Knowledge Transfer for Dynamic Multi-objective Optimization with a Changing Number of Objectives

Gan Ruan, Leandro L. Minku, *Senior Member, IEEE,* Stefan Menzel, Bernhard Sendhoff, *Fellow, IEEE,* and Xin Yao, *Fellow, IEEE*

***Abstract*—Different from most other dynamic multi-objective optimization problems (DMOPs), DMOPs with a changing number of objectives usually result in expansion or contraction of the Pareto front or Pareto set manifold. Knowledge transfer has been used for solving DMOPs, since it can transfer useful information from solving one problem instance to solve another related problem instance. However, we show that the state-of-the-art transfer algorithm for DMOPs with a changing number of objectives lacks sufficient diversity when the fitness landscape and Pareto front shape present nonseparability, deceptiveness or other challenging features. Therefore, we propose a knowledge transfer dynamic multi-objective evolutionary algorithm (KTDMOEA) to enhance population diversity after changes by expanding/contracting the Pareto set in response to an increase/decrease in the number of objectives. This enables a solution set with good convergence and diversity to be obtained after optimization. Comprehensive studies using 13 DMOP benchmarks with a changing number of objectives demonstrate that our proposed KTDMOEA is successful in enhancing population diversity compared to state-of-the-art algorithms, improving optimization especially in fast changing environments.**

***Index Terms*—Evolutionary algorithms, Multi-objective optimization, Dynamic optimization, Changing objectives, Knowledge transfer**

I. Introduction

Dynamic multi-objective optimization problems (DMOPs) [1], widely existing in the real-world [2–4], are a kind of multi-objective optimization problems which comprise a series of problems whose objectives change over time [5]. Due to the dynamics in objective functions of DMOPs, the Pareto sets (PSs) and/or Pareto fronts (PFs) may change over time. Therefore, how to efficiently track the changing PSs/PFs is a key problem in solving DMOPs. Facing this challenge, many strategies have been proposed to tackle the dynamics in DMOPs. They can be classified as diversity enhancement [2, 6–9], memory techniques [10, 11], prediction strategies [12–15] and knowledge transfer-based methods [16–19].

However, few studies have been done to solve DMOPs with a changing number of objectives (NObj), even though this type of problems also widely exist in the real-world [20–23]. For instance, in project scheduling, people usually urge to minimize the makespan and total salary cost simultaneously. However, given a tighter deadline must be met, the salary cost may not be important and would not be considered as an objective any longer under the new circumstance [21–23]. In some other examples, if an application is running on a system with a wired power supply, there is no need to consider the energy consumption [24]. However, once the system is detached from the wired power supply and only relies on batteries, minimizing the energy consumption thus becomes a new objective. In the water resource management problem [25], the construction cost of the water resource management system is the main objective to be minimized. However, climate changes may result in unexpected flood, such that minimizing the expected cost induced by the flood becomes a new objective. In the car side impact problem [26], the pubic force experienced by a passenger and the average velocity of the V-pillar responsible for withstanding the impact load are usually minimized. Given a tighter budget when designing a car, minimizing the weight of the car may become a new objective. This dynamic scenario may also happen in crash safety design of vehicles [27].

One of the most recent work for solving DMOPs with a changing NObj is the proposal of the Dynamic Two Archive Evolutionary Algorithm (DTAEA) [28], in which the research significance of DMOPs with a changing NObj has been highlighted by several real-world problems in software engineering [20], project scheduling [21–23], etc [24, 29, 30]. The main idea of DTAEA was to simultaneously maintain two co-evolving populations, i.e, a convergence archive (CA) and a diversity archive (DA) during the evolution. Whenever environmental changes occur, CA and DA are reconstructed to preserve as much convergence and diversity as they can in the new environment.

Considering that the reconstruction of CA and DA in DTAEA involves copying (optimal) solutions from the past problem instance to the next after changes, DTAEA can be regarded as a kind of knowledge transfer-based algorithm, as it makes use of knowledge acquired from solving the previous problem instance to solve the new problem instance. However, in this study, we show that DTAEA cannot handle DMOPs with a changing NObj containing more complex problem features including PF shapes (convex, discontinuous and mixed shape of convex and concave) and fitness land-

Gan Ruan, Leandro L. Minku and Xin Yao are with CERCIA, School of Computer Science, University of Birmingham, UK. (Emails: gxr847@student.bham.ac.uk,l.l.minku@bham.ac.uk,xinyao@ln.edu.hk)

Stefan Menzel and Bernhard Sendhoff are with the Honda Research Institute Europe GmbH, 63073 Offenbach, Germany. (email: stefan.menzel@honda-ri.de, bernhard.sendhoff@honda-ri.de)

Xin Yao is also with Department of Computing and Decision Sciences, Lingnan University, Hong Kong SAR, China.

scapes (nonseparability and deceptiveness) well. Specifically, the knowledge transfer (i.e. CA and DA reconstruction) in DTAEA is incapable of providing enough diversity in these complex scenarios. The reason is that the change in the NObj changes the distribution of reconstructed CA on the true PF for the new problem instance and the uniformly sampled solutions in the search space by the DA reconstruction are not uniformly distributed in the objective space due to problem features in the more complex problems.

In this paper, we aim to answer the following research questions:

- How to increase diversity when solving DMOPs with a changing NObj, so as to improve knowledge transfer right after changes?
- How does knowledge transfer help the optimization process itself?

In order to answer these research questions, we propose to expand or contract the PS of the problem after NObj increases or decreases, respectively, to improve the knowledge transfer. This strategy works better than DTAEA because DMOPs with a changing NObj usually result in the expansion or contraction of the dimension of the PS manifold [28]. Experimental studies have been carried out on 13 DMOPs with a changing NObj, modified from 4 DTLZ [31] and 9 WFG [32] problems to demonstrate the effectiveness of our proposed approach.

The novel contributions of our work are summarized as follows:

- Comprehensive experiments have been carried out on representative problems with complex problem features in the fitness landscape (nonseparability and deceptiveness) and complex PF shapes (convex, discontinuous and mixed shape of convex and concave) to understand the limitations of the state-of-the-art algorithm DTAEA. Our analyses reveal that DTAEA lacks diversity when solving more complex DMOPs with a changing NObj;
- A novel knowledge transfer-based method, called knowledge transfer dynamic multi-objective evolutionary algorithm (KTDMOEA), is proposed. This method proposes PS expansion and contraction mechanisms to enhance diversity for dealing with changing NObj in DMOPs;
- Systematic computational studies have been conducted to compare our proposed KTDMOEA with 5 algorithms on 13 DMOPs with a changing NObj under different frequencies and types of changes in the NObj. Experimental results have shown that our algorithm is competitive against all compared algorithms.

The remainder of this paper is organized as follows. Section II describes related work on DMOPs with a changing NObj and evolutionary transfer optimization as well as the motivation of our proposal. The proposed knowledge transfer-based algorithm is elaborated in Section III. Section IV describes the specific experimental setup. The experimental results are presented in detail in Section V. Section VI concludes this paper and points out possible future work.

II. Related Work and Motivations

This section firstly reviews related work on DMOPs with a changing NObj and evolutionary transfer optimization. Then, a preliminary investigation of the existing work DTAEA [28] is conducted to reveal its limitations on solving DMOPs with complex problem features.

A. DMOPs with a Changing NObj

In this paper, we focus on the continuous minimized DMOPs defined as follows:

$$\begin{cases} \min \ \mathbf{F}(\mathbf{x},t) = \big(f_1(\mathbf{x},t),\dots,f_{m(t)}(\mathbf{x},t)\big)^T \\ s.t. \ \ \mathbf{x} \in \Omega, \ \ t \in \Omega_t \end{cases} \tag{1}$$

where $\Omega \subseteq R^n$ is the decision (variable) space; t is the discrete time instance; $\Omega_t \subseteq R$ is the time space. $\mathbf{F}(\mathbf{x},t)$: $\Omega \times \Omega_t \rightarrow R^{m(t)}$ is the objective function vector that evaluates a candidate solution $\mathbf{x} = (x_1, ..., x_n)$ at time t. $m(t)$ is the number of objective at time t.

Note that equation (1) is a general definition of the changing number of objectives. There are many special cases of the changing number of objectives. In particular, suppose a bi-objective problem $\big(f_1(\mathbf{x},t), f_2(\mathbf{x},t)\big)^T$ at time step t and this problem changes to a tri-objective problem $\big(f_1(\mathbf{x},t+1), f_2(\mathbf{x},t+1), f_3(\mathbf{x},t+1)\big)^T$ at time step $t+1$. The two functions in the bi-objective problem might be the exactly the same as the first two functions of the tri-objective problem, i.e., $f_1(\mathbf{x},t) = f_1(\mathbf{x},t+1)$ and $f_2(\mathbf{x},t) = f_2(\mathbf{x},t+1)$. Or, the two functions of the tri-objective optimization problem might be splitted by one of the bi-objective optimization problem, i.e., $f_1(\mathbf{x},t) = f_1(\mathbf{x},t+1) + f_2(\mathbf{x},t+1)$ and $f_2(\mathbf{x},t) = f_3(\mathbf{x},t+1)$. More generally, there may be none of the functions at time step $t+1$ which is the same as one of the functions at time step t, i.e., $\big\{f_1(\mathbf{x},t), f_2(\mathbf{x},t)\big\} \cap \big\{f_1(\mathbf{x},t+1), f_2(\mathbf{x},t+1), f_3(\mathbf{x},t+1)\big\} = \emptyset$. Most generally, the NObjs could increase or decrease by more than one each time. The approach proposed in this paper is generally applicable to different cases.

Although people have realized the importance of tackling DMOPs with a changing NObj and mention this concept in [30, 33–35], few work existed studying this problem until recently [28]. Recently, a comprehensive investigation was conducted on the challenges of DMOPs with a changing NObj in [28]. It has been experimentally demonstrated that it is a key issue of how to propel crowded solutions to cover the whole PF and how to pull unconverged solutions back to the PF with good diversity when increasing and decreasing the NObj, respectively. Bearing this challenge in mind, the authors in [28] proposed DTAEA to tackle DMOPs with a changing NObj, in which two complementary populations, CA and DA, are simultaneously maintained in the evolution process to focus on population convergence and diversity, respectively. Whenever environmental changes occur, CA and DA are reconstructed to preserve as much convergence and diversity as they can in the new environment. More specifically, when increasing the NObj, solutions in the old CA are all copied to the new CA. When decreasing the NObj, nondominated and dominated solutions of the old CA are all copied to the new CA and new DA, respectively. Therefore, DTAEA can be seen as a kind of knowledge transfer-based algorithm, as it makes use of knowledge acquired from previous solutions. Later on, two novel transfer learning-based algorithms [36, 37] were proposed for solving DMOPs with a changing NObj, in which the geodesic flow kernel and self-organizing map are used, respectively.

B. Evolutionary Transfer Optimization

Knowledge transfer has been applied to evolutionary computation to solve mutli-objective optimization and dynamic multi-objective optimization problems [38]. Specifically, knowledge transfer is able to learn useful knowledge from related problem instances to solve the targeted problem instance [39, 40]. However, evolutionary multi-tasking optimization (EMT) [39–42] differs from our scenario here because EMT considers solving multiple tasks simultaneously, while our work considers solving different problem instances sequentially as the environment, e.g., NObj, changes. At any given time, we solves only one problem instance, not multiple ones.

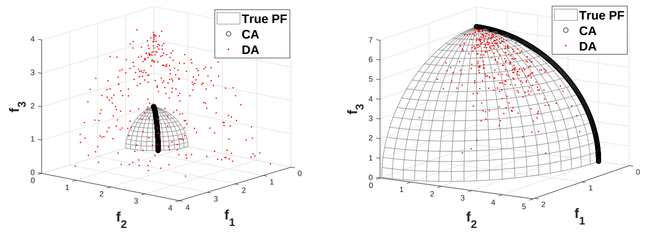
For dynamic multi-objective optimization, knowledge transfer can help to predict good solutions for the next change based on previously optimized solutions. The transfer learning-based dynamic multi-objective evolutionary algorithm (Tr-DMOEA) [16] was the first work of applying knowledge transfer to solve DMOPs, in which transfer component analysis is used to transfer solutions in the PF of the previous environment to generate an initial population for the next environment. An autoencoding evolutionary search is regarded as a knowledge transfer method to predict the moving of PSs based on nondominated solutions obtained before the change [43]. In [18], a manifold transfer learning method is applied to forecast the changing PSs over time. In general, it is always very challenging to decide what, when and how to transfer in DMOPs [44, 45].

C. Motivations

In this section, the limitations of existing algorithms for solving DMOPs are analyzed in detail.

1) Limitations of Existing DMOEAs except for DTAEA: The limitations of existing DMOEAs except for DTAEA in dealing with changing NObj are analyzed as follows.

- The diversity enhancement methods [2, 9] always randomly or heuristically produce some solutions to respond to environmental changes. However, these generated solutions can hardly provide enough diversity along the PF when increasing the NObj, as they are randomly or heuristically generated in the decision space without converging to the PF and diversifying along the PF. When decreasing the NObj, they are hardly able to increase population convergence, since they are generated to track the changing PS positions without considering PS contraction.
- The memory technique in [30] has been applied to deal with the changing NObj. Specifically, the population obtained in the previous environment is stored in a memory and then simply re-evaluated whenever the NObj changes. This strategy is not enough to provide appropriate diversity and convergence when dealing with changing NObj, since it is impossible that optimal solutions of the old problem are still optimal on the changed problem.
- Existing prediction strategies are normally proposed to learn the PSs movement in past environments and predict the PS after the change. However, they are rarely able to predict how PS expands or contracts in DMOPs with a changing NObj, since they only consider learning the PSs movement rather than PS expansion or contraction.
- Similar to the prediction-based strategies, most knowledge transfer-based DMOEAs never considered changing NObj and cannot solve DMOPs with a changing NObj, since they were designed to track the changing position and/or shape of PSs and/or PFs rather than expand or contract PS/PF.
- Recently, two knowledge transfer-based DMOEAs [36, 37] were proposed for solving DMOPs with a changing NObj. However, they do not have good performance when increasing the NObj. The reason is that the transferred solutions have poor convergence even though they are made more diversified when increasing the NObj. As a result, they need more generations to reconverge. Moreover, the transfer learning methods used in [36, 37] are more time-consuming than heuristic methods.

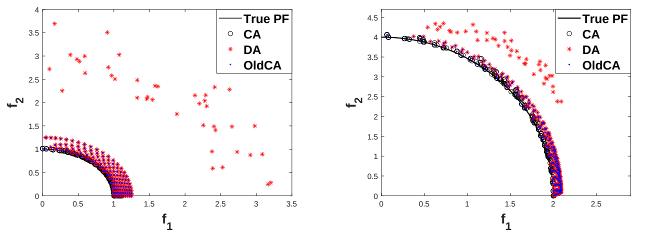


(a) F2 (b) WFG4

Fig. 1. Distribution of the reconstructed CA and DA obtained by DTAEA in the first generation right after changes when increasing NObj from 2 to 3 on F2 and WFG4.

2) A Preliminary Investigation Revealing DTAEA's Weaknesses: Even though DTAEA has been computationally demonstrated in [28] to be effective on DMOPs with a changing NObj based on knowledge transfer, the test problems that were used to evaluate DTAEA are somewhat limited, as problem features in those problems are relatively simple, such as linear or concave PF shape and fitness landscape with multimodality, bias or even nothing.

In order to evaluate whether DTAEA is able to solve DMOPs with a changing NObj and more complex problem features including PF shapes (convex, discontinuous and mixed shape of convex and concave) and fitness landscapes (nonseparability and deceptiveness), a benchmark problem WFG4 is arbitrarily selected from the WFG suite [32] as an example to conduct an experimental investigation of the performance of DTAEA. In contrast, F2 [28] is arbitrarily selected from the DTLZ suite [31]. When increasing the NObj, the problems are set as bi-objective problems and then given 1000 generations to evolve by DTAEA before increasing the NObj from 2 to 3. When decreasing the NObj, the problems are set as tri-objective problems and then given 1000 generations to evolve by DTAEA before decreasing the NObj from 3 to 2.



(a) F2 (b) WFG4

Fig. 2. Distribution of the reconstructed CA and DA obtained by DTAEA in the first generation right after changes when decreasing NObj from 3 to 2 on F2 and WFG4.

Figs. 1 and 2 show the distribution of the old CA, the reconstructed CA and DA obtained by DTAEA on the two problems F2 and WFG4 in the first generation right after changes for the cases of increasing the NObj from 2 to 3 and decreasing the NObj from 3 to 2, respectively. It should be noted that when increasing the NObj, solutions in the old CA are all copied to the new CA. Therefore, in Fig. 1, 'CA' represents both solutions in the old CA and the new CA. It is clear from Fig. 1 that when increasing the NObj from 2 to 3, the new CA does not have good diversity on both F2 and WFG4. As for the reconstructed DA, it has a good level of diversity on F2. As shown in Fig. 1, solutions randomly generated in the search space are covering the whole area over the true PF. However, on WFG4, solutions in the reconstructed DA only cover a part of the PF over it.

Similarly, it can be seen from Fig. 2 that solutions in DA also have good diversity on F2 when decreasing NObj from 3 to 2. However, on WFG4, there are some areas close to the high values of the second objective in the objective space without any solutions covered by the DA. It is clear that on both problems and the two cases (increasing and decreasing the NObj), the reconstructed CA and DA do not provide enough diversity. Therefore, the CA and DA reconstruction of DTAEA cannot provide enough diversity on DMOPs with more complex problem features. The reason is that the problem features in the more complex problems cause uniformly sampled solutions in the search space not to be uniformly distributed in the objective space.

III. Knowledge Transfer Dynamic Multi-objective Evolutionary Algorithm (KTDMOEA)

In this section, we present our proposed knowledge transfer dynamic multi-objective evolutionary algorithm (denoted as KTDMOEA), which is designed to tackle more complex DMOPs with a changing NObj. The main component of KTDMOEA is the proposed diversity enhanced knowledge transfer, which is designed to improve population diversity right after changes in DMOPs with a changing NObj. KTDMOEA's flowchart is shown in Fig. 3. As the flowchart exhibits, KTDMOEA maintains a single population. Whenever there is a change in the NObj, the process of knowledge transfer is evoked to reconstruct the population such that it has increased diversity; otherwise, other procedures in the evolution process are carried out. The novel process of knowledge transfer through PS expansion/contraction proposed in this paper will be elaborated in Section III-A. Then, Section

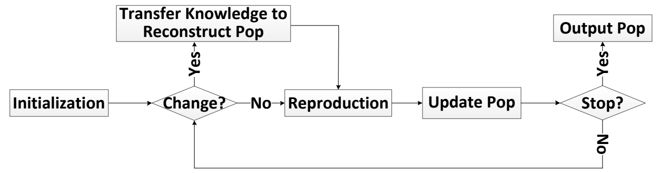
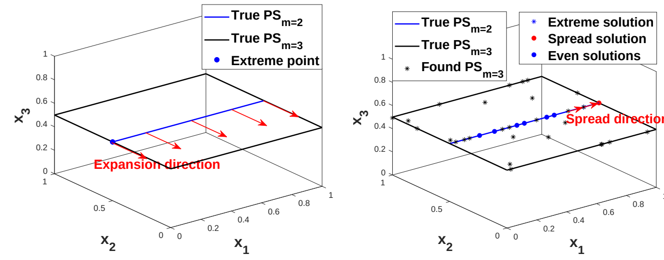


Fig. 3. Flow chart of KTDMOEA.



(a) PS expansion (b) PS contraction

Fig. 4. Brief illustration of how to expand and contract the PS for increasing and decreasing NObj, respectively.

III-B presents the overall evolutionary process of the proposed KTDMOEA.

A. Diversity Enhancing Knowledge Transfer

As DMOPs with increasing/decreasing NObj result in the expansion/contraction of PS/PF, we propose to enhance diversity through PS expansion and contraction for increasing and decreasing NObj, respectively. This strategy is targeted at enhancing knowledge transfer right after changes. In this section, the specific details of PS expansion and contraction are given in Section III-A1 and Section III-A2, respectively.

1) Expand the PS when Increasing the NObj: Increasing the NObj usually results in the expansion of the dimension of PS/PF manifold. Therefore, the PS is proposed to be expanded when increasing the NObj, so as to increase the population diversity.

The idea of PS expansion in the decision space is illustrated in Fig. 4(a). Note this figure is just drawn to demonstrate the process of PS expansion, the specific PF and expansion direction in real problems may be different. As shown in Fig. 4(a), suppose the blue point is one extreme point in the PS before the change (PS_t); blue line is the Pareto optimal set at time step t with two NObj; the expansion direction is found by generating several solutions around the blue point and connecting the blue point to the point nondominated to it; the plane formed by 4 black lines is the Pareto optimal set at time step $t+1$ with three NObj and the red arrows are the expansion directions. Solutions evenly selected in PS_t, which are the points in the starting points of the red arrows, are regarded as the PS expansion base solutions to cover the whole PS right after the change (PS_{t+1}).

The framework of PS expansion is exhibited in Algorithm 1. As Algorithm 1 presents, in order to achieve PS expansion, the first step is to search for the potential PS expansion directions, whose procedure is given in Algorithm 2 and explained as follows. Given a set of Pareto optimal solutions at the time step t ($\mathbf{PS}_t$), the algorithm firstly finds solutions with the maximum objective value for each objective as the set of extreme points (denoted as $\mathbf{P}_e$) in line 1 of Algorithm 2. The reason for selecting one extreme point is to ensure that

Algorithm 1: Expand the PS to generate transferred solutions

Input: Pareto optimal solution set at time t ($\mathbf{PS}_t$); set of the searched expansion directions $\mathbf{D}$ with size N_{dir}; population size N; the NObj for the old problem before the change M_t; number of solutions to generate along each expansion direction θ;
Output: Transferred solutions $\mathbf{P}_{tr}$

1 Search a set of expansion direction as $\mathbf{D}$ via Algorithm 2;
2 Evenly select solutions from $\mathbf{PS}_t$ in the objective space with the size of $N_{base} = \lfloor \frac{N-M_t}{N_{dir}*\theta} \rfloor$ as $\mathbf{P}_{base}$;
3 Generate θ solutions along each direction in $\mathbf{D}$ to fill $\mathbf{P}_{tr}$ through the following equation (3);
4 Evenly select solutions from $\mathbf{PS}_t$ in the objective space with the size of $N - N_{base} * \theta * N_{dir}$ to fill $\mathbf{P}_{tr}$;
Return $\mathbf{P}_{tr}$.

Algorithm 2: Search of Expansion Direction

Input: Pareto optimal solution set at time t ($\mathbf{PS}_t$)
Output: Set of the searched expansion directions $\mathbf{D}$ or NULL.

1 Find the solutions with the maximum objective value for each objective as the set of extreme points ($\mathbf{P}_e$);
2 Randomly select one solution from P_e and regard it as $\mathbf{x}_e$;
3 Produce a solution set $\mathbf{P}_{var}$ (also called detective population) with the same size as the population size through randomly generating solutions via the polynomial mutation [46] around the selected extreme point x_e;
4 Evaluate all solutions in $\mathbf{P}_{var}$ and delete dominated solutions after conducting nondominated sorting on them;
5 **if** *All solutions in $\mathbf{P}_{var}$ are nondominated by $\mathbf{PS}_t$* **then**
6   Set $\mathbf{P}_{non}$ as $\mathbf{P}_{var}$;
end
7 **else**
8   Delete solutions from $\mathbf{P}_{var}$ that are dominated by those in the $\mathbf{PS}_t$ and set the remaining solutions as $\mathbf{P}_{non}$;
end
9 Use evenly generated weight vectors following the method in [28] to estimate density of $\mathbf{P}_{non}$ and $\mathbf{PS}_t$ with the method introduced in Section III-B;
10 Delete solutions in $\mathbf{P}_{non}$ located at the same subarea with solutions of $\mathbf{PS}_t$;
11 **if** *$\mathbf{P}_{non}$ is NULL* **then**
    Return NULL.
end
12 **else**
13   Use the remaining solutions in $\mathbf{P}_{non}$ and the extreme point $\mathbf{x}_e$ to form a set of lines that represent the directions (denoted as $\mathbf{D}$): $\mathbf{D}_j = \frac{\mathbf{P}_{non}^j - \mathbf{x}_e}{\|\mathbf{P}_{non}^j - \mathbf{x}_e\|}, (j = 1, ..., |\mathbf{P}_{non}|)$;
14   Delete duplicated directions from $\mathbf{D}$;
end
Return $\mathbf{D}$.

the found expansion directions are not misleading, since the expansion directions are formed by the extreme point and its nondominated solutions. If middle points in the PF were selected, the found expansion directions might be along or within the PF of the old problem, thus wasting computational resources. Then, in line 2, a random extreme point $\mathbf{x}_e$ is selected from the set $\mathbf{P}_e$ as the initial point of expansion directions. Later on, a solution set $\mathbf{P}_{var}$ with the same size as the population size is randomly produced around $\mathbf{x}_e$ via the polynomial mutation [46] to be regarded as the candidate sets of end points of expansion directions in line 3, which is also called detective population.

Procedures from lines 4 to 10 in Algorithm 2 are conducted to make sure the remaining solutions in $\mathbf{P}_{non}$ are nondominated and in different sub-spaces from the extreme point such that the extreme point and them can form the right expansion directions. Therefore, in line 4, all solutions in $\mathbf{P}_{var}$ are evaluated in the new environment and dominated solutions are discarded after sorting them using nondominated sorting [47]. Then, if all solutions in $\mathbf{P}_{var}$ are nondominated by those in PS_t, just set $\mathbf{P}_{non}$ as $\mathbf{P}_{var}$; else delete all solutions from $\mathbf{P}_{var}$ that are dominated by those in $\mathbf{PS}_t$ and regard the set of remaining solutions as $\mathbf{P}_{non}$ in line 8. Then, in line 9, use evenly generated weight vectors following the method in [28] to estimate density of $\mathbf{P}_{non}$ and $\mathbf{PS}_t$ with the method introduced in Section III-B. Delete those solutions from $\mathbf{P}_{non}$ that are in the same subarea as those in $\mathbf{PS}_t$ in line 10. Later on, if there is no solution in $\mathbf{P}_{non}$, this means no expansion direction is found and return NULL; else, in line 13, use the points to form a set of lines that represents the directions (denoted as $\mathbf{D}$) by regarding the extreme point $\mathbf{x}_e$ in line 2 as the starting point and those solutions in $\mathbf{P}_{non}$ as the end point: $\mathbf{D}_j = \frac{\mathbf{P}_{non}^j - \mathbf{x}_e}{\|\mathbf{P}_{non}^j - \mathbf{x}_e\|}, (j = 1, ..., |\mathbf{P}_{non}|)$. Then, delete duplicated expansion directions from $\mathbf{D}$ and return $\mathbf{D}$.

After getting the expansion directions, the next step is to expand the PS to generate transferred solutions following the expansion directions. The detailed procedures of this algorithm are shown in Algorithm 1. Given the Pareto optimal solution set at time t $\mathbf{PS}_t$, evenly select solutions from it with the size of N_{base} as $\mathbf{P}_{base}$, where

$$N_{base} = \lfloor \frac{N - M_t}{N_{dir} * \theta} \rfloor \tag{2}$$

where N is the size of population; M_{t+1} is the NObj at time step t+1; N_{dir} is the number of expansion directions in $\mathbf{D}$ and θ is the number of solutions to generate along each expansion direction, which is a parameter to be set by the user. $N - M_t$ is designed to enable those M_t extreme points in $\mathbf{PS}_t$ to be preserved to the next environment. Then in line 3 of Algorithm 1, generate θ solutions along each expansion direction in $\mathbf{D}$ to fill the transferred solution set $\mathbf{P}_{tr}$ through the following Equation (3), which produces a transferred solution based on a base solution and an expansion direction.

$$\mathbf{x}_{new}^{(j-1)*N_{dir}+i} = \mathbf{x}_i + C_i^j * rand() * \mathbf{D}_j \quad (i = 1, ..., N_{base}; j = 1, ..., N_{dir}) \tag{3}$$

where $\mathbf{x}_i$ is the i-th solution in the base population $\mathbf{P}_{base}$; C_i^j is a float variable enabling some expanded solutions to reach the boundary of the decision space, whose detailed calculation will be discussed in the next paragraph; rand() is a function to generate a random number in $(0, 1]$; $\mathbf{D}_j$ is the j-th expansion direction in the set $\mathbf{D}$. After generating transferred solutions through PS expansion, if $\mathbf{P}_{tr}$ is not full, just evenly select solutions from $\mathbf{PS}_t$ in the objective space with the size of $N - N_{base} * \theta * N_{dir}$.

The calculation of C_i^j should follow the criterion that all solutions expanded from the solutions $\mathbf{x}_i$ in $\mathbf{P}_{base}$ are within the bound of each decision variable and they should reach the boundary of the search space as close as possible. Bearing this criterion in mind, we design the calculation of C_i^j. Given a base solution $\mathbf{x}_i$ and one expansion direction $\mathbf{D}_j$, suppose $para^k$ is the value that makes the k-th variable of the generated solution reach the boundary of this variable. Therefore, each $para^k$ can be calculated according to whether the expansion direction is positive or negative, via the following equation:

$$para^k = \begin{cases} \frac{upper^k - x_i^k}{D_j^k}, & D_j^k > 0 \\ \frac{lower^k - x_i^k}{D_j^k}, & D_j^k < 0 \end{cases} \tag{4}$$

where $upper^k$ and $lower^k$ are the upper bound and lower bound of the k-th dimension of the decision space; x_i^k is the k-th decision value; D_j^k is the value of the direction $\mathbf{D}_j$ at the k-th dimension. In order to ensure each generated solution is located within the region, $C_i^j = \underset{k=1,...,n}{min}\ para^k$, where n is the dimension of the decision space.

2) Contract the PS when Decreasing the NObj: It has been observed that decreasing the NObj usually results in the contraction of the dimension of PS/PF manifold. Therefore, the PS is proposed to be contracted when decreasing the number of objective.

The idea of PS contraction in the decision space is illustrated in Fig. 4(b). Note this figure is just drawn to demonstrate the process of PS contraction, the specific cases in real problems may be different. It tries to generate spread and uniform solutions given current nondominated solution set. Steps from lines 4 to 7 in Algorithm 3 are designed to help improving the spread of the population and line 8 tries to make the distribution of population more even. As illustrated in Fig. 4(b), black start points are the found optimal solutions for the problem with 3 objectives. When decreasing the NObj from 3 to 2, the black start points in the true PS of bi-DMOP (blue line) are selected as the nondominated solutions.

One solution (denoted by the red point) is generated based on the extreme point (denoted by blue start point) following the spread direction between the extreme point's closest point and itself, so as to increase the spread of population. Other solutions (denoted by the blue points) are produced from two randomly selected solutions in the nondominated solution set, to improve the even distribution of the population (suppose there is no bias in the problem).

Algorithm 3: Contract the PS to generate transferred solutions

Input: Pareto optimal solution set at time t ($\mathbf{PS}_t$);
Output: Transferred solutions P_{tr}
1 Evaluate the $\mathbf{PS}_t$ in the new environment and put the nondominated solutions to $\mathbf{P}_{non}$ after conducting the nondominated sorting on $\mathbf{PS}_t$;
2 Put all solutions in $\mathbf{P}_{non}$ to $\mathbf{P}_{tr}$;
3 Find the solutions with the maximum objective value at any objective from $\mathbf{P}_{non}$ as the set of extreme points ($\mathbf{P}_e$);
4 **for** *j = 1 to* $|\boldsymbol{P}_e|$ **do**
5   Find a closest solution $\mathbf{P}_{non}^j$ to $\mathbf{x}_e^j$ from $\mathbf{P}_{non}$ and connect $\mathbf{P}_{non}^j$ to $\mathbf{x}_e^j$ as a direction and normalize it as $\mathbf{D}_j = \frac{\mathbf{x}_e^j - \mathbf{P}_{non}^j}{|\mathbf{x}_e^j - \mathbf{P}_{non}^j|}$;
6   Produce a new solution $\mathbf{x}_{new}$ along the direction $\mathbf{D}_j$ to make it reach the boundary of the search space;
7   Put the solution $\mathbf{x}_{new}$ to $\mathbf{P}_{tr}$;
end
8 Random select two solutions ($\mathbf{x}_a$ and $\mathbf{x}_b$) from $\mathbf{P}_{tr}$ and generate one solution between them to fill $\mathbf{P}_{tr}$ until the size reaches the population size N;
Return $\mathbf{P}_{tr}$

Given the Pareto optimal solution set $\mathbf{PS}_t$, the first step in line 1 of Algorithm 3 is to evaluate all solutions in $\mathbf{PS}_t$ in the new environment and put the nondominated solutions to $\mathbf{P}_{non}$ after conducting nondominated sorting [47] on $\mathbf{PS}_t$. Then, put all solutions of $\mathbf{P}_{non}$ to the set of transferred solution $\mathbf{P}_{tr}$ in line 2. Later on, in line 3, find solutions from $\mathbf{P}_{non}$ with the maximum objective value for each objective as the set of extreme points $\mathbf{P}_e$. Subsequently, for each extreme point $\mathbf{x}_e^j$ in the set $\mathbf{P}_e$, find a closest solution $\mathbf{P}_{non}^j$ to $\mathbf{x}_e^j$ from $\mathbf{P}_{non}$ and connect $\mathbf{P}_{non}^j$ to $\mathbf{x}_e^j$ as a direction and normalize it as $\mathbf{D}_j = \frac{\mathbf{x}_e^j - \mathbf{P}_{non}^j}{|\mathbf{x}_e^j - \mathbf{P}_{non}^j|}$, as shown in line 5. Then, produce a new solution $\mathbf{x}_{new}^i$ along $\mathbf{D}_i$ to make it reach the boundary of the search space according to the following equation:

$$\mathbf{x}_{new}^j = \mathbf{x}_e^j + C_j * \mathbf{D}_j \tag{5}$$

where C_j is a float variable, whose calculation method is the same as that in the process of PS expansion, as shown in equation (4). The newly generated solution $\mathbf{x}_{new}$ is then put in $\mathbf{P}_{tr}$. Later on, in line 8, randomly select two solutions ($\mathbf{x}_a$ and x_b) from P_{tr} and generate one solution between them through equation (6) to fill P_{tr} until the size of P_{tr} reaches the population size N, using the following equation:

$$\mathbf{x}_{new} = \mathbf{x}_a + rand() * (\mathbf{x}_a - \mathbf{x}_b) \tag{6}$$

We believe this strategy of expanding/contracting the PF and PS works better over DTAEA [28] because increasing or decreasing the NObj usually results in the expansion and contraction of the dimension of the PS manifold. When increasing the NObj, the proposed PS expansion is able to find the expansion directions and generate solutions along these directions, increasing the population diversity in the new environment. When decreasing the NObj, the two mechanisms in PS contraction are targeted for improving the spread and evenness of the distribution. Therefore, those produced solutions by PS expansion and contraction may achieve better diversity than that of DTAEA.

B. KTDMOEA

As the PS expansion/contraction is designed to enhance the diversity of population after changes, the proposed KTDMOEA will not maintain a separate DA. As a result, there is no need to update DA as in DTAEA. The population update mechanism in KTDMOEA is the same as the update mechanism of CA in [28].

Algorithm 4: Framework of KTDMOEA

Input: Population size N;
Output: The found population Pop
1 Randomly generate an initial population Pop;
2 **while** *stopping criteria not satisfied* **do**
3   **if** *the number of objective changes* **then**
4   **if** *the number of objective increases* **then**
5   Conduct the process of PS expansion in Algorithm 1 ;
          end
6   **else if** *the number of objective decreases* **then**
7   Conduct the process of PS contraction in Algorithm 3;
          end
      end
8   **else**
9   Generate an offspring population A using the mating selection and genetic operators based on the parent population Pop;
10   Update Pop using offspring population A with the CA update mechanism in DTAEA [28] ;
      end
end
Return Pop

The flowchart of KTDMOEA is shown in Fig. 3. The overall framework of the proposed KTDMOEA is given in Algorithm 4. KTDMOEA starts with generating an initial population of size N, as shown in line 1. While the stopping criteria are not satisfied, carry out the following steps. Detect whether the environmental changes occur. If the NObj is detected to increase, evoke the process of PS expansion on Pop using Algorithm 1. If the NObj decreases, evoke the process of PS contraction on Pop using Algorithm 3. If there is no change detected, conduct the evolutionary optimization process on Pop. In line 9, an offspring population A is produced through the following two steps until the size of A reaches the population size N. Firstly, randomly pick two solutions from Pop as the parent solutions via the mating selection. Then, those two solutions are used to generate two offspring solutions via appropriate crossover and mutation operators. Here, we utilize the simulated binary crossover [46] and the polynomial mutation, as used by most continuous multi-objective evolutionary algorithms including DTAEA.

Lastly, the generated offspring population A is used to update Pop with the CA update mechanism in DTAEA [28]. Due to space limitation, the CA update mechanism of DTAEA is not introduced here. Interested readers can refer to [28].

C. Time Complexity Analysis

The time complexity of KTDMOEA is discussed in this subsection for one generation. When increasing or decreasing NObj, the PS expansion or contraction is evoked. In the case of increasing NObj, the main complexity of PS expansion is cost in nondominated sorting (line 4 of Algorithm 2) and density estimation (line 9 of Algorithm 2) of searching for expansion directions. The former consumes $O(N^2)$. The main time of the density estimation is cost in associating each solution in the population to one of the weight vectors by calculating the perpendicular distance between the solution and each of the weight vectors. Therefore, the density estimation costs $O(N|W(t)|NWt)$ [1], where $|W(t)|$ is the number of weight vectors at time step t and NWt is the size of each weight vector. As for the PS contraction, the nondominated sorting in line 1 of Algorithm 3 costs $O(N^2)$; the 'for' loop consumes $O(M(t)N)$, where $M(t)$ is the number of objectives at time step t. As KTDMOEA utilizes CA update mechanism in DTAEA to update the population, the population update in KTDMOEA costs $O(N^2)$ comparisons. In summary, the complexity of KTDMOEA in one generation is $O(N|W(t)|NWt)$.

IV. Experimental Setup

In this section, experimental studies are designed to verify whether the improved knowledge transfer-based approach answers the research questions mentioned in Section I. Analyses will be carried out to reveal whether existing DMOEAs for DMOPs are able to deal with a changing NObj despite not being designed to do so, and how well static MOEAs could perform on DMOPs with a changing NObj. These are important baselines.

A. Benchmark Problems

Two suites of multi-objective optimization test problems DTLZ [31] and WFG [32] are modified to be DMOPs with a changing NObj. Four DMOPs with a changing NObj from DTLZ1-DTLZ4 are renamed as F1-F4, the same as in [28]. These two suites of benchmark functions are used to verify that the proposed algorithm is able to deal with problems with both simple and complex problem features. Detailed descriptions of problems features can be found in Section I of our Supplementary File.

There are two different sequences of changes for these benchmark problems:

[1] It is implicitly assumed in [28] that the size of the vector is negligible, which leads to a complexity of $O(N|W(t)|)$.

1) The initial NObj is set as 2. It firstly increases from 2 to 7 one by one and then decreases from 7 to 2 one by one (simply denoted as '2-7-2 one by one'), which was used in [28];
2) The initial NObj is set as 7. It firstly decreases from 7 to 2 one by one and then increases from 2 to 7 one by one (simply denoted as '7-2-7 one by one').

B. Compared Algorithms

This section presents six selected compared algorithms in our experimental studies, so as to verify the performance of our proposal against the state-of-the-art. The selected compared algorithms and the reasons of selecting them are presented as follows:

- Two static MOEAs are selected due to their popularity and good performance on solving static MOPs. Whenever there is a change, the whole population of the last generation in the old environment is just copied to the next generation after changes and then re-evaluated in the new environment to respond to changes in the NObj.
  - The elitist nondominated sorting genetic algorithm (NSGA-II) [47], one representative domination-based MOEA;
  - Multi-objective evolutionary algorithm based on decomposition (MOEA/D) [48], one representative decomposition-based MOEA.

  The way they deal with the changes in the NObj is able to verify whether existing memory-based dynamic handling strategies are able to tackle the changing NObj.
- Two popular and state-of-the-art DMOEAs targeted designed for solving DMOPs with changing position and/or shape of PSs and/or PFs:
  - Dynamic version of NSGA-II (DNSGA-II) [2], one representative diversity-based DMOEA;
  - MOEA/D based on Kalman Filter (MOEA/D-KF) [15], one representative prediction-based DMOEA.

  They are selected to verify whether existing diversity-based and prediction-based DMOEAs are able to solve DMOPs with a changing NObj.
- Two state-of-the-art DMOEAs targeted designed for solving DMOPs with a changing NObj:
  - DTAEA [28], one of the popular and recently developed algorithms targeted for handling changes in the NObj;
  - A decision space information driven algorithm (DSID) [37], one of recently developed algorithms targeted for handling changes in the NObj based on knowledge transfer.

  They are selected to verify whether our proposed KTDMOEA outperforms the existing DMOEAs targeted for solving DMOPs with a changing NObj. In particular, DTAEA is chosen to verify whether our improved knowledge transfer-based approach (KTDMOEA) answers the research questions mentioned in Section I.

C. Parameter Settings

The parameters of these compared algorithms are set as follows:

- Population size: 300, the same as that of DTAEA, θ in KTDMOEA is set as 2. The impact of θ on KTDMOEA's performance will be analyzed in Section V-E;
- Several different frequencies of change: τ_t is set as 5, 25 and 50 and 200; Those parameters are set for assessing the effects of different algorithms under different frequencies of change.
- All algorithms run 31 times independently, also the same as in DTAEA's work [28];
- 1000 generations are given to each algorithm before the first change so that the population before the change can converge;
- The crossover probability was p_c = 1.0 and its distribution index was η_c = 20. The mutation probability was p_m = 1/n (where n denotes the number of decision variables) and its distribution η_m = 20. These parameters are chosen because of their good performance on solving continuous problems, which have been analyzed in [49] and [31].
- The neighbourhood size and the number n_r of solutions allowed to replace in MOEA/D were set to 20 and 2, respectively, which is the same as in the original paper [48].

D. Performance Metrics

- Hypervolume (HV) [50] comprehensively measures the convergence and diversity of solution sets; the larger the better.
- Generational Distance (GD) [6] [5] evaluates the convergence of obtained solution sets; the smaller the better.
- Maximum Spread (MS) [51] assesses the diversity of solution sets; the larger the better.

Note that these three metrics are used to measure the solution quality of a found solution set. They can be also used to measure comprehensive performance of an algorithm by averaging the metric values of all obtained solutions under multiple environmental changes.

For DMOPs with a changing NObj, an algorithm can obtain one solution set at each environment and these metrics can be used to measure this algorithm's overall performance by averaging the metrics values of all obtained solution sets under all environmental changes. For example, if we want to compare the performance of all algorithms right after changes under the sequence of change '2-7-2 one by one', each algorithm can obtain one solution set after each change. In this sequence of change, there are 10 environmental changes and after these 10 environmental changes, the NObj is equal to 3, 4, 5, 6, 7, 6, 5, 4, 3, 2, respectively, which are also the NObj of obtained solution sets. Therefore, we can compare all algorithms by comparing the averaged metric values of all (10) obtained solution sets right after changes under all (10) environmental changes (one solution set for each change).

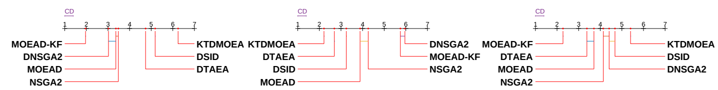
V. Experimental Results and Analyses

In order to achieve the objectives of the experiment, i.e. answering the research questions and verifying whether the existing static MOEAs and DMOEAs for solving DMOPs with fixed NObj are able to tackle DMOPs with a changing NObj, experimental results of all compared algorithms are presented in this section. Furthermore, further analyses regarding the further verification of the improved knowledge transfer, performance comparison of different NObj changing sequence and the impact of algorithm parameters are also given in this section.

Three metrics (HV, GD and MS) are used to measure the quality of the found solutions at the first generation after the change and in the last generation before the next change by six algorithms. To show the significant superiority of the proposed KTDMOEA to other algorithms across all problem instances in general, Friedman and Nemenyi statistical tests [52] are adopted across all benchmark problems regarding the three metrics (HV, GD and MS) of six compared algorithms. The larger the values of HV and MS, the better the algorithm. Therefore, the larger the Friedman ranking, the better the algorithm. Similarly, the smaller the Friedman ranking of GD, the better the algorithm. Absence/presence of a coloured horizontal line connecting algorithms indicates that significant difference has / has not been found based on Nemenyi tests.

The mean metric value of 31 independent runs that each algorithm gets on one problem with one frequency of change at each environmental change is regarded as an observation of the Friedman and Nemenyi test. Therefore, there are 520 (13 problems, 4 different frequencies of change and 10 environmental changes) observations for each algorithm in the Friedman and Nemenyi tests. Additionally, in order to show the significant superiority of the proposed KTDMOEA to other algorithms on each individual problem of each parameter setting, the Wilcoxon rank sum test at the 5% significance level is implemented on each benchmark problem regarding each metric of six compared algorithms at each parameter setting. Therefore, there are 31 observations obtained from 31 independent runs for each algorithm on each problem and parameter setting in the Wilcoxon rank sum test.

Due to the space limitation of the paper, only the results of the Friedman and Nemenyi statistical tests are presented here. The Wilcoxon rank sum test results are shown in the Supplementary File. Mean and standard deviation values of HV, GD and MS of obtained solutions in the first generation after changes and the last generation before the next change averaged across 10 environmental changes in two sequences of changes as ‘2-7-2 one by one’ and ‘2-7-2 one by one’ are also presented in the Supplementary File, respectively. Moreover, mean and standard deviation values of HV, GD and MS of obtained solutions at the first generation after changes and at the last generation before the next change at each environmental changes under 31 independent runs in those two sequences of changes as ‘2-7-2 one by one’ and ‘2-7-2 one by one’ are also recorded and presented in the Supplementary File.



(a) HV (b) GD (c) MS

Fig. 5. Friedman ranking among HV, GD and MS of obtained solutions at the first generation by 7 algorithms in the changing sequence of **firstly increasing the NObj from 2 to 7 and then decreasing it from 7 to 2**, both one by one. Larger (smaller) ranks mean better HV and MS (GD). Absence/presence of a coloured horizontal line connecting algorithms indicates that significant difference has / has not been found based on Nemenyi tests.

A. Initial Effectiveness of Knowledge Transfer

In order to verify (1) whether the proposed PS expansion/contraction mechanism is able to increase diversity so as to improve knowledge transfer after the change, and (2) whether the MOEAs not tailored for DMOPs with a changing NObj can achieve this aim after the change, the quality of the solutions obtained by all algorithms in the first generation after changes is compared.

1) NObj increasing from 2 to 7 and then decreasing from 7 to 2: Fig. 5 presents the Nemenyi post-tests results among HV, GD and MS of obtained solutions at the first generation after changes by 7 algorithms. Friedman test detects significant differences in average values for HV, GD and MS with a p-value of 3.57E-251, 9.14E-256, and 1.44E-117, respectively.

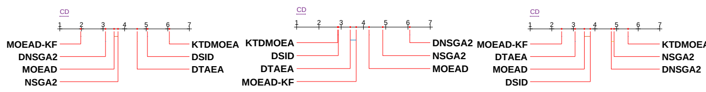
Overall, it can be observed from Fig. 5 that when comparing all algorithms, KTDMOEA significantly outperforms all others in all three metrics. More details can be found from Table 2 of the Supplementary File. This implies that the proposed knowledge transfer technique via PS expansion/contraction indeed improves the diversity and maintain the convergence of transferred solutions right after changes, under all frequencies of changes on most problems.

For readers who want to examine the details, results of mean and standard deviation values for HV, GD and MS when the NObj increase from 2 to 7 and then decrease from 7 to 2 are presented in Tables 6, 7 and 8 of the Supplementary File, respectively.

2) NObj decreasing from 7 to 2 and then increasing from 2 to 7: Fig. 6 presents the Nemenyi post-tests results among HV, GD and MS of obtained solutions at the first generation by 7 algorithms. Friedman test detects significant differences in average accuracy for HV, GD and MS with a p-value of 1.97E-240, 4.72E-221, and 6.87E-113, respectively.

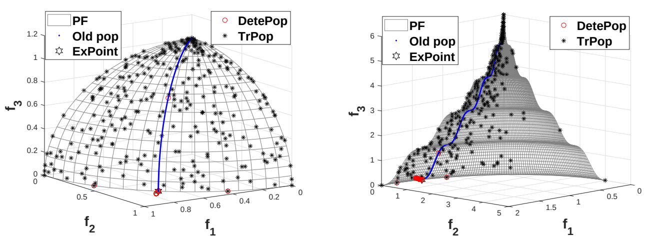
Overall, it can be found from the Friedman test results in Fig. 6 that KTDMOEA performs significantly better than all others regarding HV and MS metrics. There is no significant difference between KTDMOEA and DTAEA regarding GD. More details can be found from Table 3 of the Supplementary File. This further supports that the proposed knowledge transfer technique via PS expansion/contraction indeed improves the diversity and maintains the convergence of transferred solutions right after changes, under all frequencies of changes on most problems.

For readers who are interested in details, mean and standard deviation values for HV, GD and MS in the benchmark of decreasing the NObj from 7 to 2 and then increasing it from 2 to 7 are presented in Tables 9, 10 and 11 of the Supplementary File, respectively. The comparison results of all algorithms at



(a) HV (b) GD (c) MS

Fig. 6. Friedman ranking among HV, GD and MS of obtained solutions at the first generation by 7 algorithms in the changing sequence of **firstly decreasing the NObj from 7 to 2 and then increasing it from 2 to 7**, both one by one. Larger (smaller) ranks mean better HV and MS (GD). Absence/presence of a coloured horizontal line connecting algorithms indicates that significant difference has / has not been found based on Nemenyi tests.



(a) F2 (b) WFG1

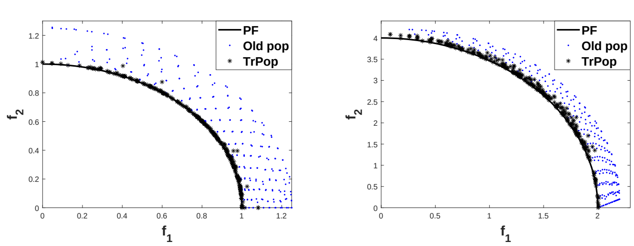
Fig. 7. The distribution of the old population ('Old pop'), the extreme point ('ExPoint'), the detective population ('DetePop') and transferred population ('TrPop') via PS expansion on F2 and WFG1 at the first generation when increasing the NObj from 2 to 3.

each NObj regarding HV, GD and MS are presented in Tables 44-53, Tables 54-63 and Tables 64-73 of the Supplementary File, respectively.

3) Why Does Knowledge Transfer Usually Get Better Solution Quality Right after Changes?: In this section, two examples are presented to elaborate the reason why the proposed PS expansion/contraction works well on most problems.

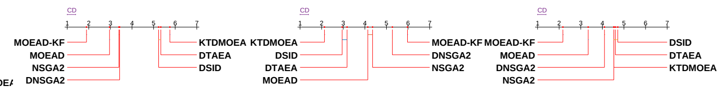
As shown in Fig. 7, the distributions of the old population, the detective population and transferred population via PS expansion on F2 and WFG1 is presented when increasing the NObj from 2 to 3. It is clear that nondominated solutions in the detective population are still nondominated by the selected extreme point. Following steps 5 to 10 in Algorithm 2, there are only two solutions in P_{var}, which are located in the areas away from that of the old population. Then, each of those two solutions is regarded as the ending point of the expansion direction, together with the extreme point as the starting point of the direction. Therefore, when evenly selecting solutions from the old population to conduct the PS expansion, almost all areas of F2 can be covered right after the change. Even for WFG1, a large area of the PF is covered by the transferred solutions. In addition, it is clear that some of the transferred solutions are able to reach the boundary of the PF.

Fig. 8 presents the distributions of the old population and the transferred solutions via PS contraction on F2 and WFG1



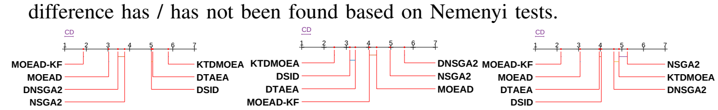
(a) F2 (b) WFG6

Fig. 8. The distribution of the old population ('Old pop') and transferred population ('TrPop') via PS contraction on F2 and WFG1 at the first generation when decreasing the NObj from 3 to 2.



(a) HV (b) GD (c) MS

Fig. 9. Friedman ranking among HV, GD and MS of optimized solutions at the last generation by 7 algorithms in the changing sequence of **firstly increasing the NObj from 2 to 7 and then decreasing it from 7 to 2**, both one by one. Larger (smaller) ranks mean better HV and MS (GD). Absence/presence of a coloured horizontal line connecting algorithms indicates that significant difference has / has not been found based on Nemenyi tests.



(a) HV (b) GD (c) MS

Fig. 10. Friedman ranking among HV, GD and MS of optimized solutions at the last generation by 7 algorithms in the changing sequence of **firstly decreasing the NObj from 7 to 2 and then increasing it from 2 to 7**, both one by one. Larger (smaller) ranks mean better HV and MS (GD). Absence/presence of a coloured horizontal line connecting algorithms indicates that significant difference has / has not been found based on Nemenyi tests.

at the first generation when decreasing the NObj from 3 to 2 in the changing NObj sequence of firstly decreasing from 7 to 2 and then increasing from 2 to 7. It is clear that on those two problems the transferred population via PS contraction has better convergence and diversity than the old population.

B. How Does Knowledge Transfer Help Optimization?

In order to verify whether the proposed KTDMOEA can find solutions with better convergence and diversity in the last generation after optimization against all other algorithms, the solution quality of all compared algorithms after optimization and before the next change is compared.

1) NObj increasing from 2 to 7 and then decreasing from 7 to 2: Fig. 9 presents the Nemenyi post-tests results among HV, GD and MS of obtained solutions at the last generation after optimization by 7 algorithms. Friedman test detects significant differences in average accuracy for HV, GD and MS with a p-value of 6.83E-215, 6.22E-160, and 4.27E-164, respectively.

Overall, it can be seen from those statistical test results that KTDMOEA performs significantly better than or the same as the other approaches. Specifically, it is clear from the Friedman ranking results in Fig. 9 that KTDMOEA gets significantly best results among all compared algorithms regarding HV and GD values. It is the equal best, together with DTAEA and NSGA2, regarding the MS value. These three algorithms outperforms other algorithms regarding the MS value. The statistical results show that the proposed knowledge transfer is able to help the optimization, which achieves better convergence and at least similar diversity compared to the start-of-the-arts when the NObj increasing from 2 to 7 and then decreasing from 7 to 2, under all frequencies of changes on most problems. More details can be seen from Table 4 in the Supplementary File.

2) NObj decreasing from 7 to 2 and then increasing from 2 to 7: Fig. 10 presents the Nemenyi post-tests results among HV, GD and MS of obtained solutions at the last generation after optimization by 7 algorithms. Friedman test detects significant differences in average values for HV, GD and MS with a p-value of 3.56E-223, 1.58E-129, and 4.98E-183, respectively.

It can be found from the Friedman test results in Fig. 10 that KTDMOEA achieves significantly better results than all other algorithms regarding HV and GD metrics. As for the MS results, KTDMOEA and DNGSA2 rank the second in the Friedman ranking test, both are outperformed by NSGA2 only. Overall, those statistical results imply that the proposed knowledge transfer is able to help the optimization in obtaining better convergence and at least similar diversity compared to the start-of-the-arts in the changing sequence of firstly decreasing from 7 to 2 and then increasing from 2 to 7, under all frequencies of changes on most problems.

3) Why Does Knowledge Transfer Help Optimization?: It has been presented in Section V-A that the proposed PS expansion/contraction has indeed enhanced the diversity of knowledge transfer, resulting in better solution quality of obtained solutions than other state-of-the-arts in the first generation after changes. In other words, given the results of better solution quality than other algorithms in the first generation after changes, our proposed KTDMOEA is able to find solutions with good convergence and diversity at at all frequencies of change, even when the frequency of change is very high. This means that our proposed approach is robust to different frequencies of change.

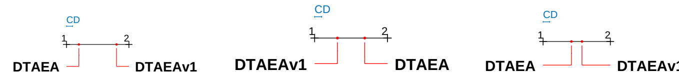
Because the transferred solutions are better distributed in the new environment with a better diversity already, KTDMOEA is able to find better solutions across different frequencies of change. This is also the reason why KTDMOEA is able to quickly respond to the changes in the NObj, since finding good solution under high frequency of change means fast response to changes. There are some problems where KTDMOEA did not perform best when the frequency of changes is large. The specific results and analyses are presented in Section III.B.3) of the Supplementary File.

C. Further Analysis of Our Knowledge Transfer Methods

In order to further verify the effectiveness of the proposed PS expansion/contraction method against the state-of-the-art method DTAEA, two pairs of comparisons are designed. First, DTAEA is compared to DTAEAv1, where the CA reconstruction after changes is replaced by the proposed PS expansion/contraction with other components of DTAEA unchanged. Second, KTDMOEA is compared to KTDMOEAv1, where the proposed PS expansion/contraction is replaced by the CA reconstruction of DTAEA with other components of KTDMOEA unchanged.

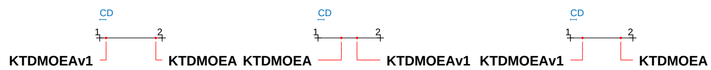
All experimental settings are set the same as in Section IV-C except for the frequency of change and the NObj changing sequence, which is set as 25 and NObj increasing from 2 to 7 and then decreasing from 7 to 2 one by one, respectively, to save space. For Friedman and Nemenyi tests, the mean metric value of 10 environmental changes that each algorithm gets on one problem with one frequency of change at each independent run is regarded as an observation of the test. Therefore, there are 403 (13 problems and 31 environmental changes) observations for each algorithm in the Friedman and Nemenyi tests.

Fig. 11 presents the Nemenyi post-tests results among HV, GD and MS of obtained solutions at the last generation after



(a) HV (b) GD (c) MS

Fig. 11. Friedman ranking among HV, GD and MS of optimized solutions at the last generation by DTAEA and DTAEAv1 in the changing sequence of **firstly increasing the NObj from 2 to 7 and then decreasing it from 7 to 2**, both one by one.



(a) HV (b) GD (c) MS

Fig. 12. Friedman ranking among HV, GD and MS of optimized solutions at the last generation by KTDMOEA and KTDMOEAv1 in the changing sequence of **firstly increasing the NObj from 2 to 7 and then decreasing it from 7 to 2**, both one by one.

optimization before the next change by DTAEA and DTAEAv1 algorithms. Friedman test detects significant differences in average values for HV, GD and MS with a p-value of 9.97E-34, 5.40E-14, 0.0017, respectively. Similarly, Fig. 12 presents the Nemenyi post-tests results among HV, GD and MS of obtained solutions at the last generation after optimization before the next change by KTDMOEA and KTDMOEAv1 algorithms. Friedman test detects significant differences in average values for HV, GD and MS with a p-value of 1.50E-57, 4.88E-07, 8.63E-35, respectively.

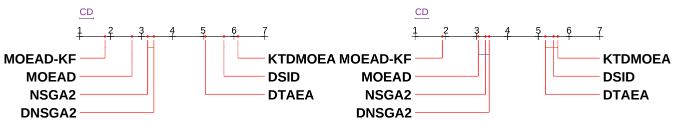
Overall, it can be observed from Figs. 11 and 12 the algorithm with our proposed knowledge transfer strategy significantly outperforms the one without it, i.e, DTAEAv1 outperforms DTAEA and KTDMOEA outperforms KTDMOEAv1. More details can be found from Tables 138-140 of the Supplementary File. From this result, we can get the conclusion that, the proposed PS expansion/contraction method works better than the knowledge transfer in DTAEA, further confirming the effectiveness of our proposed knowledge transfer method.

D. Performance Comparison on Other Changes in the NObj

In the previous experiments, the NObj only increases or decreases one by one. This section aims to verify the performance of the proposed algorithm in the scenario where the NObj increases or decreases by more than one. Two different changing sequences where the NObj increases or decreases by one or two each time are designed as follows:

- The initial NObj is set as 2. Then, NObj firstly increases from 2 to 3. Then there are four changes with the first two changes increasing the NObj by two and then two changes decreasing the NObj by two. Lastly, the NObj decreases from 3 to 2 (simply denoted as '2-3-5-7-5-3-2').
- The initial NObj is set as 7. Then, there are two changes where the NObj decreases by two. Later on, the NObj decreases from 3 to 2 and then increases from 2 to 3. In the last two changes, the NObj increases by two at each change (simply denoted as '7-5-3-2-3-5-7').

All experimental settings are set the same as in Section IV-C except for the frequency of change and the metric, which is set as 25 and HV, respectively, to save space. For Friedman and Nemenyi tests, the HV values that all algorithms get on one problem with one frequency of change at one independent run



(a) For sequence ‘2-3-5-7-5-3-2’. (b) For sequence ‘7-5-3-2-3-5-7’.

Fig. 13. Friedman ranking among HV of optimized solutions at the last generation by all algorithms in the changing sequences ‘2-3-5-7-5-3-2’ and ‘7-5-3-2-3-5-7’, respectively. Larger (smaller) ranks mean better HV and MS (GD). Absence/presence of a coloured horizontal line connecting algorithms indicates that significant difference has / has not been found based on Nemenyi tests.

of 31 runs is regarded as an observation of the test. Therefore, there are 403 (13 problems and 1 frequency of changes and 31 independent runs) observed data.

Fig.s 13(a) and 13(b) presents the Nemenyi post-tests results among HV values of optimized solutions at the last generation by all compared algorithms in two changing sequences ‘2-3-5-7-5-3-2’ and ‘7-5-3-2-3-5-7’, respectively. Friedman detects significant differences in average accuracy for HV with a p-value of 3.8407e-294 and 2.3020e-234, respectively for these two changing sequences.

Overall, it can be observed from Fig. 13 in the changing sequence of ‘2-3-5-7-5-3-2’, our proposed KTDMOEA performs the best among all compared algorithms. In another changing sequence of ‘7-5-3-2-3-5-7’, both KTDMOEA and DTAEA performs the best. More details can be found in Tables 141 and 142 in the Supplementary File.

E. Impact of Algorithm Parameters

In the process of PS expansion, there is a parameter θ to set which is the number of solutions to generate along each expansion direction. In this section, different values of this parameter will be set to verify whether different parameter settings affect the performance of KTDMOEA.

All experimental settings are set the same as in Section IV-C except for the frequency of change and the metric, which is set as 25 and HV, respectively, to save space. The changing sequence is that the NObj firstly increases from 2 to 7 and then decreases from 7 to 2 one by one. There are three KTDMOEAs (denoted as KTDMOEA-1, KTDMOEA-2 and KTDMOEA-4), which has the value of θ as 1, 2 and 4, respectively. In order to verify whether different parameter settings affect the performance of KTDMOEA, the Friedman and Nemenyi tests on 5 state-of-the-arts and 3 KTDMOEAs are conducted to indicate the significant differences among them. The HV values that all algorithms get on one problem with one frequency of change at one independent run of 31 runs is regarded as an observation of the test. Therefore, there are 403 (13 problems and 1 frequency of changes and 31 independent runs) observed data.

Fig. 14 presents the Nemenyi post-tests results among HV values of optimized solutions at the last generation by 8 algorithms in the changing sequence of firstly increasing the NObj from 2 to 7 and then decreasing it from 7 to 2, both one by one. Friedman detects significant differences in average accuracy for HV with a p-value of 2.4703e-323.

It is clear that the three KTDMOEAs get the best HV values among all algorithms. It can be found from Fig. 14 that there is no significant difference among the three

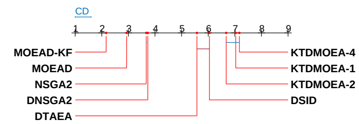


Fig. 14. Friedman ranking among HV of optimized solutions at the last generation by 6 state-of-the-arts and 3 KTDMOEAs with different values of parameters theta (1, 2 and 4) in the changing sequence of firstly increasing the NObj from 2 to 7 and then decreasing it from 7 to 2, both one by one. Larger (smaller) ranks mean better HV and MS (GD). Absence/presence of a coloured horizontal line connecting algorithms indicates that significant difference has / has not been found based on Nemenyi tests.

KTDMOEAs with different setting of the parameter θ. These results have verified that the performance of the proposed PS expansion/contraction is not sensitive to the setting of the parameter θ. The performance of the proposed KTDMOEA against the existing algorithms is not affected by the setting of different parameter values.

F. Running Time Analysis

To investigate how efficient our proposal is compared to other algorithms, we record the running time (in seconds) of all compared algorithms on all benchmark problems with the changing sequence of ‘2-7-2 one by one’ when $\tau_t = 50$ under the same hardware configuration [2]. The running time by each algorithm is recorded using the clock time of C++ programming language. Also, we use Friedman and Nemenyi statistical tests [52] with a significance level 0.05 across all benchmark problems regarding the running time of five compared algorithms to indicate the significance of the running time. The running time obtained by all algorithms on one problem under one run is regarded as an observation of the test.

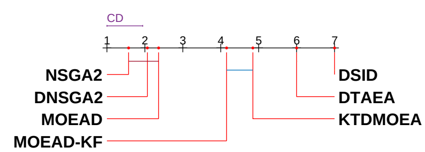
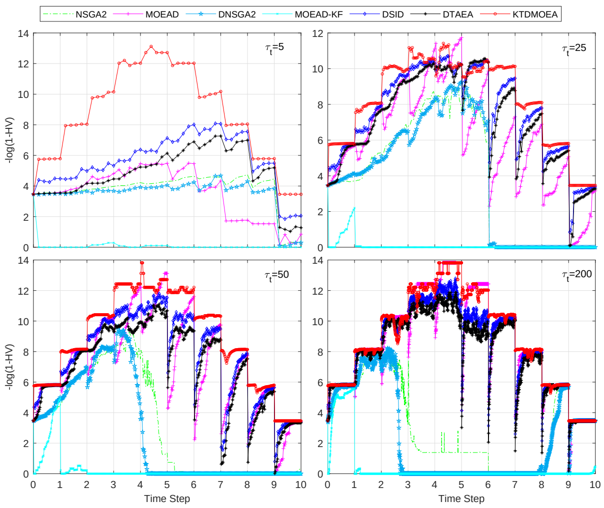


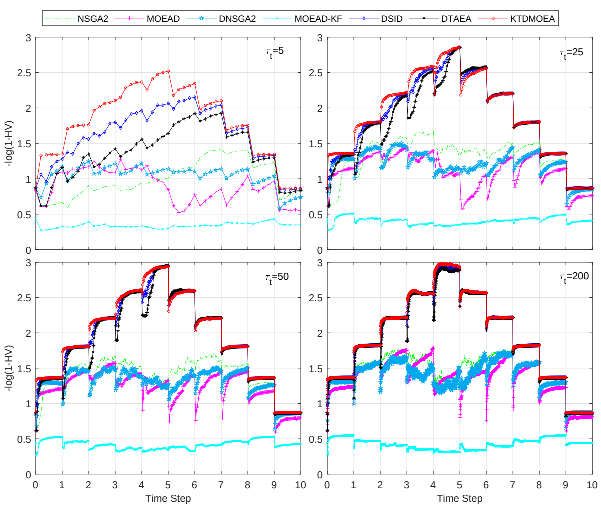
Fig. 15. Friedman ranking among running time obtained by all compared algorithm in changing sequence of ‘2-7-2 one by one’. Here, larger ranks mean larger running time. Larger (smaller) ranks mean better HV and MS (GD). Absence/presence of a coloured horizontal line connecting algorithms indicates that significant difference has / has not been found based on Nemenyi tests.

Fig. 15 presents the Nemenyi post-tests results among running time of all compared algorithms and Friedman detects significant differences in average running time with a p-value of 1.4727e-106. The specific mean values of running time (in seconds) obtained by all compared algorithms are presented in Table 24 of the Supplementary File. It is clear from Fig. 15 that our proposed KTDMOEA achieves competitive efficiency compared to algorithms tailored for changing NObj (DTAEA and DSID). In addition, our intuition is right that algorithms with one archive (KTMOEA) cost significantly less running time than that with two archives (DTAEA and DSID). In particular, DSID costs more running time than DTAEA, since it involves the manifold learning, which costs more time on training the self-organizing mapping. Meantime, algorithms

[2]The implementation environment is as follows: 2.20-GHz Intel Core i7, 8-GB DDR4 2666MHz.



(a) F1



(b) WFG4

Fig. 16. HV trajectories across the whole evolution process for the NObj changing sequence of '2-7-2 one by one' and all τ_ts.

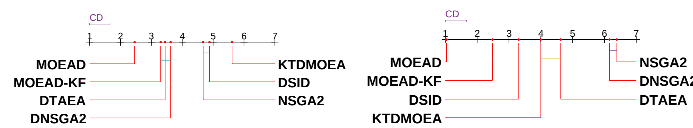
without specific strategies to cope with dynamic changes (NSGA2 and MOEAD) run faster.

G. How Does Our Proposed KTDMOEA Perform in Evolution Process?

To investigate how our proposed KTDMOEA performs against its counterparts in the evolution process, we plot the HV trajectory of all compared algorithms over the whole evolution process on all problem instances for all τ_ts. To save space, only the figures for problems F1 and WFG4 are presented in Fig. 16. The figures for all other cases are presented in Fig. 1 of the Supplementary File. It is clear from Fig. 16 that KTDMOEA converges faster than all compared algorithms, achieving better HV in a shorter time especially when τ_t is small.

H. Results on Real-world Problems

In this section, we utilize three widely used continuous multi-objective optimization real-world problems to evaluate the performance of our proposed KTDMOEA. These three real world problems are the water problem [25], Car-Side Impact Problem [26] and Crash Worthiness in Design of Vehicles [27]. The water problem has five objectives and the others have three objectives. To investigate this problems under a setting with changes in the NObj, we simulate a scenario where we first set the NObj of these problems as 2 and then increase to their own maximal NObj one by one and then decrease to 2 one by one. The frequency of changes is set as 25. All the experimental settings for all compared algorithms are the same to those in Section V-C. The same significant tests were also used. The HV values of each compared algorithm on one problem at each run were regarded as one observation for the Friedman ranking and Nemenyi test. The Friedman ranking to compare the HV of the obtained solutions in the first generation and at the last generation after optimization on the three real-world problems are presented in Fig. 17. Specific HV values of obtained solutions for all compared algorithms on these problems can be found in Table of the Supplementary File. It is clear from Fig. 17 that KTDMOEA achieved competitive performance regarding solution quality on the real-world problems, especially soon after the changes.



(a) At the first generation after changes (b) At the last generation after opt

Fig. 17. Friedman ranking among HV of obtained solutions in the first generation and solutions after τ_t generations' optimization by 7 algorithms. Here, larger ranks mean higher HV.

VI. Conclusion

It has been investigated in this paper that existing work cannot handle well DMOPs with a changing NObj and more complex PF shapes (convex, discontinuous and mixed shape of convex and concave) and fitness landscapes (nonseparability and deceptiveness). The main reason is the lack of sufficient population diversity right after dynamic changes. To solve this problem, two research questions are studied, first how to transfer knowledge so as to enhance diversity and second how the knowledge transfer helps optimization. In order to answer both research questions, inspired by the characteristic of DMOPs with a changing NObj, a dynamics handling strategy–PS expansion/contraction is proposed. As a result, a new algorithm, KTDMOEA, is designed to make use of this strategy. Experimental studies have demonstrated the effectiveness of the proposed knowledge transfer in enhancing the diversity right after changes and assisting the optimization under different number of objective changing sequences.

In comparison with the state-of-the-art in solving DMOPs with a changing NObj, our KTDMOEA achieved the best performance according to HV, GD and MS metrics across a number of test functions. We argue that is is important to use both DTLZ and WFG functions to build dynamic benchmarks because they provide rather different problem characteristics. KTDMOEA performed well on different problems under different parameter settings under different frequencies of change.

According to the obtained results, we suggest to use our proposed algorithm when solving problems where environmental changes are frequent since the results show that our proposed approach is able to quickly respond to environmental changes, i.e. achieving significantly better HV than other algorithms under such conditions.

As expected, no algorithm would be the best on all possible problems. According to the details in the Supplementary File, there are several problems on which KTDMOEA did not outperform existing algorithms. Although we have done initial analysis of these, as reported in the Supplementary File, more in-depth analysis will be our next work in the future. Moreover, proposing new algorithms on benchmark problems with variable linkages like UF [53] and LSMOP [54] after analyzing the limitations of existing algorithms could be also our future work. In addition, extending our current work to solve discrete real-world DMOPs with a changing NObj is one of our future directions.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant 62250710682; the Guangdong Provincial Key Laboratory under Grant 2020B121201001; the Program for Guangdong Introducing Innovative and Entrepreneurial Teams under Grant 2017ZT07X386; and the European Union's Horizon 2020 Research and Innovation Programme under Grant 766186.

References

[1] M. Farina, K. Deb, P. Amato, Dynamic multiobjective optimization problems: test cases, approximations, and applications, IEEE Transactions on Evolutionary Computation 8 (5) (2004) 425–442.

[2] K. Deb, S. Karthik, et al., Dynamic multi-objective optimization and decision-making using modified NSGA-II: a case study on hydro-thermal power scheduling, in: Evolutionary Multi-Criterion Optimization, Springer, 2007, pp. 803–817.

[3] J. Ding, C. Yang, Q. Xiao, T. Chai, Y. Jin, Dynamic evolutionary multiobjective optimization for raw ore allocation in mineral processing, IEEE Transactions on Emerging Topics in Computational Intelligence 3 (1) (2018) 36–48.

[4] W. Zhou, Y. Liu, M. Li, Y. Wang, Z. Shen, L. Feng, Z. Zhu, Dynamic multi-objective optimization framework with interactive evolution for sequential recommendation, IEEE Transactions on Emerging Topics in Computational Intelligence (2023).

[5] G. Ruan, G. Yu, J. Zheng, J. Zou, S. Yang, The effect of diversity maintenance on prediction in dynamic multi-objective optimization, Applied Soft Computing 58 (2017) 631–647.

[6] C.-K. Goh, K. C. Tan, A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization, IEEE Transactions on Evolutionary Computation 13 (1) (2009) 103–127.

[7] Z. Peng, J. Zheng, J. Zou, A population diversity maintaining strategy based on dynamic environment evolutionary model for dynamic multiobjective optimization, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 274–281.

[8] B. Zheng, A new dynamic multi-objective optimization evolutionary algorithm, in: ICNC, IEEE, 2007, pp. 565–570.

[9] R. Liu, W. Zhang, L. Jiao, F. Liu, J. Ma, A sphere-dominance based preference immune-inspired algorithm for dynamic multi-objective optimization, in: Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, ACM, 2010, pp. 423–430.

[10] Z. Liang, S. Zheng, Z. Zhu, S. Yang, Hybrid of memory and prediction strategies for dynamic multiobjective optimization, Information Sciences 485 (2019) 200–218.

[11] W. T. Koo, C. K. Goh, K. C. Tan, A predictive gradient strategy for multiobjective evolutionary algorithms in a fast changing environment, Memetic Computing 2 (2) (2010) 87–110.

[12] I. Hatzakis, D. Wallace, Dynamic multi-objective optimization with evolutionary algorithms: a forward-looking approach, in: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation, ACM, 2006, pp. 1201–1208.

[13] A. Zhou, Y. Jin, Q. Zhang, A population prediction strategy for evolutionary dynamic multiobjective optimization, IEEE Transactions on Cybernetics 44 (1) (2014) 40–53.

[14] A. Muruganantham, Y. Zhao, S. B. Gee, Dynamic multi-objective optimization using evolutionary algorithm with kalman filter, Procedia Computer Science 24 (2013) 66–75.

[15] A. Muruganantham, K. C. Tan, P. Vadakkepat, Evolutionary dynamic multiobjective optimization via kalman filter prediction, IEEE Transactions on Cybernetics 46 (12) (2015) 2862–2873.

[16] M. Jiang, Z. Huang, L. Qiu, W. Huang, G. G. Yen, Transfer learning-based dynamic multiobjective optimization algorithms, IEEE Transactions on Evolutionary Computation 22 (4) (2017) 501–514.

[17] M. Jiang, Z. Wang, S. Guo, X. Gao, K. C. Tan, Individual-based transfer learning for dynamic multiobjective optimization, IEEE Transactions on Cybernetics 51 (10) (2020) 4968–4981.

[18] M. Jiang, Z. Wang, L. Qiu, S. Guo, X. Gao, K. C. Tan, A fast dynamic evolutionary multiobjective algorithm via manifold transfer learning, IEEE Transactions on Cybernetics 51 (7) (2020) 3417–3428.

[19] M. Jiang, Z. Wang, H. Hong, G. G. Yen, Knee point-based imbalanced transfer learning for dynamic multiobjective optimization, IEEE Transactions on Evolutionary Computation 25 (1) (2020) 117–129.

[20] P. Kruchten, The rational unified process: an introduction, Addison-Wesley Professional, 2004.

[21] X. Shen, L. L. Minku, R. Bahsoon, X. Yao, Dynamic software project scheduling through a proactive-rescheduling method, IEEE Transactions on Software

Engineering 42 (7) (2015) 658–686.
[22] X.-N. Shen, X. Yao, Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems, Information Sciences 298 (2015) 198–224.
[23] J. Xiao, L. J. Osterweil, Q. Wang, M. Li, Dynamic resource scheduling in disruption-prone software development environments, in: International Conference on Fundamental Approaches to Software Engineering, Springer, 2010, pp. 107–122.
[24] J. Zhuo, C. Chakrabarti, An efficient dynamic task scheduling algorithm for battery powered dvs systems, in: Proceedings of the 2005 Asia and South Pacific Design Automation Conference, 2005, pp. 846–849.
[25] T. Ray, K. Tai, C. Seow, An evolutionary algorithm for multiobjective optimization, Engineering Optimization 33 (3) (2001) 399–424.
[26] H. Jain, K. Deb, An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part ii: Handling constraints and extending to an adaptive approach, IEEE Transactions on Evolutionary Computation 18 (4) (2013) 602–622.
[27] X. Liao, Q. Li, X. Yang, W. Zhang, W. Li, Multiobjective optimization for crash safety design of vehicles using stepwise regression model, Structural and Multidisciplinary Optimization 35 (2008) 561–569.
[28] R. Chen, K. Li, X. Yao, Dynamic multiobjectives optimization with a changing number of objectives, IEEE Transactions on Evolutionary Computation 22 (1) (2017) 157–171.
[29] Y.-J. Zhang, S.-F. Shao, J. Niyongabo, Cloud hyper mutation particle swarm optimization algorithm based on cloud model, Pattern Recognition and Artificial Intelligence 24 (1) (2011) 90–94.
[30] S.-U. Guan, Q. Chen, W. Mo, Evolving dynamic multi-objective optimization problems with objective replacement, Artificial Intelligence Review 23 (3) (2005) 267–293.
[31] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable test problems for evolutionary multiobjective optimization, in: Evolutionary Multiobjective Optimization, Springer, 2005, pp. 105–145.
[32] S. Huband, P. Hingston, L. Barone, L. While, A review of multiobjective test problems and a scalable test problem toolkit, IEEE Transactions on Evolutionary Computation 10 (5) (2006) 477–506.
[33] R. Azzouz, S. Bechikh, L. B. Said, A multiple reference point-based evolutionary algorithm for dynamic multi-objective optimization with undetectable changes, in: 2014 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2014, pp. 3168–3175.
[34] L. Huang, I. H. Suh, A. Abraham, Dynamic multi-objective optimization based on membrane computing for control of time-varying unstable plants, Information Sciences 181 (11) (2011) 2370–2391.
[35] S. Jiang, M. Kaiser, S. Yang, S. Kollias, N. Krasnogor, A scalable test suite for continuous dynamic multiobjective optimization, IEEE Transactions on Cybernetics 50 (6) (2019) 2814–2826.
[36] X. Ma, H. Sun, Z. Hu, L. Wei, J. Yang, Multi-spatial information joint guidance evolutionary algorithm for dynamic multi-objective optimization with a changing number of objectives, Neural Computing and Applications 35 (20) (2023) 15167–15199.
[37] X. Ma, H. Sun, Z. Hu, L. Wei, J. Yang, Decision space information driven algorithm for dynamic multiobjective optimization with a changing number of objectives, International Journal of Machine Learning and Cybernetics (2023) 1–29.
[38] K. C. Tan, L. Feng, M. Jiang, Evolutionary transfer optimization-a new frontier in evolutionary computation research, IEEE Computational Intelligence Magazine 16 (1) (2021) 22–33.
[39] A. Gupta, Y.-S. Ong, L. Feng, Multifactorial evolution: toward evolutionary multitasking, IEEE Transactions on Evolutionary Computation 20 (3) (2015) 343–357.
[40] A. Gupta, Y.-S. Ong, L. Feng, Insights on transfer optimization: Because experience is the best teacher, IEEE Transactions on Emerging Topics in Computational Intelligence 2 (1) (2017) 51–64.
[41] B. Da, A. Gupta, Y.-S. Ong, Curbing negative influences online for seamless transfer evolutionary optimization, IEEE Transactions on Cybernetics 49 (12) (2018) 4365–4378.
[42] L. Feng, L. Zhou, J. Zhong, A. Gupta, Y.-S. Ong, K.-C. Tan, A. K. Qin, Evolutionary multitasking via explicit autoencoding, IEEE Transactions on Cybernetics 49 (9) (2018) 3457–3470.
[43] L. Feng, W. Zhou, W. Liu, Y.-S. Ong, K. C. Tan, Solving dynamic multiobjective problem via autoencoding evolutionary search, IEEE Transactions on Cybernetics 52 (5) (2020) 2649–2662.
[44] G. Ruan, L. L. Minku, S. Menzel, B. Sendhoff, X. Yao, When and how to transfer knowledge in dynamic multi-objective optimization, in: 2019 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2019, pp. 2034–2041.
[45] G. Ruan, L. L. Minku, S. Menzel, B. Sendhoff, X. Yao, Computational study on effectiveness of knowledge transfer in dynamic multi-objective optimization, in: 2020 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2020, pp. 1–8.
[46] K. Deb, R. B. Agrawal, et al., Simulated binary crossover for continuous search space, Complex Systems 9 (2) (1995) 115–148.
[47] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197.
[48] Q. Zhang, H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, IEEE Transactions on Evolutionary Computation 11 (6) (2007) 712–731.
[49] K. Deb, Multi-objective optimization using evolutionary algorithms, Vol. 16, John Wiley & Sons, 2001.
[50] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, V. G.

Da Fonseca, Performance assessment of multiobjective optimizers: an analysis and review, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 117–132.

[51] M. Li, X. Yao, Quality evaluation of solution sets in multiobjective optimisation: A survey, ACM Computing Surveys (CSUR) 52 (2) (2019) 1–38.

[52] J. Demšar, Statistical comparisons of classifiers over multiple data sets, The Journal of Machine Learning Research 7 (2006) 1–30.

[53] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, S. Tiwari, Multiobjective optimization test instances for the cec 2009 special session and competition, University of Essex, Colchester, UK and Nanyang technological University, Singapore, Special Session on Performance Assessment of Multi-objective Optimization Algorithms, Technical Report (2008) 1–30.

[54] R. Cheng, Y. Jin, M. Olhofer, et al., Test problems for large-scale multiobjective and many-objective optimization, IEEE Transactions on Cybernetics 47 (12) (2016) 4108–4121.

Gan Ruan is currently pursuing the Ph.D. degree in computer science at University of Birmingham, UK. His research interest is knowledge transfer in dynamic multi-objective optimization.

Leandro L. Minku (Senior Member, IEEE) received the Ph.D. degree in computer science from the University of Birmingham, Birmingham, U.K., in 2010. He was a Lecturer in computer science with the University of Leicester, Leicester, U.K. He is currently an Associate Professor with the School of Computer Science, University of Birmingham, Birmingham, U.K. His main research interests are machine learning in non-stationary environments / data stream mining, online class imbalance learning, ensembles of learning machines, dynamic optimisation, and computational intelligence for software engineering. Dr. Minku is an Associate Editor-in-Chief of Neurocomputing, was a Senior Editor of the IEEE TNNLS, and is an Associate Editor of Journal of Systems and Software and of Empirical Software Engineering journal.

Stefan Menzel received the Dipl.-Ing. degree in civil engineering from RWTH Aachen, Germany, in 1998, and the Ph.D. degree in civil engineering from TU Darmstadt, Germany, in 2004. Since 2004, he is with the Honda Research Institute Europe, Offenbach, Germany, where he is currently Chief Scientist with the Optimization and Creativity Group. His current research interests include evolutionary optimization with special focus on adaptive design representations, machine learning for knowledge transfer, generative AI, and multidisciplinary optimization for real-world applications.

Bernhard Sendhoff (Fellow, IEEE) received the Ph.D. degree in applied physics from Ruhr-Universität Bochum, Bochum, Germany, in 1998. He was with Honda Research Institute Europe GmbH, Offenbach, Germany, from 2003 to 2010, as a Chief Technology Officer, and from 2011 to 2017, as a President. Since 2017, he has been an Operating Officer with Honda Research and Development Ltd., Tokyo, Japan, and since 2019, the CEO of the Global Network Honda Research Institutes. He is an Honorary Professor with the Technical University of Darmstadt, Darmstadt, Germany. He has authored or coauthored over 180 scientific publications. Dr. Sendhoff is a Senior Member of ACM and a member of SAE.

Xin Yao (Fellow, IEEE) received the his BSc from the University of Science and Technology of China (USTC) in 1982, his MSc from the North China Institute of Computing Technology in 1985, and his Ph.D. from USTC, in 1990. He is currently Tong Tin Sun Chair Professor of Machine Learning at Lingnan University in Hong Kong, and a part-time Professor of computer science with the University of Birmingham, U.K. His major research interests include evolutionary computation, ensemble learning, and their applications. His work won the 2001 IEEE Donald G. Fink Prize Paper Award; 2010, 2016 and 2017 IEEE Transactions on Evolutionary Computation Outstanding Paper Awards; 2011 IEEE Transactions on Neural Networks Outstanding Paper Award; and many other best paper awards at conferences. He received a Royal Society Wolfson Research Merit Award in 2012, the IEEE CIS Evolutionary Computation Pioneer Award in 2013 and the 2020 IEEE Frank Rosenblatt Award.