Evolutionary Multitasking for Multiobjective Optimization With Subspace Alignment and Adaptive Differential Evolution

Zhengping Liang, Hao Dong, Cheng Liu, Weiqi Liang, and Zexuan Zhu, *Member, IEEE*

***Abstract*—In contrast to the traditional single-tasking evolutionary algorithms, evolutionary multitasking (EMT) travels in the search space of multiple optimization tasks simultaneously. Through sharing knowledge across the tasks, EMT is able to enhance solving the optimization tasks. However, if knowledge transfer is not properly carried out, the performance of EMT might become unsatisfactory. To address this issue and improve the quality of knowledge transfer among the tasks, a novel multiobjective EMT algorithm based on subspace alignment and self-adaptive differential evolution (DE), namely, MOMFEA-SADE, is proposed in this article. Particularly, a mapping matrix obtained by subspace learning is used to transform the search space of the population and reduce the probability of negative knowledge transfer between tasks. In addition, DE characterized by a self-adaptive trial vector generation strategy is introduced to generate promising solutions based on previous experiences. The experimental results on multiobjective multi/many-tasking optimization test suites show that MOMFEA-SADE is superior or comparable to other state-of-the-art EMT algorithms. MOMFEA-SADE also won the Competition on Evolutionary Multitask Optimization (the multitask multiobjective optimization track) within IEEE 2019 Congress on Evolutionary Computation.**

***Index Terms*—Differential evolution (DE), evolutionary multitasking (EMT), many-tasking optimization, multiobjective optimization, subspace learning.**

Manuscript received August 31, 2019; revised January 21, 2020 and March 9, 2020; accepted March 11, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61871272, in part by the Natural Science Foundation of Guangdong Province, China, under Grant 2020A151501479, and in part by the Shenzhen Scientific Research and Development Funding Program under Grant JCYJ20190808173617147 and Grant GGFW2018020518310863. This article was recommended by Associate Editor Y. S. Ong. *(Corresponding author: Zexuan Zhu.)*

Zhengping Liang, Hao Dong, Cheng Liu, and Weiqi Liang are with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China (e-mail: liangzp@szu.edu.cn; 1242618848@qq.com; 1553500031@qq.com; 974860726@qq.com).

Zexuan Zhu is with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518060, China, also with the Shenzhen Pengcheng Laboratory, Shenzhen 518055, China, and also with the SZU Branch, Shenzhen Institute of Artificial Intelligence and Robotics for Society, Shenzhen University, Shenzhen 518060, China (e-mail: zhuzx@szu.edu.cn).

This article has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the authors.

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2020.2980888

I. Introduction

MULTIOBJECTIVE optimization problems (MOPs) are ubiquitous in the real world [1], [2]. Without loss of generality, an MOP can be defined as

$$\begin{aligned} &\min_{\mathbf{x}}(\mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))) \\ &\text{subject to: } \mathbf{x} \in \Omega^d \end{aligned} \tag{1}$$

where $\mathbf{x} = \{x_1, \ldots, x_d\}$ denotes the decision variable in the search space Ω^d, d is the dimensionality of the decision variables, and $\mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))$ indicates the objective function vector consisting of m conflicting objective functions. Since the objective functions conflict with each other, optimizing one function could lead to deterioration of another, that is, it is impractical to identify one single optimal solution for all objectives. Instead, the tradeoff solutions based on Pareto dominance are pursued. Given two different solutions $\mathbf{x}$ and $\mathbf{y}$, if $\forall i \in \{1, \ldots, m\}$, $f_i(\mathbf{x}) \leq f_i(\mathbf{y})$ and $\mathbf{F}(\mathbf{x}) \neq \mathbf{F}(\mathbf{y})$, $\mathbf{x}$ is said to dominate $\mathbf{y}$ (denoted as $\mathbf{x} \prec \mathbf{y}$). A solution $\mathbf{x}^*$ not dominated by any other solution is called a Pareto-optimal solution. The set of all Pareto-optimal solutions is referred to as the Pareto-optimal set (PS). The corresponding projection of PS in the objective space is called the Pareto front (PF).

Multiobjective evolutionary algorithms (MOEAs), capable of obtaining a set of nondominated solutions in one run, have been widely used to solve MOPs [3]. Traditional MOEAs can be generally divided into three categories, that is, domination-based algorithms [4]–[6], decomposition-based algorithms [7]–[12], and indicator-based algorithms [13]–[15]. These MOEAs were designed to handle one MOP independently at a time and they must be run from scratch to deal with new problems. However, many MOPs in the real world are related with each other, that is, beneficial knowledge collected in optimizing one MOP can be used to solve other relevant problems.

Inspired by the capability of the human brain to process transactions in parallel, Gupta *et al.* [16] proposed a new paradigm called evolutionary multitasking (EMT) to deal with multiple optimization tasks simultaneously. EMT algorithms tend to obtain better solutions for the optimization tasks than the counterpart single-task-based algorithms thanks to the potential synergistic effect between tasks. EMT has gained increasing attention and achieved successes in various complex optimization problems including MOPs [17]–[25]. For example, based on NSGA-II [4], Gupta *et al.* [26] proposed

a multiobjective EMT algorithm, namely, MO-MFEA to solve multiple MOPs simultaneously with one population. MO-MFEA is characterized by knowledge transfer across the distinct MOPs via assortative mating and vertical cultural transmission. Tuan *et al.* [27] introduced the local search algorithm into MO-MFEA to further improve the quality of elite individuals. Feng *et al.* [28] presented EMT-EGT using different solvers for multiple MOPs based on the autoencoder. Yang *et al.* [29] put forward TMO-MFEA to deal with diversity variables and convergence variables separately in multiple MOPs. Lin *et al.* [30] suggested a knowledge transfer approach for multiobjective multitasking optimization by involving the neighbors of nondominated transferred solutions.

Although EMT has shown promising results in MOPs, especially when the optimized tasks are correlated or share a high similarity, it is impractical to assume a high degree of similarity between every two tasks and EMT could suffer from negative transfer [31], that is, knowledge sharing across tasks lead to negative impact. Moreover, the evolutionary experience could serve as the best teacher [32], yet the archive of nondominant solutions in the evolutionary process is not fully used in the existing multiobjective EMT algorithms. To address these issues, this article proposes an EMT algorithm, namely, MOMFEA-SADE, based on subspace alignment (SA) and self-adaptive differential evolution (DE). Particularly, SA is applied in unified space to minimize the distribution difference between any two tasks, such that the population of one task can be well adapted to guide the search of another task, leading to more efficient knowledge transfer across tasks. Self-adaptive DE [33] is an optimization algorithm with diverse strategies applicable to different periods of evolution. The strategies are featured by different search behavior. The frequency of each DE strategy producing nondominated solutions is collected and stored as the experience, based on which the probability of using a specific strategy can be estimated. To verify the efficacy of MOMFEA-SADE, comprehensive empirical studies are conducted on two multiobjective multitasking optimization benchmarks and one multiobjective many-tasking optimization test suite. The experimental results demonstrate that MOMFEA-SADE is superior or comparable to other advanced EMT algorithms. MOMFEA-SADE is also the champion of the Competition on Evolutionary Multitask Optimization (the multitask multiobjective optimization track) within IEEE 2019 Congress on Evolutionary Computation (CEC 2019) [34]. The main contributions of this article are highlighted as follows.

1) A new search space mapping mechanism, namely, SA is designed to enable efficient and high-quality knowledge transfer among different tasks.
2) An adaptive DE strategy is introduced to enable the algorithm to select the appropriate trial vector generation method. SA is embedded in the adaptive DE to enhance knowledge transfer.
3) The performance of MOMFEA-SADE is investigated in various benchmark problems and compared with three state-of-the-art EMT algorithms, that is, MO-MFEA, EMT-EGT, and TMO-MFEA, and a baseline single-tasking algorithm NSGA-II. Both strength and weakness of the proposed method are discussed.

The remainder of this article is constructed as follows. Section II reviews the relevant works of the proposed algorithm. Section III introduces the details of MOMFEA-SADE. The experimental studies are presented in Section IV. Finally, Section V summarizes this article and prospects the potential future research directions.

II. Related Work

A. Evolutionary Multitasking Algorithms

Facing multiple optimization tasks, the majority of the existing evolutionary algorithms process one task at a time (also known as single-tasking algorithms) and hardly use knowledge information of one task to facilitate the solving of another. To address this issue, transfer learning and multitasking learning have been introduced to learn the knowledge across the related tasks. The purpose of transfer learning is to improve the performance in the target task by transferring knowledge from the source task. Multitasking learning considers the internal relationships of multiple related tasks at the same time to improve the performance of all tasks. Gupta *et al.* [16] first applied multitasking learning to evolutionary computation leading to a new paradigm EMT and proposed a multifactorial evolutionary algorithm (MFEA), which aims to exploit the potential synergies between different optimization problems. Thereafter, many EMT algorithms have been successfully proposed to address various complex multitasking problems.

The existing EMT algorithms were designed to improve the solving of multiple optimization tasks based on different philosophies. For example, from the perspective of computation resource redistribution of EMT, Wen and Ting [35] proposed a computational resource allocation strategy in EMT, where the frequency of knowledge transfer is proportional to the probability of knowledge transfer that can produce improved solutions. Gong *et al.* [36] put forward an EMT algorithm, namely, MTO-DRA to enable dynamic resources allocation according to task requirements, such that more computing resources can be assigned to complex tasks.

Toward adaptive knowledge transfer, Zheng *et al.* [37] defined a novel notion of ability vector to capture the correlations between tasks and changed the intensity of knowledge transfer to enhance the performance of the EMT algorithm. Bali *et al.* [38] introduced an improved MFEA to detect the similarity between tasks through online learning and accordingly adjust the frequency of knowledge transfer. Yang *et al.* [29] presented the TMO-MFEA algorithm, in which decision variables are divided into two types, namely, diversity variables and convergence variables. The knowledge transfer on diversity variables is intensified to obtain evenly distributed solutions over the PF, whereas the knowledge transfer on convergence variables is restrained to maintain the convergence of the solution population toward the PF. Zhou *et al.* [39] proposed the MFEA-AKT where the crossover operator used to achieve knowledge transfer is self-adapted according to the information collected along the evolutionary search process.

Based on search space mapping among the tasks, Ding *et al.* [40] presented GMFEA to transfer the locations of decision variables in different problem spaces and extended the basic MFEA to handle expensive problems. Bali *et al.* [41] embedded a linear domain adaptation method in MFEA to map two tasks into a higher order representation space where similarity sharing is easier. Feng *et al.* [28] implemented search space mapping by autoencoder. The method is characterized by explicit knowledge transfer with which search mechanisms with different preferences can be embedded together to take advantage from each of them. They further improved the explicit knowledge transfer to address combinatorial optimization problems based on the weighted l_1-norm-regularized learning process for capturing the transfer mapping in [42].

To handle many-tasking problems, Liaw and Ting [43] introduced a new EMT framework to control knowledge transfer and select candidate solutions in optimizing more than three tasks simultaneously. Tang *et al.* [44] suggested a group-based MFEA to solve many-tasking problems by clustering tasks with similar characteristics and transferring knowledge between tasks within the same group. Chen *et al.* [45] came up with a many-tasking framework called MATEA that identifies the most likely beneficial task for knowledge transfer based on the Kullback–Leibler (KL) divergence and transfers knowledge via a new crossover method.

Thanks to the effectiveness of the multitasking algorithms, they have been successfully applied in many practical scenarios. EMT was used to solve semantic Web service composition in [46], where the similarity of different service composition requests is considered to improve the quality of the final solution. Bao *et al.* [23] proposed an EMT algorithm, namely, EMA-CCSC to efficiently solve the computing service composition problem in cloud computing. Rauniyar *et al.* [47] put forward an MFEA to solve the pollution-routing problem. In [42], the proposed EMT algorithm, namely, EEMTA, showed promising performance on capacitated vehicle routing problems.

The majority of the existing EMT algorithms were designed for single-objective problems. More efforts should be put into multiobjective multi/many-tasking problems and there remain many open questions in EMT for multiobjective optimization. This article presents an attempt to address multiobjective multitasking problems by enhancing search space mapping and acknowledge transfer in the EMT framework.

B. MO-MFEA

MO-MFEA [16] is a basic implementation of EMT for MOPs which incorporates NSGA-II into the traditional EMT framework. The procedure of MO-MFEA is outlined in Algorithm 1. To enable different tasks to transfer knowledge seamlessly in MO-MFEA, a platform is needed to compare solutions associated with different tasks. First, a unified search space $\mathbf{Y}$ defined in (2) is established to encode the individuals from all tasks

$$y_i = (x_i - L_i)/(U_i - L_i) \tag{2}$$

Algorithm 1 General Framework of MO-MFEA

Input: N, the number of individuals in the population.
1: Generate N individuals in unified space $\mathbf{Y}$ to form the initial population $\mathbf{O}$;
2: Calculate the Skill Factor, Factorial Rank, and Scalar Fitness of each individual;
3: **while** stopping conditions are not satisfied **do**
4: Perform assortative mating to evolve the population and obtain the offspring;
5: Determine the task of the offspring according to the vertical cultural transmission, and then evaluate the offsprings;
6: Carry out environmental selection, eliminate poor individuals, and obtain new populations;
7: **end while**

where x_i is the value of the ith dimension of an individual in the original space, U_i and L_i represent the upper and lower limits of an individual in the ith dimension, respectively, and y_i is the value of x_i in the unified space $\mathbf{Y}$. The decision variables of all individuals are mapped to [0, 1] according to (2). The normalized individuals can be used in all tasks, which guarantee assortative mating being proceeded smoothly. The dimensionality of $\mathbf{Y}$ is denoted as $D_{\text{multitask}} = \max\{D_t\}, t = 1, 2, \ldots, T$, where D_t is the dimensionality of the optimization task t and T is the total number of tasks. To compare individuals from different tasks in the unified space, MO-MFEA uses the following properties as defined in [16].

1) *Factorial Rank:* To distinguish the priority of an individual o_i in a task t, the factorial rank r_i^t is defined as the rank of o_i in the population in terms of some specific criteria, for example, the fitness value $f_t(\cdot)$ in a single-objective problem, or the nondominated ranking and crowded distance [4] in an MOP.
2) *Scalar Fitness:* The scalar fitness φ_i of an individual o_i is defined as $\varphi_i = 1/\min_{t\in\{1,\ldots,T\}} r_i^t$, that is, the reciprocal of the minimum factorial rank of o_i.
3) *Skill Factor:* The skill factor τ_{o_i} of an individual o_i is assigned as $\tau_{o_i} = \text{argmin}_t\{r_i^t\}$, that is, among all tasks, o_i performs best in task τ_{o_i}.

Herein, the cultural trait from parents in MO-MFEA can be inherited to offsprings by the Skill Factor. Scalar Fitness is considered as the unified performance criterion.

In Algorithm 1, the knowledge transfer is mainly reflected in the assortative mating (line 4) and the vertical cultural transmission (line 5). In assortative mating, knowledge transfer among different tasks happens in a unified space with implicit genetic transfer through the individual crossover. Knowledge transfer is also achieved by changing the Skill Factor in vertical cultural transmission. However, the search spaces of the two tasks are different. If the individuals associated with different tasks directly perform crossover to produce offsprings, the obtained offspring are likely eliminated due to the negative transfer. Moreover, in the vertical cultural transmission, each offspring is randomly assigned to a task, which may cause misclassification of the individuals.

As an example shown in Fig. 1, the obtained solutions of MO-MFEA on the two tasks T_1 and T_2 defined in the CILS problem [48] are plotted in the objective space of T_1. Let the circles represent the individuals associated with T_1 and

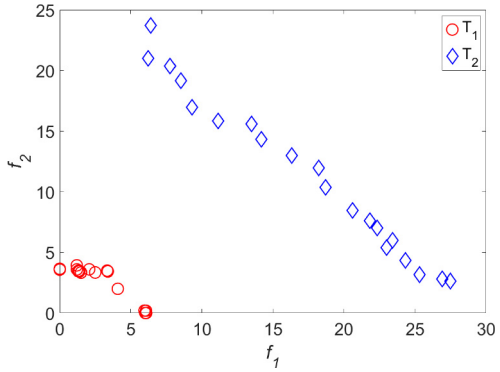


Fig. 1. Population distribution of T_1 and T_2 in T_1's objective space obtained by MO-MFEA in the CILS problem.

the diamonds be the individuals associated with T_2 projected in the objective space of T_1 by vertical cultural transmission. The population distributions in the objective space show different characteristics. In this scenario, the knowledge attained in solving one task is unhelpful or even harmful to solve the other problem. We propose a new algorithm to solve the above issue in the next section.

III. Proposed Algorithm

This section details the framework of the proposed MOMFEA-SADE and the two key components, that is, the SA and adaptive DE. The complexity of MOMFEA-SADE is also analyzed at the end of this section.

A. Framework of MOMFEA-SADE

The framework of MOMFEA-SADE is outlined in Algorithm 2. MOMFEA-SADE follows the general framework of MO-MFEA except that SA and adaptive DE strategies are introduced to enable efficient and high-quality knowledge transfer among different tasks. The algorithm initializes a population $\mathbf{O}$ of N individuals in the unified space in line 1 and then evaluates the Factorial Rank, Scalar Fitness, and Skill Factor of each individual based on nondominated ranking and crowded distance in line 2. It is important to note that all tasks share the same population throughout the evolution. In line 3, the probability vector Ψ is set to $[1/k, 1/k, \ldots, 1/k]$ (k is the total number of DE strategies) such that each DE strategy can be equally selected. In each evolution generation (lines 6 and 7), a parent is selected from $\mathbf{O}$ and a DE strategy is chosen subject to Ψ. An offspring population $\mathbf{C}$ is generated in lines 8 and 9 based on SA and Algorithm 4. Finally, in line 12, a new generation is picked out from the joint $\mathbf{O}$ and $\mathbf{C}$ with the environmental selection. Following [26], in the environmental selection, the same number of best individuals is selected for each task in terms of the Scalar Fitness. Therefore, the population $\mathbf{O}$ actually can be divided into multiple equal-sized subpopulations according to the Skill Factor and the subpopulations are used in SA to calculate the mapping matrices among different tasks. The probability vector Ψ is updated in line 13. The details of SA and adaptive DE are provided in the following sections.

Algorithm 2 Framework of MOMFEA-SADE

Input: N, the number of individuals in the population.
Output: a series of nondominated solutions.
1: Initialize an population $\mathbf{O}$ of N individuals;
2: Evaluate the Factorial Rank, Scalar Fitness, and Skill Factor of each individual in $\mathbf{O}$;
3: Initialize $\Psi = [\psi_1, ..., \psi_k]$ with $\psi_i = 1/k$ and $i = \{1, ..., k\}$;
4: **while** stopping conditions are not satisfied **do**
5: **for** $j = 1 \rightarrow N$ **do**
6: Randomly select an individual $\mathbf{o}_j$ from $\mathbf{O}$ as the parent;
7: Select a trial vector generation strategy according to Ψ;
8: Generate an offspring $\mathbf{c}_j$ based on SA and Algorithm 4;
9: $\mathbf{C} = \mathbf{C} \cup \mathbf{c}_j$;
10: **end for**
11: $\mathbf{O} = \mathbf{O} \cup \mathbf{C}$;
12: Environment selection;
13: Update Ψ;
14: **end while**

B. Subspace Alignment

In recent years, transfer learning has been proposed to solve the issue of knowledge transfer between different problems in machine learning [31], [49], [50]. Domain adaptation is an important research direction in transfer learning and subspace learning is commonly used to achieve domain adaptation [51], [52]. In this article, a subspace learning method, namely, SA [51] is applied to EMT. The aim of SA is to minimize the discrepancy between the source data and target data.

SA learns a new domain-invariant feature representation by finding a new projection space. Specifically, SA projects the source-domain data and the target-domain data into the respective subspaces based on the principal component analysis (PCA). A transformation matrix $\mathbf{M}$ is learned to map the source subspace to the target one. The process of SA is detailed as follows. First, PCA-based dimensionality reduction is used to obtain the eigenvectors of the source data and target data, respectively. The top h eigenvectors of which the corresponding eigenvalues can retain 95% of the information are used as the bases to generate the subspaces of the source data and target data. Given the corresponding subspaces $\mathbf{S}_1$ and $\mathbf{S}_2$, where $\mathbf{S}_1, \mathbf{S}_2 \in \mathbf{R}^{r\times h}$, r represents the dimensionality of the source data and target data, and h denotes the dimensionality of the subspace. The optimal alignment of $\mathbf{S}_1$ and $\mathbf{S}_2$ is achieved by optimizing the linear transform function L, namely, the Bregman matrix divergence

$$L(\mathbf{M}) = \|\mathbf{S}_1 \cdot \mathbf{M} - \mathbf{S}_2\|_F^2 \tag{3}$$

where $\|\cdot\|_F^2$ indicates the Frobenius norm and $\mathbf{M}$ represents the transformation matrix to minimize the difference between $\mathbf{S}_1$ and $\mathbf{S}_2$. The optimal solution $\mathbf{M}^*$ of $L(\mathbf{M})$ is denoted as follows:

$$\mathbf{M}^* = \arg\min_{\mathbf{M}}(L(\mathbf{M})). \tag{4}$$

To solve (4), orthonormal operations are applied to (3), such that

$$L(\mathbf{M}) = \left\|\mathbf{S}_1' \cdot \mathbf{S}_1 \cdot \mathbf{M} - \mathbf{S}_1' \cdot \mathbf{S}_2\right\|_F^2 \tag{5}$$

where $\mathbf{S}_1'$ represents the orthonormal $\mathbf{S}_1$ and $\mathbf{S}_1' \cdot \mathbf{S}_1 = \mathbf{I}$ with $\mathbf{I} \in \mathbf{R}^{h \times h}$ being an identity matrix. From (5), we can derive

$$L(\mathbf{M}) = \left\| \mathbf{M} - \mathbf{S}_1' \cdot \mathbf{S}_2 \right\|_F^2. \tag{6}$$

According to (6), the optimal solution $\mathbf{M}^*$ of (4) is obtained via

$$\mathbf{M}^* = \mathbf{S}_1' \cdot \mathbf{S}_2. \tag{7}$$

This implies that the new coordinate system of $\mathbf{S}_1$, labeled as $\mathbf{S}_1^*$, is equivalent to $\mathbf{S}_1 \cdot \mathbf{M}^*$, that is

$$\mathbf{S}_1^* = \mathbf{S}_1 \cdot \mathbf{M}^*. \tag{8}$$

The subspace $\mathbf{S}_1$ can be closest mapped to $\mathbf{S}_2$ via $\mathbf{M}^*$. The transfer of data from the source domain to the target domain can thus be conducted by the multiplication operation with $\mathbf{M}^*$.

SA is applied to EMT for establishing the connection among the tasks. All tasks in MOMFEA-SADE are mapped to the unified search space, yet they have different solution distributions and are biased in different search regions in the unified space. To alleviate the negative migration caused by the bias toward different regions, it is necessary to map the individuals of a task to the space of another task. When two tasks communicate in the assortative mating, spatial transformation is first carried out through SA. It can align the subspaces of two tasks to ensure they have similar distributions in their subspaces. Consequently, one task can be endowed with the distribution characteristics of the other. This provides a way to achieve an efficient knowledge transfer.

Given two subpopulations $\mathbf{P} = \{\mathbf{p}_1, \ldots, \mathbf{p}_n\}$ and $\mathbf{Q} = \{\mathbf{q}_1, \ldots, \mathbf{q}_n\}$ each associated with a task (where $\mathbf{P}, \mathbf{Q} \in \mathbf{R}^{n \times d}$, n denotes the number of the individuals in a subpopulation, and d represents the dimensionality of the decision variables), $\mathbf{P}$ is considered as the source data, and $\mathbf{Q}$ is the target data. Following the SA method, PCA is first applied to $\mathbf{P}$ and $\mathbf{Q}$ to construct the subspaces, with which a mapping from $\mathbf{P}$ to $\mathbf{Q}$ can be done via subspace transformation. Let $\mathbf{W}_P$ and $\mathbf{W}_Q$ denote the covariance matrices of $\mathbf{P}$ and $\mathbf{Q}$, respectively, as follows:

$$\mathbf{W}_P = \frac{1}{n}\mathbf{P}^{\mathrm{T}}\mathbf{P} \tag{9}$$

$$\mathbf{W}_Q = \frac{1}{n}\mathbf{Q}^{\mathrm{T}}\mathbf{Q}. \tag{10}$$

The eigenvectors and eigenvalues of $\mathbf{W}_P$ and $\mathbf{W}_Q$ are obtained using eigendecomposition

$$\mathbf{E}_P^{\mathrm{T}}\mathbf{W}_P\mathbf{E}_\mathbf{P} = \Lambda_P = \mathrm{diag}\left(\lambda_1^P, \lambda_2^P, \ldots, \lambda_d^P\right) \tag{11}$$

$$\mathbf{E}_Q^{\mathrm{T}}\mathbf{W}_Q\mathbf{E}_\mathbf{Q} = \Lambda_Q = \mathrm{diag}\left(\lambda_1^Q, \lambda_2^Q, \ldots, \lambda_d^Q\right) \tag{12}$$

where $\mathbf{E}_P$ and $\mathbf{E}_Q$ ($\mathbf{E}_P$, $\mathbf{E}_Q \in \mathbf{R}^{d \times d}$) consist of the set of all eigenvectors of $\mathbf{W}_P$ and $\mathbf{W}_Q$, respectively, with one eigenvector per column. Λ_P and Λ_P are the diagonal matrices of the eigenvalues $\{\lambda_1^P, \ldots, \lambda_d^P\}$ and $\{\lambda_1^Q, \ldots, \lambda_d^Q\}$ of the two covariance matrices, respectively. From $\mathbf{E}_P$ and $\mathbf{E}_Q$, the eigenvectors corresponding to the largest h eigenvalues that can retain 95% of the information are selected to construct the subspaces of $\mathbf{P}$ and $\mathbf{Q}$, that is, $\mathbf{S}_P$ and $\mathbf{S}_Q$ ($\mathbf{S}_P$, $\mathbf{S}_Q \in R^{d \times h}$)

$$\mathbf{S}_P = \mathbf{E}_P^{\mathrm{sort}}(1 : h) \tag{13}$$

Algorithm 3 SA

Input: $\mathbf{P}$ and $\mathbf{Q}$, the two subpopulations of source task and target task, respectively; $\mathbf{x}$, a parent from $\mathbf{P}$.
Output: $\mathbf{x}^*$, the transformed individual from $\mathbf{P}$ to $\mathbf{Q}$;
1: Construct subspace $\mathbf{S}_P$ of $\mathbf{P}$ according to Eq. (13);
2: Construct subspace $\mathbf{S}_Q$ of $\mathbf{Q}$ according to Eq. (14);
3: Calculate $\mathbf{M}^*$ based on Eq. (7);
4: Obtain $\mathbf{x}^*$ based on Eq. (16);

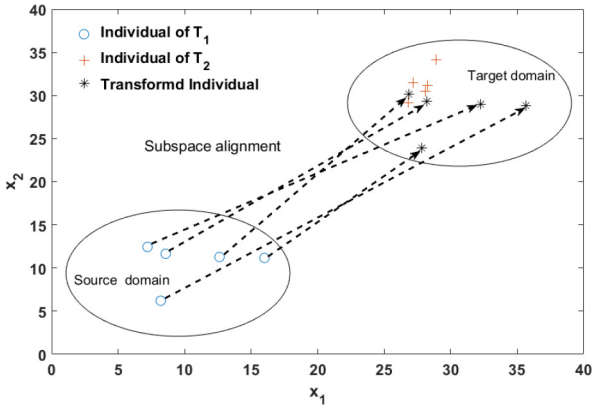


Fig. 2. Example of SA in a 2-D search space.

$$\mathbf{S}_Q = \mathbf{E}_Q^{\mathrm{sort}}(1 : h) \tag{14}$$

where $\mathbf{E}_{P(Q)}^{\mathrm{sort}}(1 : h)$ sorts the eigenvectors in $\mathbf{E}_{P(Q)}$ in the descending order in terms of the eigenvalues and retrieves the top h vectors. Afterward, the transformation matrix $\mathbf{M}^*$ of mapping $\mathbf{S}_P$ to $\mathbf{S}_Q$ is obtained according to (7). As such, $\mathbf{S}_P$ can be aligned to $\mathbf{S}_Q$ as follows:

$$\mathbf{S}_P^* = \mathbf{S}_P \cdot \mathbf{M}^* \tag{15}$$

where $\mathbf{S}_P^*$ represents the new coordinate system that transforms $\mathbf{S}_P$ into $\mathbf{S}_Q$.

As $\mathbf{M}^*$ represents the mapping from the subspace $\mathbf{S}_P$ of $\mathbf{P}$ to the subspace $\mathbf{S}_Q$ of $\mathbf{Q}$, the connection of $\mathbf{P}$ and $\mathbf{Q}$ can be established by (15). Particularly, an individual $\mathbf{x}$ in $\mathbf{P}$ can be mapped to a corresponding individual $\mathbf{x}^*$ in $\mathbf{Q}$, that is

$$\mathbf{x}^* = \mathbf{x} \cdot \mathbf{S}_P \cdot \mathbf{M}^* \cdot \mathbf{S}_Q^{\mathrm{T}} = \mathbf{x} \cdot \mathbf{S}_P^* \cdot \mathbf{S}_Q^{\mathrm{T}} \tag{16}$$

where $\mathbf{x}$ is first projected to the subspace $\mathbf{S}_P$ via $\mathbf{x} \cdot \mathbf{S}_P$, and then converted to the subspace $\mathbf{S}_Q$ via $\mathbf{x} \cdot \mathbf{S}_P^*$. Finally, the mapping is restored to the original space of $\mathbf{Q}$ through $\mathbf{x} \cdot \mathbf{S}_P^* \cdot \mathbf{S}_Q^{\mathrm{T}}$. The pseudocode of SA is provided in Algorithm 3.

As the example shown in Fig. 2, there could be a big difference in a 2-D search space between two tasks, namely, T_1 and T_2. Negative transfer is likely to occur when performing assortative mating and vertical cultural transmission, which results in a deterioration of algorithm performance. After the SA mapping process, the population of T_1 moves closer to the population of T_2 and the impact of negative transfer is weakened.

Note that both SA and autoencoding [28] implement the mapping of individuals in one task to another to counteract negative transfer. The differences between these two methods are two-fold: 1) autoencoding uses a learned and fixed

autoencoder in the process of evolution, whereas SA dynamically updates the mapping matrix during the evolution, which enables SA to capture the subtask population distribution more accurately in the course of the search and 2) the order of the individuals in autoencoding affects the final learned model, whereas SA is independent of the individual order thanks to the use of PCA-based dimensionality reduction. The order of the individual does not affect the calculations of the eigenvalues and eigenvectors in PCA.

C. Offspring Generation Using Adaptive DE

DE algorithm [53] is a simple yet efficient population-based evolutionary search technique. DE algorithms have been successfully applied to solve various complex problems [54], [55]. In each evolution generation of DE, a parent vector (or called target vector) selected from the current generation and a mutant vector (also known as donor vector) obtained through the differential mutation operation are recombined to generate trial vectors under different strategies [56]. The generated trial vectors are further used to generate new offspring with crossover operation. The generation of trial vectors is critical to the performance of DE. Many strategies have been developed to generate a trial vector for different scenarios. Adaptively choosing the best trail vector generation strategy in different evolutionary stages is more desirable.

According to [33], the evolution experience can be beneficial to the current search. To utilize the historical experience and achieve knowledge transfer, this article proposes an adaptive DE strategy based on the framework of SaDE [33]. The evolution experience gained from all tasks is used to adaptively choose the appropriate trial vector generation strategy. Particularly, the trial vector generation strategies with different search behaviors are dynamically selected according to their historical performance, such that promising strategies are more preferable in the course of evolutionary search. The adaptive DE is easy to implement and incorporate with the SA method. The details of the adaptive DE are described as follows.

In each evolution generation, the times the ith trial vector generation strategy generating a nondominated solution is recorded as c_i. The experience period e is introduced to control the generation window within which the evolution experience is obtained. Given k different strategies, a_i records the times the ith trial vector generation strategy generating a nondominated solution between $g-e$ and $g-1$ generation, as defined in (17). A larger a_i indicates the better effectiveness of the ith trial vector generation strategy. Based on the evolution experience, the probability of using each trial vector generation strategy at generation g is defined as (18). It is worth noting that for $g < e$ the probability of each strategy is set to $1/k$

$$a_i = \sum_{g-e}^{g-1} c_i \tag{17}$$

$$\psi_i = \frac{a_i}{\sum_{i=1}^{k} a_i}, \quad \text{subject to } \sum_{i=1}^{k} \psi_i = 1. \tag{18}$$

Theoretically, many trial vector strategies can be applied to the adaptive DE. In this article, the following three representative strategies with different search behaviors are considered.

DE/best/1/bin

$$\mathbf{v}_{j,g} = \mathbf{x}_{\text{best},g} + \alpha * \left(\mathbf{x}_{r_1,g} - \mathbf{x}_{r_2,g}\right). \tag{19}$$

DE/rand/1/bin

$$\mathbf{v}_{j,g} = \mathbf{x}_{r_1,g} + \alpha * \left(\mathbf{x}_{r_2,g} - \mathbf{x}_{r_3,g}\right). \tag{20}$$

DE/current-to-rand/1

$$\mathbf{v}_{j,g} = \mathbf{x}_{j,g} + \beta * \left(\mathbf{x}_{r_1,g} - \mathbf{x}_{j,g}\right) + \alpha * \left(\mathbf{x}_{r_2,g} - \mathbf{x}_{r_3,g}\right) \tag{21}$$

where $\mathbf{v}_{j,g} = \{\mathrm{v}_1, \mathrm{v}_2, \ldots, \mathrm{v}_d\}$ denotes the trial vector generated for each individual $\mathbf{x}_j$ at generation g and d is the dimensionality of the problem encountered. r_1, r_2, and r_3 are mutually exclusive integers and random produced in the range $[1, N]$. N is the population size. $\mathbf{x}_{\text{best}}$ indicates the solution equipped with the best fitness value found so far. α and β are two scaling factors to fluctuate donor vector.

DE/best/1/bin shows strong global search capability and can lead to a better convergence rate, but it is relatively weak in local search [57]. DE/rand/1/bin possesses a slow global convergence speed but good local search ability [53]. DE/current-to-rand/1 is a rotation-invariant strategy competitive to solve MOPs [58]. In the early stage of the evolutionary search, the global search strategy tends to generate more nondominated solutions, so its probability of being selected is larger and premature convergence can be avoided. As the evolutionary search progresses, the local search strategy becomes more preferable. An efficient algorithm should be able to select more favorable strategy at different search stages.

To relieve the impact from negative transfer caused by different search spaces of the tasks, SA is introduced to trial vector strategies resulting in three corresponding modified strategies as follows.

Modified DE/best/1/bin

$$\mathbf{v}_{j,g} = \mathbf{x}_{\text{best},g} + \alpha * \left(\mathbf{x}^*_{r_1,g} - \mathbf{x}^*_{r_2,g}\right). \tag{22}$$

Modified DE/rand/1/bin

$$\mathbf{v}_{j,g} = \mathbf{x}^*_{r_1,g} + \alpha * \left(\mathbf{x}^*_{r_2,g} - \mathbf{x}^*_{r_3,g}\right). \tag{23}$$

Modified DE/current-to-rand/1

$$\mathbf{v}_{j,g} = \mathbf{x}_{j,g} + \beta * \left(\mathbf{x}^*_{r_1,g} - \mathbf{x}_{j,g}\right) + \alpha * \left(\mathbf{x}^*_{r_2,g} - \mathbf{x}^*_{r_3,g}\right) \tag{24}$$

where $\mathbf{x}^*_{r_1,g}$, $\mathbf{x}^*_{r_2,g}$, and $\mathbf{x}^*_{r_3,g}$ are the mappings of $\mathbf{x}_{r_1,g}$, $\mathbf{x}_{r_2,g}$, and $\mathbf{x}_{r_3,g}$ using SA (as described in Algorithm 3), respectively.

The pseudocode of adaptive DE is shown in Algorithm 4. In line 1 of Algorithm 4, a random number *rand* is drawn from [0, 1]. If *rand* is smaller than a predefined threshold *rmp*, the offspring is generated according to lines 2 and 3 via one strategy defined in (22)–(24), that is, SA is applied and knowledge transfer is performed. Otherwise, the offsprings are generated in lines 5 and 6 using one strategy defined in (19)–(21). In this case, the individuals used to produce the trial vector are all from the same task, so the Skill Factors of these individuals are equal to their corresponding parents, that is, no genetic material or knowledge transfer occurs. It is worth noting that if *rand* < *rmp*, the vertical culture transmission proposed in

Algorithm 4 Offspring Generation of MOMFEA-SADE

Input: **P**, the individual population; *DEnum*, the type of the selected strategy.
Output: $\mathbf{u}_{j,g}$, offspring.
1: **if** *rand* < *rmp* **then**
2: Generate a donor vector $\mathbf{v}_{j,g} = \{v_{j,g}^1, ..., v_{j,g}^d\}$ using one of the Eqs. (22)–(24) decided by *DEnum*;
3: Create a trial vector $\mathbf{u}_{j,g} = \{u_{j,g}^1, ..., u_{j,g}^d\}$ via a binomial crossover;
4: **else**
5: Generate a donor vector $\mathbf{v}_{j,g} = \{v_{j,g}^1, ..., v_{j,g}^d\}$ using one of the Eqs. (19)–(21) decided by *DEnum*;
6: Create a trial vector $\mathbf{u}_{j,g} = \{u_{j,g}^1, ..., u_{j,g}^d\}$ via a binomial crossover;
7: **end if**

MO-MFEA is kept to play a role. Differently, the individuals involved in knowledge transfer are mapped to the same search space based on SA, so the Skill Factor of the parent can be directly assigned to the offspring. In many-tasking problems, the knowledge used to generate offspring can come from multiple tasks, and SA is used to impair the impact from negative transfer, which greatly enhances the advantages brought by knowledge transfer.

D. Complexity Analysis

In this section, the computational complexity of an iteration of MOMFEA-SADE is discussed. The computational cost of MOMFEA-SADE is mainly involved in: 1) the computation of mapping matrices among the tasks; 2) the knowledge transfer and offspring generation; and 3) the environmental selection. SA is used to calculate the mapping matrices with a computational complexity of $O(T(T-1)\min(d^3, n^3))$, where the time complexity of PCA dimensionality reduction is $O(\min(d^3, n^3))$ (d is the dimensionality of the decision variables and n is the size of the subpopulation associated with a task) and $T(T-1)$ times of PCA are required to obtain the mapping matrices of T tasks. The time complexity of knowledge transfer and generating offspring is $O(dN)$, where N indicates the population size. In environmental selection, the nondominated sorting and crowding distance account for computational complexities of $O(mN^2)$ and $O(mN\log(N))$ [4], respectively, where m is the number of objectives. Overall, the computational complexity of MOMFEA-SADE is $O(\max(T(T-1)\min(d^3, n^3), mN^2))$. Noted that the computational complexity of MOMFEA-SADE is dominated by the environmental selection in small-scale problems, whereas in large-scale problems, the computational complexity of the algorithm mainly depends on the variable dimension and/or the number of tasks.

IV. Experimental Design and Results

In this section, the performance of the proposed MOMFEA-SADE is verified on various test problems, including the multiobjective multitasking test suite [56], the multiobjective complex two-task benchmark problems [34], and the multiobjective many-tasking (50-tasks) benchmark problems [34]. The effects of the SA and adaptive DE strategies, as well as parameter sensitivity are also investigated in this section.

A. Test Problems and Compared Algorithms

According to the degree of task intersection, the multiobjective multitasking test suite [56] can be divided into three groups [48], that is, complete intersection (CI), partial intersection (PI), and no intersection (NI). Moreover, based on the degree of similarity among the problems, the CI, PI, and NI groups can be further partitioned into high similarity (HS), medium similarity (MS), and low similarity (LS). Therefore, there are nine sets of problems, namely, CIHS, CIMS, CILS, PIHS, PIMS, PILS, NIHS, NIMS, and NILS. Each problem encompasses two multiobjective problems with two or three objectives that have been well studied in the literature. The detailed information of the multiobjective multitasking test suite is provided in Table SI in the supplementary material and more information can be found in [48].

The multiobjective complex two-task benchmark problems, namely, CPLX, and the multiobjective many-task (50-tasks) benchmark problems, called MATP, were introduced in the IEEE CEC 2019 Competition on Evolutionary Multitask Optimization [34]. CPLX contains ten multiobjective multitask benchmark problems of which the details are summarized in Tables SII and SIII in the supplementary material. MATP contains six multiobjective 50-task benchmark problems generated by multiplying the ZDT problems [59] or DTLZ problems [60] by different rotation matrices. The details of MATP are available in Table SIV in the supplementary material.

Three state-of-the-art multiobjective EMT algorithms, that is, MO-MFEA [26], TMO-MFEA [29], and EMT-EGT [28], are used in the comparison with MOMFEA-SADE on multiobjective multitasking problems. MO-MFEA is a basic multiobjective EMT algorithm serving as the baseline. TMO-MFEA adopts different *rmp* values for diversity-related variables and convergence-related variables. EMT-EGT is equipped with multiple solvers (i.e., SPEA2 and NSGA-II) to achieve explicit knowledge via autoencoder. NSGA-II [4] is a single-tasking algorithm serving as another baseline algorithm. NSGA-II is run on each task separately for a multitasking problem subject to the same total computation budget as the other algorithms. On the many-tasking test suite MATP, MOMFEA-SADE is compared with MO-MFEA and the state-of-the-art evolutionary many-tasking algorithm MATGA [45].

B. Parameter Settings

Following [28], SPEA2 is used for one task and NSGA-II is used for the other task in EMT-EGT. The interval of explicit transmission is set to ten according to [28]. The parameters of TMO-MFEA are set according to [29], that is, *rmp* is set to 1 for the diversity variables and 0.3 for the convergence variables. For a fair comparison, in MO-MFEA, EMT-EGT, TMO-MFEA, and MOMFEA-SADE, the scale of the population is set to 200, and the population size for each task in NSGA-II is set to 100. All algorithms are terminated when the maximum number of fitness evaluations 200 000 is reached. The maximum number of fitness evaluations for NSGA-II is, therefore, 100 000 for each task. The other parameter settings of the algorithms are summarized in Table I.

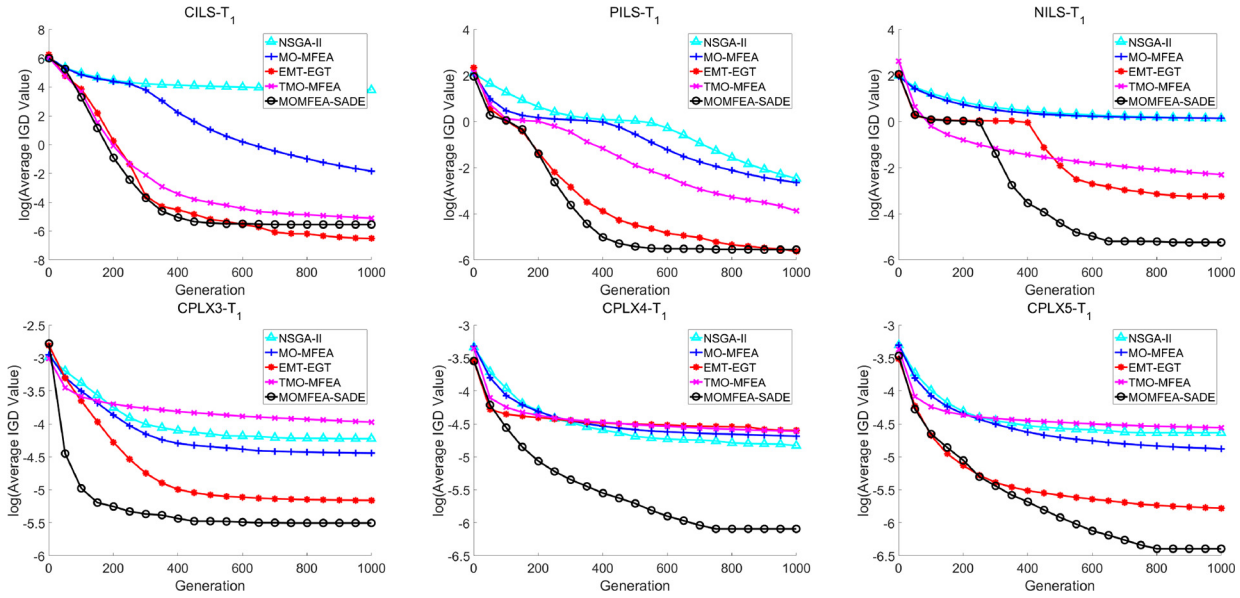


Fig. 3. Average IGD numerical curves of NSGA-II, MO-MFEA, EMT-EGT, TMO-MFEA, and MOMFEA-SADE over 20 independent runs on representative benchmark problems.

TABLE I
PARAMETER SETTING FOR THE MULTITASKING MULTIOBJECTIVE EXPERIMENT

Algorithm	*rmp*	SBX η_c	PM η_m	α	β	N	e	Max FEs
NSGA-II	-	20	20	-	-	200	-	200,000
MOMFEA	0.3	20	20	-	-	200	-	200,000
TMO-MFEA	1 for *DV* and 0.3 for *CV*	20	20	-	-	200	-	200,000
EMT-EGT	-	20	20	-	-	200	-	200,000
MOMFEA-SADE	0.3	-	-	0.6	0.5	200	60	200,000

TABLE II
PARAMETER SETTING FOR THE MANY-TASKING MULTIOBJECTIVE EXPERIMENT

Parameter	value
rmp	0.3
N	5,000
Max FEs	2,500,000
SBX η_c (MO-MFEA and MATGA)	20
PM η_m (MO-MFEA and MATGA)	20
e (MOMFEA-SADE)	60
α (MOMFEA-SADE)	0.6
β (MOMFEA-SADE)	0.5

The parameter settings of MOMFEA-SADE, MO-MFEA, and MATGA in the many-tasking problems MATP are summarized in Table II.

C. Performance Metric

In this section, we used the inverted generational distance (IGD) [61] to measure the performance of the algorithms. IGD is one of the most widely used indicators to evaluate both the exploration and exploitation capabilities of MOEAs

$$\text{IGD} = \frac{\sum_{x \in P^*} \text{dist}(x, A)}{|P^*|} \tag{25}$$

where $\text{dist}(x, A)$ refers to the nearest Euclidean distance between a reference point x and the set of nondominated solutions A generated by the algorithm in the objective space, and P^* represents a predefined set of reference points on the PF. Smaller IGD values indicate better convergence and diversity of the obtained solutions. The calculation of IGD is implemented in [62].

D. Results on the Multitasking Test Problems

The experimental results of NSGA-II, MO-MFEA, EMT-EGT, TMO-MFEA, and MOMFEA-SADE on the two tasks of the multiobjective multitasking test suites [48] are tabulated in Tables III and IV, respectively. The average IGD values over 20 independent runs of the algorithm are reported and the best result on each problem is highlighted in boldface. The Wilcoxon rank-sum test with 95% confidence level is used to check the significant difference of the results. It can be observed that all EMT algorithms perform better than NSGA-II, which demonstrates that knowledge transfer among the tasks can improve the quality of the final solutions. MOMFEA-SADE shows superior performance to the compared algorithms on most of the test problems, thanks to the amelioration of negative transfer endowed by SA and adaptive DE strategies. EMT-EGT obtains best results on T_1 of PIHS, T_1 of PILS, and T_2 of NIMS. EMT-EGT uses multiple solvers (i.e., NSGA-II and SPEA2), which allows it to take full advantage of the different search solvers. The autoencoder mapping the solutions of different search solvers also can reduce the possibility of negative transfer between tasks. MOMFEA-SADE is defeated by TMO-MFEA on T_2

TABLE III
AVERAGED IGD VALUE AND STANDARD DEVIATION OBTAINED BY NSGA-II, MO-MFEA, EMT-EGT, TMO-MFEA, AND MOMFEA-SADE OF TASK_1 ON NINE PROBLEMS

Problem	NSGA-II	MO-MFEA	EMT-EGT	TMO-MFEA	MOMFEA-SADE
CIHS	2.46E+00(4.17E-01)(-)	1.71E-02(2.28E-04)(-)	**1.19E-03(1.22E-04)**(+)	8.06E-03(4.21E-05)(-)	3.89E-03(6.04E-06)
CIMS	4.64E+00(1.45E+00)(-)	1.27E-03(1.30E+00)(-)	5.03E+00(1.46E+00)(-)	4.18E+00(3.50E+02)(-)	**3.26E-04(3.22E-06)**
CILS	4.46E+01(7.34E+00)(-)	3.02E-02(1.08E-02)(-)	**1.38E-03(1.46E-04)**(+)	2.57E-02(5.70E-02)(-)	3.91E-03(8.09E-06)
PIHS	2.28E+00(6.96E-01)(-)	2.62E-01(8.03E-01)(-)	2.04E-02(3.59E-03)(-)	3.54E-03(6.03E-04)(-)	**4.72E-04(5.54E-05)**
PIMS	2.29E+00(3.81E-01)(-)	3.10E-01(1.09E-01)(-)	5.83E-02(1.76E-02)(-)	1.26E+00(3.99E-02)(-)	**2.39E-02(1.77E-03)**
PILS	8.51E-02(1.20E-02)(-)	3.36E-02(7.04E-03)(-)	1.13E-02(3.85E-03)(-)	2.08E-02(4.29E-04)(-)	**1.10E-02(1.59E-04)**
NIHS	6.87E+02(1.88E+02)(-)	5.64E+01(1.21E+03)(-)	4.73E+01(1.71E-01)(-)	4.64E+01(2.12E-01)(-)	**4.46E+01(7.73E-03)**
NIMS	3.94E+01(2.92E+01)(-)	1.72E+01(4.90E-01)(-)	1.64E+01(1.36E-01)(-)	1.75E+01(3.18E-01)(-)	**1.41E+01(1.15E-01)**
NILS	1.16E+00(3.39E-02)(-)	1.09E+00(3.67E-03)(-)	3.37E-02(9.04E-03)(-)	3.92E-02(1.08E-02)(-)	**1.67E-02(4.21E-05)**

+/-/≈ indicates that the statistically performance of the corresponding algorithm is significantly better than, significantly worse than, and comparable to that of MOMFEA-SADE, respectively, at the 5% significance level under Wilcoxon rank-sum test.

TABLE IV
AVERAGED IGD VALUE AND STANDARD DEVIATION OBTAINED BY NSGA-II, MO-MFEA, EMT-EGT, TMO-MFEA, AND MOMFEA-SADE OF TASK_2 ON NINE PROBLEMS

Problem	NSGA-II	MO-MFEA	EMT-EGT	TMO-MFEA	MOMFEA-SADE
CIHS	1.03E+00(1.54E-01)(-)	6.83E-01(1.61E-03)(-)	1.52E-03(6.39E-04)(-)	9.19E-02(1.84E-02)(-)	**5.04E-04(3.16E-05)**
CIMS	1.13E+00(1.04E+00)(-)	1.48E-01(8.66E-03)(-)	3.42E-01(2.32E-01)(-)	5.12E-03(5.50E-02)(-)	**8.25E-04(1.08E-05)**
CILS	1.50E-01(2.57E-01)(-)	7.62E-03(5.82E-05)(-)	1.82E-03(2.03E-04)(-)	2.35E-03(2.59E-04)(-)	**5.01E-04(2.58E-06)**
PIHS	8.62E+01(2.75E+01)(-)	9.25E+01(1.73E-01)(-)	5.77E-01(2.54E-01)(-)	**1.85E-01(6.32E-02)**(+)	2.22E-01(6.89E-02)
PIMS	1.48E+03(5.33E+02)(-)	1.33E+03(5.38E+00)(-)	5.66E+01(6.49E+00)(-)	5.06E+02(3.24E+02)(-)	**1.52E+01(6.50E+0)**
PILS	2.11E+01(3.38E-02)(-)	5.23E+00(1.25E-02)(-)	1.80E+00(1.51E-02)(-)	2.23E+00(2.34E-01)(-)	**1.72E+00(1.44E-01)**
NIHS	1.26E+00(3.73E-01)(-)	6.41E-01(1.26E-03)(-)	1.52E-03(5.08E-04)(-)	5.65E-02(3.46E+02)(-)	**4.68E-04(7.11E-06)**
NIMS	2.08E+00(3.94E+00)(-)	1.30E-01(1.04E-02)(-)	**5.22E-04(1.36E-04)**(+)	3.71E-03(3.30E-02)(-)	1.53E-03(9.14E-04)
NILS	2.00E+01(8.42E-02)(-)	2.01E+01(1.41E-03)(-)	2.00E+01(8.93E+00)(-)	2.10E+01(1.43E+00)(-)	**1.92E+01(1.79E-02)**

+/-/≈ indicates that the statistically performance of the corresponding algorithm is significantly better than, significantly worse than, and comparable to that of MOMFEA-SADE, respectively, at the 5% significance level under Wilcoxon rank-sum test.

TABLE V
AVERAGED IGD VALUE AND STANDARD DEVIATION OBTAINED BY NSGA-II, MO-MFEA, EMT-EGT, TMO-MFEA, AND MOMFEA-SADE OF TASK_1 ON CPLX1–CPLX10

problem	NSGA-II	MO-MFEA	EMT-EGT	TMO-MFEA	MOMFEA-SADE
CPLX1	3.44E-04(4.09E-05)(-)	3.36E-04(4.97E-05)(-)	**2.69E-04(4.75E-05)**(+)	1.57E-03(1.63E-03)(-)	2.89E-04(3.78E-05)
CPLX2	3.45E-04(4.53E-05)(-)	2.69E-04(4.66E-05)(-)	**2.58E-04(2.43E-05)**(+)	1.73E-03(1.12E-03)(-)	2.79E-04(5.41E-05)
CPLX3	4.16E-03(2.18E-03)(-)	5.35E-03(2.10E-03)(-)	3.74E-03(2.45E-03)(-)	9.35E-03(3.77E-03)(-)	**1.87E-03(1.78E-03)**
CPLX4	5.77E-03(2.55E-03)(-)	4.34E-03(1.69E-03)(-)	7.90E-03(4.82E-03)(-)	1.02E-02(2.61E-03)(-)	**1.86E-03(1.53E-03)**
CPLX5	2.30E-03(2.27E-03)(-)	2.35E-03(1.82E-03)(-)	6.26E-03(3.72E-03)(-)	2.40E-03(1.29E-03)(-)	**2.02E-03(5.62E-04)**
CPLX6	3.32E-03(1.68E-03)(-)	1.97E-03(7.02E-04)(-)	4.36E-03(3.82E-03)(-)	5.91E-03(1.29E-03)(-)	**2.03E-03(6.38E-04)**
CPLX7	1.84E-03(4.28E-04)(-)	1.81E-03(3.99E-04)(-)	2.24E-03(4.36E-04)(-)	2.21E-03(1.01E-03)(-)	**1.34E-03(2.37E-04)**
CPLX8	4.21E-03(1.19E-03)(-)	2.54E-03(1.75E-03)(-)	5.26E-03(1.97E-03)(-)	1.87E-03(1.24E-03)(-)	**9.23E-04(7.02E-04)**
CPLX9	7.53E-03(2.44E-03)(-)	9.33E-03(1.40E-03)(-)	**3.23E-03(2.24E-04)**(+)	1.40E-02(3.95E-03)(-)	4.08E-03(2.70E-04)
CPLX10	1.86E-02(5.48E-03)(-)	1.29E-02(2.65E-03)(-)	1.59E-02(8.46E-03)(-)	1.67E-02(2.27E-03)(-)	**3.05E-03(3.73E-03)**

+/-/≈ indicates that the statistically performance of the corresponding algorithm is significantly better than, significantly worse than, and comparable to that of MOMFEA-SADE, respectively, at the 5% significance level under Wilcoxon rank-sum test.

of PIHS. The main reason is that the PIHS problem has high intertask similarity, and the occurrence possibility of negative transfer is small. The introduced new strategies might not show an advantage in such cases. Moreover, TMO-MFEA uses different *rmp* values for convergency variables and diversity variables, which could lead to better convergence and uniform diversity.

The average IGD values and variance of MO-MFEA, EMT-EGT, TMO-MFEA, and MOMFEA-SADE on CPLX problems over 20 independent runs are summarized in Tables V and VI. The Wilcoxon rank-sum test with a 95% confidence level is also conducted and the best results are highlighted in boldface on each problem. MOMFEA-SADE again achieves the best performance on most CPLX problems thanks to the introduction of SA and adaptive DE strategies. EMT-EGT uses SPEA2 to generate promising solutions and share these solutions with another task through the autoencoder. It outperforms other algorithms on CPLX9 and T_1 of CPLX1 and CPLX2.

In MOMFEA-SADE, SA is applied in both assorting mating and vertical cultural communication to map an individual in the search space of one task to another, which reduces the negative migration caused by knowledge exchange between different tasks with lower similarity and helps the offspring of one task to better adapting the search space of another task. Moreover, the adaptive DE enables more reasonable choice of the trial vector generation strategy. The two mechanisms work well together and lead to the superior performance of MOMFEA-SADE.

TABLE VI
AVERAGED IGD VALUE AND STANDARD DEVIATION OBTAINED BY NSGA-II, MO-MFEA, EMT-EGT, TMO-MFEA, AND MOMFEA-SADE OF TASK_2 ON CPLX1–CPLX10

problem	NSGA-II	MO-MFEA	EMT-EGT	TMO-MFEA	MOMFEA-SADE
CPLX1	3.01E-03(1.57E-03)(-)	2.26E-03(1.02E-03)(-)	4.52E-03(2.83E-03)(-)	3.34E-03(6.42E-04)(-)	**1.16E-03(1.23E-03)**
CPLX2	7.54E-03(4.75E-03)(-)	7.29E-04(3.29E-04)(-)	1.12E-02(6.86E-03)(-)	5.69E-03(3.58E-03)(-)	**6.72E-04(2.89E-04)**
CPLX3	1.86E-03(5.15E-04)(-)	1.75E-03(3.39E-04)(-)	1.51E-03(3.78E-04)(-)	2.39E-03(9.28E-04)(-)	**1.43E-03(4.22E-04)**
CPLX4	5.44E-03(2.32E-03)(-)	4.72E-03(1.63E-03)(-)	8.28E-03(3.42E-03)(-)	1.01E-02(1.34E-03)(-)	**2.14E-03(1.62E-03)**
CPLX5	7.87E-03(2.13E-03)(-)	9.20E-03(2.24E-03)(-)	1.82E-02(9.26E-04)(-)	1.17E-02(5.50E-04)(-)	**3.87E-03(2.37E-04)**
CPLX6	4.36E-03(2.63E-03)(-)	5.35E-03(2.00E-03)(-)	5.32E-03(2.42E-03)(-)	1.12E-02(3.94E-03)(-)	**3.62E-03(1.55E-03)**
CPLX7	5.54E-03(1.17E-03)(-)	2.12E-03(9.80E-04)(-)	2.00E-03(2.83E-03)(-)	1.75E-03(1.08E-03)(-)	**1.38E-03(2.68E-04)**
CPLX8	4.85E-02(5.66E-03)(-)	1.11E-02(1.52E-03)(-)	7.47E-03(1.26E-03)(-)	1.32E-02(1.73E-03)(-)	**1.93E-03(4.79E-03)**
CPLX9	4.52E-03(2.28E-03)(-)	5.44E-03(2.49E-03)(-)	4.44E-03(1.03E-03)(-)	1.09E-02(6.15E-04)(-)	**4.04E-03(9.84E-04)**
CPLX10	1.17E-02(2.20E-03)(-)	**1.93E-03(2.08E-03)**(≈)	2.29E-03(4.32E-04)(-)	1.18E-02(2.82E-03)(-)	1.94E-03(2.21E-03)

+/-/≈ indicates that the statistically performance of the corresponding algorithm is significantly better than, significantly worse than, and comparable to that of MOMFEA-SADE, respectively, at the 5% significance level under Wilcoxon rank-sum test.

TABLE VII
AVERAGED IGD VALUE OBTAINED BY MATGA, MOMFEA-SADE OF TASK_1–TASK_{50} ON MATP1–MATP6 PROBLEMS AFTER 20 INDEPENDENT RUNS, WHERE THE BOLD IS THE BEST RESULT OF MATGA AND MOMFEA-SADE

	MATGA						MOMFEA-SADE					
task	MATP1	MATP2	MATP3	MATP4	MATP5	MATP6	MATP1	MATP2	MATP3	MATP4	MATP5	MATP6
T_1	3.47E-02	2.69E-02	1.56E-01	1.44E+00	7.23E-04	3.35E-03	**6.65E-03**	**7.18E-04**	**9.09E-02**	**1.22E+00**	**2.83E-04**	**1.26E-03**
T_2	3.20E-02	2.72E-02	1.81E-01	1.16E+00	7.16E-04	3.52E-03	**4.71E-03**	**7.07E-04**	**9.06E-02**	**1.13E+00**	**2.79E-04**	**9.25E-04**
T_3	3.17E-02	2.67E-02	1.73E-01	1.30E+00	7.23E-04	3.69E-03	**5.24E-03**	**6.83E-04**	**8.76E-02**	**1.13E+00**	**2.79E-04**	**1.01E-03**
T_4	3.20E-02	2.87E-02	1.81E-01	1.12E+00	7.26E-04	3.58E-03	**4.96E-03**	**7.05E-04**	**9.33E-02**	**1.12E+00**	**2.81E-04**	**1.01E-03**
T_5	3.13E-02	2.94E-02	1.82E-01	1.33E+00	7.32E-04	3.55E-03	**5.92E-03**	**7.02E-04**	**9.22E-02**	**1.15E+00**	**2.91E-04**	**1.16E-03**
T_6	2.93E-02	2.87E-02	1.98E-01	**1.08E+00**	7.21E-04	3.41E-03	**5.74E-03**	**7.08E-04**	**9.35E-02**	1.15E+00	**2.80E-04**	**1.41E-03**
T_7	2.94E-02	2.88E-02	1.80E-01	1.18E+00	7.32E-04	3.30E-03	**5.57E-03**	**7.06E-04**	**9.30E-02**	**1.17E+00**	**2.83E-04**	**1.17E-03**
T_8	3.38E-02	2.82E-02	1.88E-01	1.34E+00	7.13E-04	3.38E-03	**5.81E-03**	**7.15E-04**	**9.19E-02**	**1.17E+00**	**2.80E-04**	**1.34E-03**
T_9	2.94E-02	2.95E-02	1.64E-01	1.58E+00	7.19E-04	3.58E-03	**5.45E-03**	**7.04E-04**	**9.07E-02**	**1.14E+00**	**2.76E-04**	**1.12E-03**
T_{10}	3.29E-02	2.87E-02	1.79E-01	1.26E+00	7.43E-04	3.50E-03	**5.53E-03**	**7.37E-04**	**8.86E-02**	**1.16E+00**	**2.79E-04**	**1.19E-03**
T_{11}	2.98E-02	3.09E-02	1.75E-01	1.26E+00	7.29E-04	3.26E-03	**5.80E-03**	**6.08E-04**	**8.65E-02**	**1.15E+00**	**2.89E-04**	**1.59E-03**
T_{12}	3.27E-02	2.76E-02	1.77E-01	1.41E+00	7.34E-04	3.47E-03	**3.87E-03**	**6.87E-04**	**9.16E-02**	**1.10E+00**	**2.84E-04**	**8.79E-04**
T_{13}	3.16E-02	2.72E-02	2.08E-01	1.11E+00	7.07E-04	3.35E-03	**4.10E-03**	**7.02E-04**	**9.23E-02**	**1.08E+00**	**2.94E-04**	**8.65E-04**
T_{14}	3.05E-02	2.74E-02	1.78E-01	1.48E+00	7.23E-04	3.48E-03	**3.81E-03**	**6.52E-04**	**9.23E-02**	**1.07E+00**	**2.88E-04**	**8.63E-04**
T_{15}	3.29E-02	2.92E-02	1.86E-01	1.30E+00	7.21E-04	3.70E-03	**4.10E-03**	**6.48E-04**	**9.19E-02**	**1.10E+00**	**2.82E-04**	**9.32E-04**
T_{16}	2.97E-02	2.80E-02	1.40E-01	1.15E+00	7.27E-04	3.31E-03	**4.05E-03**	**6.69E-04**	**9.29E-02**	**1.10E+00**	**2.78E-04**	**9.83E-04**
T_{17}	3.22E-02	2.87E-02	1.82E-01	1.40E+00	7.11E-04	3.17E-03	**4.61E-03**	**6.93E-04**	**9.23E-02**	**1.09E+00**	**2.79E-04**	**9.07E-04**
T_{18}	3.49E-02	2.95E-02	1.69E-01	1.20E+00	7.33E-04	3.79E-03	**4.07E-03**	**6.52E-04**	**9.29E-02**	**1.10E+00**	**2.84E-04**	**9.49E-04**
T_{19}	3.19E-02	2.84E-02	1.91E-01	1.20E+00	7.21E-04	3.33E-03	**4.50E-03**	**6.54E-04**	**9.34E-02**	**1.07E+00**	**2.76E-04**	**1.08E-03**
T_{20}	2.96E-02	3.04E-02	1.84E-01	1.27E+00	7.43E-04	3.68E-03	**4.55E-03**	**6.80E-04**	**9.27E-02**	**1.11E+00**	**2.91E-04**	**1.02E-03**
T_{21}	2.81E-02	2.87E-02	1.58E-01	1.38E+00	7.22E-04	3.30E-03	**6.33E-03**	**7.22E-04**	**8.61E-02**	**1.13E+00**	**2.57E-04**	**1.39E-03**
T_{22}	3.52E-02	2.73E-02	1.67E-01	1.12E+00	7.03E-04	3.37E-03	**4.15E-03**	**6.73E-04**	**8.74E-02**	**1.10E+00**	**2.73E-04**	**9.82E-04**
T_{23}	3.26E-02	2.86E-02	1.18E-01	**1.05E+00**	7.28E-04	3.47E-03	**4.28E-03**	**6.72E-04**	**8.99E-02**	1.08E+00	**2.75E-04**	**1.01E-03**
T_{24}	3.13E-02	2.84E-02	1.69E-01	1.56E+00	7.43E-04	3.19E-03	**4.39E-03**	**6.41E-04**	**8.26E-02**	**1.10E+00**	**2.87E-04**	**8.99E-04**
T_{25}	3.37E-02	2.91E-02	1.82E-01	1.68E+00	7.00E-04	3.42E-03	**3.98E-03**	**6.51E-04**	**8.98E-02**	**1.11E+00**	**2.81E-04**	**1.11E-03**
T_{26}	3.17E-02	2.73E-02	2.16E-01	1.13E+00	7.33E-04	3.40E-03	**5.18E-03**	**6.65E-04**	**8.98E-02**	**1.07E+00**	**2.85E-04**	**8.22E-04**
T_{27}	3.11E-02	3.05E-02	1.77E-01	1.30E+00	7.08E-04	3.43E-03	**5.00E-03**	**6.61E-04**	**9.03E-02**	**1.09E+00**	**2.85E-04**	**8.02E-04**
T_{28}	3.25E-02	2.73E-02	2.03E-01	1.59E+00	7.47E-04	3.61E-03	**4.78E-03**	**6.43E-04**	**8.61E-02**	**1.07E+00**	**2.76E-04**	**9.11E-04**
T_{29}	3.29E-02	2.94E-02	2.18E-01	**1.03E+00**	7.37E-04	3.45E-03	**5.37E-03**	**6.88E-04**	**8.76E-02**	1.09E+00	**2.76E-04**	**1.05E-03**
T_{30}	3.14E-02	2.84E-02	1.71E-01	1.28E+00	7.16E-04	3.14E-03	**5.35E-03**	**6.63E-04**	**8.89E-02**	**1.12E+00**	**2.74E-04**	**1.06E-03**
T_{31}	3.23E-02	2.77E-02	1.41E-01	1.36E+00	7.05E-04	3.40E-03	**4.87E-03**	**7.24E-04**	**9.20E-02**	**1.18E+00**	**2.70E-04**	**1.21E-03**
T_{32}	3.03E-02	2.88E-02	1.76E-01	**1.03E+00**	7.02E-04	3.29E-03	**4.11E-03**	**6.73E-04**	**9.29E-02**	1.06E+00	**2.77E-04**	**9.52E-04**
T_{33}	3.13E-02	2.92E-02	1.75E-01	1.48E+00	7.37E-04	3.54E-03	**4.80E-03**	**6.90E-04**	**9.30E-02**	**1.06E+00**	**2.84E-04**	**7.63E-04**
T_{34}	3.38E-02	2.98E-02	1.82E-01	1.57E+00	7.24E-04	3.44E-03	**4.75E-03**	**7.03E-04**	**9.18E-02**	**1.09E+00**	**2.89E-04**	**1.02E-03**
T_{35}	3.65E-02	2.59E-02	1.76E-01	1.58E+00	7.25E-04	3.44E-03	**4.30E-03**	**6.56E-04**	9.52E-02	**1.08E+00**	**2.71E-04**	**8.73E-04**
T_{36}	3.34E-02	3.00E-02	1.52E-01	1.64E+00	7.32E-04	3.32E-03	**4.47E-03**	**6.45E-04**	**9.39E-02**	**1.05E+00**	**2.87E-04**	**9.14E-04**
T_{37}	3.39E-02	2.81E-02	1.63E-01	1.36E+00	7.24E-04	3.46E-03	**4.69E-03**	**6.95E-04**	**9.41E-02**	**1.09E+00**	**2.89E-04**	**8.79E-04**
T_{38}	3.14E-02	2.99E-02	1.29E-01	1.28E+00	7.32E-04	3.37E-03	**5.09E-03**	**6.89E-04**	**9.50E-02**	**1.09E+00**	**2.93E-04**	**8.58E-04**
T_{39}	3.39E-02	2.86E-02	1.63E-01	1.23E+00	7.09E-04	3.42E-03	**4.89E-03**	**6.84E-04**	**9.35E-02**	**1.08E+00**	**2.88E-04**	**1.07E-03**
T_{40}	3.32E-02	2.91E-02	1.48E-01	1.23E+00	7.14E-04	3.42E-03	**5.08E-03**	**6.93E-04**	**9.56E-02**	**1.08E+00**	**2.76E-04**	**8.79E-04**
T_{41}	3.04E-02	2.89E-02	2.02E-01	1.31E+00	7.26E-04	3.28E-03	**6.25E-03**	**7.73E-04**	**9.22E-02**	**1.15E+00**	**2.83E-04**	**1.52E-03**
T_{42}	3.29E-02	2.84E-02	2.45E-01	1.18E+00	7.36E-04	3.70E-03	**4.02E-03**	**6.78E-04**	**9.20E-02**	**1.09E+00**	**2.82E-04**	**9.97E-04**
T_{43}	3.18E-02	3.16E-02	1.64E-01	1.29E+00	7.26E-04	3.23E-03	**4.29E-03**	**6.81E-04**	**9.33E-02**	**1.11E+00**	**2.80E-04**	**9.94E-04**
T_{44}	3.37E-02	2.87E-02	1.99E-01	1.18E+00	7.28E-04	3.23E-03	**4.04E-03**	**6.96E-04**	**8.98E-02**	**1.11E+00**	**2.89E-04**	**1.14E-03**
T_{45}	3.44E-02	2.76E-02	1.52E-01	1.31E+00	7.20E-04	3.46E-03	**4.93E-03**	**6.98E-04**	**9.42E-02**	**1.11E+00**	**2.83E-04**	**1.08E-03**
T_{46}	3.14E-02	2.67E-02	1.76E-01	1.50E+00	7.10E-04	3.63E-03	**4.73E-03**	**6.41E-04**	**9.15E-02**	**1.08E+00**	**2.85E-04**	**1.24E-03**
T_{47}	3.23E-02	2.87E-02	1.92E-01	1.15E+00	7.28E-04	3.56E-03	**4.51E-03**	**6.86E-04**	**9.26E-02**	**1.11E+00**	**2.89E-04**	**1.20E-03**
T_{48}	3.32E-02	2.74E-02	1.61E-01	1.10E+00	7.37E-04	3.38E-03	**4.88E-03**	**6.73E-04**	**9.39E-02**	**1.10E+00**	**2.80E-04**	**1.19E-03**
T_{49}	3.04E-02	2.63E-02	1.59E-01	1.20E+00	7.36E-04	3.41E-03	**5.06E-03**	**6.92E-04**	**9.14E-02**	**1.09E+00**	**2.79E-04**	**1.22E-03**
T_{50}	2.97E-02	2.95E-02	1.64E-01	1.43E+00	7.03E-04	3.68E-03	**5.00E-03**	**7.07E-04**	**9.02E-02**	**1.07E+00**	**2.88E-04**	**1.14E-03**

Fig. 3 shows the IGD curves of the algorithm on the representative problems, that is, CILS-T_1, PILS-T_2, and NILS-T_1, and CPLX3-T_1, CPLX4-T_1, and CPLX5-T_1. It can be observed that MOMFEA-SADE converges fast in the early stages, which is attributed to the fact that MOMFEA-SADE uses SA to reduce the impact of negative transfer and selects

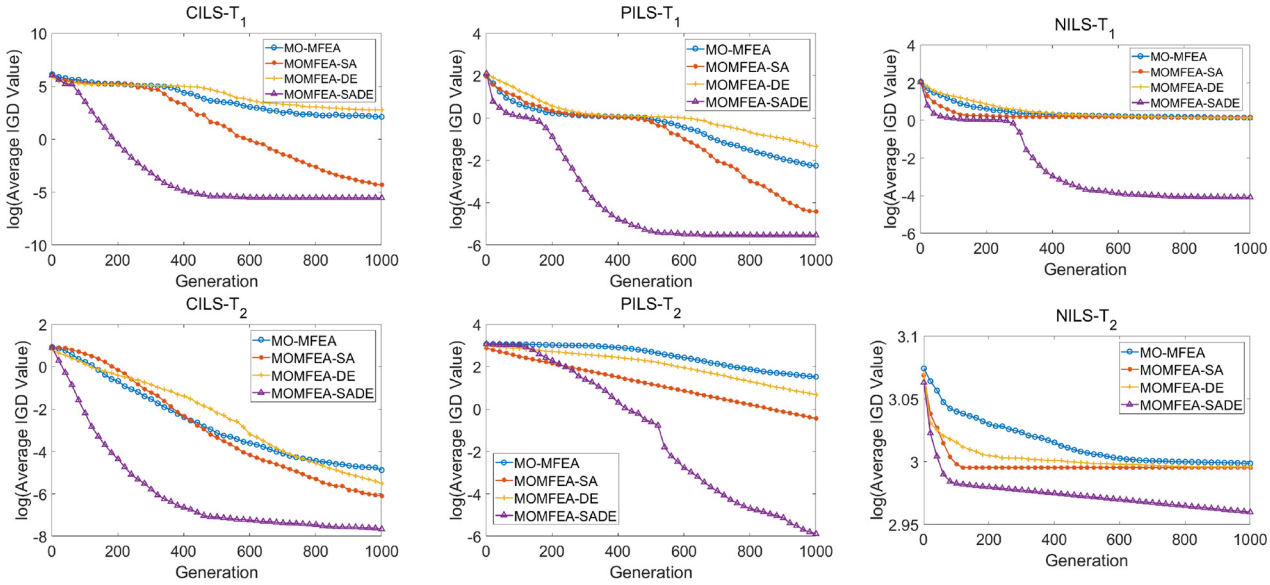


Fig. 4. Average IGD numerical curves of NSGA-II, MO-MFEA, EMT-EGT, TMO-MFEA, and MOMFEA-SADE over 20 independent runs on representative benchmark problems.

the global trial vector generation strategy. In the late stages, when the solutions are basically nondominated, the diversity of the population can be enhanced by using the trial vector generation strategy. Due to the page limit, the average IGD numerical curves of all algorithms on multiobjective multitasking test suite and the multiobjective complex two-task problems are provided in Figs. SI–SIV in the supplementary material.

E. Results on the Many-Tasking Test Problems

The experimental results of MATGA and MOMFEA-SADE on MATP test suite are presented in Table VII, where the best result on each problem is emphasized in boldface and the Wilcoxon rank-sum test with 95% confidence level is also conducted to test significant difference between algorithms. The KL divergence is used to measure the similarity between the tasks in MATGA, which selects the tasks with the highest similarity for knowledge transfer. MOMFEA-SADE uses SA to map an individual of one task to another task. Both MATGA and MOMFEA-SADE can reduce the possibility of negative transfer. On top of that, MOMFEA-SADE learns a mass of experience generated by the many-tasking evolution process and chooses an appropriate trial vector generation strategy for each task when generating offspring. As can be seen in Table VII, MOMFEA-SADE obtains better performance than MATGA on most of the MATP problems. Due to the page limit, the comparison results of MO-MFEA and MOMFEA-SADE on the MATP test suite are presented in Table SIX in the supplementary material, where MOMFEA-SADE also achieves better performance than MO-MFEA. The reason is that MO-MFEA can only transfer knowledge between two tasks and there is no restriction on negative transfer, whereas MOMFEA-SADE converts the population of different tasks into the same problem domain through SA, which enables more efficient knowledge transfer. Moreover, the adaptive DE selects the trial vector generation strategy more suitable for the evolutionary process, which speeds up the convergence in the early stage and improves the diversity in the later stage. The average IGD numerical curves of MO-MFEA and MOMFEA-SADE on MATP test suite can be found in Figs. SV–SVIII in the supplementary material.

F. Effects of the SA and Adaptive DE Strategies

To explore the influence of the SA and adaptive DE strategies, this section investigates the algorithm performance using the two strategies separately on CILS, PILS, and NILS problems. Since the three problems are featured by low task similarity, the negative transfer is more likely to occur in these problems. CILS, PILS, and NILS serve as good test problems to investigate the effects of the SA and adaptive DE strategies. Let MOMFEA-SA denote the algorithm using only SA and MOMFEA-DE indicates the algorithm with only the adaptive DE. MOMEFA-SADE is compared with MO-MFEA (i.e., without using SA or adaptive DE), MOMFEA-SA, and MOMFEA-DE on the three problems. The average IGD values of the compared algorithms over 20 independent runs on the three problems are shown in Fig. 4. The results of MOMFEA, MOMFEA-SA, MOMFEA-DE, and MOMFEA-SADE on CIHS, CIMS, PIHS, PIMS, NIHS, and NIMS problems are provided in Figs. SX–SXV in the supplementary material. As can be seen in the figures, MOMFEA-SA and MOMFEA-SADE with the SA strategy achieve better results than the other two algorithms, especially in the three LS problems, where a negative transfer is well handled by SA. MOMFEA and MOMFEA-DE without the SA strategy fail to relieve the impact of negative transfer and thus result in the deteriorated performance.

MOMFEA-SA obtains better performance than MO-MFEA as SA can reduce the impact of negative transfer. MOMFEA-DE outperforms MO-MFEA because of the use of evolutionary experience to dynamically choose the suitable strategy to generate the trial vector. MOMFEA-SADE takes

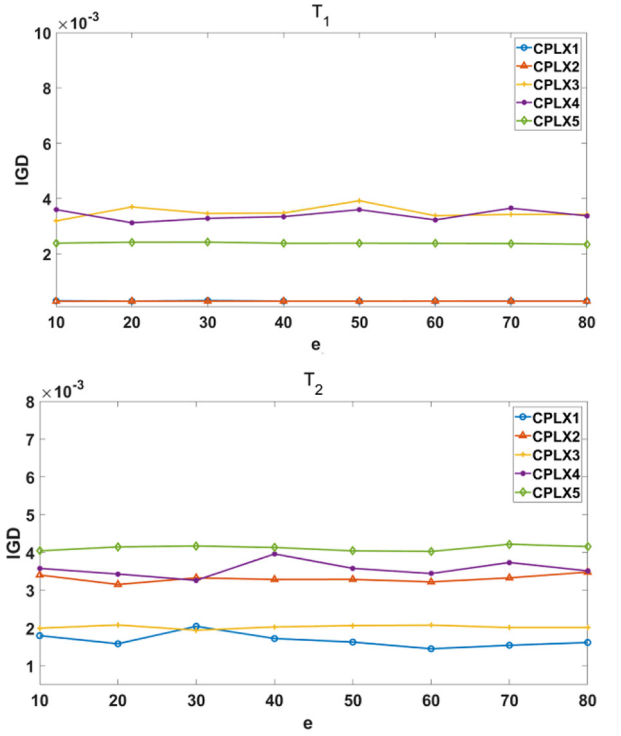


Fig. 5. Average IGD numerical curves of MOMFEA-SADE with different e values over 20 independent runs on CPLX1, CPLX2, CPLX3, CPLX4, and CPLX5.

the advantage from both SA and adaptive DE strategies and therefore achieving a better result than the counterpart compared algorithms. The effects of using different trial vector generation strategies on the performance of the algorithm are investigated in Tables SV–SVIII in the supplementary material where the adaptive strategy is shown to be the best option.

G. Parameters Sensitivity Analysis

The experience period e is the key parameter to MOMFEA-SADE. To verify the robustness of the proposed algorithm, the representative problems CPLX1, CPLX2, CPLX3, CPLX4, and CPLT5, which cover the cases of HS, MS, and LS, are selected for the parameter sensitivity analysis. MOMFEA-SADE is run with different e values on the select test problems. Particularly, the experiments are carried out on the representative test problems with $e = 10$, 20, 30, 40, 50, 60, 70, and 80. Fig. 5 shows the average IGD numerical curves obtained by MOMFEA-SADE with different e values over 20 runs on CPLX1, CPLX2, CPLX3, CPLX4, and CPLX5 problems. It can be observed that the final results of MOMFEA-SADE do not show a significant difference on each problem with different e values, that is, MOMFEA-SADE, is not very sensitive to the parameter e. According to the results shown in Fig. 5, MOMFEA-SADE shows slightly more preferable performance when $e = 60$, which is also adopted in the previous experiments.

V. Conclusion

This article proposes a new EMT algorithm for multiobjective optimization, namely, MOMFEA-SADE. The main differences between MOMFEA-SADE and the existing multiobjective EMT algorithms lie in the novel SA strategy and adaptive DE strategy. The SA strategy establishes the connection between two tasks using two transforming matrices, which achieves search space mapping and reduces the probability of negative transfer. The adaptive DE selects the suitable trial vector generation strategy automatically by recording the ratio of the trial vector generation strategy to produce a nondominated solution in all tasks. The SA is embedded in the adaptive DE to improve the communication efficiency among tasks in many tasking. The effectiveness of MOMFEA-SADE is empirically assessed and compared with the other state-of-the-art multiobjective EMT algorithms on two multitasking test suites and one many-tasking test suite. The experimental results demonstrate that MOMFEA-SADE is superior or comparable to the compared algorithms. SA and adaptive DE strategies work well together to improve the efficiency of MOMFEA-SADE. Nevertheless, there still are a few issues could be considered in future work. For example, SA based on PCA could be time consuming in large-scale problems. More efforts should be put into finding more computationally efficient methods for SA. In the current adaptive DE, the trial vector generation strategies are evaluated across all tasks. However, different tasks might prefer different trial vector generation strategies. Maintaining a probability vector Ψ for each task could be a promising direction to improve the algorithm performance. The source code of MOMFEA-SADE written in MATLAB is provided at https://github.com/CIA-SZU/DH.

References

[1] J. Wang, Y. Zhou, Y. Wang, J. Zhang, C. L. P. Chen, and Z. Zheng, "Multiobjective vehicle routing problems with simultaneous delivery and pickup and time windows: Formulation, instances, and algorithms," *IEEE Trans. Cybern.*, vol. 46, no. 3, pp. 582–594, Mar. 2016.

[2] F. Sarro, F. Ferrucci, M. Harman, A. Manna, and J. Ren, "Adaptive multi-objective evolutionary algorithms for overtime planning in software projects," *IEEE Trans. Softw. Eng.*, vol. 43, no. 10, pp. 898–917, Oct. 2017.

[3] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. N. Suganthan, and Q. Zhangd, "Multiobjective evolutionary algorithms: A survey of the state of the art," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 32–49, 2011.

[4] D. Kalyanmoy, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[5] V. Palakonda and R. Mallipeddi, "Pareto dominance-based algorithms with ranking methods for many-objective optimization," *IEEE Access*, vol. 5, pp. 11043–11053, 2017.

[6] Y. Tian, R. Cheng, X. Zhang, Y. Su, and Y. Jin, "A strengthened dominance relation considering convergence and diversity for evolutionary many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 2, pp. 331–345, Apr. 2019.

[7] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.

[8] Y. Qi, X. Ma, F. Liu, L. Jiao, J. Sun, and J. Wu, "MOEA/D with adaptive weight adjustment," *Evol. Comput.*, vol. 22, no. 2, pp. 231–264, Jun. 2014.

[9] H.-L. Liu, F. Gu, and Q. Zhang, "Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 3, pp. 450–455, Jun. 2014.

[10] R. Wang, Q. Zhang, and T. Zhang, "Pareto adaptive scalarising functions for decomposition based algorithms," in *Proc. Int. Conf. Evol. Multi Crit. Opt.*, 2015, pp. 248–262.

[11] M. Wu, K. Li, S. Kwong, Y. Zhou, and Q. Zhang, "Matching-based selection with incomplete lists for decomposition multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 4, pp. 554–568, Aug. 2017.

[12] R. Wang, Z. Zhou, H. Ishibuchi, T. Liao, and T. Zhang, "Localized weighted sum method for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 3–18, Feb. 2018.

[13] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, 2004, pp. 832–842.

[14] J. Bader and E. Zitzler, "HypE: An algorithm for fast hypervolume-based many-objective optimization," *Evol. Comput.*, vol. 19, no. 1, pp. 45–76, 2011.

[15] W. Hong, K. Tang, A. Zhou, H. Ishibuchi, and X. Yao, "A scalable indicator-based evolutionary algorithm for large-scale multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 3, pp. 525–537, Jun. 2019.

[16] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: Toward evolutionary multitasking," *IEEE Trans. Evol. Comput.*, vol. 20, no. 3, pp. 343–357, Jun. 2016.

[17] J. Mo, Z. Fan, W. Li, Y. Fang, Y. You, and X. Cai, "Multi-factorial evolutionary algorithm based on M2M decomposition," in *Proc. Simul. Evol. Learn.*, 2017, pp. 134–144.

[18] C. Yang, J. Ding, Y. Jin, C. Wang, and T. Chai, "Multitasking multiobjective evolutionary operational indices optimization of beneficiation processes," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 3, pp. 1046–1057, Jul. 2019.

[19] R. Hashimoto, H. Ishibuchi, N. Masuyama, and Y. Nojima, "Analysis of evolutionary multi-tasking as an island model," in *Proc. Genet. Evol. Comput. Conf. Companion*, Jul. 2018, pp. 1894–1897.

[20] J. Zhong, L. Feng, W. Cai, and Y.-S. Ong, "Multifactorial genetic programming for symbolic regression problems," *IEEE Trans. Syst., Man, Cybern., Syst.*, early access.

[21] P. D. Thanh, D. A. Dung, T. N. Tien, and H. T. T. Binh, "An effective representation scheme in multifactorial evolutionary algorithm for solving cluster shortest-path tree problem," in *Proc. IEEE Congr. Evol. Comput.*, 2018, pp. 1–8.

[22] R. Chandra, "Dynamic cyclone wind-intensity prediction using co-evolutionary multi-task learning," in *Proc. Int. Conf. Neural Inf. Process.*, 2017, pp. 618–627.

[23] L. Bao *et al.*, "An evolutionary multitasking algorithm for cloud computing service composition," in *Proc. World Congr. Servery*, Jun. 2018, pp. 130–144.

[24] Q. Chen, X. Ma, Y. Sun, and Z. Zhu, "Adaptive memetic algorithm based evolutionary multi-tasking single-objective optimization," in *Proc. Int. Conf. Simul. Evol. Learn.*, 2017, pp. 462–472.

[25] E. O. Scott and K. A. De Jong, "Multitask evolution with cartesian genetic programming," Apr. 2017. [Online]. Available: arXiv:1702.02217.

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

[27] N. Q. Tuan, T. D. Hoang, and H. T. T. Binh, "A guided differential evolutionary multi-tasking with powell search method for solving multi-objective continuous optimization," in *Proc. IEEE Congr. Evol. Comput.*, Jul. 2018, pp. 1–8.

[28] L. Feng *et al.*, "Evolutionary multitasking via explicit autoencoding," *IEEE Trans. Cybern.*, vol. 49, no. 9, pp. 3457–3470, Jun. 2018.

[29] C. Yang, J. Ding, K. C. Tan, and Y. Jin, "Two-stage assortative mating for multi-objective multifactorial evolutionary optimization," in *Proc. IEEE Annu. Conf. Decis. Control*, Dec. 2017, pp. 76–81.

[30] J. Lin, H.-L. Liu, K. C. Tan, and F. Gu, "An effective knowledge transfer approach for multiobjective multitasking optimization," *IEEE Trans. Cybern.*, early access.

[31] K. Weiss, T. M. Khoshgoftaar, and D. D. Wang, "A survey of transfer learning," *J. Big Data*, vol. 3, no. 1, p. 9, May 2016.

[32] A. Gupta, Y.-S. Ong, and L. Feng, "Insights on transfer optimization: Because experience is the best teacher," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 51–64, Feb. 2018.

[33] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.

[34] L. Feng, K. Qin, A. Gupta, Y. Yuan, Y.-S. Ong, and X. Chi. (2019). *IEEE CEC 2019 Competition on Evolutionary Multi-Task Optimization*. [Online]. Available: http://cec2019.org/programs/competitions.html#cec-02

[35] Y.-W. Wen and C.-K. Ting, "Parting ways and reallocating resources in evolutionary multitasking," in *Proc. IEEE Congr. Evol. Comput.*, 2017, pp. 2404–2411.

[36] M. Gong, Z. Tang, H. Li, and J. Zhang, "Evolutionary multitasking with dynamic resource allocating strategy," *IEEE Trans. Evol. Comput.*, vol. 23, no. 5, pp. 858–869, Oct. 2019.

[37] X. Zheng, A. K. Qin, M. Gong, and D. Zhou, "Self-regulated evolutionary multi-task optimization," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 16–28, Feb. 2020.

[38] K. K. Bali, Y.-S. Ong, A. Gupta, and P. S. Tan, "Multifactorial evolutionary algorithm with online transfer parameter estimation: MFEA-II," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 69–83, Feb. 2020.

[39] L. Zhou *et al.*, "Toward adaptive knowledge transfer in multifactorial evolutionary computation," *IEEE Trans. Cybern.*, early access.

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

[41] K. K. Bali, A. Gupta, L. Feng, and Y.-S. Ong, "Linearized domain adaptation in evolutionary multitasking," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2017, pp. 1295–1302.

[42] L. Feng *et al.*, "Explicit evolutionary multitasking for combinatorial optimization: A case study on capacitated vehicle routing problem," *IEEE Trans. Cybern.*, early access.

[43] R.-T. Liaw and C.-K. Ting, "Evolutionary many-tasking based on biocoenosis through symbiosis: A framework and benchmark problems," in *Proc. IEEE Congr. Evol. Comput.*, 2017, pp. 2266–2273.

[44] J. Tang, Y. Chen, Z. Deng, Y. Xiang, and C. P. Joy, "A group-based approach to improve multifactorial evolutionary algorithm," in *Proc. Int. Joint Conf. Artif. Intell.*, 2018, pp. 3870–3876.

[45] Y. Chen, J. Zhong, L. Feng, and J. Zhang, "An adaptive archive-based evolutionary framework for many-task optimization." *IEEE Trans. Emerg. Topics Comput. Intell.*, early access.

[46] C. Wang, H. Ma, G. Chen, and S. Hartmann, "Evolutionary multitasking for semantic Web service composition," in *Proc. IEEE Congr. Evol. Comput.*, Jun. 2019, pp. 2490–2497.

[47] A. Rauniyar, R. Nath, and P. K. Muhuri, "Multi-factorial evolutionary algorithm based novel solution approach for multi-objective pollution-routing problem," *Comput. Ind. Eng.*, vol. 130, pp. 757–771, Apr. 2019.

[48] Y. Yuan *et al.*, "Evolutionary multitasking for multiobjective continuous optimization: Benchmark problems, performance metrics and baseline results," Jun. 2017. [Online]. Available: arXiv:1706.02766.

[49] Q. Xu and Q. Yang, "A survey of transfer and multitask learning in bioinformatics," *J. Comput. Sci. Eng.*, vol. 5, no. 3, pp. 257–268, Sep. 2011.

[50] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *J. Mach. Learn. Res.*, vol. 10, pp. 1633–1685, Jul. 2009.

[51] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, "Unsupervised visual domain adaptation using subspace alignment," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2014, pp. 2960–2967.

[52] B. Sun and K. Saenko, "Deep coral: Correlation alignment for deep domain adaptation," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 443–450.

[53] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[54] J. Ilonen, J.-K. Kamarainen, and J. Lampinen, "Differential evolution training algorithm for feed-forward neural networks," *Neural Process. Lett.*, vol. 17, no. 1, pp. 93–105, Feb. 2003.

[55] A. R. Hota and A. Pat, "An adaptive quantum-inspired differential evolution algorithm for 0–1 knapsack problem," in *Proc. IEEE World Congr. Nat. Biol. Inspired Comput.*, 2010, pp. 703–708.

[56] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[57] K. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. New York, NY, USA: Springer, 2006.

[58] A. W. Iorio and X. Li, "Solving rotated multi-objective optimization problems using differential evolution," in *Proc. Aust. Joint. Conf. Artif. Intell.*, vol. 3339, Dec. 2004, pp. 861–872.

[59] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, Feb. 2000.

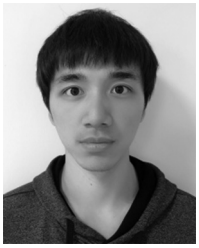
[60] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable multi-objective optimization test problems," in *Proc. IEEE Congr. Evol. Comput.*, vol. 1, 2002, pp. 825–830.

[61] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Trans. Evol. Comput.*, vol. 7, no. 2, pp. 117–132, Apr. 2003.

[62] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization," *IEEE Comput. Intell. Mag.*, vol. 12, no. 4, pp. 73–87, Nov. 2017.

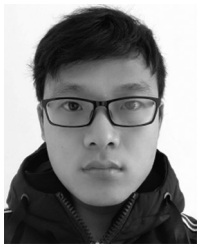
Zhengping Liang received the B.S. degree in computer science and technology from Hunan Normal University, Changsha, China, in 2001, and the Ph.D. degree in computer science and technology from Wuhan University, Wuhan, China, in 2006.

He is currently an Associate Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His main research interests include computational intelligence, multiobjective optimization, and big data analysis.



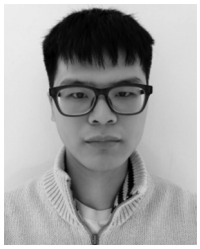
Hao Dong received the B.S. degree from Huizhou University, Huizhou, China, in 2017. He is currently pursuing the M.S. degree at Shenzhen University, Shenzhen, China.

His current research interests include transfer learning, multiobjective optimization, and evolutionary multitasking optimization.



Cheng Liu received the B.S. degree from Jiangsu University, Zhenjiang, China, in 2018. He is currently pursuing the M.S. degree with Shenzhen University, Shenzhen, China.

His current research interests include evolutionary computation, multitask optimization, and multiobjective optimization.



Weiqi Liang received the B.S. degree from the Zhongkai University of Agriculture and Engineering, Guangzhou, China, in 2017. He is currently pursuing the M.S. degree with Shenzhen University, Shenzhen, China.

His current research interests include evolutionary computation, multiobjective optimization, and evolutionary multitasking optimization.

Zexuan Zhu (Member, IEEE) received the B.S. degree in computer science and technology from Fudan University, Shanghai, China, in 2003, and the Ph.D. degree in computer engineering from Nanyang Technological University, Singapore, in 2008.

He is currently a Professor with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China. His research interests include computational intelligence, machine learning, and bioinformatics.

Prof. Zhu is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE. He is also the Chair of the IEEE Computational Intelligence Society Emergent Technologies Task Force on Memetic Computing.