Multiple source transfer learning for dynamic multiobjective optimization⋆

Yulong Ye[a], Qiuzhen Lin[a,*], Lijia Ma[a], Ka-Chun Wong[b], Maoguo Gong[c] and Carlos A. Coello Coello[d,e]

[a]College of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China
[b]Department of Computer Science, City University of Hong Kong, Hong Kong, China
[c]School of Electronic Engineering, Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, International Research Center for Intelligent Perception and Computation, Xidian University, No. 2 South TaiBai Road, Xi'an 710071, China
[d]CINVESTAV-IPN, Department of Computer Science, Mexico, D.F. 07360, Mexico
[e]Basque Center for Applied Mathematics (BCAM) & Ikerbasque, Spain

ARTICLE INFO

Keywords:
Dynamic optimization
Domain adaptation
Transfer learning
Evolutionary algorithm
Multiobjective optimization

ABSTRACT

Recently, dynamic multiobjective evolutionary algorithms (DMOEAs) with transfer learning have become popular for solving dynamic multiobjective optimization problems (DMOPs), as the used transfer learning methods in DMOEAs can effectively generate a good initial population for the new environment. However, most of them only transfer non-dominated solutions from the previous one or two environments, which cannot fully exploit all historical information and may easily induce negative transfer as only limited knowledge is available. To address this problem, this paper presents a multiple source transfer learning method for DMOEA, called MSTL-DMOEA, which runs two transfer learning procedures to fully exploit the historical information from all previous environments. First, to select some representative solutions for knowledge transfer, one clustering-based manifold transfer learning is run to cluster non-dominated solutions of the last environment to obtain their centroids, which are then fed into the manifold transfer learning model to predict the corresponding centroids for the new environment. After that, multiple source transfer learning is further run by using multisource TrAdaboost, which can fully exploit information from the above centroids in new environment and old centroids from all previous environments, aiming to construct a more accurate prediction model. This way, MSTL-DMOEA can predict an initial population with better quality for the new environment. The experimental results also validate the superiority of MSTL-DMOEA over several competitive state-of-the-art DMOEAs in solving various kinds of DMOPs.

1. Introduction

Dynamic multiobjective optimization problems (DMOPs) contain multiple (often conflicting) objectives that are changed over time. Various application problems in engineering, economics, and industry can be modeled as DMOPs [3]. For example, in the energy-saving and environmental optimization problem [33], the total energy should be maximized while the pollution emissions in power generation should be minimized, with the dynamically changed total power demand over time.

In recent years, a number of dynamic multiobjective evolutionary algorithms (DMOEAs) have been proposed, which can be classified into three main kinds, i.e., maintaining-diversity-based DMOEAs [22, 6, 30], memory-based DMOEAs [2, 28], and prediction-based DMOEAs [45, 1, 37]. Specifically, the maintaining-diversity-based DMOEAs attempt to improve the population's diversity in a new environment by introducing more randomly generated solutions or running more mutations with high probability, which can help to avoid getting trapped in a local optimum. However, too much diversity added into the population will be equivalent to restarting the optimization with little historical information used, while too little diversity added has less effect to speed up the convergence in a new environment. The memory-based DMOEAs usually preserve the approximate Pareto-optimal sets (POSs) in all historical environments and reuse one of them as an initial population for the new environment when a similar environmental change is detected. They perform very well for tracking DMOPs with periodic changes [32], but become not so effective when similar changes do not appear again, due to the lack of capability to predict the change tendency. The prediction-based

*Corresponding author
ORCID(s):

DMOEAs will learn prediction models from historical environments, which can handle various dynamic changes in DMOPs by predicting their change tendency. They have been validated to perform better than other types of DMOEAs for solving MOPs with more kinds of dynamic changes [43, 50]. However, in most prediction-based DMOEAs, the historical solutions used to construct prediction models are assumed to obey an independent identical distribution (IID), which may not always hold true in some practical cases.

Recently, some prediction-based DMOEAs using the idea of transfer learning have been proposed and shown promising performance for solving various kinds of DMOPs [18, 17, 48, 13]. These DMOEAs can release the IID condition for solutions in historical environments and allow the distributions of data used in training and testing to be different [34]. In Tr-DMOEA [17], a framework based on transfer learning was proposed, which reuses the search experiences of approximate POF in objective space to generate a high-quality initial population for guiding the evolutionary search in the new environment. Moreover, the memory mechanism is combined with manifold transfer learning in MMTL-DMOEA [21] to predict promising solutions for the new environment. In KT-DMOEA [20], a small number of high-quality solutions (knee points) are integrated with the imbalance transfer learning technique to improve the computational efficiency. In IT-DMOEA [19], a pre-search strategy is designed to reduce the possibility of negative transfer, and in AE-DMOEA [13], an autoencoder is modified to predict the moving of POS based on the nondominated solutions obtained before the dynamic occurs.

These DMOEAs can transfer effective knowledge from historical experience to predict a good initial population for the new environment. However, they still have much room for improving the effect of transfer learning. First, as pointed out in [20], transferring the whole approximate POS from the previous environments will consume a large amount of computational resources and some low-quality solutions may cause negative transfer. Second, most of them only transfer knowledge from the approximate POSs of the previous one or two environments, which may neglect effective knowledge from earlier times. Moreover, due to the very limited knowledge available, negative transfer may easily appear and the prediction models may show poor accuracy as long-term change trends from early times can not be predicted.

To overcome the aforementioned issues, a multiple source transfer learning method is tailored for DMOEA, called MSTL-DMOEA, so that more effective knowledge from all historical environments can be fully transferred, which is our main innovation different from the existing methods [20, 21, 17, 19, 13] that only learn the search experiences from the previous one or two environments. To select some representative solutions for transfer, the approximate POS from each historical environment is clustered to obtain the corresponding centroids, which are used for transferring knowledge to the new environment by using the proposed cluster-based manifold transfer learning (CMTL) and multiple source transfer learning (MSTL). By this way, our method can construct a more accurate prediction model to generate a good initial population for the new environment.

To summarize, the main contributions of this paper are clarified as follows:

1) This paper proposes the CMTL method by using a hierarchical clustering method (HCM) [25] to find the centroids of approximate POS in each environment, which are then fed into the manifold transfer learning (MTL) model to transfer the knowledge to the new environment. Instead of using the whole approximate POS for transferring knowledge to the new environment [17], our method only transfers some representative centroids, which can enhance the transfer effect as validated by our experiments.

2) This paper designs the MSTL method by using multisource TrAdaboost to construct the prediction model, which can transfer knowledge from the centroids of approximate POSs in all historical environments to the new environment. Instead of transferring knowledge from the previous one or two environments to the new environment [17],[20], this method can fully exploit all historical knowledge to construct a more accurate prediction model, without a large demand of computational resources due to the use of the above CMTL.

To validate the performance of MSTL-DMOEA, three suits of test DMOPs (DF suit [24], FDA suit [12], and F suit [49]) are adopted and four competitive DMOEAs (SVR-MOEA/D[4], KT-MOEA/D[20], MMTL-MOEA/D[21], and IT-MOEA/D[19]) are considered for comparison in this paper. The experiments validate the advantages of MSTL-DMOEA for solving various kinds of DMOPs.

The rest of this paper is organized as follows: In Section 2, some basic concepts of DMOPs and some representative DMOEAs are introduced. In Section 3, the details of our proposed algorithm MSTL-DMOEA are described. Section 4 shows the experimental results of MSTL-DMOEA with four competitive DMOEAs on three well-known test suits, while Section 5 gives the conclusions of this paper and discusses some future research directions.

Table 1
The specific mathematical formulations of representative test instances used in this paper

Problems	Objective Functions	Variable Bounds	True PS and PF
DF1	$\begin{cases} f_1(x) = x_1, \\ f_2(x) = g(x)\left(1 - \left(\frac{x_1}{g(x)}\right)^{H(t)}\right), \\ g(x) = 1 + \sum_{i=2}^{n}\left(x_i - G(t)\right)^2, \\ G(t) = \lvert\sin(0.5\pi t)\rvert, H(t) = 1.25 + 0.75\sin(0.5\pi t). \end{cases}$	$x_1, \ldots, x_n \in [0,1]$.	$\text{PS}(t): \begin{array}{l} x_1 \in [0,1], \\ x_{i\neq 1} = G(t). \end{array}$ $\text{PF}(t): \begin{array}{l} f_1 \in [0,1], \\ f_2 = 1 - f_1^{H(t)}. \end{array}$
DF2	$\begin{cases} f_1(x) = x_r, \\ f_2(x) = g(x)\left(1 - \sqrt{f_1/g}\right), \\ g(x) = 1 + \sum_{i=\{1,\ldots,n\}/\{r\}}\left(x_i - G(t)\right)^2, \\ G(t) = \lvert\sin(0.5\pi t)\rvert, r = 1 + \lfloor(n-1)G(t)\rfloor. \end{cases}$	$x_1, \ldots, x_n \in [0,1]$.	$\text{PS}(t): \begin{array}{l} x_r \in [0,1], \\ x_{i\neq r} = G(t). \end{array}$ $\text{PF}(t): \begin{array}{l} f_1 \in [0,1], \\ f_2 = 1 - \sqrt{f_1}. \end{array}$

2. Related background

2.1. Dynamic multiobjective optimization problems

This paper considers DMOPs as time-varying problems. Without loss of generality, the mathematical form of DMOPs can be formulated as follows:

$$\begin{aligned} &\min F(\mathbf{x}, t) = [f_1(\mathbf{x}, t), f_2(\mathbf{x}, t), \ldots, f_m(\mathbf{x}, t)]^T, \\ &\text{s.t. } x \in \Omega, \end{aligned} \tag{1}$$

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

Definition 1: (*Dynamic Decision Vector Domination*) At time t, a decision vector $\mathbf{x}_1$ is said to Pareto dominate another decision vector $\mathbf{x}_2$, denoted by $\mathbf{x}_1 \succ_t \mathbf{x}_2$, if and only if

$$\begin{cases} \forall i = 1, \ldots, m, & f_i\left(\mathbf{x}_1, t\right) \leq f_i\left(\mathbf{x}_2, t\right), \\ \exists j = 1, \ldots, m, & f_j\left(\mathbf{x}_1, t\right) < f_j\left(\mathbf{x}_2, t\right). \end{cases} \tag{2}$$

Definition 2: (*Dynamic Pareto-Optimal Set, DPOS*) If a decision vector $\mathbf{x}^*$ at time t satisfies

$$\mathbf{DPOS}_t = \left\{\mathbf{x}^* \mid \nexists \mathbf{x}, \mathbf{x} \succ_t \mathbf{x}^*\right\}, \tag{3}$$

then $\mathbf{x}^*$ is called a dynamic Pareto-optimal solution, and the set including dynamic Pareto-optimal solutions is called the dynamic POS (DPOS).

Definition 3: (*Dynamic Pareto-Optimal Front, DPOF*) At time t, the dynamic Pareto-optimal front (DPOF) is the corresponding objective vectors of the DPOS, i.e.,

$$\mathbf{DPOF}_t = \left\{F\left(\mathbf{x}^*, t\right) \mid \mathbf{x}^* \in \mathbf{DPOS}_t\right\}. \tag{4}$$

For instances, the specific mathematical formulations of DF1 and DF2 in the IEEE CEC2018 DMOP benchmark [23] are given in Table 1.

2.2. Related works

In recent years, there are a number of DMOEAs designed for solving various kinds of DMOPs. According to the techniques they use, most of them can be classified into three main categories: maintaining-diversity-based DMOEAs, memory-based DMOEAs, and prediction-based DMOEAs.

Maintaining population diversity in DMOEAs is one of the effective approaches to address DMOPs, as this can preserve more diversified decision variables in search space and avoid prematurely falling into local optimal. In [22],

Jiang and Yang reported a steady-state and generational evolutionary algorithm for solving DMOPs, called SGEA. When an environmental change is detected, SGEA merges the outdated solutions with high diversity and a number of solutions generated by the steady-state method as an initial population to quickly adapt to the new environment. In [6], Chen *et al*. presented a novel evolutionary algorithm for solving DMOPs with constraints, in which various operators were designed to handle infeasible solutions and promote the generation of nondominated solutions. In [30], Ma *et al*. introduced a multiregional co-evolutionary algorithm for solving DMOPs, called MRCDMO. This method combines a multi-region prediction strategy and a multi-region diversity maintenance mechanism, which can predict new solutions using different prediction models and obtain diverse individuals within sub-regions. Indeed, maintaining population diversity in DMOEAs helps to solve DMOPs in most cases. However, too much diversity added into the initial population will be equivalent to restarting the optimization with little historical information used, while too little diversity added has less effect to speed up the convergence in the new environment.

The memory-based DMOEAs store the best solutions found from all historical environments and reuse them to initialize the population in a new environment when similar environmental changes are detected. In [2], Azzouz *et al*. presented a DMOEA with a change severity-based adaptive population management strategy, which could adaptively hybridize memory mechanism, random and local search strategies. According to the intensity of the changes, this method can adaptively adjust the number of memory and random solutions. In [28], Liang *et al*. proposed a hybrid strategy of memory and prediction methods for solving DMOPs. In this approach, the memory-based method will reuse the past best solutions to guide the prediction of new locations of POS when a similar change is detected. However, these memory-based DMOEAs show some drawbacks, e.g., the large storage resources and computational resources are required to store and analyze the similarity of historical changes, respectively. Especially, the outstanding solutions stored from the historical environments may become obsolete, which will lead to negative effects in reuse and waste of storage resources.

The prediction-based DMOEAs involve exploiting historical information from the past environments to construct a prediction model, which is capable of predicting an initial population to quickly adapt to the new environment. By this way, this kind of DMOEAs can accommodate the changes in advance. In [49], Zhou *et al*. introduced a population-based prediction strategy (PPS). This method employs an autoregressive model to predict center points by using a sequence of center points from the previous environment and then uses the previous manifolds to estimate the next manifold. In [31], Muruganantham *et al*. presented a DMOEA using Kalman filter prediction. The Kalman filter is used to generate a new initial population once the environmental change is detected, which helps to guide the search for POS in the new environment. In [36], Rong *et al*. proposed a multidirectional prediction approach (MDP) to predict the moving location of POS. This approach divides the population into different groups by using a clustering algorithm and obtains a more diverse population as different groups may have different predicted directions. Meanwhile, the number of clusters is adjusted adaptively according to the severity of environmental change. In [4], Cao *et al*. designed a novel DMOEA with support vector regression (SVR) predictor, called MOEA/D-SVR, which could deal with DMOPs effectively even when the solutions collected in sequential time periods have nonlinear correlations. However, the performance of the SVR predictor is dependent heavily on the quality of historical solutions. In [35], Rong *et al*. presented a multi-model prediction approach (MMP) for DMOPs. This method detects the type of change and then selects an appropriate prediction model to generate an initial population for the new environment. Indeed, these prediction-based DMOEAs can estimate the change trends in different environments and have shown promising performance in solving various kinds of DMOPs [16, 50]. However, most of them assume that the data used to construct prediction models obey the IID condition, which may not always hold true in some practical cases and will affect the prediction accuracy. Thus, it is difficult to obtain the desired results with simple linear prediction methods.

Recently, the prediction-based DMOEAs are extended with the idea of transfer learning, which can release the IID condition and allow the distributions of data used in training and testing to be different. Some representative DMOEAs based on transfer learning are respectively introduced below. In [17], a framework based on transfer learning, called Tr-DMOEA, was proposed to predict an effective initial population for solving DMOPs. The first step of Tr-DMOEA is to find a latent space via the transfer learning method, so that the distributions of solutions at different times will be as similar as possible in this latent space. Then, Tr-DMOEA maps the Pareto-optimal front (POF) of the previous environment into the latent space, which is used to generate a good initial population to search POS in the new environment. However, Tr-DMOEA often requires very large computational resources as it maps the objective value of solutions into higher dimensional spaces. Inspired by Tr-DMOEA, some more efficient transfer learning techniques are embedded into DMOEAs, such as KT-DMOEA [20], MMTL-DMOEA [21], IT-DMOEA [19], and AE-DMOEA [13]. In KT-DMOEA [20], a knee point-based imbalanced transfer learning method, called

KT-DMOEA, was presented for solving DMOPs. Probably because low-quality solutions in the transfer process occupy a lot of computational resources, most of the existing transfer learning-based DMOEAs run slowly. Thus, this method considers knee points as high-quality solutions and only transfers knee points to predict an initial population in the new environment. However, in this approach, the distributions of knee points may also change significantly in different environments, so it is difficult to accurately predict the knee points in a new environment. In MMTL-DMOEA [21], the manifold transfer learning (MTL) model and memory mechanisms were combined to predict an initial population for the new environment. This method uses an archive to store non-dominated solutions found in the past several environments. When an environmental change is detected, partial solutions selected from this archive are regarded as the source domain of the MTL model. Then, solutions that are most similar to the samples of the source domain in the manifold space are predicted from a large number of randomly generated solutions as a predictive population, which are combined with non-dominated solutions from the last environment as an initial population for the new environment. However, transferring a large number of outdated solutions in this method will also increase the probability of negative transfer, thus affecting the prediction accuracy. In IT-DMOEA [19], a pre-search strategy is used to filter out some high-quality solutions with good diversity for transferring, thereby avoiding the negative transfer caused by solution aggregation. Moreover, in AE-DMOEA [13], an autoencoder is modified to generate high-quality solutions by learning the mapping relationships between non-dominated solutions in the previous two environments.

Learning from past experiences [15, 38] has been validated to be effective for generating high-quality initial solutions in the new environment and the transfer learning-based DMOEAs have shown promising performance for solving various DMOPs [17]. However, as clarified in [19], in the process of knowledge transfer, negative transfer can easily occur due to solution aggregation. In addition, in the above transfer learning-based DMOEAs, only the approximate POSs from the previous one or two environments are transferred to predict an initial population for the new environment, which may easily induce negative transfer, as very limited knowledge can be transferred and other knowledge from earlier environments are neglected.

To address the above problems, this paper proposes a multiple source transfer learning method for DMOEA, which uses the CMTL method to transfer diverse knowledge (i.e., the clustering centroids of approximate POS) and employs the MSTL method to fully transfer knowledge from all historical environments. By this way, our method can construct a more accurate prediction model without consuming too much computational cost and shows some advantages over the above DMOEAs for solving various DMOPs.

3. The proposed algorithm

In this section, the details of the proposed MSTL-DMOEA algorithm are introduced. The outline of the proposed algorithm is provided in Fig. 1. Briefly, MSTL-DMOEA includes two main transfer learning procedures: clustering-based manifold transfer learning (CMTL) and multisource transfer learning (MSTL). In CMTL, the HCM [25] method is first used to find clustering centroids from non-dominated solutions in the previous environment, which are considered as superior solutions and fed into the MTL model to predict the transfer centroids in the new environment. In MSTL, the clustering centroids and some random solutions are considered as source domains, while the transfer centroids and other random solutions are considered as the target domain. To fully exploit knowledge from all previous environments, the source domain of each environment will be maintained. Finally, multisource TrAdaboost is used to construct a prediction model by using all the source domains and the target domain, which can predict a high-quality initial population for the new environment. To clearly introduce MSTL-DMOEA, its overall framework is first given in Section 3.1, while the details of CMTL and MSTL are provided in Section 3.2 and Section 3.3, respectively.

3.1. The overall framework

To clarify the running of MSTL-DMOEA, its pseudo-code is given in **Algorithm 1** with the input: the target DMOP. In the initial environment, a set **S** and a set **PS** are both initialized as empty sets, which are used to reserve the source domains and the approximate POSs from all environments, respectively. The environmental variable t is initialized as 0. In line 2, a population **initPop** is initialized by including N randomly generated solutions, where N is the population size. While the termination criterion is not reached in line 3, the following evolutionary process in lines 4-14 is run. At first, $\mathbf{POS}_t$ is found by using the static multiobjective evolutionary algorithm (SMOEA) to obtain the approximate POS at time t in line 4, which is saved into the set **PS** in line 5. If the environment changes in line 6, the environmental variable t is increased by 1 in line 7. Then, the clustering centroids ($\mathbf{C}^{HCM}$) of $\mathbf{POS}_{t-1}$ in the previous environment are found and the corresponding transfer centroids ($\mathbf{C}^{MTL}$) are predicted by CMTL in line 8,

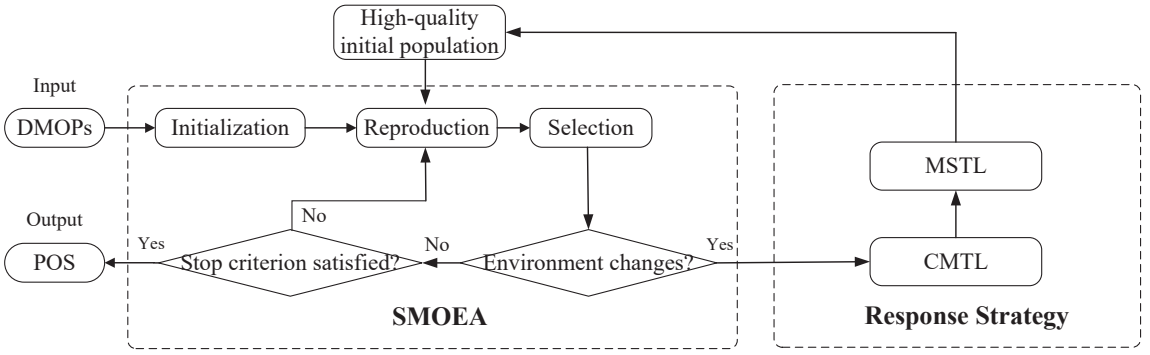

Fig. 1: The outline of the proposed MSTL-DMOEA for solving DMOPs.

Algorithm 1 MSTL-DMOEA

Input: The dynamic optimizaion problem $F_t(x)$.
Output: The approximate POS set **PS** in different moments.

1: $\mathbf{PS} = \emptyset$, $t = 0$, a set of source domains $\mathbf{S} = \emptyset$;
2: Initialize randomly a population **initPop**;
3: **while** termination criterion is not met **do**
4: $\mathbf{POS}_t = \mathbf{SMOEA}(F_t(x), \mathbf{initPop})$;
5: $\mathbf{PS} = \mathbf{PS} \cup \mathbf{POS}_t$;
6: **if** environment changes **then**
7: $t = t + 1$;
8: $(\mathbf{C}^{HCM}, \mathbf{C}^{MTL}) = \mathbf{CMTL}(\mathbf{POS}_{t-1}, F_t(x))$;
9: Randomly generate two populations $\mathbf{P}_t$, $\mathbf{P}_{t-1}$;
10: $\mathbf{X}_{so} = \mathbf{C}^{HCM} \cup \mathbf{P}_{t-1}$;
11: $\mathbf{X}_{ta} = \mathbf{C}^{MTL} \cup \mathbf{P}_t$;
12: $\mathbf{S} = \mathbf{S} \cup \mathbf{X}_{so}$;
13: $\mathbf{initPop} = \mathbf{MSTL}\ (\mathbf{S}, \mathbf{X}_{ta})$;
14: **end if**
15: **end while**

the details of which will be introduced in Section 3.2. Then, two random populations $\mathbf{P}_t$ and $\mathbf{P}_{t-1}$ are generated in line 9 as the poor solution sets at time t and $t-1$. The clustering centroids and one random population $\mathbf{P}_{t-1}$ are merged into the source domain training set $\mathbf{X}_{so}$ in line 10, while the transfer centroids and other random population $\mathbf{P}_t$ are then merged into the target domain training set $\mathbf{X}_{ta}$ in line 11. In order to efficiently transfer knowledge from all historical environments, the source domain at time $t-1$ is added into the set **S** in line 12. Then, in line 13, a new initial population **initPop** for the new environment will be predicted by MSTL, the details of which will be given in Section 3.3. The above evolutionary process in lines 4-14 will be iteratively run until the termination criterion is met in line 3. At last, the approximate POS set **PS** for different environments will be outputted.

3.2. Cluster-based manifold transfer learning (CMTL)

The main idea of CMTL is to obtain the transfer centroids by using HCM and MTL, which can guide the learning of MSTL in the subsequent step. The use of HCM can obtain those solutions with better diversity and thus avoid negative transfer due to solution aggregation [19]. Specifically, HCM is used to cluster non-dominated solutions in approximate POS at time t-1 into several groups and find the centroid of each group. Those centroids are considered as superior solutions and fed into the MTL model, which can save the computational resources and improve the transfer effect. The procedures of CMTL are plotted in Fig. 2.

To clarify the running of CMTL, its pseudo-code is given in **Algorithm 2** with the inputs: the POS of previous environment $\mathbf{POS}_{t-1}$ and the target DMOP $F_t(x)$. In line 1, the subspace dimension d is set to m-1 (m is the number

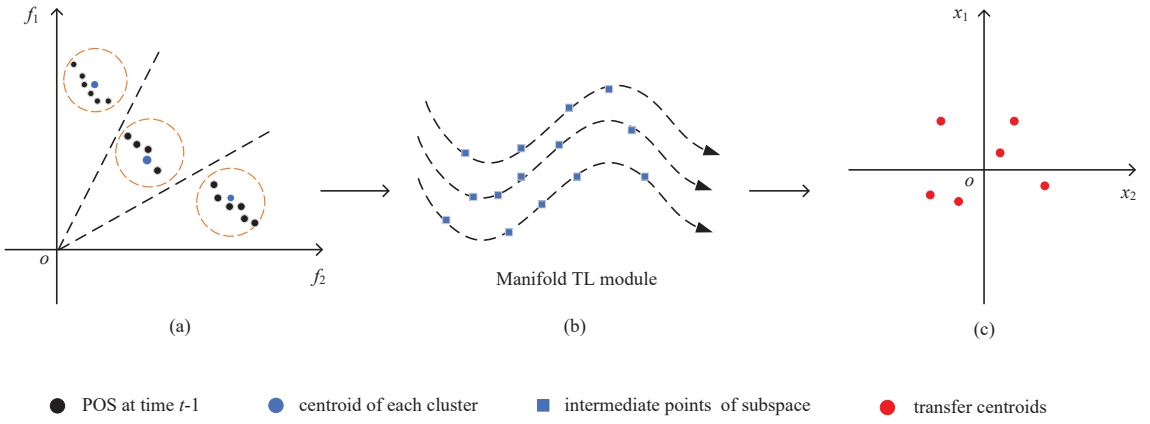

Fig. 2: A simple example of the CMTL method. (a): The black points are the objective values corresponding to non-dominated solutions in the approximate POS at environment t-1, while the blue points are the centroids of different clusters obtained by running HCM on the black points, (b): the intermediate points of subspace in the MTL model, and (c): the transfer centroids are predicted by the MTL for new environment t.

Algorithm 2 CMTL

Input: The approximate POS of last environment $\mathbf{POS}_{t-1}$,
the dynamic optimization function $F_t(x)$.
Output: The set of clustering centroids and transfer centroids:
$\mathbf{C}^{HCM}$,$\mathbf{C}^{MTL}$.

1: Initialization: dimension $d = m$-1, $\mathbf{C}^{HCM}$=∅, $\mathbf{C}^{MTL}$=∅, set the number of random individuals to Q;
2: $\mathbf{C}^{HCM} = \mathbf{HCM}(\mathbf{POS}_{t-1})$;
3: Use PCA for $\mathbf{C}^{HCM}$ to get Ps;
4: Generate a set $\mathbf{T}$ with Q random solutions of $F_t(x)$;
5: Use PCA for $\mathbf{T}$ to get P_T;
6: Construct the geodesic flow by Eq. (5);
7: **for** each $x \in \mathbf{C}^{HCM}$ **do**
8: Map x to $\phi(\cdot)$ and get $\bar{x}$;
9: $x' = \arg\min_{x'} \left\| x'^T \phi(\cdot) - \bar{x} \right\|$;
10: $\mathbf{C}^{MTL} = \mathbf{C}^{MTL} \cup x'$;
11: **end for**

of objectives), the sets of clustering centroids ($\mathbf{C}^{HCM}$) and transfer centroids ($\mathbf{C}^{MTL}$) are initialized as empty set, and the number of random individuals Q used in MTL is set to $2\times N$. Then, in line 2, the set of clustering centroids ($\mathbf{C}^{HCM}$) is obtained by running HCM on non-dominated solutions of approximate POS at time t-1 . These clustering centroids obtained by HCM can be considered as m-1 dimensional manifold [47]. Then, in line 3, principal component analysis (PCA) [26] is employed to obtain the basis (P_S) of $\mathbf{C}^{HCM}$ in the d dimensional subspace, where $P_S \in R^{n\times d}$. Similarly, after generating a set $\mathbf{T}$ including Q random solutions in line 4, the basis (P_T) of these random individuals in the d dimensional subspace can be obtained by PCA in line 5, where $P_T \in R^{n\times d}$. In line 6, the geodesic flow $\phi(\cdot)$ [14] can be constructed by $\Phi(k)$ with $\Phi(0) = P_S$ and $\Phi(1) = P_T$ when $0 < k < 1$, as follows:

$$\Phi(k) = P_S U_1 \Gamma(k) - R_S U_2 \Sigma(k), 0 < k < 1, \tag{5}$$

where U_1 and U_2 are a pair of orthogonal matrices obtained by singular value decomposition (SVD) [39], $R_S \in R^{n\times(n-d)}$ is the orthogonal complement of P_S, $\Gamma(k)$ 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 and P_T), d is the dimension of subspaces, and n is the original dimension of source domain and target domain.

Algorithm 3 HCM

Input: The POS of last environment: $\mathbf{POS}_{t-1}$.
Output: The clustering centroid set: $\mathbf{C}^{HCM}$.
1: Set $p_i \in \mathbf{POS}_{t-1}$ as a cluster C_i and centroid c_i;
2: Initialize the number of cluster N_k, $size = |\mathbf{POS}_{t-1}|$, $\mathbf{C}^{HCM} = \emptyset$;
3: **while** $size > N_k$ **do**
4: Find the two clusters with the smallest angle among all clusters by Eq. (8), noted by C_{t1} and C_{t2};
5: $C_{t1} = C_{t1} \cup C_{t2}$;
6: Remove C_{t2};
7: Update the objective values of new centroid c_{t1} by Eq. (11);
8: $size$--;
9: **end while**
10: **for** $i = 1$ to N_k **do**
11: Obtain centroid c_i of C_i by Eq. (7);
12: $\mathbf{C}^{HCM} = \mathbf{C}^{HCM} \cup c_i$;
13: **end for**

After obtaining the geodesic model, the samples in the source domain and target domain mapped to the geodesic model can be calculated by

$$x_k = x^T \phi(k), k \in (0, 1), \tag{6}$$

where $x \in R^n$ is the original data (the sample of the source or target domain), x^T is the transpose of x, and x_k is the feature data on the intermediate subspace of x corresponding to position k on the geodesic flow. The number of intermediate subspaces k is set as 5 in this paper [21].

Therefore, for each solution x in line 7, the mapped data $\bar{x}$ can be obtained by mapping $x \in \mathrm{C}^{HCM}$ into the geodesic flow in line 8. In the dynamic environment, the source domain and target domain in the manifold space may be similar, so the mapped target domain data $x'^T \phi(\cdot)$ is similar to $\bar{x}$. Consequently, in line 9, a solution x' is found such that the mapped data $x'^T \phi(\cdot)$ on the geodesic flow is close to $\bar{x}$. Then, by iteratively running lines 7-10, a set of transfer centroids can be obtained. Finally, two sets respectively including clustering centroids ($\mathbf{C}^{HCM}$) and transfer centroids ($\mathbf{C}^{MTL}$) will be outputted by the CMTL method.

To further describe the specific process of HCM, the details of HCM are introduced below. At first, HCM initially treats each solution as a cluster and then iteratively merges similar clusters into one cluster. In the used HCM [29], the vector angle of solutions in the objective space is used as an indicator to estimate the similarity between solutions. Here, the centroid c_i of the cluster C_i can be obtained, as follows:

$$c_i = \frac{1}{|C_i|} \sum_{p_i \in C_i} p_i, \tag{7}$$

where $|C_i|$ represents the cardinality of cluster C_i and p_i indicates each solution belonging to cluster C_i. To clarify the running of HCM, its pseudo-code is given in **Algorithm 3**. In line 1, each solution $p_i \in \mathbf{POS}_{t-1}$ is set as a cluster C_i with its centroid c_i ($i = 1, 2, \ldots, |\mathbf{POS}_{t-1}|$). In line 2, the number of clusters N_k is set as 10, $size$ is initialized as $|\mathbf{POS}_{t-1}|$ that indicates the number of clusters at first, and the set of clustering centroids $\mathbf{C}^{HCM}$ is initialized as $\emptyset$. Then, two clusters with the smallest angle among all clusters can be identified by comparing the vector angle between each pair of clusters. Here, the vector angle between two centroids c_i and c_j can be computed by Eq. (8), which is an indicator to evaluate the similarity of two clusters C_i and C_j, as follows:

$$\text{angle}\left(c_i, c_j\right) \triangleq \arccos\left|A_{ij}\right|, \tag{8}$$

where

$$A_{ij} = \frac{\sum_{l=1}^{m} f_l'\left(c_i\right) \cdot f_l'\left(c_j\right)}{\sqrt{\sum_{l=1}^{m} f_l'\left(c_i\right)^2} \cdot \sqrt{\sum_{l=1}^{m} f_l'\left(c_j\right)^2}}, \tag{9}$$

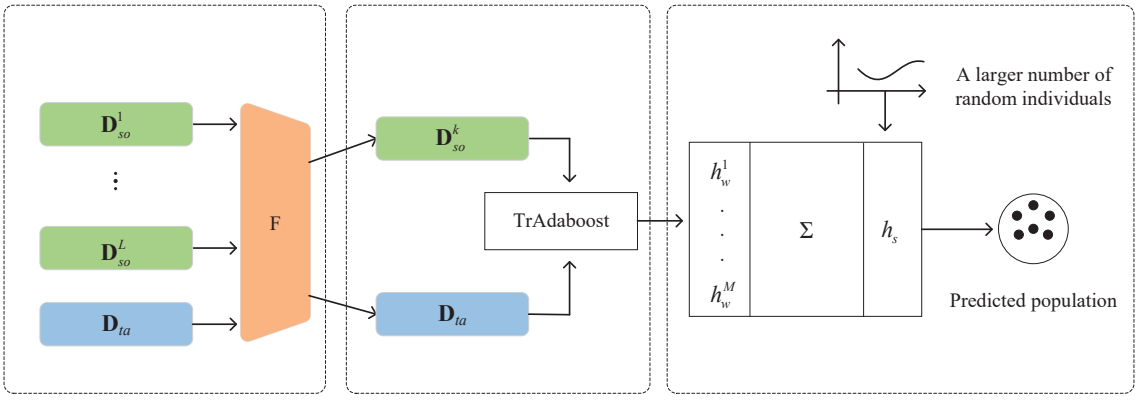

Fig. 3: A flowchart of MSTL, in which (a): a set of source domains and target domain are inputted into a base classifier F, (b): the most similar source domain found by F and the target domain are used to train TrAdaboost, and (c): a number of weak classifiers are combined by a weighted voting mechanism to compose a strong classifier to predict an initial population for the new environment.

and $f_l'(c_i)$ indicates the normalized value of the lth objective value for c_i by

$$f_l'(c_i) = \frac{f_l(c_i) - f_l^{min}}{f_l^{max} - f_l^{min}}, \tag{10}$$

where $l = 1, 2, \ldots, m$, f_l^{max} and f_l^{min} are, respectively, the maximum and minimum values of the lth objective among all solutions in $\mathbf{POS}_{t-1}$. When $size$ is larger than N_k, the procedures of clustering in lines 4-8 will be run iteratively. In line 5, two most similar clusters (C_{t_1} and C_{t_2}) are merged into C_{t1}, and the corresponding value of the new centroid c_{t1} in the objective space is updated by

$$f_i(c_{t1}) = \frac{\sum f_i(p)}{|C_{t1}|}, \tag{11}$$

where $i = 1, 2, \ldots, m$, and each $p \in C_{t1}$. Then, remove the cluster C_{t2} and decrease $size$ by 1 in lines 6 and 8 respectively. After getting the above N_k clusters, the centroid of each cluster by Eq. (7) is obtained iteratively in lines 10-13 and finally a set of clustering centroids ($\mathbf{C}^{HCM}$) is outputted.

3.3. Multisource transfer learning (MSTL)

Since the number of transfer centroids obtained by CMTL are limited, these small number of transfer centroids are not sufficient to guide the population to search POS in the new environment. Thus, MSTL is further run by using the multisource TrAdaboost to improve the accuracy of prediction.

The main idea of TrAdaboost [7] is to establish a weight adjustment mechanism through the Boosting method, which increases the weight of effective data (i.e., target domain) and decreases the weight of invalid data (i.e., source domain). This way, the data in the source domain that are not similar to the samples of the target domain can be filtered out. To do this, TrAdaboost generates a strong classifier by combining several weak classifiers and each weak classifier maps samples $X \in \{\mathbf{X}_{so} \cup \mathbf{X}_{ta}\}$ to a label $Y \in \{-1, 1\}$, where $\mathbf{X}_{so}$ and $\mathbf{X}_{ta}$ refer to the samples in the source domain and target domain, respectively. Indeed, TrAdaboost can effectively utilize the similar distributions in both the source and target domains to help classify the data in the target domain. However, when the distributions of the source domain are not similar to that of the target domain, the accuracy of TrAdaboost will be decreased, which may lead to negative effect on transfer learning.

To solve the above problem, the multisource TrAdaboost [44] is used in this paper, which can effectively reduce the possibility of negative transfer by fully exploiting effective knowledge from all historical environments. In a dynamic environment, the solutions of the previous environments often have some relationship with that of the

Algorithm 4 MSTL

Input: The set of source domain $\mathbf{S}$, target domain $\mathbf{D}_{ta}$.
Output: Predicted population $\mathbf{POP}_{predicted}$.

1: Set L as the size of $\mathbf{S}$;
2: Set the maximal number of iterations M;
3: **for** $i = 1$ to L **do**
4: $\quad \mathbf{D}^i_{so} = \mathbf{S}_i$;
5: $\quad$ Initialize the weight vector w_1 by Eq. (12);
6: $\quad$ Train a weak classifier h^i_w with $(\mathbf{D}^i_{so}, \mathbf{D}_{ta})$ and w_1;
7: $\quad$ Compute the error of h^i_w on $\mathbf{D}_{ta}$ by Eq. (13);
8: **end for**
9: Identify the source domain ($\mathbf{D}^k_{so}$) with the minimal ϵ;
10: Initialize the weight vector w_1 by Eq. (12);
11: **for** $i = 1$ to M **do**
12: $\quad$ Train a weak classifier h^i_w with $(\mathbf{D}^k_{so}, \mathbf{D}_{ta})$ and w_i;
13: $\quad$ Calculate the weighted error ϵ_i by Eq. (13);
14: $\quad$ Calculate β_i and β by Eq. (14) and Eq. (15);
15: $\quad$ Update the weight vector w_{i+1} by Eq. (16);
16: **end for**
17: Get a strong classifier h_s by Eq. (17);
18: Generate a large number of test solutions $\mathbf{X}_{test}$;
19: Select $x \in \mathbf{X}_{test}$ which $h_s(x) = 1$ as $\mathbf{POP}_{predicted}$;

current environment. Therefore, the source domain of each environment should be preserved and exploited. When the environmental changes are detected, the source domains from all historical environments with the target domain are fed into a base classifier, where one source domain with the smallest error is selected as the most suitable one for training TrAdaboost. Fig. 3 depicts a flowchart of the MSTL method. During the transfer process, the clustering centroids and random solutions are regarded as the source domain $\mathbf{D}_{so}$, while the transfer centroids and random solutions are regarded as the target domain $\mathbf{D}_{ta}$. In the source domain $\mathbf{D}_{so}$, the clustering centroids and the random solutions are respectively labeled by +1 and -1, which indicate good samples and poor samples. In the target domain $\mathbf{D}_{ta}$, the transfer centroids and the random solutions are respectively labeled by +1 and -1, which also indicate good samples and poor samples.

To clarify the running of MSTL, its pseudo-code is given in **Algorithm 4** with the inputs: the set of source domain $\mathbf{S}$ and the target domain $\mathbf{D}_{ta}$. Firstly, set L as the size of $\mathbf{S}$ in line 1 and set the maximal number of iterations M as 7 in line 2. Then, for each source domain from set $\mathbf{S}$ in line 3, the weights of samples in the source and target domains are set as follows:

$$w_1(x) = \begin{cases} \frac{1}{|\mathbf{D}_{so}|}, x \in \mathbf{X}_{so}, \\ \frac{1}{|\mathbf{D}_{ta}|}, x \in \mathbf{X}_{ta}, \end{cases} \tag{12}$$

where $\mathbf{D}_{so}$ and $\mathbf{D}_{ta}$ indicate the source domain and target domain, respectively. Then, train a classifier h^i_w for each set $(\mathbf{D}^i_{so}, \mathbf{D}_{ta})$, $i = 1, \ldots, L$ in line 6, and calculate the error of the corresponding classifier to the target domain in line 7. Then, select the source domain (i.e., the k-th source domain in $\mathbf{S}$, $\mathbf{D}^k_{so}$) corresponding to the classifier with the smallest error as the source domain of TrAdaboost in line 9. After that, reinitialize the weights of all solutions from $\mathbf{D}^k_{so}$ and $\mathbf{D}_{ta}$ in line 10 and train M weak classifiers in lines 12-15. Specifically, in line 13, in the ith (i =1,… ,M) iteration, the error rate of the weak classifier h^i_w on $\mathbf{D}_{ta}$ is calculated, as follows:

$$\epsilon_i = \sum_{x \in X_{ta}} \frac{w_i(x) \cdot |h^i_w(x) - y(x)|}{\sum_{x \in X_{ta}} w_i(x)}, \tag{13}$$

and the coefficients of the weights in $\mathbf{D}_{ta}$ and $\mathbf{D}_{so}^{k}$ are calculated respectively by

$$\beta_i = \frac{\epsilon_i}{1-\epsilon_i}, \tag{14}$$

and

$$\beta = 1/\left(1+\sqrt{2\ln\left|X_{so}^{k}\right|/M}\right), \tag{15}$$

where M is the maximal number of iterations. The weights of the samples in the source and target domains can be updated by:

$$w_{i+1}(x) = \begin{cases} w_i(x)\cdot\beta^{|h_w^i(x)-y(x)|}, & x\in\mathbf{D}_{so}^{k}, \\ w_i(x)\cdot\beta_i^{-|h_w^i(x)-y(x)|}, & x\in\mathbf{D}_{ta}, \end{cases} \tag{16}$$

For each iteration in lines 12-15, a weak classifier can be obtained, which will become more and more accurate when the weights are adjusted. Finally, in line 17, a weighted voting mechanism is used to combine these weak classifiers into a strong classifier, which is expressed by

$$h_s(x) = \operatorname{sign}\left(\sum_{i=1}^{N}\beta_i h_w^i(x)\right). \tag{17}$$

After obtaining the strong classifier, a large number of test solutions ($\mathbf{X}_{test}$) are generated in line 18. To make full use of effective information from $\mathbf{C}^{MTL}$, the solutions in $\mathbf{C}^{MTL}$ are evolved once in the new environment and then also added into $\mathbf{X}_{test}$. Then, $\mathbf{X}_{test}$ is inputted into the strong classifier and those solutions predicted as good ones by the strong classifier will be selected to form an initial population in line 19. It is worth noting that if the number of predicted solutions is smaller than the population size, the initial population will randomly add some non-dominated solutions from the previous environment.

3.4. Complexity analysis

The computational cost of the dynamic response part comes from the CMTL and MSTL algorithms. In CMTL, the computational costs involve HCM, constructing the geodesic flow, and identifying transfer centroids by using the interior-point algorithm. For HCM, the total computational complexity of the clustering process is $O(N\times N_k)$. In addition, mapping data via PCA and constructing the geodesic flow spends $O(d^2)$ and $O(1)$, respectively. The computational cost of identifying transfer centroids by using an interior-point algorithm consumes $O(m^3\times N_k)$. In MSTL, the computational cost only involves training weak classifiers, and thus acquiring a strong classifier takes $O(N^2\times D)$, where D is the dimension of decision variables.

The space cost of the proposed MSTL-DMOEA comes from storing the source domain of all historical environments, where the source domain is the POS of each environment. Therefore, the total space cost of MSTL-DMOEA is $O(N\times t)$, where t is the number of environmental changes.

4. Experimental studies

4.1. Performance indicators

In this paper, two indicators are used to evaluate the performance of the compared DMOEAs when solving various kinds of DMOPs, as introduced below.

4.1.1. Mean Inverted Generational Distance (MIGD)

This MIGD metric calculates the average of inverted generational distance (IGD) [47] values in all historical environments over a run, as follows:

$$\mathrm{MIGD}(\mathbf{POF}_t^*,\mathbf{POF}_t) = \frac{1}{|\mathbf{T}|}\sum_{t\in\mathbf{T}}\mathrm{IGD}(\mathbf{POF}_t^*,\mathbf{POF}_t), \tag{18}$$

Table 2
Types of DMOPs

Type I	POS changes over time, but POF remains unchanged.
Type II	Both POS and POF change over time.
Type III	POF changes over time, but POS remains unchanged.
Type IV	Both POS and POF remain unchanged, but the environment may change.

where $\mathbf{POF}_t^*$ is the true POF of a DMOP at time t, $\mathbf{POF}_t$ is an approximate POF set obtained by a DMOEA at time t, $\mathbf{T}$ is a set of discrete-time points in a run, $|\mathbf{T}|$ is the cardinality of $\mathbf{T}$, and the IGD metric measures the average of the closest distance of the solutions in $\mathbf{POF}_t^*$ to that in $\mathbf{POF}_t$, as follows:

$$\mathrm{IGD}(\mathbf{POF}^*, \mathbf{POF}) = \frac{1}{|\mathbf{POF}^*|} \sum_{p^* \in \mathbf{POF}^*} \min_{p \in \mathbf{POF}} \left\| p^* - p \right\|^2. \tag{19}$$

Thus, a smaller MIGD value indicates a better performance of DMOEA regarding the convergence and diversity.

4.1.2. Mean Hypervolume (MHV)

The MHV metric calculates the average of hypervolume (HV) [42] values in all historical environments over a run, as follows:

$$\mathrm{MHV}(\mathbf{POF}) = \frac{1}{|\mathbf{T}|} \sum_{t \in \mathbf{T}} \mathrm{HV}(\mathbf{POF}_t), \tag{20}$$

where $\mathbf{T}$ is a set of discrete-time points in a run, $|\mathbf{T}|$ is the cardinality of $\mathbf{T}$, and the HV metric measures the hypervolume of the space construed by the approximate PF set $\mathbf{POF}_t$ and the reference point, which can reflect both convergence and diversity of these non-dominated solutions, as follows:

$$\mathrm{HV}(\mathbf{POF}_t) = \mathrm{VOL}\left(\bigcup_{x \in \mathbf{POF}_t} \left[f_1(x), z_1^t \right] \times \cdots \times \left[f_m(x), z_m^t \right] \right), \tag{21}$$

where VOL is the lebesgue measure and z_i^t $(i = 1, 2, \ldots, m)$ is the reference point for the computation of HV.

4.2. Test problems and parameter settings

In this paper, three popular test suites for DMOPs, i.e., DF series [24], FDA series [12], and F series [49], are adopted to challenge the performance of MSTL-DMOEA. In general, the DF, FDA, and F test problems cover almost all types of environmental changes as shown in Table 2. The environmental parameter in these test problems is set as $t = \left(1/n_t\right) \lfloor \tau/\tau_t \rfloor$, where n_t, τ_t and τ are the severity of the environmental changes, the frequency of environmental changes, and the maximum generation for solving DMOPs, respectively.

To verify the effectiveness of the proposed MSTL-DMOEA, four competitive DMOEAs are considered for comparison, i.e., SVR-MOEA/D [4], KT-MOEA/D[20], MMTL-MOEA/D[21], and IT-MOEA/D[19]. For static optimization solvers, there are many excellent MOEA solvers that can be chosen [40, 41, 46], and MOEA/D [46] is used as the static optimization solver in this paper. Please note that all the parameters in these compared algorithms are set according to the original references. Other common parameters are briefly summarized in Table 3.

For environmental parameters, there are four pairs of dynamic configurations for test DMOPs in our experiments: $(n_t = 5, \tau_t = 10)$, $(n_t = 10, \tau_t = 5)$, $(n_t = 5, \tau_t = 5)$, $(n_t = 10, \tau_t = 10)$, which are commonly used in the experimental comparisons of DMOEAs [20],[4],[21],[27]. As suggested in [10], the maximum number of generations τ is fixed to be $20 \times \tau_t$, which ensures that there are 20 environmental changes in each run. Each algorithm is run 20 times independently for each test case.

For the static optimizer MOEA/D, the population size N is set to 100 for biobjective problems and 150 for triobjective problems, and the number of decision variables is set to 10 for all benchmark problems according to the reference [20]. The simulated binary crossover (SBX) [8] is used as a crossover operator, whose crossover distribution

Table 3
Parameter settings

Parameter Variables	Parameter Values
Dynamic settings	$n_t = \{5,10\}$, $\tau_t = \{5,10\}$.
Termination criterion	$\tau = 20 \times \tau_t$.
The number of runs	20 times.
The population size, N	$N = 100$ or 150.
The number of decision variables, D	$D = 10$.
The decomposition method	Tchebycheff [46].
The SBX crossover	$\eta_c = 20$, $p_c = 0.8$.
The PM mutation	$\eta_m = 20$, $p_m = 1/D$.
Weak classifier	SVM.
The number of clusters N_k	10.

index η_c and crossover possibility p_c are set as 20 and 0.8, respectively. The polynomial mutation (PM) [9] is used as a mutation operator, whose mutation distribution index η_m and mutation possibility p_m are set as 20 and $1/D$, respectively.

In the proposed MSTL-DMOEA, the support vector machine (SVM) [5] is used as weak classifier. The number of clusters N_k used in Algorithm 3 is set to 10 according to parameter analysis in section 4.5.

4.3. Performance comparisons with other algorithms

The statistical results of MIGD from 20 runs are collected in Table 4 for solving the DF test problems and Table 5 for solving the FDA and F test problems. In these tables, (+), (=), and (-) after the statistical results respectively indicate that MSTL-MOEA/D performs better than, similarly with, and worse than the compared algorithms according to Wilcoxon [11] ranksum test with a 0.05 significance level.

Table 4 gives the MIGD values of all the compared algorithms when solving the DF test problems. As observed from the last row in Table 4, MSTL-MOEA/D can achieve the best results in 29 out of 56 cases, while SVR-MOEA/D, KT-MOEA/D, MMTL-MOEA/D, and IT-MOEA/D respectively obtain the best results in 12, 3, 2, and 10 cases. Specifically, MSTL-MOEA/D is superior to the other compared algorithms on DF1, DF2, DF3, DF4, DF5, DF9, and DF12 with the most dynamic configurations, which indicates that MSTL-MOEA/D can find the final solutions with better convergence and diversity at different environments on most test cases. This is mainly because the proposed CMTL can effectively transfer knowledge from the clustering centroids of approximate POS and the proposed MSTL can fully transfer knowledge from all historical environments, which constructs a more accurate prediction model to generate a good initial population for the new environment. On DF6 and DF8, MSTL-MOEA/D is only slightly worse than the corresponding competitors with the best performance. Moreover, MSTL-MOEA/D shows significantly worse performance than SVR-MOEA/D on DF7 and DF13, and than IT-MOEA/D on DF10 and DF14, mainly due to some specific characteristics in these problems. The POS of DF7 is dynamically changed in different environments, but its centroid remains unchanged. Thus, it is not so effective for MSTL-MOEA/D by using the past centroids of POS to train a prediction model. For DF11, DF13, and DF14, as their PFs are irregularly changed, it is very difficult for MSTL-MOEA/D to exploit valid information from the historical environments, while SVR-MOEA/D constructs a regression model for each decomposed subproblem, which can give a more correct prediction for each subproblem and perform relatively better on these three cases.

To comprehensively show the performance of all the compared algorithms, their IGD values in each environment are plotted in Fig. 4. As observed from Fig. 4, MSTL-MOEA/D obtains the better IGD results in most environments when compared to other algorithms. Moreover, MSTL-MOEA/D also shows a better stability, as its IGD curve is smoother, while the IGD curve of other compared algorithms may rise suddenly, e.g., the performance of KT-MOEA/D on DF1, DF3, DF5, DF12, and DF13 fluctuates significantly at some specific environments, as well as that of SVR-MOEA/D on DF6 and DF9. For KT-MOEA/D, probably because the distribution of knee points varies dramatically in different environments, it is difficult to accurately predict the knee points. For SVR-MOEA/D, probably because the predictor is highly dependent on the qualities of past solutions, the accuracy of the predictor will be affected when their qualities are poor.

Table 4
Mean and standard deviation values of MIGD metric obtained by the compared algorithms under the DF problems

Problems	(n_t, τ_t)	SVR-MOEA/D	KT-MOEA/D	MMTL-MOEA/D	IT-MOEA/D	MSTL-MOEA/D
DF1	(5, 10)	0.0325±7.58e-03(=)	0.0382±5.47e-03(+)	0.0301±4.05e-03(=)	0.0429±5.67e-03(+)	**0.0294±3.38e-03**
	(10, 5)	0.0780±1.34e-02(+)	0.1254±8.65e-03(+)	0.0790±1.13e-02(+)	0.0817±1.01e-02(+)	**0.0611±5.79e-03**
	(5, 5)	0.1116±2.04e-02(+)	0.1245±1.23e-02(+)	0.0849±1.03e-02(=)	0.1386±1.91e-02(+)	**0.0801±5.21e-03**
	(10, 10)	0.0270±5.00e-03(+)	0.0386±5.12e-03(+)	0.0274±4.60e-03(+)	0.0280±4.58e-03(+)	**0.0236±3.09e-03**
DF2	(5, 10)	0.0404±9.40e-03(+)	0.0352±4.11e-03(+)	0.0332±5.20e-03(+)	0.0430±6.33e-03(+)	**0.0304±4.43e-03**
	(10, 5)	0.1019±1.77e-02(+)	0.1025±8.73e-03(+)	0.0837±9.12e-03(+)	0.0746±6.86e-03(+)	**0.0617±5.96e-03**
	(5, 5)	0.1214±2.24e-02(+)	0.1053±1.06e-02(+)	0.0838±1.10e-02(+)	0.1097±1.74e-02(+)	**0.0764±4.30e-03**
	(10, 10)	0.0367±7.01e-03(+)	0.0352±4.50e-03(+)	0.0321±5.85e-03(+)	0.0342±7.31e-03(+)	**0.0261±3.04e-03**
DF3	(5, 10)	0.1676±5.74e-02(+)	0.1721±1.66e-02(+)	0.1688±1.39e-02(+)	0.1478±2.33e-02(+)	**0.1155±2.12e-02**
	(10, 5)	0.2487±3.24e-02(+)	0.3533±3.26e-02(+)	0.3199±1.46e-02(+)	0.2718±5.09e-02(+)	**0.1961±4.52e-02**
	(5, 5)	0.3182±3.41e-02(+)	0.3186±2.14e-02(+)	0.2568±1.87e-02(+)	0.2214±2.97e-02(+)	**0.1913±5.05e-02**
	(10, 10)	0.1308±2.58e-02(+)	0.2160±1.36e-02(+)	0.2321±1.82e-02(+)	0.1871±2.39e-02(+)	**0.0993±3.06e-02**
DF4	(5, 10)	0.1212±1.23e-02(+)	0.1360±1.05e-02(+)	0.1047±6.14e-03(+)	0.1415±1.85e-02(+)	**0.0992±7.62e-03**
	(10, 5)	0.3491±8.02e-02(+)	0.4840±5.60e-02(+)	0.2222±3.80e-02(+)	0.5340±5.40e-02(+)	**0.1952±1.60e-02**
	(5, 5)	0.2533±4.03e-02(+)	0.3724±4.07e-02(+)	0.1719±1.96e-02(+)	0.3624±6.04e-02(+)	**0.1440±1.62e-02**
	(10, 10)	0.1863±2.37e-02(+)	0.1907±1.29e-02(+)	**0.1594±8.68e-03(-)**	0.2209±3.36e-02(+)	0.1667±1.18e-02
DF5	(5, 10)	0.0279±6.88e-03(+)	0.0387±6.85e-03(+)	0.0231±3.21e-03(=)	0.0261±3.74e-03(+)	**0.0228±2.52e-03**
	(10, 5)	0.0600±1.01e-02(+)	0.1541±2.39e-02(+)	0.0574±1.02e-02(+)	0.0461±7.33e-03(=)	**0.0425±3.93e-03**
	(5, 5)	0.0953±2.17e-02(+)	0.2087±4.02e-02(+)	0.0814±1.96e-02(+)	0.0747±8.44e-03(+)	**0.0675±9.77e-03**
	(10, 10)	0.0226±3.88e-03(+)	0.0318±4.82e-03(+)	0.0216±2.46e-03(+)	0.0227±3.56e-03(+)	**0.0195±2.41e-03**
DF6	(5, 10)	1.4911±5.51e-01(+)	0.5268±5.91e-02(+)	0.8024±6.09e-02(+)	0.6794±3.96e-01(+)	**0.4346±8.93e-02**
	(10, 5)	**0.7778±1.99e-01(=)**	1.0265±1.55e-01(+)	1.3155±3.63e-01(+)	1.0974±5.88e-01(=)	0.7817±2.57e-01
	(5, 5)	2.0049±6.82e-01(+)	1.0337±1.41e-01(+)	1.2027±4.22e-01(+)	1.3477±4.25e-01(+)	**0.7906±1.03e-01**
	(10, 10)	**0.4233±1.19e-01(=)**	0.4753±9.40e-02(=)	0.8842±3.42e-01(+)	0.5801±2.78e-01(=)	0.5541±2.56e-01
DF7	(5, 10)	**0.1770±3.95e-02(-)**	0.2882±3.34e-02(+)	0.3185±4.54e-02(+)	0.3262±2.89e-02(+)	0.2208±5.64e-02
	(10, 5)	**0.2454±3.22e-02(=)**	0.3132±2.16e-02(+)	0.3684±5.17e-02(+)	0.3235±4.11e-02(+)	0.2505±3.75e-02
	(5, 5)	**0.2855±5.84e-02(-)**	0.3523±1.82e-02(=)	0.4017±3.25e-02(+)	0.3642±2.85e-02(=)	0.3470±4.74e-02
	(10, 10)	0.1866±2.86e-02(+)	0.2325±2.22e-02(+)	0.2705±3.73e-02(+)	0.2577±3.43e-02(+)	**0.1680±3.05e-02**
DF8	(5, 10)	0.0758±1.02e-02(+)	0.0783±9.01e-03(+)	**0.0617±9.14e-03(=)**	0.0802±1.18e-02(+)	0.0671±1.17e-02
	(10, 5)	**0.0664±1.26e-02(=)**	0.1324±1.34e-02(+)	0.1074±1.64e-02(+)	0.0935±1.24e-02(+)	0.0745±1.34e-02
	(5, 5)	0.0790±9.05e-03(=)	0.1274±1.13e-02(+)	0.1018±1.30e-02(+)	0.0855±1.08e-02(+)	**0.0755±1.46e-02**
	(10, 10)	**0.0625±1.12e-02(=)**	0.0801±8.28e-03(+)	0.0639±8.19e-03(=)	0.0751±1.39e-02(=)	0.0679±1.02e-02
DF9	(5, 10)	0.2926±1.02e-01(+)	**0.1566±3.37e-02(=)**	0.2388±4.51e-02(+)	0.1906±8.70e-02(+)	0.1666±4.85e-02
	(10, 5)	0.6828±1.48e-01(+)	0.3203±5.21e-02(+)	0.4173±1.01e-01(+)	0.3187±1.09e-01(+)	**0.1833±3.86e-02**
	(5, 5)	0.6797±1.59e-01(+)	0.3688±6.05e-02(+)	0.4558±1.43e-01(+)	0.4354±1.16e-01(+)	**0.2729±4.41e-02**
	(10, 10)	0.2886±5.15e-02(+)	0.1541±3.07e-02(+)	0.2238±4.38e-02(+)	0.1496±7.07e-02(=)	**0.0979±3.24e-02**
DF10	(5, 10)	0.2265±1.90e-02(=)	0.2372±1.99e-02(=)	0.2200±2.02e-02(=)	**0.2107±1.13e-02(-)**	0.2336±2.90e-02
	(10, 5)	0.2384±2.09e-02(+)	0.2693±1.51e-02(+)	0.2151±1.74e-02(=)	**0.1878±1.35e-02(-)**	0.2192±2.29e-02
	(5, 5)	0.2292±2.20e-02(=)	0.2659±1.48e-02(+)	0.2365±2.48e-02(=)	**0.2161±3.65e-02(=)**	0.2393±2.31e-02
	(10, 10)	0.2460±2.31e-02(+)	0.2353±1.46e-02(+)	0.2197±3.00e-02(=)	**0.1957±1.32e-02(-)**	0.2197±1.46e-02
DF11	(5, 10)	0.1156±3.35e-03(=)	**0.1126±2.47e-03(-)**	0.1182±5.14e-03(=)	0.1128±2.00e-03(-)	0.1171±4.16e-03
	(10, 5)	0.1508±5.24e-03(=)	0.1745±8.10e-03(+)	0.1523±6.36e-03(=)	**0.1435±5.72e-03(=)**	0.1551±1.05e-02
	(5, 5)	0.1489±2.68e-03(-)	0.1775±8.41e-03(+)	0.1540±5.49e-03(=)	**0.1468±5.81e-03(=)**	0.1539±8.51e-03
	(10, 10)	0.1167±3.48e-03(=)	**0.1117±3.16e-03(-)**	0.1151±3.60e-03(=)	0.1152±3.60e-03(=)	0.1168±3.42e-03
DF12	(5, 10)	0.1883±1.54e-02(+)	0.3177±1.96e-02(+)	0.3236±6.85e-02(+)	0.1887±9.29e-03(+)	**0.1729±3.17e-02**
	(10, 5)	0.3111±6.50e-02(+)	0.3754±2.49e-02(+)	0.3187±4.21e-02(+)	0.2090±1.06e-02(=)	**0.1985±1.97e-02**
	(5, 5)	0.3329±1.15e-01(+)	0.3874±2.25e-02(+)	0.3951±4.82e-02(+)	**0.2288±7.77e-03(=)**	0.2313±2.38e-02
	(10, 10)	0.1765±8.24e-03(+)	0.2701±2.81e-02(+)	0.2556±2.37e-02(+)	0.1589±1.29e-02(+)	**0.1384±8.06e-03**
DF13	(5, 10)	**0.2586±1.59e-02(=)**	0.2986±1.98e-02(+)	0.2599±1.01e-02(=)	0.2652±1.01e-02(=)	0.2616±1.15e-02
	(10, 5)	**0.2438±1.28e-02(-)**	0.3803±2.88e-02(+)	0.2697±1.39e-02(=)	0.2491±5.09e-03(-)	0.2680±1.32e-02
	(5, 5)	**0.2853±2.35e-02(-)**	0.3749±1.65e-02(+)	0.3155±4.06e-02(=)	0.2924±1.46e-02(=)	0.3196±4.17e-02
	(10, 10)	**0.2472±1.02e-02(-)**	0.2817±1.27e-02(+)	0.2644±1.34e-02(=)	0.2532±7.29e-03(-)	0.2604±1.51e-02
DF14	(5, 10)	**0.0805±2.10e-03(-)**	0.0887±3.30e-03(+)	0.0932±2.94e-02(=)	0.0809±3.83e-03(=)	0.0853±3.80e-03
	(10, 5)	0.0928±2.45e-03(-)	0.1468±1.38e-02(+)	0.1042±3.53e-03(-)	**0.0907±2.42e-03(-)**	0.1117±1.02e-02
	(5, 5)	0.1054±5.57e-03(-)	0.1505±1.19e-02(+)	0.1970±1.00e-01(+)	**0.1012±3.77e-03(-)**	0.1169±1.19e-02
	(10, 10)	0.0794±2.36e-03(-)	0.0899±3.17e-03(+)	0.0817±2.81e-03(-)	**0.0785±1.40e-03(-)**	0.0846±5.06e-03
+/=/-		33/13/10	50/4/2	35/18/3	32/15/9	—
best/all		12/56	3/56	2/56	10/56	29/56

Table 5 gives the MIGD values of all the compared algorithms when solving the FDA and F test problems. It is observed that the proposed MSTL-MOEA/D demonstrates a better performance when compared to other algorithms. As observed from the last row of Table 5, MSTL-MOEA/D performs best on 32 out of total 44 cases, while SVR-MOEA/D, KT-MOEA/D, MMTL-MOEA/D, and IT-MOEA/D respectively obtain the best results in 3, 1, 6, and 2 cases. Specifically, MSTL-MOEA/D performs significantly better than the other compared algorithms on FDA1, FDA2, FDA4, F5, F7, F9, and F10 with the most dynamic configurations, indicating that MSTL-MOEA/D has an outstanding performance in dynamically tracking POS of these problems. On F6 and F8, MSTL-MOEA/D is slightly

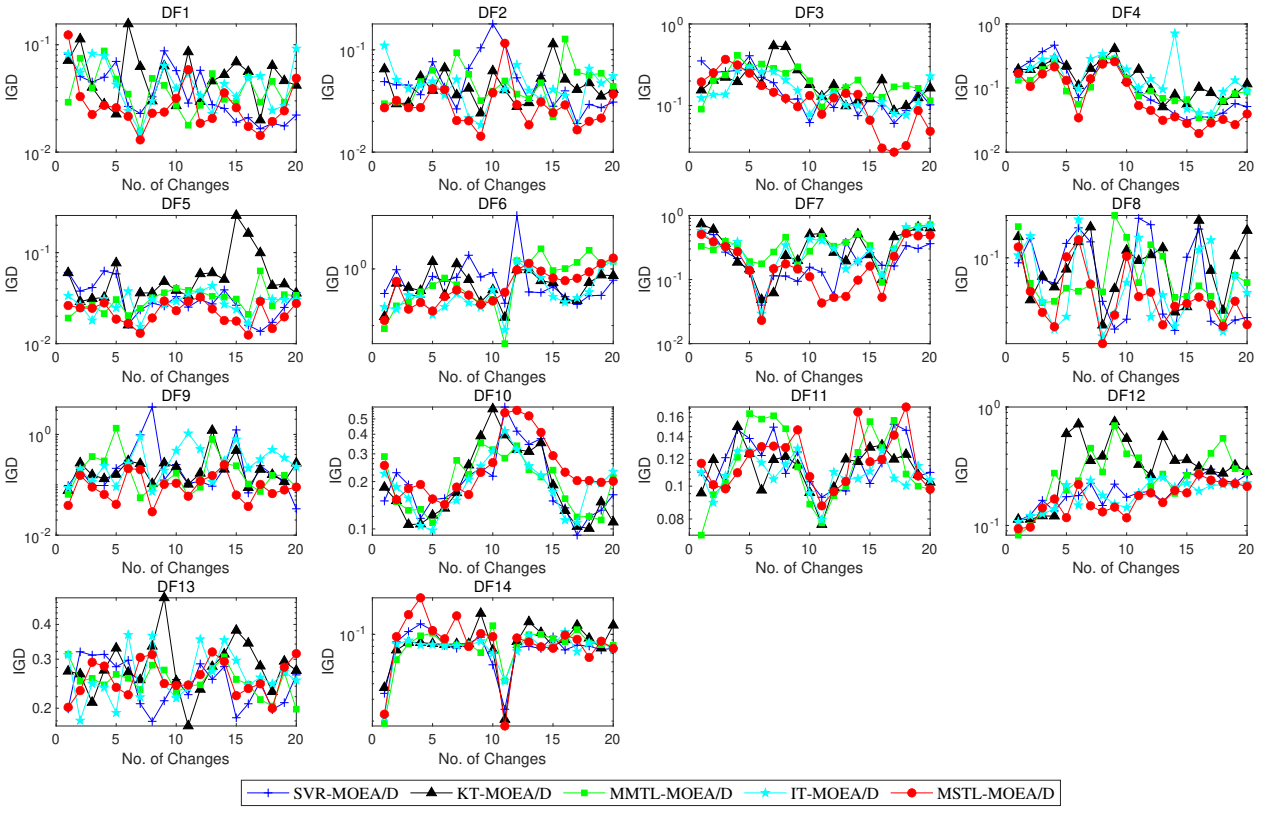

Fig. 4: The IGD values of all the compared algorithms with the configuration ($n_t = 5$ and $\tau_t = 10$).

worse than the corresponding competitors with the best performance. Only on FDA3 and FDA5, MSTL-MOEA/D performs significantly worse than the compared algorithms, mainly due to some specific characteristics in these problems. FDA3 is a biobjective problem with different densities of solutions on the dynamic POF and with the overlapped POS dynamically changing over time, which causes a significant challenge to find good transfer solutions in the procedure of CMTL. FDA5 is a triobjective problem, where both POS and POF dynamically change over time. As a result, the POS and POF of FDA5 become very complicated, which makes it very difficult to exploit valid knowledge in transfer learning.

Based on the above experimental results, MSTL-MOEA/D can effectively generate a good initial population for most test problems when the environment changes. However, for some test problems with specific characteristics, such as DF7, DF11, DF13, DF14, FDA3, and FDA5, MSTL-MOEA/D still needs further improvement, as it is not so effective to reuse past experiences to predict a high-quality initial population for the new environment, mainly due to the severe POS changes in these problems. In addition, when the environmental changes do not occur in a similar manner, MSTL may degenerate into a single source domain for transfer learning and thus is not so effective. Generally, the performance of transfer learning based DMOEAs will be significantly degraded when the POSs in different environments do not share a certain correlation.

Moreover, to further verify the performance of our algorithm in terms of both convergence and diversity, MHV is adopted as another metric to evaluate the final approximate POSs obtained by all the compared algorithms. Due to page limitations, the experimental results are presented in Table S-1 for solving the FDA and F test problems and Table S-2 for solving the DF test problems in the supplementary material. As observed from Table S-1 and Table S-2, MSTL-MOEA/D also outperforms other compared DMOEAs in most of the cases, which further validates that MSTL-MOEA/D shows certain advantages for solving different types of DMOPs.

Table 5
Mean and standard deviation values of MIGD metric obtained by the compared algorithms under the FDA and F problems

Problems	(n_t, τ_t)	SVR-MOEA/D	KT-MOEA/D	MMTL-MOEA/D	IT-MOEA/D	MSTL-MOEA/D
FDA1	(5, 10)	0.0303±2.52e-03(+)	0.0481±4.15e-03(+)	0.0416±4.63e-03(+)	0.0284±2.47e-03(=)	**0.0281±1.68e-03**
	(10, 5)	0.0654±6.66e-03(+)	0.1552±1.23e-02(+)	0.0734±5.69e-03(+)	0.0520±5.93e-03(=)	**0.0510±5.80e-03**
	(5, 5)	0.1004±1.26e-02(+)	0.1597±1.29e-02(+)	0.1275±1.41e-02(+)	0.0765±7.30e-03(=)	**0.0746±7.54e-03**
	(10, 10)	0.0254±1.48e-03(+)	0.0476±4.70e-03(+)	0.0282±2.29e-03(+)	0.0244±2.18e-03(=)	**0.0242±2.33e-03**
FDA2	(5, 10)	0.0199±1.11e-03(+)	0.0233±2.65e-03(+)	0.0293±4.92e-03(+)	0.0206±2.46e-03(+)	**0.0181±1.37e-03**
	(10, 5)	0.0482±4.32e-03(+)	0.0513±3.83e-03(+)	0.0737±1.50e-02(+)	0.0399±1.07e-02(+)	**0.0259±2.33e-03**
	(5, 5)	0.0527±4.21e-03(+)	0.0521±3.97e-03(+)	0.0763±1.22e-02(+)	0.0387±4.70e-03(+)	**0.0333±5.99e-03**
	(10, 10)	0.0193±1.28e-03(+)	0.0231±1.99e-03(+)	0.0282±3.81e-03(+)	0.0193±1.89e-03(+)	**0.0175±1.24e-03**
FDA3	(5, 10)	0.0461±1.10e-02(=)	0.0440±5.78e-03(=)	**0.0386±8.88e-03(-)**	0.0470±9.11e-03(=)	0.0448±7.65e-03
	(10, 5)	0.0991±1.94e-02(+)	0.1201±1.99e-02(+)	0.0637±1.21e-02(=)	0.0745±1.12e-02(+)	**0.0589±9.09e-03**
	(5, 5)	0.0776±1.97e-02(+)	0.0986±1.56e-02(+)	**0.0497±1.16e-02(=)**	0.0741±1.50e-02(+)	0.0520±5.85e-03
	(10, 10)	0.0988±2.36e-02(+)	0.0699±8.80e-03(-)	**0.0657±1.56e-02(-)**	0.0666±1.06e-02(-)	0.0816±1.82e-02
FDA4	(5, 10)	0.0976±2.75e-03(=)	0.1026±3.74e-03(+)	0.0966±3.10e-03(=)	0.0984±2.36e-03(+)	**0.0961±2.77e-03**
	(10, 5)	0.1336±3.89e-03(=)	0.1620±7.55e-03(+)	0.1458±7.35e-03(+)	0.1412±6.44e-03(+)	**0.1311±6.73e-03**
	(5, 5)	0.1619±1.55e-02(+)	0.1618±5.23e-03(+)	**0.1420±7.58e-03(-)**	0.1536±6.63e-03(=)	0.1521±7.49e-03
	(10, 10)	0.0930±2.57e-03(=)	0.1009±2.83e-03(+)	0.0958±3.04e-03(+)	0.0947±2.63e-03(+)	**0.0919±1.94e-03**
FDA5	(5, 10)	0.0855±1.47e-02(+)	**0.0666±4.17e-03(-)**	0.0714±7.39e-03(=)	0.0725±6.37e-03(=)	0.0749±8.11e-03
	(10, 5)	0.1077±1.04e-02(+)	0.1002±1.05e-02(+)	0.0959±8.58e-03(=)	**0.0899±8.07e-03(=)**	0.0930±8.66e-03
	(5, 5)	0.1114±1.02e-02(+)	0.1101±4.99e-03(+)	**0.0899±1.09e-02(-)**	0.1076±1.03e-02(=)	0.1036±8.85e-03
	(10, 10)	0.0770±7.64e-03(=)	0.0690±5.68e-03(=)	**0.0645±5.85e-03(-)**	0.0666±4.58e-03(-)	0.0734±8.28e-03
F5	(5, 10)	1.1083±1.82e-01(+)	0.5935±9.50e-02(+)	1.7362±1.20e-01(+)	0.7828±2.76e-01(+)	**0.4813±9.28e-02**
	(10, 5)	0.7982±1.60e-01(+)	0.9243±1.28e-01(+)	2.1409±3.97e-01(+)	0.7703±2.59e-01(+)	**0.5542±1.00e-01**
	(5, 5)	1.7953±1.36e-01(+)	0.9407±9.61e-02(+)	2.2594±2.99e-01(+)	1.2618±3.14e-01(+)	**0.8592±1.01e-01**
	(10, 10)	0.4102±1.15e-01(+)	0.5846±6.28e-02(+)	1.2076±2.66e-01(+)	0.3983±1.87e-01(+)	**0.2515±7.00e-02**
F6	(5, 10)	**0.4240±1.12e-01(-)**	0.5345±9.60e-02(=)	0.7968±1.96e-01(+)	0.4590±7.73e-02(=)	0.4816±8.74e-02
	(10, 5)	0.5645±8.01e-02(+)	0.8880±1.36e-01(+)	1.2068±2.41e-01(+)	0.6807±2.07e-01(+)	**0.5035±6.67e-02**
	(5, 5)	0.7895±1.30e-01(=)	0.9797±1.38e-01(+)	0.9655±2.30e-01(+)	0.7923±1.80e-01(=)	**0.7786±1.14e-01**
	(10, 10)	**0.2664±5.59e-02(=)**	0.5007±8.31e-02(+)	0.8160±2.22e-01(+)	0.4344±1.15e-01(+)	0.2904±5.61e-02
F7	(5, 10)	0.6263±1.67e-01(+)	0.4070±5.97e-02(+)	1.0675±3.13e-01(+)	0.4818±1.11e-01(+)	**0.3643±5.68e-02**
	(10, 5)	0.6346±7.02e-02(+)	0.7825±8.78e-02(+)	1.3903±6.17e-01(+)	0.6882±1.71e-01(+)	**0.4495±7.84e-02**
	(5, 5)	1.1142±1.86e-01(+)	0.7832±8.17e-02(+)	1.4244±2.51e-01(+)	0.9052±2.01e-01(+)	**0.6089±7.71e-02**
	(10, 10)	0.3190±7.83e-02(=)	0.4482±9.10e-02(+)	0.6150±2.02e-01(+)	0.3779±7.45e-02(+)	**0.2721±4.49e-02**
F8	(5, 10)	**0.1972±1.93e-02(-)**	0.2654±2.24e-02(=)	0.2472±3.33e-02(-)	0.2174±1.90e-02(-)	0.2713±2.39e-02
	(10, 5)	0.2793±1.77e-02(+)	0.4191±3.14e-02(+)	0.3338±1.73e-02(+)	0.2613±1.50e-02(=)	**0.2591±2.04e-02**
	(5, 5)	0.3442±3.47e-02(-)	0.3882±2.53e-02(=)	0.3659±3.76e-02(=)	**0.3271±3.75e-02(-)**	0.3877±3.05e-02
	(10, 10)	0.1946±1.18e-02(+)	0.2627±2.66e-02(+)	0.2454±1.89e-02(+)	0.1961±1.17e-02(+)	**0.1840±1.92e-02**
F9	(5, 10)	1.5377±6.11e-02(+)	0.6234±1.32e-01(+)	1.1861±3.30e-01(+)	1.1490±2.61e-01(+)	**0.3934±6.30e-02**
	(10, 5)	1.4116±2.13e-01(+)	0.9615±1.54e-01(+)	1.1730±1.95e-01(+)	1.1600±3.83e-01(+)	**0.6196±1.16e-01**
	(5, 5)	1.9171±1.19e-01(+)	0.9975±1.26e-01(=)	1.4540±3.61e-01(+)	1.6630±3.18e-01(+)	**0.9275±1.84e-01**
	(10, 10)	0.8102±9.21e-02(+)	0.6142±7.80e-02(+)	0.9079±1.70e-01(+)	0.5414±1.63e-01(+)	**0.2967±3.63e-02**
F10	(5, 10)	0.9988±7.85e-02(+)	0.5314±9.85e-02(+)	1.1142±1.66e-01(+)	0.7586±1.83e-01(+)	**0.4351±7.32e-02**
	(10, 5)	2.5510±1.98e-01(=)	2.7684±2.16e-01(+)	3.0806±2.14e-01(+)	2.7028±2.49e-01(+)	**2.4481±1.95e-01**
	(5, 5)	1.3686±1.46e-01(+)	0.9108±1.83e-01(=)	1.5140±1.46e-01(+)	1.3025±3.11e-01(+)	**0.9067±2.00e-01**
	(10, 10)	2.7018±2.16e-01(=)	2.9455±9.76e-02(+)	2.8056±1.18e-01(+)	2.8518±1.41e-01(+)	**2.6776±1.90e-01**
+/=/-		31/10/3	35/7/2	32/6/6	28/12/4	—
best/all		3/44	1/44	6/44	2/44	32/44

4.4. Ablation study

The main purpose of MSTL-MOEA/D is to exploit all past valid information by using the MSTL method to predict a good initial population for the new environment. On the one hand, MSTL-MOEA/D can improve the transfer effect by only selecting clustering centroids for transfer learning in CMTL. On the other hand, MSTL-MOEA/D can help to reduce negative effect by fully exploiting the effective knowledge from all approximate POSs in past environments using MSTL. To verify the effectiveness of each component (CMTL and MSTL) in the algorithm design, more experiments are conducted by comparing the performance of MSTL-MOEA/D, MSTL-MOEA/D$_a$, and MSTL-MOEA/D$_b$ with $n_t = 5, \tau_t = 10$. MSTL-MOEA/D$_a$ removes the CMTL part and uses non-dominant solutions in the previous environment for running MTL and that in all historical environments for running MSTL, which aims to verify the effectiveness of CMTL. MSTL-MOEA/D$_b$ removes the MSTL part and just uses the source domain in the last environment for running transfer learning by TrAdaboost, which aims to verify the effectiveness of MSTL.

Table 6
Mean and standard deviation values of MIGD and MHV metric obtained by MSTL-MOEA/D_a, MSTL-MOEA/D_b, and MSTL-MOEA/D

Problems	Metrics	MSTL-MOEA/D_a	MSTL-MOEA/D_b	MSTL-MOEA/D
DF1	MIGD	0.0347±3.16e-03(+)	0.0353±4.91e-03(+)	**0.0294±3.38e-03**
	MHV	0.5098±4.25e-03(+)	0.5087±6.45e-03(+)	**0.5172±5.11e-03**
DF2	MIGD	0.0382±4.51e-03(+)	0.0416±3.08e-03(+)	**0.0304±4.43e-03**
	MHV	0.6611±6.72e-03(+)	0.6527±4.89e-03(+)	**0.6758±6.23e-03**
DF3	MIGD	0.1192±1.77e-02(=)	**0.0894±1.36e-02(-)**	0.1155±2.12e-02
	MHV	0.3973±1.41e-02(=)	**0.4229±1.08e-02(-)**	0.4001±1.82e-02
DF4	MIGD	0.1003±9.11e-03(=)	0.1012±8.87e-03(=)	**0.0992±7.62e-03**
	MHV	0.6597±3.12e-03(=)	0.6597±2.79e-03(=)	**0.6600±3.35e-03**
DF5	MIGD	0.0245±5.87e-03(+)	0.0242±2.14e-03(+)	**0.0228±2.52e-03**
	MHV	0.5400±7.80e-03(+)	0.5427±3.36e-03(+)	**0.5520±3.30e-03**
DF6	MIGD	0.6243±9.83e-02(+)	0.4531±1.20e-01(+)	**0.4346±8.93e-02**
	MHV	0.3523±2.82e-02(+)	0.3644±9.07e-02(+)	**0.3791±8.05e-02**
DF7	MIGD	0.2298±4.12e-02(+)	0.2355±4.37e-02(+)	**0.2208±5.64e-02**
	MHV	0.3946±1.61e-02(=)	0.3846±1.36e-02(+)	**0.3965±2.27e-02**
DF8	MIGD	0.0692±6.99e-03(=)	**0.0656±8.27e-03(=)**	0.0671±1.17e-02
	MHV	0.6027±2.64e-03(+)	**0.6058±2.65e-03(=)**	0.6045±2.81e-03
DF9	MIGD	0.1789±4.66e-02(+)	0.1695±3.70e-02(=)	**0.1666±4.85e-02**
	MHV	0.3924±2.76e-02(+)	0.3953±2.42e-02(+)	**0.4032±2.42e-02**
DF10	MIGD	0.2265±3.01e-02(=)	**0.2256±2.53e-02(=)**	0.2336±2.90e-02
	MHV	0.5968±2.52e-02(=)	**0.5992±2.10e-02(=)**	0.5929±2.66e-02
DF11	MIGD	0.1206±5.40e-03(=)	0.1196±4.80e-03(=)	**0.1171±4.16e-03**
	MHV	0.2522±4.03e-03(+)	0.2521±2.49e-03(+)	**0.2547±2.15e-03**
DF12	MIGD	0.1646±1.42e-02(=)	**0.1593±6.05e-03(=)**	0.1729±3.17e-02
	MHV	0.6069±2.11e-02(=)	**0.6400±1.10e-02(-)**	0.6057±2.05e-02
DF13	MIGD	**0.2574±1.37e-02(-)**	0.2613±1.55e-02(=)	0.2616±1.15e-02
	MHV	0.5834±2.10e-02(+)	0.5821±1.66e-02(+)	**0.5868±1.12e-02**
DF14	MIGD	0.0860±4.80e-03(=)	0.0886±3.94e-03(+)	**0.0853±3.80e-03**
	MHV	0.5556±4.20e-03(=)	0.5494±5.36e-03(=)	**0.5570±3.68e-03**
+/=/-		14/13/1	14/11/3	—
best/all		1/28	8/28	19/28

The experimental results are collected in Table 6 when solving the DF test problems. Obviously, MSTL-MOEA/D performs significantly better than MSTL-MOEA/D_a and MSTL-MOEA/D_b, as MSTL-MOEA/D obtains the best results in 19 out of 28 cases. To be specific, MSTL-MOEA/D is better than MSTL-MOEA/D_a and MSTL-MOEA/D_b both in 14 cases. However, MSTL-MOEA/D is only outperformed by MSTL-MOEA/D_a and MSTL-MOEA/D_b in 1 and 3 cases, respectively. Thus, it is reasonable to conclude that selecting clustering centroids for transfer learning is more effective than simply transferring all non-dominated solutions and the MSTL method is effective to construct a prediction model by exploiting effective knowledge from all historical environments. Other experiments with other dynamic settings are all conducted on the DF test problems, which are provided in Table S-3 and Table S-4 of the supplementary material due to page limitations. To summarize briefly, the experimental results in these two tables also support the above conclusion that both CMTL and MSTL are effective in improving the performance of MSTL-MOEA/D.

4.5. Influence of parameter N_k in MSTL-DMOEA

In MSTL-MOEA/D, the parameter N_k plays an important role to decide the number of transfer centroids. The setting of the parameter N_k will affect the performance of our algorithm. If the parameter N_k is set too large, it will generally lead to the waste of computational resources and may cause negative effect in transfer learning. However, if it is set too small, the historical information may not be fully exploited. To study the effect of parameter N_k on the performance of our algorithm, more experiments are conducted by running MSTL-MOEA/D with different values of N_k. Fig. 5 shows the average value of MIGD obtained by MSTL-MOEA/D on the DF test problems. To summarize, the overall performance of MSTL-MOEA/D becomes a little better when the number of clusters is enlarged from 1 to

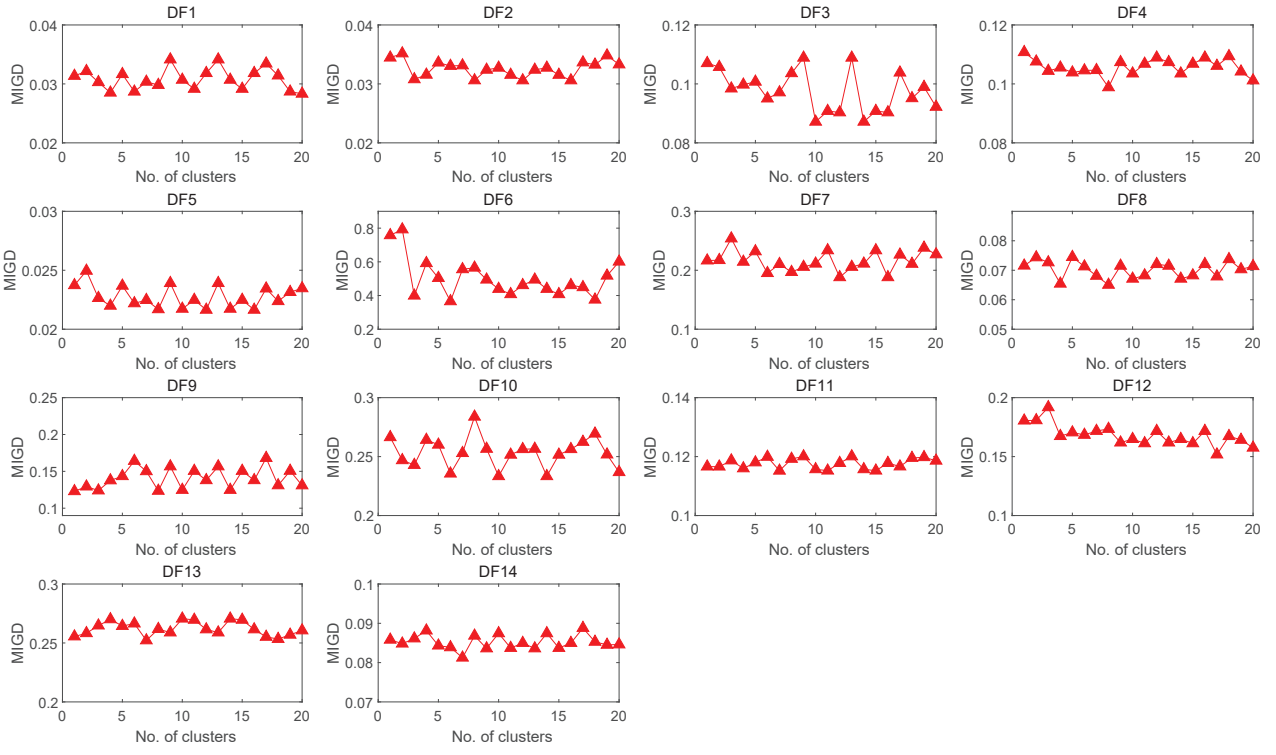

Fig. 5: The mean MIGD values obtained by MSTL-MOEA/D with different numbers of clusters on the DF problems with the configuration ($n_t = 5$ and $\tau_t = 10$).

10. However, in some cases, such as DF8, DF11, and DF13, the number the clusters has little effect on the performance of MSTL-MOEA/D, as it is difficult to accurately find transfer centroids in these problems. Through these extensive experiments, it is also found that the performance of MSTL-MOEA/D won't be improved significantly after the number of clusters has exceeded 10. Therefore, the number of clusters is suggested as 10 in MSTL-MOEA/D.

4.6. Running time cost

To study the computational cost of all the compared algorithms in solving the DF test problems, the running times of SVR-MOEA/D, KT-MOEA/D, MMTL-MOEA/D, IT-MOEA/D, and MSTL-MOEAD are plotted in Fig. 6, where the computer system is windows 10 and the running software is Matlab 2020a. As shown in Fig. 6, SVR-MOEA/D consumes less time on most of the cases among all the compared algorithms. Although MSTL-MOEA/D includes two transfer learning procedures, it still runs faster than the other three transfer learning-based DMOEAs (i.e., KT-MOEA/D, MMTL-MOEA/D, and IT-MOEA/D) in most of the cases. Thus, considering the promising performance of MSTL-MOEA/D regarding MIGD and MHV metrics, MSTL-MOEA/D still seems very competitive.

5. Conclusion

Applying transfer learning to address DMOPs have been validated to be promising, but the existing transfer learning-based DMOEAs intend to transfer knowledge from the approximate POS in the previous one or two environments, which may ignore effective knowledge from earlier times and may induce negative effect in transfer learning due to limited knowledge available. To address the above problems, a multiple source transfer learning algorithm, called MSTL-DMOEA, has been proposed, which enhances the transfer effect by only selecting clustering centroids for transfer. Moreover, a multisource TrAdaboost method is adopted to exploit effective knowledge from

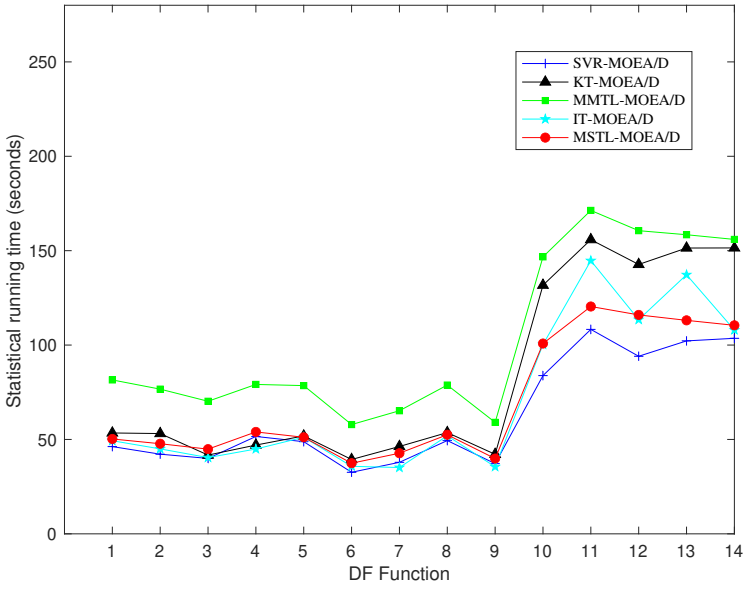

Fig. 6: Running times (seconds) of five DMOEAs on the DF test problems.

all historical environments, in which one best source domain from the source domain set will be selected with the target domain to train classifiers. As a result, a more accurate prediction model can be constructed to predict a good initial population for the new environment. The extensive experiments demonstrate some advantages of our proposed algorithm over four competitive DMOEAs (i.e., SVR-MOEA/D, KT-MOEA/D, MMTL-MOEA/D, and IT-MOEA/D).

Although the proposed algorithm can effectively exploit the historical knowledge to construct a more accurate predictor, its idea is still simple to select one best source domain based on the similarity between the source and target domains by using the error rate from the base classifier, which is then combined with the target domain to train a strong classifier. In addition, the accuracy of the weak classifier can greatly affect the optimization performance of the algorithm. For future work, there is still a huge challenge to improve the positive knowledge transfer for solving DMOPs. Hence, we will further study an incremental positive transfer learning method, which can incrementally exploit the historical knowledge from all the source domains for solving DMOPs. Moreover, the applications of our algorithm in solving some real-world DMOP applications will also be our future work.

CRediT authorship contribution statement

Yulong Ye: Conceptualization, Methodology, Software, Writing. **Qiuzhen Lin:** Supervision, Validation. **Lijia Ma:** Formal analysis, Writing-review. **Ka-Chun Wong:** Writing-reviewing and editing. **Maoguo Gong:** Writing-reviewing and editing. **Carlos A. Coello Coello:** Supervision, Writing-review.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (NSFC) under Grant 61876110, the Joint Funds of the NSFC under Key Program Grant U1713212, the Shenzhen Scientific Research and Development Funding Program under Grant JCYJ20190808164211203, the Guangdong “Pearl River Talent Recruitment Program” under Grant 2019ZT08X603, and Shenzhen Science and Technology Innovation Commission (R2020A045). Prof. Coello Coello gratefully acknowledges support from CONACyT grant no. 2016-01-1920 (Investigación en Fronteras de la Ciencia 2016). He was also partially supported by the Basque Government through the BERC 2022-2025 program and by Spanish Ministry of Economy and Competitiveness MINECO: BCAM Severo Ochoa excellence accreditation SEV-2017-0718.

References

[1] Ali Ahrari, Saber Elsayed, Ruhul Sarker, Daryl Essam, and Carlos A. Coello Coello. Weighted pointwise prediction method for dynamic multiobjective optimization. *Inf. Sci.*, 546:349–367, 2021.

[2] Radhia Azzouz, Slim Bechikh, and Lamjed Ben Said. A dynamic multi-objective evolutionary algorithm using a change severity-based adaptive population management strategy. *Soft Comput.*, 21(4):885–906, 2017.

[3] Radhia Azzouz, Slim Bechikh, and Lamjed Ben Said. Dynamic multi-objective optimization using evolutionary algorithms: a survey. In *Recent Adv. Evol. multi-objective Optim.*, pages 31–70. Springer, 2017.

[4] Leilei Cao, Lihong Xu, Erik D Goodman, Chunteng Bao, and Shuwei Zhu. Evolutionary dynamic multiobjective optimization assisted by a support vector regression predictor. *IEEE Trans. Evol. Comput.*, 24(2):305–319, 2019.

[5] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: a library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):1–27, 2011.

[6] Qingda Chen, Jinliang Ding, Shengxiang Yang, and Tianyou Chai. A novel evolutionary algorithm for dynamic constrained multiobjective optimization problems. *IEEE Trans. Evol. Comput.*, 24(4):792–806, 2019.

[7] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. Boosting for Transfer Learning. In *Proc. 24th Int. Conf. Mach. Learn.*, pages 193–200. Association for Computing Machinery, 2007.

[8] Kalyanmoy Deb and Ram Bhushan Agrawal. Simulated binary crossover for continuous search space. *Complex Syst.*, 9(2):115–148, 1995.

[9] Kalyanmoy Deb, Mayank Goyal, et al. A combined genetic adaptive search (geneas) for engineering design. *Comput. Sci. informatics*, 26:30–45, 1996.

[10] Kalyanmoy Deb, S Karthik, and Others. Dynamic multi-objective optimization and decision-making using modified NSGA-II: A case study on hydro-thermal power scheduling. In *Int. Conf. Evol. multi-criterion Optim.*, pages 803–817. Springer, 2007.

[11] Joaquín Derrac, Salvador García, Daniel Molina, and Francisco Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm Evol. Comput.*, 1(1):3–18, 2011.

[12] Marco Farina, Kalyanmoy Deb, and Paolo Amato. Dynamic multiobjective optimization problems: test cases, approximations, and applications. *IEEE Trans. Evol. Comput.*, 8(5):425–442, 2004.

[13] Liang Feng, Wei Zhou, Weichen Liu, Yew-Soon Ong, and Kay Chen Tan. Solving dynamic multiobjective problem via autoencoding evolutionary search. *IEEE Trans. Cybern.*, pages 1–14, 2020.

[14] Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *2011 Int. Conf. Comput. Vis.*, pages 999–1006. IEEE, 2011.

[15] Abhishek Gupta, Yew-Soon Ong, and Liang Feng. Insights on transfer optimization: Because experience is the best teacher. *IEEE Trans. Emerg. Top. Comput. Intell.*, 2(1):51–64, 2018.

[16] Yaru Hu, Jinhua Zheng, Juan Zou, Shengxiang Yang, Junwei Ou, and Rui Wang. A dynamic multi-objective evolutionary algorithm based on intensity of environmental change. *Inf. Sci.*, 523:49–62, 2020.

[17] Min Jiang, Zhongqiang Huang, Liming Qiu, Wenzhen Huang, and Gary G Yen. Transfer learning-based dynamic multiobjective optimization algorithms. *IEEE Trans. Evol. Comput.*, 22(4):501–514, 2017.

[18] Min Jiang, Liming Qiu, Zhongqiang Huang, and Gary G Yen. Dynamic multi-objective estimation of distribution algorithm based on domain adaptation and nonparametric estimation. *Inf. Sci.*, 435:203–223, 2018.

[19] Min Jiang, Zhenzhong Wang, Shihui Guo, Xing Gao, and Kay Chen Tan. Individual-based transfer learning for dynamic multiobjective optimization. *IEEE Trans. Cybern.*, 51(10):4968–4981, 2021.

[20] Min Jiang, Zhenzhong Wang, Haokai Hong, and Gary G Yen. Knee Point-Based Imbalanced Transfer Learning for Dynamic Multiobjective Optimization. *IEEE Trans. Evol. Comput.*, 25(1):117–129, 2020.

[21] Min Jiang, Zhenzhong Wang, Liming Qiu, Shihui Guo, Xing Gao, and Kay Chen Tan. A fast dynamic evolutionary multiobjective algorithm via manifold transfer learning. *IEEE Trans. Cybern.*, 51(7):3417–3428, 2021.

[22] Shouyong Jiang and Shengxiang Yang. A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization. *IEEE Trans. Evol. Comput.*, 21(1):65–82, 2016.

[23] Shouyong Jiang, Shengxiang Yang, Xin Yao, Kay Chen Tan, and Marcus Kaiser. Benchmark Problems for CEC2018 Competition on Dynamic Multiobjective Optimisation. *CEC2018 Compet.*, pages 1–18, 2018.

[24] Shouyong Jiang, Shengxiang Yang, Xin Yao, Kay Chen Tan, Marcus Kaiser, and Natalio Krasnogor. Benchmark problems for CEC2018 competition on dynamic multiobjective optimisation. In *CEC2018 Competition.*, pages 1–18, 2018.

[25] Stephen C Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

[26] Ian Jolliffe. Principal component analysis. *Encycl. Stat. Behav. Sci.*, 2005.

[27] Jianqiang Li, Tao Sun, Qiuzhen Lin, Min Jiang, and Kay Chen Tan. Reducing negative transfer learning via clustering for dynamic multiobjective optimization. *IEEE Trans. Evol. Comput.*, pages 1–1, 2022.

[28] Zhengping Liang, Shunxiang Zheng, Zexuan Zhu, and Shengxiang Yang. Hybrid of memory and prediction strategies for dynamic multiobjective optimization. *Inf. Sci.*, 485:200–218, 2019.

[29] Qiuzhen Lin, Songbai Liu, Ka-Chun Wong, Maoguo Gong, Carlos A Coello Coello, Jianyong Chen, and Jun Zhang. A clustering-based evolutionary algorithm for many-objective optimization problems. *IEEE Trans. Evol. Comput.*, 23(3):391–405, 2018.

[30] Xuemin Ma, Jingming Yang, Hao Sun, Ziyu Hu, and Lixin Wei. Multiregional co-evolutionary algorithm for dynamic multiobjective optimization. *Inf. Sci.*, 545:1–24, 2021.

[31] Arrchana Muruganantham, Kay Chen Tan, and Prahlad Vadakkepat. Evolutionary dynamic multiobjective optimization via Kalman filter prediction. *IEEE Trans. Cybern.*, 46(12):2862–2873, 2015.

[32] Hidehiro Nakano, Masataka Kojima, and Arata Miyauchi. An artificial bee colony algorithm with a memory scheme for dynamic optimization problems. In *2015 IEEE Congr. Evol. Comput.*, pages 2657–2663. IEEE, 2015.

[33] Akira Oyama, Taku Nonomura, and Kozo Fujii. Data mining of Pareto-optimal transonic airfoil shapes using proper orthogonal decomposition. *J. Aircr.*, 47(5):1756–1762, 2010.

[34] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Trans. Knowl. Data Eng.*, 22(10):1345–1359, 2009.

[35] Miao Rong, Dunwei Gong, Witold Pedrycz, and Ling Wang. A multimodel prediction method for dynamic multiobjective evolutionary optimization. *IEEE Trans. Evol. Comput.*, 24(2):290–304, 2019.

[36] Miao Rong, Dunwei Gong, Yong Zhang, Yaochu Jin, and Witold Pedrycz. Multidirectional prediction approach for dynamic multiobjective optimization problems. *IEEE Trans. Cybern.*, 49(9):3362–3374, 2018.

[37] Hao Sun, Anran Cao, Ziyu Hu, Xiaxia Li, and Zhiwei Zhao. A novel quantile-guided dual prediction strategies for dynamic multi-objective optimization. *Inf. Sci.*, 579:751–775, 2021.

[38] Kay Chen Tan, Liang Feng, and Min Jiang. Evolutionary transfer optimization-a new frontier in evolutionary computation research. *IEEE Comput. Intell. Mag.*, 16(1):22–33, 2021.

[39] Michael E Wall, Andreas Rechtsteiner, and Luis M Rocha. Singular value decomposition and principal component analysis. In *A Pract. approach to microarray data Anal.*, pages 91–109. Springer, 2003.

[40] Zhenkun Wang, Qingfu Zhang, Maoguo Gong, and Aimin Zhou. A replacement strategy for balancing convergence and diversity in moea/d. In *Proc. IEEE Congr. Evol. Comput. (CEC)*, pages 2132–2139, 2014.

[41] Zhenkun Wang, Qingfu Zhang, Aimin Zhou, Maoguo Gong, and Licheng Jiao. Adaptive replacement strategies for moea/d. *IEEE Trans. Cyber.*, 46(2):474–486, 2016.

[42] Lyndon While, Philip Hingston, Luigi Barone, and Simon Huband. A faster algorithm for calculating hypervolume. *IEEE Trans. Evol. Comput.*, 10(1):29–38, 2006.

[43] Huipeng Xie, Juan Zou, Shengxiang Yang, Jinhua Zheng, Junwei Ou, and Yaru Hu. A decision variable classification-based cooperative coevolutionary algorithm for dynamic multiobjective optimization. *Inf. Sci.*, 560:307–330, 2021.

[44] Yi Yao and Gianfranco Doretto. Boosting for transfer learning with multiple sources. In *2010 IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pages 1855–1862, 2010.

[45] Huan Zhang, Jinliang Ding, Min Jiang, Kay Chen Tan, and Tianyou Chai. Inverse gaussian process modeling for evolutionary dynamic multiobjective optimization. *IEEE Trans. Cybern.*, early access, 2021. doi: 10.1109/TCYB.2021.3070434.

[46] Qingfu Zhang and Hui Li. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.*, 11(6):712–731, 2007.

[47] Qingfu Zhang, Aimin Zhou, and Yaochu Jin. RM-MEDA: A regularity model-based multiobjective estimation of distribution algorithm. *IEEE Trans. Evol. Comput.*, 12(1):41–63, 2008.

[48] Qi Zhao, Bai Yan, Yuhui Shi, and Martin Middendorf. Evolutionary dynamic multiobjective optimization via learning from historical search process. *IEEE Trans. Cybern.*, early access, 2021. doi: 10.1109/TCYB.2021.3059252.

[49] Aimin Zhou, Yaochu Jin, and Qingfu Zhang. A population prediction strategy for evolutionary dynamic multiobjective optimization. *IEEE Trans. Cybern.*, 44(1):40–53, 2013.

[50] Fei Zou, Gary G. Yen, and Chen Zhao. Dynamic multiobjective optimization driven by inverse reinforcement learning. *Inf. Sci.*, 575:468–484, 2021.