A Fast Dynamic Evolutionary Multiobjective Algorithm via Manifold Transfer Learning

Min Jiang, *Senior Member, IEEE*, Zhenzhong Wang, Liming Qiu, Shihui Guo, *Member, IEEE*, Xing Gao, *Member, IEEE*, and Kay Chen Tan, *Fellow, IEEE*

***Abstract*—Many real-world optimization problems involve multiple objectives, constraints, and parameters that may change over time. These problems are often called dynamic multiobjective optimization problems (DMOPs). The difficulty in solving DMOPs is the need to track the changing Pareto-optimal front efficiently and accurately. It is known that transfer learning (TL)-based methods have the advantage of reusing experiences obtained from past computational processes to improve the quality of current solutions. However, existing TL-based methods are generally computationally intensive and thus time consuming. This article proposes a new memory-driven manifold TL-based evolutionary algorithm for dynamic multiobjective optimization (MMTL-DMOEA). The method combines the mechanism of memory to preserve the best individuals from the past with the feature of manifold TL to predict the optimal individuals at the new instance during the evolution. The elites of these individuals obtained from both past experience and future prediction will then constitute as the initial population in the optimization process. This strategy significantly improves the quality of solutions at the initial stage and reduces the computational cost required in existing methods. Different benchmark problems are used to validate the proposed algorithm and the simulation results are compared with state-of-the-art dynamic multiobjective optimization algorithms (DMOAs). The results show that our approach is capable of improving the computational speed by two orders of magnitude while achieving a better quality of solutions than existing methods.**

***Index Terms*—Dynamic multiobjective, manifold learning, transfer learning (TL).**

Manuscript received November 1, 2019; revised February 8, 2020 and April 5, 2020; accepted April 8, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61673328 and Grant 61876162, in part by the Shenzhen Scientific Research and Development Funding Program under Grant JCYJ20180307123637294, and in part by the Research Grants Council of the Hong Kong under Grant CityU11202418 and Grant CityU11209219. This article was recommended by Associate Editor Y. S. Ong. *(Corresponding authors: Liming Qiu; Kay Chen Tan.)*

Min Jiang, Zhenzhong Wang, Shihui Guo, and Xing Gao are with the School of Informatics, Xiamen University, Xiamen 361005, China (e-mail: liming.qiu@traxretail.com).

Liming Qiu is with the LenzTech Company, Beijing 100026, China (e-mail: liming.qiu@traxretail.com).

Kay Chen Tan is with the Department of Computer Science, City University of Hong Kong, Hong Kong, and also with the City University of Hong Kong Shenzhen Research Institute, Shenzhen, China (e-mail: kaytan@cityu.edu.hk).

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.2020.2989465

I. Introduction

Dynamic multiobjective optimization problems (DMOPs) have received increasing attention in recent years because many real-world problems can be formulated as DMOPs [1], [2]. An example is a dynamic portfolio optimization problem [3], which maximizes financial profit while minimizing risk. A variety of factors needs to be considered, including profit expectations, share/option prices, and risk obligations. These factors change over time, and it is a common demand to solve this optimization at high frequency. Therefore, accelerating the process of solving DMOPs is one of the research focuses on computational intelligence.

Among the recent progress of dynamic multiobjective optimization algorithms (DMOAs), the learning-based methods are particularly promising. This class of approaches allows the heuristic algorithms to improve their performance with machine-learning techniques. Generally, a prediction model can be created via machine-learning techniques, and assist the standard DMOAs by reducing the computational cost while maintaining a comparable performance even if the environment changes over time. For example, researchers proposed a memory-based evolutionary algorithm (EA) by introducing two prediction models: the first one used a regression model to predict the occurrence timing of environmental changes; and the second model was based on Markov chains and used to forecast changes [4]. Another work suggested integrating motion information into an EA so that the algorithm can track a time-changing optimum [5].

However, it is still an open challenge to solve DMOPs with a demanding goal of minimal time cost (e.g., optimizing the investment portfolio in high-frequency trading). One of the reasons is that the solutions of a given DMOP at different times are nonindependent identically distributed (Non-IID). This indicates that although there may exist a latent relationship between the solution distributions at different times, they are not identical.

An algorithmic framework, called Tr-DMOEA [6], addresses this challenge by integrating transfer learning (TL) [7] and population-based EAs. This approach speeds up the evolutionary process by reusing past experiences with TL and effectively improving the quality of the initial population. This framework can benefit general population-based multiobjective algorithms without requiring extensive modifications. The experimental results confirm the effectiveness of the proposed design for DMOPs. However, this solution still suffers from two significant defects that prevent its application in real-world tasks. First, the TL technique [7] solves the Non-IID problem by searching for a latent space, and the determination of the optimal latent space involves choosing the optimal values of numerous hyperparameters. This process of parameter tuning usually requires trial and error and often takes a significant amount of time. Second, even if the optimal latent space is found, *Tr-DMOEA* still needs to invoke the optimization algorithm to produce the initial population, which also consumes excessive computing resources.

The motivation of this article is to accelerate DMOPs-solving without deteriorating the solution quality. We propose a method for solving DMOPs by combining a *memory* mechanism with *manifold TL*. To the best of our knowledge, this is the first attempt to explore an integrative approach of both the techniques. These two approaches complement each other and this integration provides two benefits. First, the proposed method compresses high-dimensional data in a low-dimensional space with a special TL technique and it requires a few hyperparameters. Second, computing resources are spent on knowledge transfer of elite individuals via the memory mechanism, which greatly improves the speed of finding the solutions in evolutionary optimization.

The remainder of this article is organized as follows. Section II introduces the fundamental concepts of dynamic optimization problems and discusses the existing works. Section III presents background on manifold TL and explains the details of our method memory-driven manifold TL-based EA for dynamic multiobjective optimization (MMTL-DMOEA). In Section IV, we present the experimental results of the proposed algorithm on different test functions and compare with state-of-the-art algorithms. Section V draws a summary of this article and outlines future research directions.

II. Preliminaries and Related Works

A. Dynamic Multiobjective Optimization Problems

The essential characteristic of DMOPs is that the objectives change over time or under different environments. Without loss of generality, the general form of DMOPs is defined as

$$\begin{aligned} \min\ & F(x,t) = <f_1(x,t), f_2(x,t), \ldots, f_m(x,t)> \\ \text{s.t.}\ & x \in \mathcal{X} \end{aligned} \tag{1}$$

where $x = \langle x_1, x_2, \ldots, x_n \rangle$ is the n-dimensional decision variable, and $\mathcal{X} \in R^n$ is the decision space. t is the time (environment) variable, $f_1, f_2, \ldots, f_m$ are objective functions, and m is the number of objectives.

A set of solutions is defined as a dynamic Pareto-optimal set (POS) if this solution set exists in a DMOP at a given time t, and the decision vector in the set cannot be further optimized in any dimension.

Definition 1 (Dynamic Decision Vector Domination): At time t, a decision vector x_1 Pareto dominates another vector x_2, denoted by $x_1 \succ_t x_2$, if and only if

$$\begin{cases} \forall i = 1, \ldots, m, & f_i(x_1,t) \le f_i(x_2,t) \\ \exists i = 1, \ldots, m, & f_i(x_1,t) < f_i(x_2,t). \end{cases} \tag{2}$$

Definition 2 [Dynamic Pareto-Optimal Set (DPOS)]: Both x and x^* are decision vectors, and if a decision vector x^* is said to be nondominated at time t if and only if there is no other decision vector x such that $x \succ_t x^*$ at time t. The DPOS is the set of all Pareto-optimal solutions at time t, that is

$$\text{DPOS} = \left\{ x^* \mid \ \nexists x,\ x \succ_t x^* \right\}.$$

Definition 3 [Dynamic Pareto-Optimal Front (DPOF)]: At time t, the DPOF is the corresponding objective vectors of the DPOS

$$\text{DPOF} = \left\{ F\left(x^*, t\right) \mid x^* \in \text{DPOS} \right\}.$$

The purpose of a DMOA is to produce a set of solutions as close as possible to the changing POF while maximizing the diversity of the solution.

B. Manifold-Based Transfer Learning

1) Manifold and Geodesic: A *manifold* is a topological space that is locally homeomorphic to a Euclidean space. In other words, at any point of a d-dimensional manifold, it is locally homeomorphic to the Euclidean space R^d. For example, the surface of the Earth as a 2-D manifold, and any part of it can be considered as a 2-D plane. A geodesic is a smooth curve on the manifold, that is locally the shortest curve connecting two points. A geodesic can be considered as a “bent” straight line on a manifold. Calculating the distance between two points on the manifold is to calculate the geodesic distance. For more details about manifold learning and geodesic, please refer to [8].

2) Sample Geodesic Flow: The manifold-based TL algorithm used in this article is sample geodesic flow (SGF) [9]. The SGF algorithm is inspired by incremental learning: a learning system can continuously learn from the samples and keep most knowledge of them. In order to apply the knowledge learned from the source domain to the target domain, the SGF algorithm constructs a geodesic path between the two domains and then transports the knowledge through this path to the target domain. In other words, the source and target domains are mapped to the starting and ending points, and the algorithm constructs a geodesic flow between the two points in the manifold. After that, it will select k intermediate points on the geodesic flow, and finally, transform the data of the source and target domains to these intermediate points.

Fig. 1 presents an illustrative diagram of the SGF method. First, the algorithm maps the source and target domains to the two points S_0 and S_1 on a Grassmann manifold $G(d,n)$, and the algorithm will construct a geodesic flow from S_0 to S_1 (shown as a dotted line in the figure). Next, it will select k intermediate points, the red dot $S_{0.2}$, $S_{0.4}$, $S_{0.6}$, $S_{0.8}$ on the geodesic flow, and transform the training and testing data to these intermediate subspaces. The data onto these subspaces will be analyzed to perform classification.

C. Related Works

Existing DMOAs can be divided into the following categories: maintaining diversity methods, multipopulation-based methods, memory-based methods, and prediction-based methods.

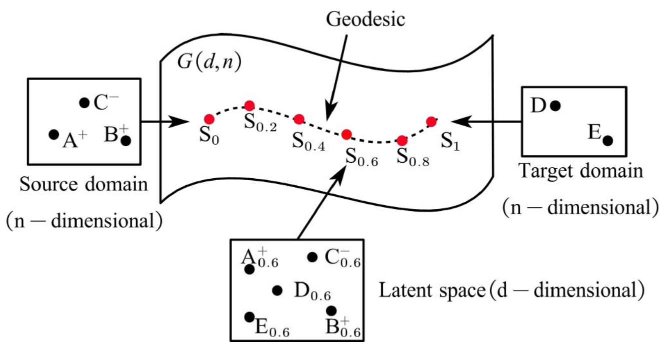


Fig. 1. Illustrative diagram of the SGF method. *A, B*, and *C* are the labeled samples from the source domain. *D* and *E* are the samples from the target domain that need to be classified.

Maintaining population diversity is an effective approach for minimizing the likeliness of the algorithm to be trapped in local optima. This strategy assumes that a population with higher diversity should have better adaptability and aims to maintain population diversity. A representative work in this category is the dynamic NSGA-II (D-NSGA-II) proposed by Deb *et al.* [10]. This algorithm has two versions, and the first version is DNSGA-II-A, which increases diversity by replacing some individuals in the population, $\zeta\%$ of the population, with randomly generated individuals. The second version, DNSGA-II-B, ensures diversity by substituting mutated individuals for $\zeta\%$ of the population. The actual performance of these two versions depends on the choice of ζ. Considering the disconnectedness of the POF of DMOPs with constraints may cause the population to sink into local regions, decreasing the population diversity. Chen *et al.* [11] designed a selection operator which can obtain nondominated solutions with diversity when the environment changes. The mating selection strategy and population selection operator can handle infeasible solutions adaptively.

Recently, Ruan *et al.* [12] also proposed an algorithm based on maintaining population diversity. The method uses the position and direction of the center point of the optimal solution at the previous moment to predict future solutions. Jiang and Yang [13] proposed a strategy based on steady state and population diversity, called SGEA. This algorithm detects the incidence of environmental changes and uses a steady-state method to adapt to the new environment quickly.

The multipopulation strategy is an efficient solution for DMOPs, especially in the case of multiple peaks and competing peaks. Goh and Tan [14] proposed a coevolutionary multiobjective algorithm (dCOEA) based on competition and collaboration, which breaks the problem into several subproblems. Yazdani *et al.* [15] proposed a cooperative coevolutionary multipopulation framework for solving large-scale dynamic optimization problems. The framework breaks a large-scale dynamic optimization problem into a set of lower dimensional components and controls the budget assignment to components [16] for tracking multiple moving optima. Gong *et al.* [17] proposed a framework in which all the decision variables are divided into two subpopulation according to the interval similarity between each decision variable and interval parameters.

The memory-based methods use additional storage to implicitly or explicitly store useful information from previous generations to improve the algorithm performance. Wang and Li [18] proposed a multistrategy dynamic multiobjective EA (MS-MOEA). It uses a progeny generation mechanism based on adaptive genetics and differential operators to accelerate convergence. Chen *et al.* [16] implemented a dynamic two-archive EA that maintains two co-evolving populations simultaneously. Two complementary populations concern about convergence and diversity, respectively.

The prediction-based DMOAs have received much attention in the field of DMOPs, and this class of methods predicts the state of the changing environment and then makes a decision such that the algorithms can accommodate the changes in advance. The prediction typically introduces machine-learning techniques to utilize existing information in the memory.

Rong *et al.* [19] presented a multidirectional prediction strategy to predict the moving location of the POS accurately. A classification strategy clusters the population into a number of representative groups and adapts the number of clusters according to the intensity of the environmental change. Koo *et al.* [20] proposed a prediction strategy called the dynamic gradient prediction strategy. It defines a set of gradient prediction vectors that can be used to correlate the current solution with the previously obtained optimal solution. Rong *et al.* [21] presented a multimodel prediction approach (MMP), and the method detects the type of change and then selects an appropriate prediction model to generate an initial population.

Zhou *et al.* [22] proposed a method called a population prediction strategy (PPS), for predicting the entire population. When a change in the environment is detected, the next center point is predicted by an autoregressive model using a series of center points throughout the search process, and at the same time, the previous manifold is used to estimate the next manifold. The main problem of this method is the lack of sufficient historical information at the beginning stage, which may lead to poor convergence.

Muruganantham *et al.* [23] proposed a predictive dynamic multiobjective EA (MOEA/D-KF) based on the Kalman filter. The Kalman filter guides the search for a new POS and generates a new initial population. The optimal solutions can be found via a decomposition-based differential evolution algorithm (MOEA/D-DE) [24]. Zou *et al.* [25] proposed a prediction strategy based on center points and knee points (CKPSs) to predict the initial population of the next moment. Xu *et al.* [26] introduced a method in which all decision variables are partitioned into two subcomponents according to their interrelation with the environment. The two subcomponents are optimized with two prediction methods. Woldesenbet and Yen [27] distinguished decision variables by their average sensitivities to the change in the objective space and relocated individuals.

The prediction-based algorithms show their advantages in maintaining the population quality, however, it may not produce satisfactory results in problems with Non-IID. An inaccurate prediction model will likely lead the search process in the wrong direction, which means actual results will be worse than a method that does not use the prediction technique.

TL is a powerful weapon that solves the dilemma caused by the Non-IID hypothesis, and recent research has also proved that transferring acquired knowledge in the optimization process is promising [28], especially for solving dynamic [6] or multiple optimization tasks simultaneously [29]–[31]. Jiang *et al.* [6] proposed a framework called Tr-DMOEA for solving DMOPs. This framework integrates between TL and classical evolutionary multiobjective optimization algorithms. TL effectively generates a high-quality initial population pool via reusing past experience to speed up the evolutionary process.

Bali *et al.* [31] proposed an online transfer parameter estimation approach for the multitasking EA. The proposed algorithm evaluates the similarities between different task instances, therefore, the extent of knowledge transfer can be adapted based on the optimal mixture of probabilistic models. Da *et al.* [32] demonstrated a framework for black-box transfer optimization. The framework measures the similarities between the source instance and the target instance and reveals latent synergies during optimizing. When faced with multiple sources, the positive and negative transfer is automatically identified so as to curb the negative transfer, thereby the excellent performance is kept.

Inspired by the recent autoencoding evolutionary search [29], Zhou *et al.* [33] proposed an improved adaptive indicator-based EA. The proposed algorithm is capable of transferring the past search information by adapting some obtained solutions with high hypervolume value. Min *et al.* [34] designed a transfer stacking of the Gaussian process surrogate models by reusing the acquired knowledge. The adaptive knowledge gained from expensive problem-solving information is introduced to improve the performance of the optimization process of the target task. Liu *et al.* [35] proposed a neural-network-based information transfer method to reuse past solutions. When the environment changes, the proposed algorithm collects the solutions from both the previous environment and the new environment. Then, the neural-network transfers acquired solutions for guiding the search in the new environment.

Introducing TL into the field of dynamic optimization shows its potential. However, at the same time, some new problems arise. For example, Tr-DMOEA often leads to diminishing diversity in solutions. Researchers proposed a dynamic multiobjective estimation of distribution algorithm, called DANE-EDA [36]. This design combines the Monte Carlo method with the TL technique, and this combination maintains a tradeoff of exploration–exploitation from temporal and spatial perspectives. Although researchers made progress in the DMOP domain, existing methods are generally time consuming and not suitable for some applications in the real world.

III. MMTL-DMOEA Algorithm

This article proposes the method of memory-driven manifold TL (MMTL-DMOEA) to address DMOPs. The purpose of this method is to accelerate the process of solution searching while ensuring the solution quality. The main contribution of MMTL-DMOEA is the integration of manifold TL with the elite memory mechanism, to produce an initial population for the next generation. This initial population can help any population-based multiobjective optimization algorithms find the POS of the optimization problem faster.

Process 1 MMTL-DMOEA
```
Input: The dynamic optimization function: $F(x, t)$; the size of
    the population: $N$; an external storage $P$ and its capacity
    $C$; the number of clusters: $L$.
Output: The $Solutions_t$ at time $t$.
 1: Initialization;
 2: $Solutions_0 = \mathbf{SMOA}(POP_0, F(x, 0), N)$;
 3: Store $Solutions_0$ into $P$;
 4: while Change detected do
 5:     $t = t + 1$;
 6:     $LastBestSol = \mathbf{FindBestSol}(P, F(x, t), N)$;
 7:     $TransSol = \mathbf{Transfer}(LastBestSol, F(x, t), N, L)$;
 8:     $POP_t = LastBestSol \cup TransSol$;
 9:     $(Solutions_t) = \mathbf{SMOA}(POP_t, F(x, t), N)$;
10:     if size$(P \cup Solutions_t) \leq C$ then
11:         Store $Solutions_t$ into $P$;
12:     else
13:         Replace the earliest stored individuals with
    $Solutions_t$;
14:     end if
15: end while
16: return $Solutions_t$;
```

Fig. 2 shows the algorithm flow of MMTL-DMOEA. This algorithm first *stores* the optimal solutions obtained at different times in external memory. Once the environment changes are detected, the algorithm *selects* a certain number of individuals from external storage according to predefined criteria. In the third step, the algorithm predicts individuals through the principal component analysis (PCA) and the manifold TL algorithm. After generating the predicted individuals, the elite individuals selected from the memory are *merged* with them to constitute an initial population.

The detailed steps of MMTL-DMOEA are shown in Process 1. It is worth pointing out that any population-based static multiobjective algorithm (SMOA) can be used as a routine for processing static multiobjective optimization problems in our proposed algorithm.

Remark 1: The parameter C refers to the maximum number of individuals that can be stored in the external memory.

Remark 2: When the external memory overflows, the algorithm replaces the earliest stored individuals with the newly generated Solutions$_t$.

Our proposed algorithm follows the elite reservation mechanism. In other words, we will preserve the optimal solutions obtained at previous moments. Although solutions at different times are varying, there should be some inherent relationship between these solutions. Therefore, it is possible to obtain a better initial population by reserving the best individuals and making a reasonable transformation of these individuals.

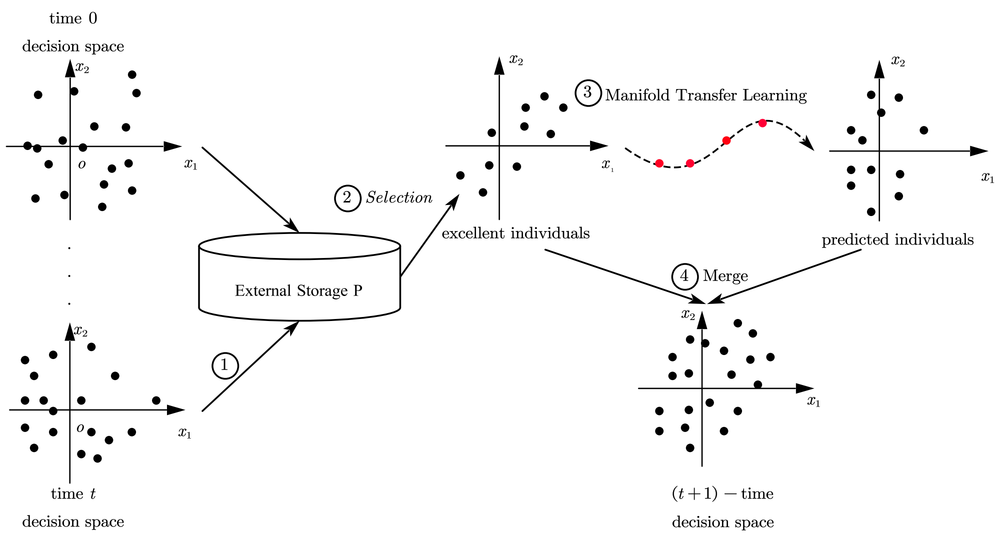


Fig. 2. Main idea of MMTL-DMOEA. Step 1: MMTL-DMOEA will preserve the nondominated solutions obtained from time 0 to time t. Step 2: Some individuals are selected from the stored elite set according to the predefined mechanism. Step 3: The selected individuals are mapped to the predicted individuals by using the manifold-based TL method. Step 4: MMTL-DMOEA generates an initial population by merging predicted and selected individuals.

Lines 10–14 in Process 1 implement our elite reserving strategy. When the external memory overflows, the algorithm replaces the earliest stored individuals with the newly generated ones. This straightforward strategy builds on the intuition that the optimal solution for the next moment may be more relevant to the recently obtained individuals.

Selecting the most suitable individual from external memory is critical in MMTL-DMOEA. The results of the selection algorithm (Process 2) provide samples of the source domain for TL, and part of selected solutions directly constitute the final population. When changes are detected, to avoid extra function evaluations of individuals in the external storage, an estimator for estimating the objective values of individuals from the external memory is constructed by support vector regression (SVR) with only a small number of individuals sampled from the decision space in the new environment. Then, according to the estimated objective values, all individuals in the external storage are ranked by nondominated sorting, and the nondominated solutions are chosen. In order to ensure the diversity of solutions, the selection algorithm will eliminate individuals with a higher degree of congestion. When the number of individuals in the nondominated solution set is too large, the algorithm will randomly select some solutions to remove from the set; otherwise, some random noise will be generated to supplement the solution set.

MMTL-DMOEA also introduces TL to predict the initial population of the next moment, as shown by *Transfer* in line 7 of Process 1. The *Transfer* algorithm is an improved version of the TL method SGF.

In SGF, the difficulty lies in constructing a geodesic model. First, the PCA method is used to produce the feature vector corresponding to first d eigenvalues. Descending eigenvalues sort the covariance matrix as a set of subspaces, which are denoted as P_S and P_T, respectively. $P_S \in R^{n\times d}$, $P_T \in R^{n\times d}$. $R_S \in R^{n\times(nd)}$ is the orthogonal complement of P_S.

The geodesic from the source domain to the target domain can be represented by $\phi(k)$ with $\phi(0) = P_S$ and $\phi(1) = P_T$

```
Process 2 FindBestSol
Input: The dynamic optimization function: $F(x, t)$;
       Number of individuals: $N$;
       External storage: $P$.
Output: Excellent individuals $LastBestSol$.
1:  Uniformly sample $n_s$ solutions $X_T$ from the decision space;
2:  Call a SVR to construct the estimator $E$ with $\{X_T, F(X_T, t)\}$;
3:  Estimate objectives of $P$: $Y = E(P)$;
4:  Find non-dominated solutions $LastBestSol$ in $Y$;
5:  while $N/2 < size(LastBestSol)$ do
6:      Delete individual in LastBestSol;
7:  end while
8:  while $N/2 > size(LastBestSol)$ do
9:      Add Gaussian noise with individuals in $LastBestSol$ to $LastBestSol$;
10: end while
11: return $LastBestSol$;
```

when $0 < k < 1$

$$\phi(k) = P_S U_1 \Gamma(k) - R_S U_2 \Sigma(k) \tag{3}$$

where U_1 and U_2 are a pair of orthogonal matrices obtained by SVD decomposition. $\Gamma(k)$ and $\Sigma(k)$ are $d \times d$ dimensional diagonal matrices with diagonal elements $\cos(k\theta_i)$ and $\sin(k\theta_i)$, $i = 1, 2, \ldots, d$, and $\theta_i \in [0, \pi/2]$ are the principal angle of P_S, P_T.

After obtaining the geodesic model, the point where the original data are mapped to the geodesic can be calculated through

$$x_k = x^T \phi(k), \quad k \in (0, 1) \tag{4}$$

where $x \in R^n$ is the original data (training data or testing data), and x_k is the feature data on the middle subspace where x corresponds to the position p on the geodesic flow (e.g., $A_{0.6}$ in Fig. 1). The number of subspaces p is predefined.

Process 3 Transfer

```
Input: The elite individuals: $LastBestSol$;
       The dynamic optimization function: $F(x, t)$;
       Number of individuals: $N$;
       Clustering number: $L$.
Output: Predicted individuals: $TransSol$.
1:  Initialization: dimension $d = m - 1$; $TransSol = \emptyset$;
2:  Clustering $LastBestSol$: $LastBestSol^1$, ..., $LastBestSol^L$ by LPCA;
3:  for j = 1 to L do
4:      Use PCA for $LastBestSol^j$ to get $P_S$;
5:      Generate set $T$ containing $N$ individuals of $F(x, t)$;
6:      Use PCA for $T$ to get $P_T \in \mathbb{R}^{d \times N}$;
7:      Construct the geodesic flow $\phi(k) = P_S U_1 \Gamma(k) - R_S U_2 \Sigma(k), k \in (0, 1)$;
8:      for $x \in LastBestSol^j$ do
9:          Project $x$ to $\phi(\cdot)$ and get $\bar{x}$;
10:         $x' = \arg\min_{x'} \|x'^T \phi(\cdot) - \bar{x}\|$;
11:         $TransSol = x' \cup TransSol$;
12:     end for
13: end for
14: return $TransSol$;
```

The detailed steps for *Transfer* are shown in Process 3. We believe that the estimated nondominated solutions *LastBestSol* can be viewed as $m - 1$-dimensional segmented manifolds [37], where m is the number of objectives. Hence, first, the *LastBestSol* is clustered by LocalPCA (LPCA) [37] into $m - 1$-dimensional L segmented manifolds: $\text{LastBestSol}^1, \ldots, \text{LastBestSol}^L$ in line 2. For each cluster, the geodesic flow $\phi(\cdot)$ are constructed in lines 4–7. Then, $x \in \text{LastBestSol}^j$ is mapped into the geodesic flow in line 9, and we can obtain the mapped data $\bar{x}$. In the dynamic environment, the manifolds LastBestSol^j in the source domain may be similar to those in the target domain, so the mapped target domain data $x'^T\phi(\cdot)$ is similar to $\bar{x}$. Therefore, in line 10, we find a solution x', such that the mapped data $x'^T\phi(\cdot)$ on the geodesic flow is closet to $\bar{x}$. This is a single-objective optimization problem, and any single-objective optimization algorithm can be applied to solve the problem. In this article, we use the interior-point method to solve the problem.

A. Computational Complexity

In the *FindBestSol*, constructing the SVR estimator with n_s samples needs $O(n_s^2 d)$, where d is the dimension of decision variables. Using SVR to estimate the objective values consumes $O(N^2 d)$, where N is the population size. According to the estimated objective values, the cost of finding nondominated solutions by the fast nondominated sorting is $O(N^2 m)$, where m is the number of objectives. Because $N \gg n_s$ and $d \gg m$, therefore, the computational complexity of *FindBestSol* is $O(N^2 d)$.

The computational cost of *Transfer* involves clustering, constructing the geodesic flow, and finding solutions by using the interior-point algorithm. Clustering *LastBestSol* by LPCA calls for $O(d^2)$. For each cluster, mapping data via PCA and constructing the geodesic flow spends $O(d^2)$ and $O(1)$, respectively. The computational cost of finding solutions by using the interior-point algorithm consumes $O(m^3 n)$ [6]. Because the number of clusters is a constant set by the user, therefore, the computational complexity of *Transfer* is $O(m^3 n)$. The total computational complexity of the proposed prediction is thus $O(m^3 n)$.

IV. Experiments

In order to verify the effectiveness of MMTL-DMOEA, we carried out three sets of experiments. The first experiment compares the proposed MMTL-DMOEA with the state-of-the-art DMOAs. The second experiment verifies the effectiveness of integrating the components of memory and manifold TL. The third experiment compares with Tr-DMOEA [6] and analyzes performance improvement.

A. Comparison With Other DMOPs

The test functions used are FDA series [38], dMOP series [14], and DF series [39]. The time parameter in the test functions is $t = (1/n_t)\lfloor \tau/\tau_t \rfloor$, where n_t, τ_t, and τ are the severity of the time change, the frequency of change, and the maximum generation of the problems, respectively.

The algorithms used for comparison in this experiment are as follows: MOEA/D-SVR [40], Tr-DMOEA [6], MOEA/D-KF [23], PPS [22], and the baseline algorithm MOEA/D [24], which are modified as RI-MOEA/D to adapt to dynamic change, that is, 10% of the population is randomly reinitialized when the environment changes. For a fair comparison, the baseline algorithm in all the compared algorithms are replaced by MOEA/D [24], and these compared algorithms are denoted as KT-MOEA/D, SVR-MOEA/D, Tr-MOEA/D, KF-MOEA/D, and PPS-MOEA/D, respectively, and most of the parameters of these algorithms are set according to their original references.

There are three metrics used in this group of comparison experiments, and three metrics are listed as follows.

Inverted generational distance (IGD) measures the convergence and diversity of the solutions. A smaller value of IGD indicates a better convergence of the solution obtained by the algorithm and the higher the diversity. This metric is computed as

$$\text{IGD} = \frac{1}{n_{P^t}} \sum_{p \in P^t} \min_{p^* \in P^*} \|p - p*\|^2 \quad (5)$$

where P^t represents the true POF of the problem, P^* represents the obtained POF by testing algorithms, and n_{P^t} represents the number of solutions in the true POF.

The MIGD metric is a variant of IGD and is defined as the mean IGD values in time steps

$$\text{MIGD}\big(\text{POF}_t^*, \text{POF}_t\big) = \frac{1}{|T|} \sum_{t \in T} \text{IGD}\big(\text{POF}_t^*, \text{POF}_t\big) \quad (6)$$

where T is a set of discrete-time points in a run and $|T|$ is the cardinality of T.

Schott's spacing metric (SP) measures the solution uniformity. A smaller SP value reflects a more uniform distribution

TABLE I
MEAN AND STANDARD DEVIATION VALUES OF MIGD METRIC OBTAINED BY COMPARED ALGORITHMS FOR DIFFERENT DYNAMIC TEST SETTINGS

Problems	n_t, τ_t	MMTL-MOEA/D	KF-MOEA/D	PPS-MOEA/D	SVR-MOEA/D	Tr-MOEA/D	RI-MOEA/D
FDA1	5,10	**0.1214±1.07e-01**	0.4670±3.38e-01	0.2485±1.40e-01	0.3745±3.12e-01	0.3381±2.14e-01	0.3166±3.58e-01
	10,10	**0.1199±7.93e-02**	0.2659±1.23e-01	0.2141±1.22e-01	0.2332±1.66e-01	0.3592±3.41e-01	0.2733±1.83e-01
	10,20	**0.0658±3.64e-02**	0.1635±9.12e-02	0.1018±1.25e-01	0.2168±2.03e-01	0.1778±2.47e-01	0.1959±2.36e-01
FDA2	5,10	**0.0740±3.53e-02**	0.1695±6.51e-02	0.1023±1.09e-01	0.2062±1.66e-01	0.1241±4.72e-02	0.2127±1.49e-01
	10,10	**0.0842±3.34e-02**	0.1906±7.00e-02	0.1200±2.00e-01	0.1965±1.31e-01	0.1243±4.27e-02	0.2528±1.34e-01
	10,20	**0.0662±3.63e-02**	0.1335±4.02e-02	0.0719±9.86e-02	0.1810±1.88e-01	0.0785±3.37e-02	0.1678±1.44e-01
FDA3	5,10	**0.1428±1.11e-01**	0.2685±2.66e-01	0.3142±2.14e-01	0.2250±1.81e-01	0.2925±2.44e-01	0.3493±4.27e-01
	10,10	**0.0914±9.77e-02**	0.1429±7.49e-02	0.2072±1.38e-01	0.1994±1.93e-01	0.2529±2.75e-01	0.2530±3.05e-01
	10,20	**0.0749±5.08e-02**	0.1349±1.02e-01	0.2286±1.76e-01	0.1409±1.94e-01	0.1442±8.24e-02	0.1361±7.58e-02
FDA4	5,10	**0.1523±9.67e-02**	0.1578±7.21e-02	0.2114±1.48e-01	0.1866±7.83e-02	0.2335±1.21e-01	0.1702±4.11e-02
	10,10	0.1594±5.77e-02	**0.1311±4.03e-02**	0.1848±1.75e-01	0.1709±5.15e-02	0.2180±1.05e-01	0.1787±8.33e-02
	10,20	0.1336±3.89e-02	0.1250±4.06e-02	0.1765±2.02e-01	**0.1234±2.36e-02**	0.1998±9.90e-02	0.1253±2.66e-02
FDA5	5,10	0.2081±6.47e-02	0.2683±8.65e-02	0.2036±7.28e-02	0.2120±1.05e-01	**0.1737±4.19e-02**	0.2184±1.01e-01
	10,10	0.1892±5.19e-02	0.2369±7.79e-02	0.2305±1.04e-01	0.1862±9.43e-02	**0.1752±4.89e-02**	0.2140±1.01e-01
	10,20	**0.1642±6.06e-02**	0.1818±5.76e-02	0.1895±8.11e-02	0.1729±9.00e-02	0.1879±4.56e-02	0.1968±7.64e-02
dMOP1	5,10	**0.0589±3.82e-02**	0.1857±9.13e-02	0.1269±2.37e-01	0.2237±8.15e-02	0.2345±6.53e-02	0.2421±1.33e-01
	10,10	**0.0543±5.52e-02**	0.1565±7.39e-02	0.0965±2.18e-01	0.3266±1.99e-01	0.2507±8.15e-02	0.2734±1.46e-01
	10,20	**0.0252±9.00e-03**	0.1145±5.03e-02	0.0690±1.95e-01	0.1938±1.25e-01	0.1204±9.13e-02	0.1606±1.63e-01
dMOP2	5,10	**0.0494±1.59e-02**	0.2258±1.31e-01	0.1265±1.34e-01	0.1302±8.99e-02	0.1311±6.02e-02	0.1505±1.58e-01
	10,10	**0.0717±4.20e-02**	0.1646±8.01e-02	0.1102±1.00e-01	0.1142±8.98e-02	0.1157±6.03e-02	0.1586±1.33e-01
	10,20	**0.0261±8.53e-03**	0.1208±8.70e-02	0.0771±1.12e-01	0.0541±4.82e-02	0.0795±4.89e-02	0.0609±4.64e-02
dMOP3	5,10	**0.0593±3.10e-02**	0.1132±8.72e-02	0.1136±8.84e-02	0.0987±7.16e-02	0.1203±4.29e-02	0.0729±3.87e-02
	10,10	**0.0683±4.26e-02**	0.1431±5.58e-02	0.0736±6.38e-02	0.0897±4.56e-02	0.1057±5.18e-02	0.0850±5.68e-02
	10,20	**0.0260±5.56e-03**	0.0730±4.91e-02	0.0563±6.87e-02	0.0510±3.52e-02	0.0575±3.22e-02	0.0401±2.57e-02

of the solution obtained by the algorithm. This metric is calculated as

$$\mathrm{SP} = \sqrt{\frac{1}{n_{\mathrm{POF}^*} - 1} \sum_{i=1}^{n_{\mathrm{POF}^*}} \left(D_i - \overline{D}\right)^2} \tag{7}$$

where D_i represents the Euclidean distance between the ith individual in POF* and its closest solution, and $\overline{D}$ represents the mean of D_i. This metric is similarly modified to the IGD to act as a performance metric for evaluating DMOAs.

Maximum spread (MS) measures the coverage extent of obtained solutions in the true POF. A larger MS value indicates a higher coverage by the obtained solution in the real POF. This metric is calculated as

$$\mathrm{MS} = \sqrt{\frac{1}{M} \sum_{k=1}^{M} \left[\frac{\min\left[\overline{P_k^t}, \overline{P_k^*}\right] - \max\left[\underline{P_k^t}, \underline{P_k^*}\right]}{\overline{P_k^t} - \underline{P_k^t}} \right]^2}$$

where $\overline{P_k^t}$ and $\underline{P_k^t}$ represent the maximum and minimum of the kth objective in true POF, respectively, and $\overline{P_k^*}$ and $\underline{P_k^*}$ represent the maximum and minimum of the kth objective, in the obtained POF, respectively. This metric is similarly modified to the IGD to act as a performance metric for evaluating DMOAs.

The experiment parameters are set as follows. The size of population N is 100; and the external storage size C is set to $10 \times N$. Most of the parameters of the SVR in *FindBsetSol* are set by default [41]; the number of sampling individuals n_s in *FindBsetSol* is 30; the number of manifold segments L in *Transfer* is 4; and the number of intermediate subspaces p in *Transfer* is 5.

The experimental results are shown in Tables I–III. The data in the table are the MIGD, SP, and MS values of compared algorithms under different configurations. The bold data in these tables are the best metrics obtained by the compared algorithm.

The experimental results of Table I show that the MMTL-MOEA/D algorithm has achieved the best results in 20 out of 24 test cases based on the MIGD values, indicating that the optimal solution obtained by MMTL-MOEA/D is better than the existing algorithm in terms of convergence. Fig. 3 plots the IGD values obtained by different algorithms after each change. It shows that the curves obtained by the proposed method are at the bottom in most cases, and the curves of the proposed method are smoother, which means that the method not only performs better but also is more stable. Table II presents the average SP metric values on the FDA and dMOP series of test functions. The results show that MMTL-MOEA/D is far better than the other five algorithms in 15 out of 24 test cases. This good performance shows that the distribution of solutions obtained by MMTL-MOEA/D is relatively uniform. Table III shows the average MS of the solutions obtained for the six algorithms. The results show that the MMTL-MOEA/D has a higher degree of coverage of the true POF than the other five algorithms on most of the test cases.

To see whether the proposed prediction model could be integrated into other MOEAs, we also replace the baseline SMOA by RM-MEDA [37] and these compared algorithms are denoted as MMTL-RM-MEDA, SVR-RM-MEDA, Tr-RM-MEDA, KF-RM-MEDA, and PPS-RM-MEDA, respectively. The experimental results are presented in Tables I and II in the supplementary material. From these tables, we can obverse that our proposed model is still competitive in terms of MIGD, SP, and MS metric values. We also plot the IGD values obtained by different algorithms in Fig. 1 in the supplementary material.

We conducted experiments on the DF series test functions under the configuration of $n_t = 10$ and $\tau_t = 10$. All DF functions have a changing POS, which belong to type I (POS changes, but POF does not change) or type II (POS and POF

TABLE II
MEAN AND STANDARD DEVIATION VALUES OF SP METRIC OBTAINED BY COMPARED ALGORITHMS FOR DIFFERENT DYNAMIC TEST SETTINGS

Problems	n_t, τ_t	MMTL-MOEA/D	KF-MOEA/D	PPS-MOEA/D	SVR-MOEA/D	Tr-MOEA/D	RI-MOEA/D
FDA1	5,10	**1.0768±7.55e-02**	1.0889±1.16e-01	1.1965±1.25e-01	1.1036±7.10e-02	1.0837±1.18e-01	1.1423±1.33e-01
	10,10	1.1239±1.01e-01	**1.0742±2.32e-01**	1.1900±1.23e-01	1.1211±1.22e-01	1.0890±7.96e-02	1.1481±1.33e-01
	10,20	1.0302±4.61e-02	1.0400±1.01e-01	1.0894±8.41e-02	1.0550±5.76e-02	**1.0266±5.00e-02**	1.0458±7.90e-02
FDA2	5,10	1.1011±1.05e-01	**1.0429±7.14e-02**	1.0697±1.10e-01	1.0894±9.38e-02	1.0705±5.09e-02	1.0795±1.24e-01
	10,10	1.0841±8.77e-02	1.0549±4.18e-02	**1.0532±1.27e-01**	1.0658±1.03e-01	1.0815±6.87e-02	1.0650±7.38e-02
	10,20	**0.9660±9.24e-02**	1.0116±3.42e-02	1.0338±9.03e-02	0.9896±7.64e-02	0.9825±8.70e-02	1.0118±6.17e-02
FDA3	5,10	**1.0791±7.25e-02**	1.0844±7.60e-02	1.2019±1.01e-01	1.1110±1.19e-01	1.0866±6.39e-02	1.0891±1.28e-01
	10,10	**1.1273±7.70e-02**	1.1401±1.68e-01	1.2693±2.05e-01	1.1719±1.66e-01	1.1672±1.89e-01	1.1292±1.41e-01
	10,20	**1.0184±7.19e-02**	1.0532±4.43e-02	1.1343±7.87e-02	1.0790±1.10e-01	1.0453±5.88e-02	1.0555±7.24e-02
FDA4	5,10	**0.7983±1.08e-01**	0.8628±1.08e-01	0.9481±1.27e-01	0.9313±1.19e-01	0.9446±1.49e-01	0.9342±1.40e-01
	10,10	**0.8211±1.10e-01**	0.8240±1.33e-01	0.9648±1.74e-01	0.8758±8.77e-02	0.9432±1.21e-01	0.8978±1.01e-01
	10,20	**0.7157±5.81e-02**	0.7404±1.12e-01	0.8986±1.61e-01	0.7503±7.61e-02	0.7556±7.35e-02	0.7487±6.58e-02
FDA5	5,10	1.1046±1.10e-01	1.0565±9.64e-02	1.2018±1.59e-01	1.1335±1.11e-01	**1.1043±9.17e-02**	1.0983±9.88e-02
	10,10	**1.0400±7.64e-02**	1.0416±5.34e-02	1.2324±1.95e-01	1.1464±8.54e-02	1.0780±7.90e-02	1.1413±1.37e-01
	10,20	1.0392±6.34e-02	**1.0051±5.87e-02**	1.2975±1.35e-01	1.0599±8.72e-02	1.0324±7.69e-02	1.0697±9.21e-02
dMOP1	5,10	1.0492±9.94e-02	1.0446±3.62e-02	**1.0054±1.07e-01**	1.0900±6.90e-02	1.0920±5.20e-02	1.0609±5.61e-02
	10,10	**1.0554±1.25e-01**	1.0627±6.75e-02	1.0677±1.32e-01	1.0606±5.43e-02	1.0911±4.90e-02	1.0780±4.35e-02
	10,20	**0.8787±8.40e-02**	1.0230±4.42e-02	0.9862±1.02e-01	1.0104±3.24e-02	1.0129±5.70e-02	1.0082±4.79e-02
dMOP2	5,10	**1.0202±9.83e-02**	1.0572±3.86e-02	1.1170±8.34e-02	1.0661±6.70e-02	1.0816±5.21e-02	1.0696±6.62e-02
	10,10	**0.9988±1.05e-01**	1.0505±5.92e-02	1.0933±7.98e-02	1.0669±7.23e-02	1.0554±7.07e-02	1.0430±5.52e-02
	10,20	**0.8635±6.87e-02**	1.0237±1.30e-01	1.0195±9.80e-02	0.9820±7.46e-02	0.9717±7.98e-02	0.9503±9.71e-02
dMOP3	5,10	1.1163±1.09e-01	**1.0935±4.93e-02**	1.1714±9.43e-02	1.1313±8.20e-02	1.1094±8.57e-02	1.1225±9.38e-02
	10,10	1.1254±1.09e-01	1.0932±6.55e-02	1.1445±9.94e-02	1.1122±7.53e-02	1.0962±9.37e-02	**1.0909±8.35e-02**
	10,20	**0.9770±5.67e-02**	0.9871±4.13e-02	1.0582±1.07e-01	1.0046±3.55e-02	0.9873±5.50e-02	0.9918±7.47e-02

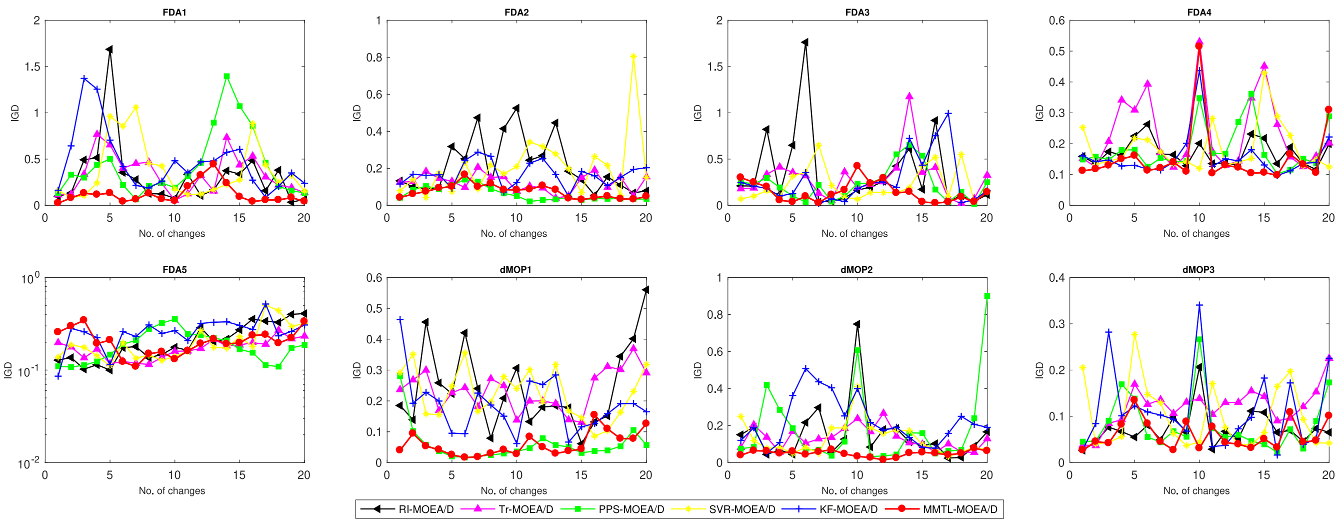


Fig. 3. IGD values of DMOAs at the configuration ($\tau_t = 10$ and $n_t = 10$).

change as well) DMOPs and meet the Non-IID problems. The statistical results of MIGD, MS, and SP are shown in Table IV in the supplementary material. It can be observed that in 20 out of 42 cases, MMTL-MOEA/D performs the best among all the algorithms, eight for RI-MOEA/D, six for KF-MOEA/D, four for SVR-MOEA/D, three for Tr-MOEA/D, and only one for PPS-MOEA/D. The comparative studies on DF test functions can also verify the effectiveness of our proposed MMTL-DMOEA in solving Non-IID problems.

From the experimental results, it is obvious that MMTL-MOEA/D performs not very well on the FDA5 problem. FDA5 is triobjective problem and the POS and the POF change as well, so FDA5 has more complicated POF and POS. Hence, clustering solutions FDA5 into low-dimensional segmented manifolds meets challenges and leads to a poor performance of *Transfer*.

B. Ablation Study

The main contribution of MMTL-DMOEA is to use the optimal solution information obtained from the previous moment to generate the initial population at the next moment. Specifically, our method produces the initial population by two different mechanisms: 1) memory-based one and 2) TL-based one. In order to verify the effectiveness of these two mechanisms, we designed an ablation experiment.

In this experiment, we propose two modified algorithms by using only one out of the two mechanisms mentioned above. For the cases in which the memory or the TL mechanism alone generates all individuals of the initial population, we respectively refer the case as MMTL-MOEA/D$_M$ or MMTL-MOEA/D$_T$. During the operation of the algorithm, if the number of the predicted initial individuals less than the population size, the MMTL-MOEA/D$_M$ and

TABLE III
MEAN AND STANDARD DEVIATION VALUES OF MS METRIC OBTAINED BY COMPARED ALGORITHMS FOR DIFFERENT DYNAMIC TEST SETTINGS

Problems	n_t, τ_t	MMTL-MOEA/D	KF-MOEA/D	PPS-MOEA/D	SVR-MOEA/D	Tr-MOEA/D	RI-MOEA/D
FDA1	5,10	**0.8545±1.19e-01**	0.5412±1.43e-01	0.7602±1.35e-01	0.7568±1.19e-01	0.6482±1.84e-01	0.8086±1.31e-01
	10,10	**0.9291±7.05e-02**	0.5571±1.15e-01	0.7468±1.74e-01	0.8121±1.26e-01	0.7337±1.48e-01	0.7614±1.27e-01
	10,20	**0.9445±6.64e-02**	0.6429±1.39e-01	0.8563±1.35e-01	0.8179±1.16e-01	0.8042±1.10e-01	0.8720±1.26e-01
FDA2	5,10	**0.9524±8.39e-02**	0.6348±1.95e-01	0.8203±1.69e-01	0.7198±2.74e-01	0.7157±1.08e-01	0.6596±3.24e-01
	10,10	**0.9640±6.84e-02**	0.5737±1.78e-01	0.8322±2.15e-01	0.6585±2.93e-01	0.7168±9.03e-02	0.6995±2.66e-01
	10,20	**0.9819±3.43e-02**	0.6208±8.74e-02	0.8915±1.45e-01	0.6999±3.38e-01	0.8199±1.47e-01	0.7058±3.02e-01
FDA3	5,10	**0.8942±1.09e-01**	0.6000±1.05e-01	0.7471±1.85e-01	0.7546±1.22e-01	0.6545±1.85e-01	0.6960±1.65e-01
	10,10	**0.9070±1.00e-01**	0.6082±1.07e-01	0.7242±2.47e-01	0.6441±2.05e-01	0.7197±1.94e-01	0.7095±1.67e-01
	10,20	**0.8560±1.56e-01**	0.6325±1.32e-01	0.8227±1.33e-01	0.7055±1.50e-01	0.7106±1.52e-01	0.7549±1.62e-01
FDA4	5,10	**1.0000±00**	0.9931±1.73e-02	0.9994±2.04e-03	1.0000±00	0.9998±8.19e-04	0.9999±4.72e-04
	10,10	**1.0000±00**	0.9891±2.35e-02	0.9980±7.63e-03	1.0000±00	0.9984±6.94e-03	0.9956±1.43e-02
	10,20	**1.0000±00**	0.9848±4.63e-02	0.9992±2.78e-03	1.0000±00	1.0000±00	1.0000±00
FDA5	5,10	0.8682±2.32e-01	0.9554±8.44e-02	0.8788±1.71e-01	0.9897±2.46e-02	0.9931±1.91e-02	**0.9968±9.67e-03**
	10,10	0.8875±1.18e-01	0.9367±1.12e-01	0.8743±1.53e-01	**1.0000±6.94e-05**	0.9869±3.39e-02	0.9996±1.49e-03
	10,20	0.9619±6.76e-02	0.9897±2.81e-02	0.9170±9.40e-02	**0.9964±1.06e-02**	0.9720±6.64e-02	0.9934±2.47e-02
dMOP1	5,10	**0.9916±5.27e-03**	0.5678±1.06e-01	0.8241±1.15e-01	0.7656±1.20e-01	0.7595±4.93e-02	0.7153±1.38e-01
	10,10	**0.9838±2.43e-02**	0.5514±1.52e-01	0.8733±1.19e-01	0.7172±1.62e-01	0.7614±6.96e-02	0.7282±1.39e-01
	10,20	**0.9929±1.46e-02**	0.6205±1.85e-01	0.9232±1.38e-01	0.8179±1.00e-01	0.8740±7.89e-02	0.8801±5.57e-02
dMOP2	5,10	**0.9415±3.47e-02**	0.5682±1.84e-01	0.8692±7.59e-02	0.7930±1.33e-01	0.7668±1.03e-01	0.8484±9.33e-02
	10,10	**0.9142±6.54e-02**	0.6223±2.07e-01	0.8569±8.50e-02	0.8355±9.56e-02	0.8283±9.46e-02	0.8113±1.27e-01
	10,20	**0.9783±1.78e-02**	0.6874±1.91e-01	0.8929±8.85e-02	0.9194±8.75e-02	0.8831±7.66e-02	0.9047±6.06e-02
dMOP3	5,10	**0.9614±4.35e-02**	0.7634±1.59e-01	0.8494±1.04e-01	0.8383±1.14e-01	0.8215±8.86e-02	0.8888±7.76e-02
	10,10	**0.9447±4.33e-02**	0.7123±1.29e-01	0.8704±9.58e-02	0.8564±8.92e-02	0.8494±8.67e-02	0.8903±8.60e-02
	10,20	**0.9669±3.50e-02**	0.7934±1.25e-01	0.8314±1.01e-01	0.8821±1.26e-01	0.8952±7.91e-02	0.9439±4.66e-02

TABLE IV
MEAN AND STANDARD DEVIATION VALUES OF MIGD, SP, AND MS METRICS OBTAINED BY MMTL-MOEA/D, MMTL-MOEA/D$_M$, MMTL-MOEA/D$_T$, AND RI-MOEA/D

Problems	Metrics	MMTL-MOEA/D	MMTL-MOEA/D$_M$	MMTL-MOEA/D$_T$	RI-MOEA/D
FDA1	MIGD	**0.1199±7.93e-02**	0.1397±7.02e-02	0.1311±8.59e-02	0.2733±1.83e-01
	SP	1.1239±1.01e-01	**1.1146±9.26e-02**	1.1167±1.04e-01	1.1481±1.33e-01
	MS	**0.9291±7.05e-02**	0.9177±6.23e-02	0.8574±1.19e-01	0.7614±1.27e-01
FDA2	MIGD	**0.0842±3.34e-02**	0.0993±4.22e-02	0.0870±3.10e-02	0.2528±1.34e-01
	SP	1.0841±8.77e-02	1.1392±1.99e-01	**1.0238±7.36e-02**	1.0650±7.38e-02
	MS	**0.9640±6.84e-02**	0.9589±4.92e-02	0.8751±1.03e-01	0.6995±2.66e-01
FDA3	MIGD	**0.0914±9.77e-02**	0.1115±6.93e-02	0.1845±1.56e-01	0.2530±3.05e-01
	SP	1.1273±7.70e-02	**1.1196±1.60e-01**	1.1445±1.60e-01	1.1201±1.41e-01
	MS	**0.9070±1.00e-01**	0.8716±1.19e-01	0.8064±1.76e-01	0.7095±1.67e-01
FDA4	MIGD	0.1594±5.77e-02	**0.1453±3.33e-02**	0.1613±4.15e-02	0.1787±8.33e-02
	SP	**0.8281±1.10e-01**	0.8290±8.37e-02	0.8327±1.10e-01	0.8978±1.01e-01
	MS	**1.0000±00**	0.9985±4.92e-03	0.9994±2.68e-03	0.9956±1.43e-02
FDA5	MIGD	**0.1892±5.19e-02**	0.2018±6.19e-02	0.1989±9.53e-02	0.2140±1.01e-01
	SP	**1.0400±7.64e-02**	1.1186±1.26e-01	1.1020±1.44e-01	1.1413±1.37e-01
	MS	0.8875±1.18e-01	0.9633±7.57e-02	0.9887±4.04e-02	**0.9996±1.49e-03**
dMOP1	MIGD	0.0543±5.52e-02	**0.0382±1.64e-02**	0.0392±1.64e-02	0.2734±1.46e-01
	SP	1.0554±1.25e-01	0.9732±7.57e-02	**0.9468±8.26e-02**	1.0480±4.35e-02
	MS	**0.9838±2.43e-02**	0.9809±9.34e-03	0.9214±8.65e-02	0.7282±1.39e-01
dMOP2	MIGD	**0.0717±4.20e-02**	0.0750±3.50e-02	0.0817±5.60e-02	0.1586±1.33e-01
	SP	**0.9988±1.05e-01**	1.0432±9.98e-02	1.0432±7.32e-02	1.0430±5.52e-02
	MS	0.9142±6.54e-02	**0.9366±3.72e-02**	0.8883±8.49e-02	0.8113±1.27e-01
dMOP3	MIGD	0.0683±4.26e-02	**0.0564±2.20e-02**	0.0586±4.00e-02	0.0850±5.68e-02
	SP	1.1254±1.09e-01	**1.0620±7.89e-02**	1.1142±1.09e-01	1.0909±8.35e-02
	MS	**0.9447±4.33e-02**	0.9417±5.46e-02	0.9285±4.59e-02	0.8903±8.60e-02

MMTL-MOEA/D$_T$ both add random individuals to replenish the population so as to ensure sufficient diversity of the initial population.

We compare these two algorithms with MMTL-MOEA/D and the baseline algorithm RI-MOEA/D. The settings of experimental parameters are consistent with the above experiments, and the FDA series and dMOP series test functions are used.

Table IV lists the average MIGD, MS, and SP metric values for the experiment. Comparing the three metrics of the MOEA/D, the MMTL-MOEA/D$_M$, and the MMTL-MOEA/D$_T$, we can find that when the initial population of the next moment is generated using the TL or the memory-based mechanism, the metric of MIGD is significantly improved, compared with the RI-MOEA/D algorithm. It shows that when the problem environment changes, the two mechanisms can adapt the evolution algorithm to the new environment faster and eventually improve the convergence and diversity of the solution.

At the same time, the comparison among the three columns of MMTL-MOEA/D, MMTL-MOEA/D$_M$, and MMTL-MOEA/D$_T$ show that the combination of the two

TABLE V
RUNNING TIME OF TWO ALGORITHMS ON DIFFERENT BENCHMARK FUNCTIONS (UNIT: SECONDS)

	FDA1	FDA2	FDA3	FDA4	FDA5	dMOP1	dMOP2	dMOP3
Tr-MOEA/D	42.54	45.27	57.71	132.59	115.52	80.23	73.53	75.55
MMTL-MOEA/D	6.23	5.4	5.09	10.06	9.94	7.05	7.74	5.63
Increasing Rate	682.83%	838.33%	1133.79%	1317.99%	1162.17%	1138.01%	950.00%	1341.92%

TABLE VI
RUNNING TIME OF DMOAS ON DIFFERENT FUNCTIONS (UNIT: SECONDS)

	MMTL-MOEA/D	KF-MOEA/D	PPS-MOEA/D	SVR-MOEA/D	Tr-MOEA/D
FDA1	5.83	3.55	3.11	4.85	42.54
FDA2	5.40	3.46	3.08	4.63	45.27
FDA3	5.09	3.92	3.64	4.78	57.71
FDA4	10.06	9.78	13.32	9.66	132.59
FDA5	9.94	10.24	14.83	11.52	115.52
dMOP1	7.05	7.25	8.96	8.34	80.23
dMOP2	7.74	4.02	4.93	4.47	73.53
dMOP3	5.63	3.52	4.15	4.61	75.55

mechanisms is superior to the use of only one mechanism. One thing we need to point out is that in this experiment, the MMTL-MOEA/D$_M$ algorithm is superior to the MMTL-MOEA/D$_T$ algorithm.

We think one of the reasons for this difference is that the MMTL-MOEA/D$_T$ is more sensitive to parameters, and it is possible to improve the performance of the MMTL-MOEA/D$_T$ by setting reasonable parameters, but how to find the optimal parameters is a tough problem.

C. Compared With Tr-DMOEA

The MMTL-DMOEA stems from the framework of the Tr-DMOEA, therefore we want to prove that MMTL-DMOEA outperforms the original algorithm in terms of the evolution speed and convergence.

In order to compare evolution speed, we recorded the running time of the proposed Tr-MOEA/D and MMTL-MOEA/D under the same hardware configuration.[1] Table V shows the mean running time of each environment of Tr-MOEA/D and MMTL-MOEA/D in solving the FDA series and dMOP series test problems. The experimental results show that the proposed method has greatly improved the speed of operation, and the acceleration rate is hundreds of times.

Two reasons are accountable for this performance improvement. First, the TL method in the proposed method spent much less time than the Tr-MOEA/D algorithm. Second, MMTL-MOEA/D directly predicts the optimal solutions in the decision space, while the Tr-MOEA/D method uses the POF to make the prediction. After locating the individual in the objectives space, Tr-MOEA/D needs extra time to find the corresponding individuals in the decision space. To ensure the fairness of the comparison, we use the same settings mentioned above to conduct the experiment.

[1]The implementation environment is as follows: 1.6-GHz Intel Core i5, 8-GB 1600-MHz DDR3.

Tables I–IV in the supplementary material record the MIGD, SP, and MS values of the Tr-MOEA/D and MMTL-MOEA/D on FDA, dMOP, and DF problems. The above experimental results show that the manifold transfer-based population prediction is effective in solving DMOPs. The reasons for the superior performance of the proposed MMTL-MOEA/D are two-fold. First, it compresses high-dimensional variables in a low-dimensional space, so MMTL-MOEA/D is more effective; second, the proposed design can transfer information gradually over geodesic flow so that it can improve its adaptability to solution space under the new environment. Besides, from the previous ablation study, the memory-based mechanism and the TL-based mechanism can both help improve performance, so the combination of these two mechanisms can achieve better performance in solving DMOPs.

Both MMTL-MOEA/D and Tr-MOEA/D are prediction methods based on TL. Compared with Tr-MOEA/D, one of the differences is that Tr-MOEA/D requires a significant amount of computation to obtain latent space to find optimal solutions, while our MMTL-MOEA/D directly predicts the optimal solutions in the decision space. Therefore, MMTL-MOEA/D takes less time but achieves better performance.

D. Running Time Cost

The good performance of MMTL-MOEA/D comes with a small price. We compare the mean running time cost of each environment of different algorithms and present the results in Table VI. As shown in Table VI, KF-MOEA/D and PPS-MOEA/D consume less time than the other compared algorithms. MMTL-MOEA/D is faster than the other compared algorithms on FDA5 and dMOP1 test functions. In other functions, MMTL-MOEA/D is only 1–2 s slower than PPS-MOEA/D or KF-MOEA/D. But considering the good performance of MMTL-MOEA/D in MIGD, SP and MS metrics, MMTL-MOEA/D is still very competitive as compared to other algorithms.

V. Conclusion

TL-based methods have shown their promising applications in solving DMOPs. However, existing methods are faced with various challenges particularly on the speed of convergence. To overcome the issue, a memory-driven manifold TL algorithm, called MMTL-DMOEA, has been proposed. The MMTL-DMOEA combines the manifold-based TL algorithm with a memory-based prediction method to generate the initial population during the evolution. We conducted systematic experiments and showed that the solutions obtained by the proposed MMTL-DMOEA are better than state-of-the-art algorithms in terms of convergence, diversity, uniformity, and convergence speed in particular. The integration of both memory and manifold TL can retain the effective strategy of elitism in traditional EA, while at the same time making full use of the advantages in TL.

References

[1] R. Azzouz, S. Bechikh, and L. B. Said, "Dynamic multi-objective optimization using evolutionary algorithms: A survey," in *Recent Advances in Evolutionary Multi-Objective Optimization*. Cham, Switzerland: Springer, 2017, pp. 31–70.

[2] Z. Yang, Y. Jin, and K. Hao, "A bio-inspired self-learning coevolutionary dynamic multiobjective optimization algorithm for Internet of Things services," *IEEE Trans. Evol. Comput.*, vol. 23, no. 4, pp. 675–688, Aug. 2019.

[3] J. Xu, P. B. Luh, F. B. White, E. Ni, and K. Kasiviswanathan, "Power portfolio optimization in deregulated electricity markets with risk management," *IEEE Trans. Power Syst.*, vol. 21, no. 4, pp. 1653–1662, Nov. 2006.

[4] A. Simões and E. Costa, "Prediction in evolutionary algorithms for dynamic environments," *Soft Comput*, vol. 18, no. 8, pp. 1471–1497, Oct. 2013.

[5] C. Rossi, M. Abderrahim, and J. C. Díaz, "Tracking moving optima using Kalman-based predictions," *Evol. Comput.*, vol. 16, no. 1, pp. 1–30, Mar. 2008.

[6] M. Jiang, Z. Huang, L. Qiu, W. Huang, and G. G. Yen, "Transfer learning-based dynamic multiobjective optimization algorithms," *IEEE Trans. Evol. Comput.*, vol. 22, no. 4, pp. 501–514, Aug. 2018.

[7] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.

[8] T. Lin and H. Zha, "Riemannian manifold learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 5, pp. 796–809, May 2008.

[9] R. Gopalan, R. Li, and R. Chellappa, "Domain adaptation for object recognition: An unsupervised approach," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, 2011, pp. 999–1006.

[10] K. Deb, U. N. B. Rao, and S. Karthik, "Dynamic multi-objective optimization and decision-making using modified NSGA-II: A case study on hydro-thermal power scheduling," in *Proc. Int. Conf. Evol. Multi Crit. Optim.*, 2007, pp. 803–817.

[11] Q. Chen, J. Ding, S. Yang, and T. Chai, "A novel evolutionary algorithm for dynamic constrained multiobjective optimization problems," *IEEE Trans. Evol. Comput.*, early access, Dec. 6, 2019, doi: 10.1109/TEVC.2019.2958075.

[12] G. Ruan, G. Yu, J. Zheng, J. Zou, and S. Yang, "The effect of diversity maintenance on prediction in dynamic multi-objective optimization," *Appl. Soft Comput.*, vol. 58, pp. 631–647, Sep. 2017.

[13] S. Jiang and S. Yang, "A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 21, no. 1, pp. 65–82, Feb. 2017.

[14] C.-K. Goh and K. C. Tan, "A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 1, pp. 103–127, Feb. 2009.

[15] D. Yazdani, M. N. Omidvar, J. Branke, T. T. Nguyen, and X. Yao, "Scaling up dynamic optimization problems: A divide-and-conquer approach," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 1–15, Feb. 2020.

[16] R. Chen, K. Li, and X. Yao, "Dynamic multiobjectives optimization with a changing number of objectives," *IEEE Trans. Evol. Comput.*, vol. 22, no. 1, pp. 157–171, Feb. 2018.

[17] D. Gong, B. Xu, Y. Zhang, Y. Guo, and S. Yang, "A similarity-based cooperative co-evolutionary algorithm for dynamic interval multi-objective optimization problems," *IEEE Trans. Evol. Comput.*, vol. 24, no. 1, pp. 142–156, Feb. 2020.

[18] Y. Wang and B. Li, "Multi-strategy ensemble evolutionary algorithm for dynamic multi-objective optimization," *Memetic Comput.*, vol. 2, no. 1, pp. 3–24, 2010.

[19] M. Rong, D. Gong, Y. Zhang, Y. Jin, and W. Pedrycz, "Multidirectional prediction approach for dynamic multiobjective optimization problems," *IEEE Trans. Cybern.*, vol. 49, no. 9, pp. 3362–3374, Sep. 2019.

[20] W. T. Koo, C. K. Goh, and K. C. Tan, "A predictive gradient strategy for multiobjective evolutionary algorithms in a fast changing environment," *Memetic Comput.*, vol. 2, no. 2, pp. 87–110, 2010.

[21] M. Rong, D. Gong, W. Pedrycz, and L. Wang, "A multi-model prediction method for dynamic multi-objective evolutionary optimization," *IEEE Trans. Evol. Comput.*, vol. 24, no. 2, pp. 290–304, Apr. 2020.

[22] A. Zhou, Y. Jin, and Q. Zhang, "A population prediction strategy for evolutionary dynamic multiobjective optimization," *IEEE Trans. Cybern.*, vol. 44, no. 1, pp. 40–53, Jan. 2014.

[23] A. Muruganantham, K. C. Tan, and P. Vadakkepat, "Evolutionary dynamic multiobjective optimization via Kalman filter prediction," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 2862–2873, Dec. 2016.

[24] 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.

[25] J. Zou, Q. Li, S. Yang, H. Bai, and J. Zheng, "A prediction strategy based on center points and knee points for evolutionary dynamic multi-objective optimization," *Appl. Soft Comput.*, vol. 61, pp. 806–818, Dec. 2018.

[26] B. Xu, Y. Zhang, D. Gong, Y. Guo, and M. Rong, "Environment sensitivity-based cooperative co-evolutionary algorithms for dynamic multi-objective optimization," *IEEE/ACM Trans. Comput. Biol. Bioinformat.*, vol. 15, no. 6, pp. 1877–1890, Nov. 2018.

[27] Y. Woldesenbet and G. Yen, "Dynamic evolutionary algorithm with variable relocation," *IEEE Trans. Evol. Comput.*, vol. 13, no. 3, pp. 500–513, Jun. 2009.

[28] A. Gupta, Y. 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.

[29] L. Feng *et al.*, "Evolutionary multitasking via explicit autoencoding," *IEEE Trans. Cybern.*, vol. 49, no. 9, pp. 3457–3470, Sep. 2019.

[30] L. Zhou *et al.*, "Toward adaptive knowledge transfer in multifactorial evolutionary computation," *IEEE Trans. Cybern.*, early access, Mar. 6, 2020, doi: 10.1109/TCYB.2020.2974100.

[31] K. K. Bali, Y. 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.

[32] B. Da, A. Gupta, and Y. Ong, "Curbing negative influences online for seamless transfer evolutionary optimization," *IEEE Trans. Cybern.*, vol. 49, no. 12, pp. 4365–4378, Dec. 2019.

[33] W. Zhou *et al.*, "A preliminary study of adaptive indicator based evolutionary algorithm for dynamic multiobjective optimization via autoencoding," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2018, pp. 1–7.

[34] A. T. W. Min, Y. Ong, A. Gupta, and C. Goh, "Multiproblem surrogates: Transfer evolutionary multiobjective optimization of computationally expensive problems," *IEEE Trans. Evol. Comput.*, vol. 23, no. 1, pp. 15–28, Feb. 2019.

[35] X.-F. Liu *et al.*, "Neural network-based information transfer for dynamic optimization," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jul. 19, 2019, doi: 10.1109/TNNLS.2019.2920887.

[36] M. Jiang, L. Qiu, Z. Huang, and G. G. Yen, "Dynamic multi-objective estimation of distribution algorithm based on domain adaptation and nonparametric estimation," *Inf. Sci.*, vol. 435, pp. 203–223, Apr. 2018.

[37] Q. Zhang, A. Zhou, and Y. Jin, "RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm," *IEEE Trans. Evol. Comput.*, vol. 12, no. 1, pp. 41–63, Feb. 2008.

[38] M. Farina, K. Deb, and P. Amato, "Dynamic multiobjective optimization problems: Test cases, approximations, and applications," *IEEE Trans. Evol. Comput.*, vol. 8, no. 5, pp. 425–442, Oct. 2004.

[39] S. Jiang, S. Yang, X. Yao, K. Tan, M. Kaiser, and N. Krasnogor, "Benchmark problems for cec2018 competition on dynamic multiobjective optimisation," in *Proc. CEC2018 Competition*, 2018, pp. 1–8.

[40] L. Cao, L. Xu, E. D. Goodman, C. Bao, and S. Zhu, "Evolutionary dynamic multiobjective optimization assisted by a support vector regression predictor," *IEEE Trans. Evol. Comput.*, vol. 24, no. 2, pp. 305–319, Apr. 2020.

[41] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 1–27, May 2011.

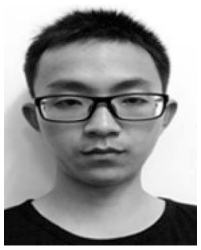
Min Jiang (Senior Member, IEEE) received the bachelor's and Ph.D. degrees in computer science from Wuhan University, Wuhan, China, in 2001 and 2007, respectively.

He was a Postdoctoral Fellow with the Department of Mathematics, Xiamen University, Xiamen, China, where he is currently a Professor with the Department of Artificial Intelligence. His main research interests are machine learning, computational intelligence, and robotics. He has a special interest in dynamic multiobjective optimization, transfer learning, and the software development and in the basic theories of robotics.

Prof. Jiang received the Outstanding Reviewer Award from the IEEE TRANSACTIONS ON CYBERNETICS in 2016. He is the Chair of the IEEE CIS Xiamen Chapter. He is currently serving as an Associate Editor for the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS and the IEEE TRANSACTIONS ON COGNITIVE AND DEVELOPMENTAL SYSTEMS.

Zhenzhong Wang received the bachelor's degree in computer science and technology from Northeastern University, Shenyang, China, in 2017. He is currently pursuing the master's degree with the School of Informatics, Xiamen University, Xiamen, China.

His research interests include computational intelligence and machine learning.



Liming Qiu received the bachelor's degree from Fujian Agriculture and Forestry University, Fuzhou, China, in 2015, and the master's degree from the School of Informatics, Xiamen University, Xiamen, China, in 2018.

He is currently an Algorithm Engineer with LenzTech, Beijing, China. His research interests include computational intelligence and machine learning.

Shihui Guo (Member, IEEE) received the B.S. degree in electrical engineering from Peking University, Beijing, China, in 2010, and the Ph.D. degree in computer animation from the National Centre for Computer Animation, Bournemouth University, Poole, U.K., in 2015.

He is an Associate Professor with the School of Informatics, Xiamen University, Xiamen, China. His research interests include computational intelligence and human–computer interaction.

Xing Gao (Member, IEEE) received the Ph.D. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 2009.

He is currently an Associate Professor with the School of Informatics, Xiamen University, Xiamen, China. His research interests include computing intelligence and computer graphics.

Kay Chen Tan (Fellow, IEEE) received the B.Eng. degree (First Class Hons.) in electronics and electrical engineering and the Ph.D. degree in evolutionary computation and control systems from the University of Glasgow, Glasgow, U.K., in 1994 and 1997, respectively.

He is a Full Professor with the Department of Computer Science, City University of Hong Kong, Hong Kong. He has published over 200 refereed articles and six books.

Prof. Tan is the Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, was the Editor-in-Chief of the *IEEE Computational Intelligence Magazine* from 2010 to 2013, and currently serves as the Editorial Board Member of over ten journals. He was an Elected Member of IEEE CIS AdCom from 2017 to 2019.