theta=4\)), \(\varphi=0.9\_stop\) (\(N\varphi=5\)), and \(\mu=2\%\) (\(N\mu=1\)).

In order to compare the performance of MTDE-VT algorithms under diverse parameter settings, we rank the tested algorithms as their reward points. For each algorithm, the reward point is the obtained rankings with respect to each component task of each MTO problem. Then, we accumulate the obtained reward points over all component tasks of all MTO problems to obtain the final score.

For instance, Tables 3 and 4 give an illustrative example of scoring procedure. Specifically, Table 3 gives the median fitness values obtained by all algorithms with different parameters. The smaller the fitness value is, the better the performance of the algorithm under the given parameter is. Then, we get the reward points of all algorithms with respect to each component task, as depicted in Table 4. According to their final scores, we can conclude that Algorithm A is the best algorithm and B is the worst one over three MTO test problems.

Generally speaking, poor individuals are rare but have a great impact on the mean fitness values. In consideration of both statistical features and fairness, we use the median fitness rather than the mean fitness, to rank and score the MTDE-VT algorithms with diverse parameter settings. The total score (reward points) of all algorithms are illustrated in Figure 2. Note that, the x-axis represents the serial number of MTDE-VT algorithms with different parameter settings. For example, the “96” in Figure 2 means that \(\mu=2\%\), \(\varphi=0.9\_stop\), and \(\theta=0.001\) as mentioned above.

As can be observed from Figure 2, MTDE-VT under the “76” parameter setting (\(\mu=2\%\), \(\varphi=0.2\_start\), and \(\theta=0.001\)) gets the highest score (1276 points), which means it offers the best algorithm performance over all MTO problems. The median best fitness and the corresponding ranking are recorded in the last row in Table 1. We can see that MTDE-VT (optimal) performs significantly better than the original MTDE-VT on 12 out of 18 tasks and is worse than it on 6 tasks. The original MTDE-VT (“34” parameter setting) studied in Section 4.2 gets 1149 reward points, and ranks 57 among all test parameter settings.

The detail results of the best fitness are displayed in Table 5. Taken individually, the MTDE-VT algorithm with “81” parameter setting exhibits the best performance on 10 tasks, and “76” parameter setting does well on 2 tasks.

Furthermore, the algorithm performance trends of three control parameters are investigated, as shown in Figures 3–5. The algorithm performance is measured by the mean reward points over all tasks and all parameter combinations. For example, for the mean reward points of \(\mu\) (e.g., 2\%), we should take into account all 20 parameter combinations of \(\varphi\) (namely, \(0.2\_strat, 0.1\_start, 0, 0.8\_stop,\) and \(0.9\_stop\)) and \(\theta\) (namely, 0.05, 0.02, 0.01, and 0.001).

As mentioned above, \(\mu\) represents the top percent best solutions in the current population. A smaller \(\mu\) means a few but excellent individuals are chosen to calculate the
estimated optimal solution. As a result, the estimated optimal solution can lead the whole population to search for more potential areas. Certainly, in the extreme case (e.g., $\mu = 2\%$ in this paper), only the best solution are used to estimate the optimal solution of the whole population, which may destroy the diversity of the population. Therefore, its performance drops sharply, as shown in Figure 3.

Moreover, $\theta$ is another key parameter of variable transformation strategy. A large number means few variable transformation strategies are executed, while a small number means more frequent operation. From Figure 4, it is reasonable to set a smaller value of $\theta$ (e.g., $\theta = 0.01$). It indicates that the proposed variable transformation strategy can help the population to evolve towards the global optimal solution. On the contrary, this strategy cannot be used too frequently because it is possible to fast fall into local optimal solution. For example, when $\theta = 0.001$, the variable transformation strategy is executed every generation. Unfortunately, its mean reward points had actually fallen relative to the optimal case ($\theta = 0.01$ in this paper).

The third parameter, $\varphi$, can decide when to start and stop the variable transformation strategy. The parameter values we studied here can be divided into three groups. The first group is 0, which means the variable transformation strategy is executed during the whole evolution process. The second one is 0.1_start and 0.2_start, which mean it is executed from 0.1 $\times$ Max$\_gen$ or 0.2 $\times$ Max$\_gen$ to the end, respectively. The third one is 0.8_stop and 0.9_stop, which mean it is executed from the beginning to 0.8 $\times$ Max$\_gen$ or 0.9 $\times$ Max$\_gen$, respectively. We can see from Figure 5, MTDE-VT under the second parameter group illustrates the best performance compared with other two parameter groups. There is reason to believe that it is good choice to run the variable transformation strategy after
some generations. In other words, the function of the proposed strategy will be more obvious at the beginning of population evolution than the ending part.

From Figures 3–5, it can be observed that the optimal parameters should be set as $\mu = 0.1$, $\varphi = 0.2_{\text{start}}$, and $\theta = 0.01$. In fact, MTDE-VT under this parameter combination ranks fifth out of 100 algorithms. Although it is not the best one, it is a satisfactory good recommendation in most cases.

5. Conclusions

In this paper, MTO problem with different optimal solutions has been investigated. It is found that, within the range between these optimal solutions, the trends of these objective functions are in different directions, which may lead to negative knowledge transfer. To boost the task similarity and robust evolutionary multitasking performance in solving MTO problems, a novel variable transformation strategy and the corresponding inverse transformation are proposed. After using variable transformation strategy, the estimated optimal solutions of all tasks are near the center point of the unified search space. Furthermore, all individuals in the transformed space will evolve to the center point, avoiding advancing in the opposite direction. Therefore, the effectiveness of knowledge transfer will probably be positive and can help us to improve the convergence speed in this case.

To crystallize this idea, we embed the proposed variable transformation strategy into MTDE, forming a novel algorithm (called MTDE-VT algorithm). The experiment results demonstrate that variable transformation strategy is an effective approach to exploit the promising areas and it works well for all evolutionary algorithms. Furthermore, the basic principle and the good parameter combination are also provided based on massive simulated data.

For future works, we would like to further study the effectiveness of variable transformation strategy when solving more complex benchmark and practical MTO problems. Moreover, we also plan to explore the similarity measure between constituent tasks and design new strategy to boost the task similarity, which can be considered as guidance for applying variable transformation strategy into the evolutionary algorithm.

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 work was partially supported by the National Natural Science Foundation of China under Grant 61773314, Natural Science Basic Research Program of Shaanxi under Grants 2019JZ-11 and 2020JMJ709, Scientific Research Foundation of National University of Defense Technology under Grant ZK18-03-43, and Research Development Foundation of Test and Training Base under Grant 23.
References

[1] 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.

[2] C. M. Fonseca and P. J. Fleming, “An overview of evolutionary algorithms in multiobjective optimization,” Evolutionary Computation, vol. 3, no. 1, pp. 1–16, 1995.

[3] X. F. Wu, S. W. Zhang, Z. Gong, J. K. Ji, Q. Z. Lin, and J. Y. Chen, “Decomposition-based multiobjective evolutionary optimization with adaptive multiple Gaussian process models,” Complexity, vol. 2020, no. 1, 22 pages, Article ID 9643273, 2020.

[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, no. 1, 12 pages, Article ID 1805198, 2019.

[5] 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.

[6] Y.-S. Ong and A. Gupta, “Evolutionary multitasking: a computer science view of cognitive multitasking,” Cognitive Computation, vol. 8, no. 2, pp. 125–142, 2016.

[7] 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.

[8] 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.

[9] L. Zhou, L. Feng, J. H. Zhong, Z. X. Zhu, B.S. Da, and Z. Wu, “A study of similarity measure between tasks for multifactor evolutionary algorithm,” in Proceedings of the Genetic And Evolutionary Computation Conference, pp. 229–230, Kyoto, Japan, July 2018.

[10] 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.

[11] 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.

[12] R. T. Liaw and C. K. Ting, “Evolutionary many-tasking based on biocoenosis through symbiosis: a framework and benchmark problems,” in Proceedings of the IEEE Congress on Evolutionary Computation, pp. 2266–2273, San Sebastian, Spain, July 2017.

[13] G. H. Li, Q. Z. Lin, and W. F. Gao, “Multifactorial optimization via explicit multipopulation evolutionary framework,” Information Sciences, vol. 512, pp. 1555–1570, 2020.

[14] M. Gong, Z. Tang, H. Li, and J. Zhang, “Evolutionary multitasking with dynamic resource allocating strategy,” IEEE Transactions on Evolutionary Computation, vol. 23, no. 5, pp. 858–869, 2019.

[15] Y. Hou, Y.-S. Ong, L. Feng, and J. M. Zurada, “An evolutionary transfer reinforcement learning framework for multiagent systems,” IEEE Transactions on Evolutionary Computation, vol. 21, no. 4, pp. 601–615, 2017.

[16] 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.

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

[18] Z. Wang and X. P. Wang, “Multiobjective multifactorial operation optimization for continuous annealing production process,” Industrial & Engineering Chemistry Research, vol. 58, no. 41, pp. 19166–19178, 2019.

[19] L. Feng, L. Zhou, J. Zhong et al., “Evolutionary multitasking via explicit autoencoding,” IEEE Transactions on Cybernetics, vol. 49, no. 9, pp. 3457–3470, 2019.

[20] L. Feng, Y. X. Huang, L. Zhou et al., “Explicit evolutionary multitasking for combinatorial optimization: a case study on capacitated vehicle routing problem,” IEEE Transactions on Cybernetics, 2019.

[21] Z. P. Liang, J. Zhang, L. Feng, and Z. X. Zhu, “A hybrid of genetic transform and hyper-rectangle search strategies for evolutionary multi-tasking,” Expert Systems with Applications, vol. 138, no. 1, pp. 1–18, 2019.

[22] L. Zhou, L. Feng, K. C. Tan et al., “Toward adaptive knowledge transfer in multifactorial evolutionary computation,” IEEE Transactions on Cybernetics, 2020.

[23] Z. P. Liang, H. Dong, C. Liu, W. Q. Liang, and Z. X. Zhu, “Evolutionary multitasking for multiobjective optimization with subspace alignment and adaptive differential evolution,” IEEE Transactions on Cybernetics, 2020.

[24] 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.

[25] K. K. Bali, A. Gupta, Y. S. Ong, and P. S. Tan, “Cognizant multitasking in multiobjective multifactorial evolution: MO-MFEA-II,” IEEE Transactions on Cybernetics, 2020.

[26] E. Osaba, A. D. Martinez, A. Galvez, A. Iglesias, and J. D. Ser, “dMFEA-II: an adaptive multifactorial evolutionary algorithm for permutation-based discrete optimization problems,” in Proceedings of the Genetic and Evolutionary Computation Conference, pp. 1690–1696, Cancun, Mexico, 2020.

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

[28] 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.

[29] A. Rauniar, R. Nath, and P. K. Muhuri, “Multi-factorial evolutionary algorithm based novel solution approach for multi-objective pollution-routing problem,” Computers & Industrial Engineering, vol. 130, no. 1, pp. 757–771, 2019.

[30] C. Yang, J. Ding, Y. Jin, C. Wang, and T. 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.

[31] D. Singh, D. S. Sisodia, and P. Singh, “Compositional framework for multitask learning in the identification of cleavage sites of HIV-1 protease,” Journal of Biomedical Informatics, vol. 102, no. 1, pp. 1–17, Article ID 103376, 2020.

[32] H. Li, Y.-S. Ong, M. Gong, and Z. Wang, “Evolutionary multitasking sparse reconstruction: framework and case study,” IEEE Transactions on Evolutionary Computation, vol. 23, no. 5, pp. 733–747, 2019.
[33] Y. Z. Zhao, H. Li, Y. Wu, S. F. Wang, and M. G. Gong, "Endmember selection of hyperspectral images based on evolutionary multitask," in Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1–7, Glasgow, UK, July 2020.

[34] J. H. Zhong, L. Feng, W. T. Cai, and Y. S. Ong, "Multifactorial genetic programming for symbolic regression problem," IEEE Transactions on Systems, Man, and Cybernetics: Systems.

[35] R. Storn and K. Price, “Differential evolution—a simple and efficient adaptive scheme for global optimization over continuous spaces,” Journal of Global Optimization, vol. 23, no. 1, 1995.

[36] A. K. Qin and P. N. Suganthan, “Self-adaptive differential evolution algorithm for numerical optimization,” in Proceedings of the IEEE Congress on Evolutionary Computation, pp. 1785–1791, Edinburgh, UK, September 2005.

[37] B. S. Da, Y. S. Ong, L. Feng et al., Evolutionary Multitasking for Single-Objective Continuous Optimization: Bench-Mark Problems, Performance Metric and Baseline Results, Nanyang Technological University, Singapore, 2016.

[38] F. Wilcoxon, “Individual comparisons by ranking methods,” in Breakthroughs in Statistics, S. Kotz and N. L. Johnson, Eds., pp. 196–202, Springer, New York, NY, USA, 1992.