Dynamic Multi-Objectives Optimization with a Changing Number of Objectives

Renzhi Chen*, Ke Li*, *Member, IEEE* and Xin Yao, *Fellow, IEEE*

Abstract—Existing studies on dynamic multi-objective optimization focus on problems with time-dependent objective functions, while the ones with a changing number of objectives have rarely been considered in the literature. Instead of changing the shape or position of the Pareto-optimal front/set when having time-dependent objective functions, increasing or decreasing the number of objectives usually leads to the expansion or contraction of the dimension of the Pareto-optimal front/set manifold. Unfortunately, most existing dynamic handling techniques can hardly be adapted to this type of dynamics. In this paper, we report our attempt toward tackling the dynamic multi-objective optimization problems with a changing number of objectives. We implement a dynamic two-archive evolutionary algorithm which maintains two co-evolving populations simultaneously. In particular, these two populations are complementary to each other: one concerns more about the convergence while the other concerns more about the diversity. The compositions of these two populations are adaptively reconstructed once the environment changes. In addition, these two populations interact with each other via a mating selection mechanism. Comprehensive experiments are conducted on various benchmark problems with a time-dependent number of objectives. Empirical results fully demonstrate the effectiveness of our proposed algorithm.

Index Terms—Multi-objective optimization, dynamic optimization, evolutionary algorithm, changing objectives, decomposition-based method.

I. INTRODUCTION

Dynamic multi-objective optimization (DMO) is more challenging than the stationary scenarios, as it needs to deal with the trade-offs and the time-dependent objective functions or constraints simultaneously. An effective method for solving the DMO problems (DMOPs) needs to overcome certain difficulties raised by the dynamics, such as tracking the time-dependent Pareto-optimal front (PF) or Pareto-optimal set (PS), and providing diversified solutions that enable a timely adaptation to the changing environment. Due to its inherent adaptation property [1], evolutionary algorithm (EA) has been recognized as a powerful tool for DMO. According to the ways of handling dynamics, the existing techniques can be classified into the following three categories [2].

- *Diversity enhancement*: As the name suggests, its basic idea is to propel an already converged population jump out of the current optimum by increasing the population diversity. This is usually implemented by either

responding to the dynamics merely after detecting a change of the environment or maintaining the population diversity throughout the whole optimization process. The prior one is known as *diversity introduction* and is usually accompanied by a change detection technique. For example, [3]–[5] developed some hyper-mutation methods by which the solutions are expected to jump out of their current positions and thus to adapt to the changed environment. In [3], [6] and [7], some additional solutions, either generated in a random manner or by some heuristics, are injected into the current population to participate the survival competition. On the other hand, the latter one, called *diversity maintenance*, usually does not explicitly respond to the changing environment [8], [9] whereas it mainly relies on its intrinsic diversity maintenance strategy to adapt to the dynamics. In [10], a multi-population strategy is developed to enhance the population diversity in a competitive and cooperative manner.

- *Memory mechanism*: The major purpose of this sort of method is to utilize the historical information to accelerate the convergence progress. For example, [11] suggested a hybrid memory scheme that re-initializes a new population either by re-evaluating the previous non-dominated solutions or by using a local search upon the elite solutions preserved in the archive. [12] exploited the historical knowledge and used it to build up an evolutionary environment model. Accordingly, this model is used to guide the adaptation to the changing environment.

- *Prediction strategy*: This sort of method is usually coupled with a memory mechanism to re-initialize the population according to the anticipated optimal information. For example, [13] and [14] employed an autoregressive model, built upon the historical information, to seed a new population within the neighborhood of the anticipated position of the new Pareto-optimal set (PS). By utilizing the regularity property [15], [16]–[18] developed various models to capture the movement of the PS manifolds' centers under different dynamic environments. Instead of anticipating the position of the new PS when the environment changes, [19] and [20] suggested to predict the optimal moving direction of the population. More recently, [21] proposed a hybrid scheme, which uses the Kalman Filter or random re-initialization, alternatively, according to their online performance, to respond to the changing environment.

It is worth noting that the current studies on DMO mainly focus on problems with time-dependent objective functions,

which result in the change of the PF or PS over time. In addition to this type of dynamic, many real-life scenarios consider problems in which the number of objective functions may also change over time. For example, a software development life cycle usually contains four phases [22], i.e., inception, elaboration, construction and transition. Each phase involves various tasks, some of which might be different from the other phases. In other words, different phases consider various number of objectives. In order to manage changes (changing requirements, technology, resources, products and so on), the software development process needs to switch among different phases back and forth, and thus changing the number of objectives. In project scheduling, we usually want 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 [23]–[25]. 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 [26]. 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. Furthermore, in a heterogeneous multi-core system, power consumption, performance, temperature, master frequency and reliability are usually considered altogether when designing a runtime scheduler [27]–[29]. However, these objectives can be ignored or re-considered according to some external instances, e.g., the power plugins, excessive temperature. Similar situations happen in the cloud computing resource scheduling, where even more varieties of objectives will be considered, e.g., load balance, quality of service [30]–[32].

The study on how to handle the DMOP with a changing number of objectives is rather limited, although this concept was mentioned in the literature [33], [34] and [8]. To the best of our knowledge, [33] is the only one that discussed the effects of the objective increment and replacement. It proposed an inheritance strategy that re-evaluates the objective functions of the current non-dominated solutions to adapt to this type of dynamics. Generally speaking, the DMOP with a changing number of objectives has two major characteristics: 1) the PF or PS of the original problem is a subset of the one after increasing the number of objectives, and vice versa; and 2) instead of changing the position or the shape of the original PF or PS, increasing or decreasing the number of objectives usually results in the expansion and contraction of the dimension of the PF or PS manifold. Bearing these two characteristics in mind, this paper presents a dynamic two-archive EA to handle the DMOP with a changing number of objectives. More specifically, our proposed method simultaneously maintains two co-evolving populations, which have complementary effects on the search process. In particular, one population, denoted as the convergence archive (CA), mainly focuses on providing a competitive selection pressure towards the PF; while the other one, called the diversity archive (DA), maintains a set of solutions with a promising diversity. Furthermore, the compositions of the CA and the DA are adaptively reconstructed when the environment changes. In addition, the interaction between these two populations

are implemented by a restricted mating selection mechanism, where the mating parents are selected from the CA and the DA, respectively, according to the population distribution. Note that the multi-population strategy is not a brand new technique in the evolutionary multi-objective optimization (EMO) literature. For example, [35] and [36], the ancestors of this paper, used two complementary populations to balance the convergence and diversity during the search process. [37] developed a dual-population paradigm to take advantages of the strengths of the Pareto- and decomposition-based techniques. Nevertheless, none of them have taken the dynamic environment into consideration.

In the remainder of this paper, we will at first provide some preliminaries of this paper in Section II. Then, the technical details of our proposed algorithm are described step by step in Section III. Comprehensive experiments are conducted and analyzed in Section V. At last, Section VI concludes this paper and provides some future directions.

II. PRELIMINARIES

In this section, we first provide some definitions useful for this paper. Then, we describe the challenges posed by the DMOPs with a changing number of objectives. Finally, we discuss the pitfalls of the current dynamic handling techniques.

A. Basic Definitions

There are various types of dynamic characteristics that can result in different mathematical definitions. This paper focuses on the continuous DMOPs defined as follows:

$$\text{minimize} \quad \mathbf{F}(\mathbf{x}, t) = (f_1(\mathbf{x}, t), \cdots, f_{m(t)}(\mathbf{x}, t))^T$$
$$\text{subject to} \quad \mathbf{x} \in \Omega, t \in \Omega_t \quad (1)$$

where t is a discrete time defined as $t = \lfloor \frac{\tau}{\tau_t} \rfloor$, τ and τ_t represent the iteration counter and the frequency of change, respectively, and $\Omega_t \subseteq \mathbb{N}$ is the time space. $\Omega = \prod_{i=1}^{n}[a_i, b_i] \subseteq \mathbb{R}^n$ is the decision (variable) space, $\mathbf{x} = (x_1, \cdots, x_n)^T \in \Omega$ is a candidate solution. $\mathbf{F} : \Omega \to \mathbb{R}^{m(t)}$ constitutes of $m(t)$ real-valued objective functions and $\mathbb{R}^{m(t)}$ is the objective space at time step t, where $m(t)$ is a discrete function of t.

Definition 1. *At time step t, $\mathbf{x}^1$ is said to Pareto dominate $\mathbf{x}^2$, denoted as $\mathbf{x}^1 \preceq_t \mathbf{x}^2$, if and only if: $\forall i \in \{1, \cdots, m(t)\}$, $f_i(\mathbf{x}^1, t) \leq f_i(\mathbf{x}^2, t)$; and $\exists j \in \{1, \cdots, m(t)\}$, $f_j(\mathbf{x}^1, t) < f_j(\mathbf{x}^2, t)$.*

Definition 2. *At time step t, $\mathbf{x}^* \in \Omega$ is said to be Pareto-optimal, if there is no other $\mathbf{x} \in \Omega$ such that $\mathbf{x} \preceq_t \mathbf{x}^*$.*

Definition 3. *At time step t, the set of all Pareto-optimal solutions is called the t-th Pareto-optimal set (PS_t). The corresponding set of Pareto-optimal objective vectors is called the t-th Pareto-optimal front (PF_t), i.e., $PF_t = \{\mathbf{F}(\mathbf{x}, t) | \mathbf{x} \in PS_t\}$.*

Theorem 1. *When increasing the number of objectives, the PF and the PS at time step t are a subset of those at time step $t + 1$; on the contrary, when decreasing the number of objectives, the PF and the PS at time step t are a superset of those at time step $t + 1$.*

Theorem 1 supports the first characteristic given in Section I, and its proof can be found in Section I of the supplementary file[1].

Definition 4. *At time step t, the t-th ideal objective vector is* $\mathbf{z}^*(t) = (z_1^*(t), \cdots, z_{m(t)}^*(t))^T$, *where* $z_i^*(t) = \min\limits_{\mathbf{x}\in\Omega} f_i(\mathbf{x}, t), i \in \{1, \cdots, m(t)\}$.

Definition 5. *At time step t, the t-th nadir objective vector is* $\mathbf{z}^{nad}(t) = (z_1^{nad}(t), \cdots, z_{m(t)}^{nad}(t))^T$, *where* $z_i^{nad}(t) = \max\limits_{\mathbf{x}\in\Omega} f_i(\mathbf{x}, t), i \in \{1, \cdots, m(t)\}$.

Definition 6. *Under some mild conditions, the regularity property, induced from the Karush-Kuhn-Tucker condition, means that the PF and PS of a continuous m-objective MOP is an $(m-1)$-dimensional piecewise continuous manifold in both the objective space and the decision space.*

Note that an efficient EMO algorithm may employ this regularity property explicitly (e.g., [38]–[40]) or implicitly (e.g., [41]–[43]) in its implementation. This paper considers continuous DMOPs with the regularity property, according to which the PF_t and PS_t of the DMOP at time step t is a $(m(t)-1)$-dimensional piecewise continuous manifold in both $\mathbb{R}^{m(t)}$ and Ω. In addition, according to Definition 6, we can easily derive the second characteristic given in Section I.

B. Challenges of DMOP with a Changing Number of Objectives

Rather than shifting the position or adjusting the geometric shape of the PF or PS [44], changing the number of objectives usually results in the expansion or contraction of the dimensionality of the PF or PS manifold. In order to understand the challenges posed by changing the number of objectives, this subsection uses DTLZ2 [45] as the benchmark problem and multi-objective EA based on decomposition (MOEA/D) [46] as the baseline algorithm for investigations. Note that the DTLZ problems are scalable to any number of objectives and MOEA/D has been reported to show good performance on a wide range of problems [46]–[48]. To have a better visual comparison, we only discuss the challenges in the two- and three-objective scenarios while the challenges will become more severe with the growth of dimensionality.

1) When Increasing the Number of Objectives: First, we run MOEA/D on a two-objective DTLZ2 instance for 200 generations[2] and obtain a well approximation to the PF as shown in Fig. 1(a). Then, we change the underlying problem to a three-objective instance. Fig. 1(b) shows the corresponding population distribution after re-evaluating the objective values. From this subfigure, we can see that although the population still converges well to the PF, the diversity is not satisfied any longer after increasing the number of objectives (as shown in Fig. 1(b), the population crowds on a curve). Generally speaking, the most direct effect of increasing the number of objectives is the expansion of the PF or PS manifold. In

particular, the population convergence might not be influenced after increasing the number of objectives; whereas the population diversity becomes severely insufficient to approximate the expanded PF or PS manifold. These characteristics pose significant challenges to the timely adaptation to the changing environment, i.e., how to propel the population jump out of the crowded areas.
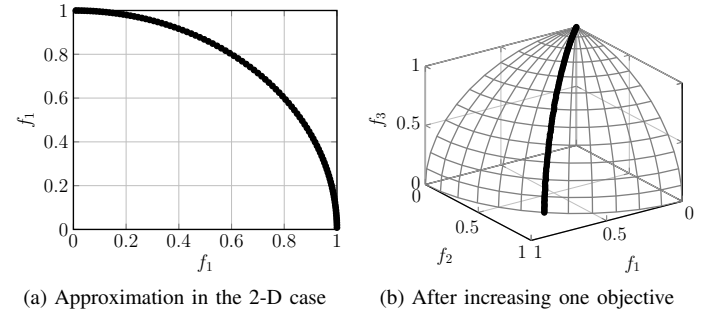

(a) Approximation in the 2-D case   (b) After increasing one objective

Fig. 1: Comparison of population distributions when increasing the number of objectives on DTLZ2.

2) When Decreasing the Number of Objectives: Here, we run MOEA/D on a three-objective DTLZ2 instance for 300 generations[3] and obtain a reasonably good approximation to the PF as shown in Fig. 2(a). Then, we change the underlying problem to a two-objective instance. Fig. 2(b) shows the corresponding population distribution after re-evaluating the objective values. In contrast to the scenario discussed in Section II-B1, the most direct effect of decreasing the number of objectives is the contraction of the PF or PS manifold. In addition, different from increasing the number of objectives, some solutions still stay on the PF after the objective reduction; while the others are moved away from the PF. Furthermore, although the spread of the population is not affected by decreasing an objective, the population diversity is not satisfied any longer (as shown in Fig. 2(b), there are many duplicate or similar solutions in the reduced objective space). These characteristics make the algorithm design difficult on both pulling the drifted solutions back to the PF and propelling the solutions jump out from their duplicates.

(a) Approximation in the 3-D case.   (b) After decreasing one objective.
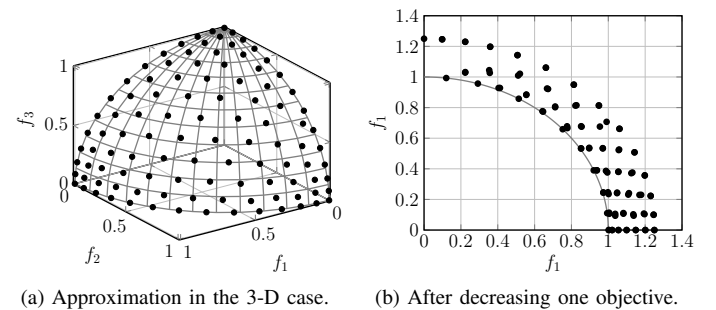
Fig. 2: Comparison of population distributions when decreasing the number of objectives on DTLZ2.

---

[1] https://coda-group.github.io/publications/suppDTAEA.pdf

[2] Here the differential evolution (DE) [49] is used as the reproduction operator, where $CR = 0.5$ and $F = 0.5$. The population size is set to 100, and the other parameters are set the same as suggested in [46].

[3] All parameters are set the same as those in Section II-B1, except for the population size which is set to 105.

C. Weaknesses of Existing Dynamic Handling Techniques

As discussed in Section I, the existing dynamic handling techniques do not take the changing number of objectives into consideration. They have the following weaknesses when encountering this sort of dynamics.

- As introduced in Section I, the diversity enhancement techniques use some randomly or heuristically injected candidates to adapt to the changing environment. However, when increasing the number of objectives, the injected candidates can hardly compete with the original population whose convergence has not be influenced. On the other hand, when decreasing the number of objectives, although the injected candidates are able to enhance the population diversity to a certain extent, they can hardly provide any selection pressure towards the PF.

- The memory mechanism has been used in [33] to handle the DMOP with a changing number of objectives. In particular, they suggested an inheritance strategy that simply re-evaluates the objective values of the original population whenever the change of the environment is detected. Obviously, this strategy is too simple to handle problems with complicated properties.

- The prediction strategy, which takes advantages of the historical information, seems to be an antidote for dynamic optimization. But unfortunately, the existing prediction strategies are usually proposed to anticipate the movement of the PS when the environment changes. However, as discussed in Section II-B, the change of the number of objectives usually leads to the expansion or contraction of the PS manifold, rather than the movement of its position. Thus, the prediction, based on the historical manifold model, can be heavily erroneous.

III. Proposed Algorithm

In this section, we develop a dynamic two-archive EA (denoted as DTAEA), whose high-level flow chart is shown in Fig. 3, for solving the DMOP with a changing number of objectives. Similar to its ancestors [50], [51] and [36], DTAEA maintains two co-evolving populations simultaneously: one, called the convergence archive (CA), is used to provide a constantly competitive selection pressure towards the optima; the other, called the diversity archive (DA), is used to provide diversified solutions as much as possible. It is worth noting that the size of the CA and the DA is equal and fixed to a constant N a priori. In each iteration, if a change of the environment is not occurred, mating parents are separately selected from the CA and the DA for offspring reproduction according to the mechanism described in Section III-C. Afterwards, the offspring is used to update the CA and the DA according to the mechanisms described in Section III-B. On the other hand, the CA and the DA are reconstructed according to the mechanism described in Section III-A whenever the underlying environment changes. At the end, all solutions of the CA are used to form the final population.

A. Reconstruction Mechanisms

This step is the main crux to respond to the changing environment. Generally speaking, its basic idea is to reconstruct the CA and the DA whenever the environment changes. Since the challenges posed by increasing and decreasing the number of objectives are different, as discussed in Section II-B, the reconstruction mechanisms are different accordingly.

1) When Increasing the Number of Objectives: As discussed in Section II-B1, the convergence of the population is not affected when increasing the number of objectives; while the population diversity becomes severely inadequate to approximate the expanded PF or PS manifold. To avoid sacrificing any selection pressure towards the optima, we use all optimal solutions in the last CA before increasing the number of objectives to form the new CA. In the meanwhile, all members of the DA are replaced by the randomly generated solutions. In particular, to provide as much diversified solutions as possible, here we employ the canonical Latin hypercube sampling (LHS) method [52] to uniformly sample N random solutions from the decision space. The pseudo-code of this reconstruction mechanism is given in Algorithm 1.

Algorithm 1: Reconstruction Mechanism After Increasing the Number of Objectives

Input: The last CA before decreasing the number of objectives P

Output: CA, DA

1   CA ← P;

2   Use the LHS to sample N random solutions to form the DA;

3   **return** CA and DA

---

2) When Decreasing the Number of Objectives: As discussed in Section II-B2, the convergence of some, if not all, population members might be impaired when decreasing the number of objectives; while the population diversity is also adversely affected due the existence of many similar or duplicate solutions in the reduced objective space. To keep the selection pressure towards the optima as much as possible, all non-dominated solutions in the last CA before decreasing the number of objectives are used to form the new CA. If the new CA is not full, we perform the polynomial mutation [53] on some selected competitive non-dominated solutions to generate some mutants to fill the gap in the new CA. In particular, the non-dominated solutions used for mutation are chosen by a binary tournament selection, where the non-dominated solution located in a less crowded area is preferred. Note that the density estimation method will be discussed in detail in Section III-B1. Since decreasing the number of objectives does not affect the spread of the population, instead of reconstructing the DA from scratch, we at first use all the dominated solutions in the last CA before decreasing the number of objectives to form the new DA. To provide enough diversified information to help the solutions jump out from their duplicates, we use the canonical LHS method to generate $N - |DA|$ random solutions to fill the gap in the new DA. The pseudo-code of this reconstruction mechanism is given in Algorithm 2.
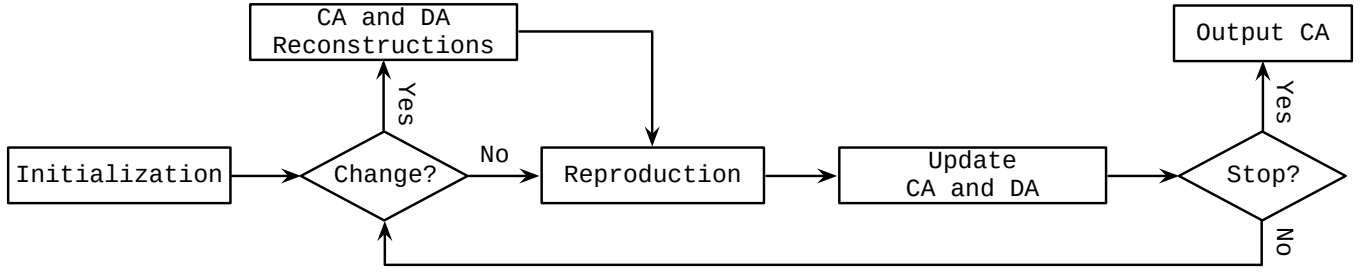
Fig. 3: Flow chart of DTAEA.

Algorithm 2: Reconstruction Mechanism After Decreasing the Number of Objectives

Input: The last CA before decreasing the number of objectives P

Output: CA, DA

1 CA ← NonDominationSelection(P);
  // choose all non-dominated solutions
2 DA ← P \ CA;
3 **while** $|\text{CA}| < N$ **do**
4 | $\mathbf{x}^c$ ← BirnaryTournamentSelection(CA);
5 | $\mathbf{x}^m$ ← PolynomialMutation($\mathbf{x}^c$);
6 | CA ← CA $\bigcup \{\mathbf{x}^m\}$;
7 Use the LHS to sample $N - |\text{DA}|$ random solutions to fill the DA;
8 **return** CA and DA

Algorithm 3: Binary Tournament Selection

Input: Solution set S

Output: Selected solution $\mathbf{x}^c$

1 Randomly select two solutions $\mathbf{x}^1$ and $\mathbf{x}^2$ from S;
2 **if** Density($\mathbf{x}^1$) < Density($\mathbf{x}^2$) **then**
3 | $\mathbf{x}^c$ ← $\mathbf{x}^1$;
4 **else if** Density($\mathbf{x}^1$) > Density($\mathbf{x}^2$) **then**
5 | $\mathbf{x}^c$ ← $\mathbf{x}^2$;
6 **else**
7 | $\mathbf{x}^c$ ← Randomly pick one between $\mathbf{x}^1$ and $\mathbf{x}^2$;
8 **return** $\mathbf{x}^c$

B. Update Mechanisms

As described at the outset of this section, the CA and the DA have complementary effects on the search process. Accordingly, their update mechanisms are different and will be described in Section III-B2 and Section III-B3, separately. Before that, Section III-B1 will at first introduces the density estimation method used in the update mechanisms.

1) Density Estimation Method: To facilitate the density estimation, we at first specify $N(t)$ uniformly distributed weight vectors, i.e., $\text{W}(t) = \{\mathbf{w}^1(t), \cdots, \mathbf{w}^{N(t)}(t)\}$, in $\mathbb{R}^{m(t)}$. In particular, we employ the weight vector generation method developed in [54] for this purpose, since it is scalable to the many-objective scenarios. Note that the number of weight vectors at time step t might not be equal to the size of the

CA and the DA, i.e., $N(t) \leq N$. In this case, a weight vector might be associated with more than one solution if $N(t) < N$. Accordingly, these weight vectors divide $\mathbb{R}^{m(t)}$ into $N(t)$ subspaces, i.e., $\Delta^1(t), \cdots, \Delta^{N(t)}(t)$. In particular, a subspace $\Delta^i(t)$, where $i \in \{1, \cdots, N(t)\}$ is defined as:

$$\Delta^i(t) = \{\mathbf{F}(\mathbf{x}, t) \in \mathbb{R}^{m(t)} | \langle \mathbf{F}(\mathbf{x}, t), \mathbf{w}^i(t) \rangle \leq \langle \mathbf{F}(\mathbf{x}, t), \mathbf{w}^j(t) \rangle\} \quad (2)$$

where $j \in \{1, \cdots, N(t)\}$ and $\langle \mathbf{F}(\mathbf{x}, t), \mathbf{w}(t) \rangle$ is the perpendicular distance between $\mathbf{F}(\mathbf{x}, t)$ and the reference line formed by the origin and $\mathbf{w}(t)$. After the setup of subspaces, each solution of the underlying population is associated with an unique subspace according to its position in $\mathbb{R}^{m(t)}$. Specifically, for a solution $\mathbf{x}$, the index of its associated subspace is determined as:

$$k = \operatorname*{argmin}_{i \in \{1, \cdots, N(t)\}} \langle \overline{\mathbf{F}}(\mathbf{x}, t), \mathbf{w}^i(t) \rangle \quad (3)$$

where $\overline{\mathbf{F}}(\mathbf{x}, t)$ is the normalized objective vector of $\mathbf{x}$, and its i-th objective function is calculated as:

$$\overline{f}_i(\mathbf{x}, t) = \frac{f_i(\mathbf{x}, t) - z_i^*(t)}{z_i^{nad}(t) - z_i^*(t)} \quad (4)$$

where $i \in \{1, \cdots, m(t)\}$. Based on the association relationship between solutions and subspaces, the density of a subspace is evaluated as the number of its associated solutions. Fig. 4 gives a simple example to illustrate this density estimation method. In particular, five weight vectors divide the underlying objective space into five subspaces, where the corresponding density of each subspace is 2, 3, 1, 1 and 3, respectively.

2) Update Mechanism of the CA: The effect of the CA is to provide a constantly competitive selection pressure towards the optima. To achieve this, we at first combine the CA and the offspring population into a mixed population R. Afterwards, we employ the fast non-dominated sorting method developed in [55] to divide R into several non-domination levels, i.e., F_1, F_2 and so on. Then, starting from F_1, each non-domination level is selected one at a time to construct the new CA. This procedure continues until the size of the new CA equals to or for the first time exceeds its predefined threshold. Let us denote the last included non-domination level as F_l, while all solutions from F_{l+1} onwards are not taken into consideration any longer. If the size of the new CA equals its predefined threshold, this update procedure terminates. Otherwise, we use the density estimation method introduced in Section III-B1 to evaluate the density information of the current CA. Then, we

eliminate one worst solution from the most crowded subspace at a time until the CA is trimmed to its predefined threshold. In particular, for a subspace $\Delta^i(t)$, the worst solution $\mathbf{x}^w$ is defined as:

$$\mathbf{x}^w = \underset{\mathbf{x} \in \Delta^i(t)}{\operatorname{argmax}}\{g^{tch}(\mathbf{x}|\mathbf{w}^i(t), \mathbf{z}^*(t))\} \quad (5)$$

where

$$g^{tch}(\mathbf{x}|\mathbf{w}^i(t), \mathbf{z}^*(t)) = \max_{1 \le j \le m(t)}\{|f_j(\mathbf{x}, t) - z_j^*(t)|/w_j^i(t)\}. \quad (6)$$

The pseudo-code of the update mechanism of the CA is given in Algorithm 4. Fig. 4 gives an example to illustrate this update mechanism. Assume that the black and green circles indicate the solutions of the original CA and the offspring population respectively. Since the first non-domination level already contains 8 solutions, $\mathbf{x}^3$ and $\mathbf{x}^7$, which belong to the second non-domination level, are not considered any longer. In the first iteration, the worst solution $\mathbf{x}^8$ in the most crowded subspace, i.e., $\Delta^5(t)$, is eliminated. Analogously, $\mathbf{x}^2$ and $\mathbf{x}^9$ will be eliminated in the latter iterations.
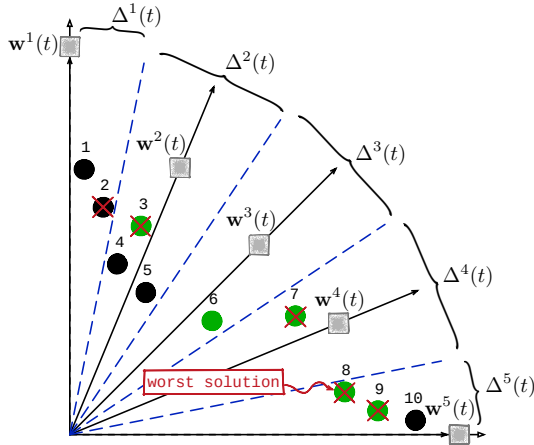

Fig. 4: An example of the CA's update mechanism. Note that $\mathbf{x}^i$ is denoted as its index i for short.

3) Update Mechanism of the DA: Comparing to the CA, the DA has a complementary effect which aims at providing diversified solutions as much as possible, especially in the areas under exploited by the CA. Similar to the update mechanism of the CA, we at first combine the DA and the offspring population into a mixed population R; in the meanwhile, we also take the up to date CA as the reference set. Afterwards, we employ the density estimation method introduced in Section III-B1 to build up the association relationship between solutions in R and the subspaces. Then, based on the association relationship, we iteratively investigate each subspace for which we try to keep itr ($1 \le$ itr $\le N$) solutions at the itr-th iteration. In particular, at the itr-th iteration, if the CA already has itr solutions or there is no solution in R associated with the currently investigating subspace, we stop considering this subspace during this iteration and move to the next subspace directly. Otherwise, the best non-dominated solution in R associated with this currently investigating subspace is chosen

---

Algorithm 4: Update Mechanism of the CA

Input: CA, offspring population Q, weight vector set W
Output: Updated CA

1 S $\leftarrow \emptyset$, $i \leftarrow 1$;
2 R $\leftarrow$ CA $\bigcup$ Q;
3 $\{F_1, F_2, \cdots\} \leftarrow$ NonDominatedSorting(R);
4 **while** $|\text{S}| \le N$ **do**
5 $\quad$ S $\leftarrow$ S $\bigcup F_i$, $i \leftarrow i + 1$;
6 **if** $|\text{S}| = N$ **then**
7 $\quad$ CA $\leftarrow$ S;
8 **else**
9 $\quad$ **foreach** $\mathbf{x} \in$ S **do**
10 $\quad\quad$ $\overline{\mathbf{F}}_k(\mathbf{x}, t) = \frac{\mathbf{F}(\mathbf{x}, t) - \mathbf{z}^*(t)}{\mathbf{z}^{nad}(t) - \mathbf{z}^*(t)}$;
11 $\quad$ $\{\Delta^1(t), \cdots, \Delta^{|W|}(t)\} \leftarrow$ Association(S, W);
12 $\quad$ **while** $|\text{S}| > N$ **do**
13 $\quad\quad$ Find the most crowded subspace $\Delta^i(t)$;
14 $\quad\quad$ $\mathbf{x}^w \leftarrow \underset{\mathbf{x} \in \Delta^i(t)}{\operatorname{argmax}}\{g^{tch}(\mathbf{x}|\mathbf{w}^i(t), \mathbf{z}^*(t))\}$;
15 $\quad\quad$ S $\leftarrow$ S $\setminus \{\mathbf{x}^w\}$;
16 $\quad$ CA $\leftarrow$ S;
17 **return** CA

Algorithm 5: Association Operation

Input: Solution set S, weight vector set W
Output: Subspaces $\Delta^1(t), \cdots, \Delta^{|W|}(t)$

1 **foreach** $\mathbf{x} \in$ S **do**
2 $\quad$ **foreach** $\mathbf{w} \in$ W **do**
3 $\quad\quad$ Compute $d^\perp(\mathbf{x}, \mathbf{w}) = \mathbf{x} - \mathbf{w}^T\mathbf{x}/\|\mathbf{w}\|$;
4 $\quad$ $k \leftarrow \underset{\mathbf{w} \in W}{\operatorname{argmin}} d^\perp(\mathbf{x}, \mathbf{w})$;
5 $\quad$ $\Delta^k(t) \leftarrow \Delta^k(t) \bigcup \{\mathbf{x}\}$;
6 **return** $\Delta^1(t), \cdots, \Delta^{|W|}(t)$

---

to be included into the newly formed DA. In particular, the best solution $\mathbf{x}^b$ of the currently investigating subspace $\Delta^c(t)$ is defined as:

$$\mathbf{x}^b = \underset{\mathbf{x} \in O}{\operatorname{argmin}}\{g^{tch}(\mathbf{x}|\mathbf{w}^c(t), \mathbf{z}^*(t))\} \quad (7)$$

where O constitutes of the non-dominated solutions in R associated with $\Delta^c(t)$. This iterative investigation continues until the DA is filled to its predefined threshold. The pseudo-code of the update mechanism of the DA is given in Algorithm 6. Fig. 5 gives an example to illustrate this update mechanism. Assume that the gray triangles represent the solutions of the up to date CA while the black and green circles indicate the solutions of the original DA and the offspring population. At the first iteration, the update mechanism of the DA starts from $\Delta^1(t)$. Since the CA already has two solutions in this subspace, we move to investigate $\Delta^2(t)$. As the CA does not has any solution in this subspace, the best non-dominated solution in this subspace, i.e., $\mathbf{x}^3$ is included into the newly formed DA. Thereafter, since $\Delta^3(t)$ to $\Delta^5(t)$ already have a solution in the CA, they are not considered during this

iteration. At the second iteration, since the CA already has two solutions in $\Delta^1(t)$, it is still not considered during this iteration. As for $\Delta^2(t)$, since the CA does not has any solution in this subspace, the remaining best solution $\mathbf{x}^5$ is selected this time. Then, since $\Delta^3(t)$ to $\Delta^5(t)$ only have one solution in the CA, we can choose the best solution from R to be included into the newly formed DA this time. At the end of the second iteration, the DA is filled and the update procedure terminates.
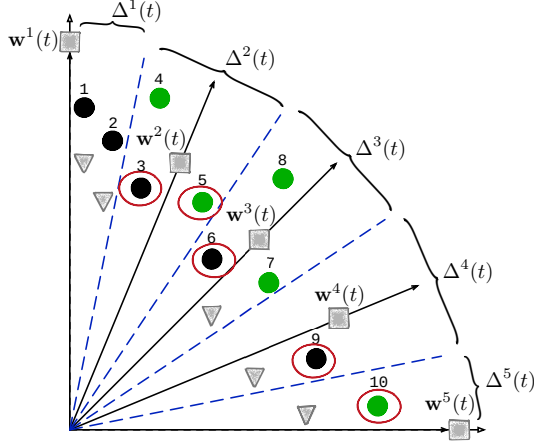

Fig. 5: An example of the DA's update mechanism. Note that $\mathbf{x}^i$ is denoted as its index i for short.

Algorithm 6: Update Mechanism of the DA

Input: CA, DA, offspring population Q, weight vector set W

Output: Updated DA

1   $\mathrm{S} \leftarrow \emptyset$, $i \leftarrow 1$;
2   $\mathrm{R} \leftarrow \mathrm{DA} \bigcup \mathrm{Q}$;
3   $\{\Delta^1(t), \cdots, \Delta^{|\mathrm{W}|}(t)\} \leftarrow \texttt{Association}(\mathrm{R}, \mathrm{W})$;
4   $\mathrm{itr} \leftarrow 1$;
5   **while** $|\mathrm{S}| \leq N$ **do**
6   **for** $i \leftarrow 1$ **to** $|\mathrm{W}|$ **do**
7   **if** $\Delta^i(t) \neq \emptyset \wedge$ CA *has less than* itr *solutions in* $\Delta^i(t)$ **then**
8         $\mathrm{O} \leftarrow \texttt{NonDominationSelection}(\Delta^i(t))$;
9         $\mathbf{x}^b \leftarrow \underset{\mathbf{x} \in \mathrm{O}}{\arg\min}\{g^{tch}(\mathbf{x}|\mathbf{w}^c(t), \mathbf{z}^*(t))\}$;
10        $\Delta^i(t) \leftarrow \Delta^i(t) \setminus \{\mathbf{x}^b\}$;
11        $\mathrm{S} \leftarrow \mathrm{S} \bigcup \{\mathbf{x}^b\}$;
12     $\mathrm{itr} \leftarrow \mathrm{itr} + 1$;
13   $\mathrm{DA} \leftarrow \mathrm{S}$;
14   **return** DA

C. Offspring Reproduction

The interaction between two co-evolving populations is a vital step in the DTAEA. To take advantage of the complementary effects between the CA and the DA, here the interaction is implemented as a restricted mating selection

mechanism that chooses the mating parents from these two co-evolving populations respectively according to the population distribution. It is worth noting that, comparing to the CA, the DA plays as an auxiliary which provides diversified information as much as possible and propels the adaptation to the changing environments. In addition, the CA and the DA become gradually asymptotic with the progress of the evolution. This results in a weakened complementary effects between these two co-evolving populations. Bearing these considerations in mind, the pseudo code of the restricted mating selection mechanism is given in Algorithm 7. More specifically, the first mating parent is randomly chosen from the CA. If the diversity of the CA is not promising, the second mating parent is randomly chosen from the DA; otherwise, it still comes from the CA. In particular, the diversity of the CA is measured by the occupation rate (denoted as I_{CA}^o) which is the percentage of the subspaces having associated solutions. It is intuitive to understand that a high I_{CA}^o indicates a well diversified distribution in the CA. Accordingly, we should give a larger chance to select the second mating parent from the CA. Based on the selected mating parents, the offspring solution can be generated by the appropriate crossover and mutation operations as desired. In particular, we use the classic simulated binary crossover [56] and the polynomial mutation in this paper.

Algorithm 7: Restricted Mating Selection Mechanism

Input: CA, DA

Output: Mating parents $\mathbf{p}_1$, $\mathbf{p}_2$

1   $\mathbf{p}_1 \leftarrow \texttt{RandomSelection}(\mathrm{CA})$;
2   **if** rnd $< I_{\mathrm{CA}}^o$ **then**
3     $\mathbf{p}_2 \leftarrow \texttt{RandomSelection}(\mathrm{CA})$;
4   **else**
5     $\mathbf{p}_2 \leftarrow \texttt{RandomSelection}(\mathrm{DA})$;
6   **return** $\mathbf{p}_1$, $\mathbf{p}_2$

D. Time Complexity Analysis

This subsection discusses the complexity of DTAEA in one generation. The CA and the DA will be reconstructed once the environment changes, i.e., when the number of objectives is increased or decreased. In the former case, i.e., when increasing the number of objectives, the reconstruction of the CA does not incur any further computation while the reconstruction of the DA costs $\mathcal{O}(N)$ function evaluations. As for the latter case, i.e., when decreasing the number of objectives, the identification of the non-dominated solutions costs $\mathcal{O}(N^2)$ comparisons. Then, filling the gap of the CA and the DA may respectively take $\mathcal{O}(N)$ function evaluations. As for the update mechanism of the CA, the most time consuming operations are the non-dominated sorting (line 3 of Algorithm 4) and the association operation (line 11 of Algorithm 4). In particular, the prior one costs $\mathcal{O}(N^2)$ comparisons while the latter one takes $\mathcal{O}(N|W|)$, where $|W|$ returns the cardinality of a W. As for the update mechanism of the DA, the association operation in line 3 of Algorithm 6

costs $\mathcal{O}(N|W|)$ comparisons. During the main while loop, the update procedure performs one by one for the subspaces. In particular, if $\Delta^i(t) \neq \emptyset$, where $i \in \{1, \cdots, |W|\}$, the most time consuming part of the for loop from line 6 to line 11 is the non-dominated sorting in line 8 of Algorithm 6, which costs $\mathcal{O}(|\Delta^i(t)|^2)$ comparisons. Since $\sum_{i=1}^{|W|} \Delta^i(t) = 2N$, this for loop costs $\mathcal{O}(N^2)$ comparisons in total. In summary, the complexity of DTAEA in one generation is $\mathcal{O}(N^2)$.

E. Further Discussions

As DTAEA is developed upon the decomposition-based framework, this subsection discusses the connections with some state-of-the-art decomposition-based EMO algorithms, MOEA/D-M2M [57], NSGA-III [58], MOEA/D-GR [59] and MOEA/D-STM [60]–[62]. Similar to MOEA/D-M2M, DTAEA uses the weight vectors to divide the objective space into several subspaces. However, the major purpose of subspace division in MOEA/D-M2M is to enforce an equal distribution of the computational resources to each subspace; while the subspace division is used to facilitate the density estimation in DTAEA. Similar to NSGA-III, MOEA/D-GR and MOEA/D-STM, each weight vector in DTAEA is associated with its related solutions to facilitate the environmental selection. It uses different strategies to maintain its two co-evolving archives. In particular, according to the association relationship, the CA trims solutions from the most crowded subspaces; while the DA gives higher survival rates to solutions resided in the subspaces under exploited by the CA. In contrast, NSGA-III merely uses the association relationship to guide the truncation through the most crowded subspaces. In MOEA/D-GR and MOEA/D-STM, the next parent population is composed of the elite solutions from each of the non-empty subspace, which has at least one associated solution.

IV. EXPERIMENTAL SETTINGS

This section introduces the benchmark problems, performance metrics and the state-of-the-art EMO algorithms used in the experimental studies of this paper.

A. Benchmark Problems

Benchmark problems play important roles in assessing and analysing the performance of an algorithm, and thus guiding its further developments. Although some dynamic multi-objective benchmark problems have been proposed in the literature, e.g., [44] and [63], most, if not all, of them merely consider the dynamically changing shape or position of the PF or PS. In [34], a dynamic multi-objective test instance with a changing number of objectives was developed, for the first time, by modifying the classic DTLZ benchmark suite [45]. In this paper, we develop a dynamic multi-objective benchmark suite as shown in Section II of the supplementary file. Note that, in addition to the changing number of objectives, F5 and F6 are also accompanied by a time-dependent change of the shape or position of the PF or PS.

B. Performance Metrics

In the field of DMO, there is still no standard metric for quantitatively evaluating the performance of algorithms. In this paper, we employ the following two metrics, adapted from two popular metrics used in the literature, for the performance assessment.

- *Mean Inverted Generational Distance (MIGD)* [16]: Let P_t^* is a set of points uniformly sampled along the PF_t, S_t is the set of solutions obtained by an EMO algorithm for approximating the PF_t, and T is a set of discrete time steps in a run,

$$\text{MIGD} = \frac{1}{|T|} \sum_{t \in T} \text{IGD}(P_t^*, S_t). \qquad (8)$$

In particular, as defined in [64],

$$\text{IGD}(P_t^*, P_t) = \frac{\sum_{\mathbf{x} \in P} dist(\mathbf{x}, P_t^*)}{|S|}, \qquad (9)$$

where $dist(\mathbf{x}, P_t^*)$ is the Euclidean distance between a point $\mathbf{x} \in S_t$ and its nearest neighbor in P_t^*. Note that the calculation of IGD requires the prior knowledge of the PF. In our experiments, we use the method suggested in [54] to sample 10,000 uniformly distributed points on the corresponding PF_t at the t-th time step where $t \in T$.
- *Mean Hypervolume (MHV)*: Let $\mathbf{z}_t^w = (z_1^w, \ldots, z_{m(t)}^w)^T$ be a worst point in the objective space that is dominated by all Pareto-optimal objective vectors at the t-th time step,

$$\text{MHV} = \frac{1}{|T|} \sum_{t \in T} \text{HV}(S_t). \qquad (10)$$

In particular, as defined in [65], HV measures the size of the objective space dominated by solutions in S_t and bounded by $\mathbf{z}_t^w$. It is calculated as:

$$\text{HV}(S_t) = \text{VOL}(\bigcup_{\mathbf{x} \in S_t} [f_1(\mathbf{x}), z_1^w] \times \ldots [f_{m(t)}(\mathbf{x}), z_{m(t)}^w])$$
$$(11)$$

where $\text{VOL}(\cdot)$ indicates the Lebesgue measure. Note that solutions, dominated by the worst point, are discarded for HV calculation. In our experiments, we set $\mathbf{z}_t^w = \underbrace{(2, \cdots, 2)}_{m(t)}^T$. For a better presentation, the HV values used in equation (10) are normalized to $[0, 1]$ by dividing $z_t = \prod_{i=1}^m z_i^w$.

C. EMO Algorithms Used in the Experimental Studies

In our empirical studies, four state-of-the-art EMO algorithms are used for comparisons: the dynamic version of the elitist non-dominated sorting genetic algorithm (DNSGA-II) [3] and MOEA/D with Kalman Filter prediction (MOEA/D-KF) [21]; and their corresponding stationary baseline NSGA-II [55] and MOEA/D [41]. They were chosen because of their popularity and good performance in both dynamic and static environments. Comparisons with the baseline algorithms are important. Because we want to check whether the dynamic algorithms outperform their static counterparts or not when handling the DMOP with a changing number of objectives.

More detailed descriptions of these algorithms are given in Section III of the supplementary file. Each algorithm is independently run 31 times on each problem instance. 300 generations are given to each algorithm before the first change. In other words, the first change occurs after the first 300 generations. To have a statistically sound conclusion, we use the Wilcoxon rank sum test at the 5% significance level in the comparisons.

V. EXPERIMENTAL RESULTS

A. Results on F1 to F4

Let us at first consider F1 to F4 in which only the number of objectives changes with time. In particular, we define the time varying number of objectives $m(t)$ as follows:

$$m(t) = \begin{cases} 3, & t = 1 \\ m(t-1) + 1, & t \in [2, 5] \\ m(t-1) - 1, & t \in [6, 10] \end{cases} \quad (12)$$

where $t \in \{1, \cdots, 10\}$ is a discrete time. The population size is constantly set as 300 while the number of weight vectors is set according to Section IV of the supplementary file. To investigate the performance of different algorithms under various frequencies of change, we set τ_t as 25, 50, 100, 200, respectively. Table I and Table II give the median and the interquartile range (IQR) of the corresponding metric values obtained by different algorithms under various circumstances. In particular, the best metric values are highlighted in the bold face with a gray background. In addition to the metric values, we also keep a record of the ranks of the IGD and HV values obtained by different algorithms at each time step. We assign a global rank to each algorithm by averaging the ranks obtained at all time steps. The experimental results clearly demonstrate that DTAEA is the best optimizer as it wins on all comparisons (64 out of 64 for MIGD and 64 out of 64 for MHV), and it is always top ranked. In the following paragraphs, we will explain these results in detail.

F1 is developed from DTLZ1 [45] which has a multi-modal property to hinder an algorithm from converging to the PF_t, $t \in \{1, \cdots, 10\}$. As shown in Table I and Table II, the performance of DTAEA is robust as it obtains the best MIGD and MHV values under all four frequencies of change. Fig. 1(a) of the supplementary file shows the trajectories of IGD values obtained by different algorithms across the whole evolution process; while Fig. 2(a) of the supplementary file shows the average ranks of IGD obtained by different algorithms at each time step. From these two figures, we clearly see that DTAEA shows the best performance at every time step. It is worth noting that the performance of DTAEA has some fluctuations when $\tau_t = 25$. This is because, under a high frequency of change, DTAEA can hardly drive the solutions fully converge to the PF_t before the environment changes. However, with the decrease of the frequency of change, i.e., the increase of τ_t, the performance of DTAEA becomes stable. As for the other four algorithms, the performance of different algorithms fluctuate a lot at different time steps. It is interesting to note that the performance of the stationary

algorithms, i.e., NSGA-II and MOEA/D, are comparable to their dynamic counterparts, i.e., DNSGA-II and MOEA/D-KF. In particular, under a high frequency of change, i.e, $\tau_t = 25$, NSGA-II and MOEA/D have shown a better performance than DNSGA-II and MOEA/D-KF at every time step. This implies that the dynamic handling techniques of DNSGA-II and MOEA/D-KF might not be capable of handling the expansion or contraction of the objective space. Even worse, these mechanisms can have an adverse effect to the population for adapting to the changing environments. In addition, we also notice that NSGA-II and DNSGA-II show better performance than MOEA/D and MOEA/D-KF at the first several time steps; whereas their performance degenerate significantly afterwards. This is because NSGA-II can have a faster convergence than MOEA/D when the number of objectives is relatively small. As discussed in Section II-B, the increase of the number of objectives does not influence the population convergence. Thus, the competitive IGD values obtained at the 2- and 3-objective cases have an accumulative effect which makes NSGA-II and DNSGA-II maintain a competitive performance at the first couple of time steps. However, the ineffectiveness of NSGA-II and DNSGA-II for handling problems with a large number of objectives [66] leads to their poor performance at the latter time steps. Even worse, their poor performance at 6- and 7-objective scenarios also disseminate a negative accumulation which results in their poor performance when decreasing the number of objectives.

F2 is developed from DTLZ2 [45] which is a relatively simple benchmark problem. From Fig. 1(b) and Fig. 2(b) of the supplementary file, we also find that DTAEA has shown the consistently best performance across all time steps. It is interesting to note that the MIGD and MHV values obtained by all five algorithms lie in the same scale. This is because F2 does not pose too much challenge to the algorithms for converging to the PF_t. Thus all algorithms are able to adapt their populations to the changing environments within a reasonable number of function evaluations. Due to this reason, as shown in Fig. 1(b) of the supplementary file, the IGD trajectories of NSGA-II and DNSGA-II surge up to a relatively high level when the number of objectives becomes large; whereas their IGD trajectories gradually go down when decreasing the number of objectives. Moreover, similar to the observations on F1, DNSGA-II and MOEA/D-KF do not show superior performance than their corresponding stationary counterparts on F2.

F3 is developed from DTLZ3 [45] which has the same PF_t, $t \in \{1, \cdots, 10\}$, as F2 but has a multi-modal property. Due to the multi-modal property, which hinders the population from approaching the PF_t, the IGD trajectories of NSGA-II and DNSGA-II can hardly drop down when decreasing the number of objectives. In the meanwhile, we still notice that the dynamic handling techniques of DNGSA-II and MOEA/D-KF do not help the population adapt to the changed environments. Even worse, the injected solutions provide a false information about the newly changed PF or PS, which is harmful to the evolution process.

F4 is developed from DTLZ4 [45] which also has the same PF_t, $t \in \{1, \cdots, 10\}$, as F2 but has a parametric mapping

that introduces a biased density of Pareto-optimal solutions towards some particular coordinates. In this case, F4 not only poses significant challenges for handling the changing number of objectives, but also requires that the algorithm can have a well balance between convergence and diversity. The superior performance of DTAEA can be attributed to the dedicated operations of two co-evolving populations for balancing convergence and diversity during the whole evolution process. Similar to the observations on the previous problem instances, the stationary algorithms still show competitive performance than their dynamic counterparts at most time steps.

B. Results on F5 and F6

Different from F1 to F4, F5 and F6 not only consider a changing number of objectives, but also have a time varying PS. Here we use equation (12) to define the time varying number of objectives; as for the parameters related to the time varying PS, we set its change frequency as $\tau_{\bar{t}} = 5$ and change severity as $n_{\bar{t}} = 10$ (these settings are widely used in the literature, e.g., [10], [67] and [68]). From the experimental results shown in Table I and Table II, we find that DTAEA has shown the best performance on almost all comparisons (30 out of 32 for MIGD and 28 out of 32 for MHV). In the following paragraphs, we will explain these results in detail.

F5 is developed from F2 and is with a time varying PS of which the position is changed with time. Although the overall performance of DTAEA is the best as shown in Table I and Table II, its performance fluctuates significantly under a high frequency of change, i.e., $\tau_t = 25$ and $\tau_t = 50$, as shown in Fig. 1(e) and Fig. 2(e) of the supplementary file. It is worth noting that DTAEA does not have any dynamic detection or handling technique for the time varying PS. Thus, under a high frequency of change, DTAEA does not make a good adaption to two kinds of dynamics. However, as discussed in Section V-A, F2 does not pose too much difficulty to the algorithm for converging to the PF. This explains the comparable performance achieved by the other four algorithms under a high frequency of change. Nevertheless, even without any specific dynamic handling technique for the time varying PS, the complementary effects of two archives of DTAEA help the population adapt to the changing environments when decreasing the frequency of change, i.e., increasing τ_t. Furthermore, NSGA-II and MOEA/D still show comparable performance than their dynamic counterparts when encountering two kinds of dynamics.

F6 is also developed from F2 but it poses more challenges to the algorithm for approaching the PF. Similar to the observations on F5, the performance of DTAEA fluctuate significantly under a high frequency of change as shown in Fig. 1(f) and Fig. 2(f) of the supplementary file. However, since the g function of F6, which controls the difficulty for converging to the PF, is more difficult than that of F5, the superiority of DTAEA is more observable than F5. In addition, although the dynamic handling technique of DNSGA-II and MOEA/D-KF are designed for handling the time varying PS, they do not show better performance than their stationary counterparts. We explain these observations as that the severity

of change of the time varying PS is smaller than the expansion or contraction of the PF or PS manifold when changing the number of objectives. Thus, due to the ineffectiveness of the dynamic reaction mechanisms of DNSGA-II and MOEA/D-KF for handling the changing number of objectives, it makes their performance be not much indifferent from their dynamic counterparts.

C. Further Analysis

From the experimental studies in Section V-A and Section V-B, it is clear to see the superiority of our proposed DTAEA for handling the DMOPs with a dynamically changing number of objectives. As introduced in Section III, DTAEA consists of three major components. The following paragraphs analyze their effects separately.

1) Effects of the Reconstruction Mechanism: As discussed in Section II-B, the population convergence might not be affected when increasing the number of objectives. However, the diversity enhancement strategy, e.g., the random injection of DNSGA-II, has an adverse effect to the population convergence. In the meanwhile, it cannot provide enough diversified information to remedy the loss of population diversity. As for the prediction strategy, e.g., the Kalman filter of MOEA/D-KF, it is usually developed to predict the change of the PS's position or shape, instead of the expansion or contraction of the PS or PF manifold. As a consequence, its large prediction errors are harmful to both the convergence and the diversity. Since the stationary EMO algorithms do not make any reaction to the changing environment, the population convergence is not affected; while the self-adaptive property of EA can also help the population gradually adapt to the changed environment. These explain the competitive performance of the stationary EMO algorithms comparing to their dynamic counterparts for handling the DMOPs with a dynamically changing number of objectives. In contrast, the CA of our proposed DTAEA, consisted of the elite solutions, preserves the population convergence; in the meanwhile, the DA of DTAEA, consisted of a set of uniformly sampled solutions from the decision space, provide as much diversified information as possible to help the population adapt to the changed environment. In Fig. 3 of the supplementary file, we keep track of the population distribution for several iterations after the number of objectives increases from two to three. From this figure, we can clearly see that neither NSGA-II nor DNSGA-II is able to spread the population across the expanded PF. As for MOEA/D and MOEA/D-KF, although the population seems to have an acceptable spread along the expanded PF, the solutions are a bit away from the PF. In contrast, our proposed DTAEA successfully spread the solutions across the PF without sacrificing any convergence property.

As discussed in Section II-B, the spread of the population might not be affected too much when decreasing the number of objectives, while some solutions are propelled away from the PF. In this case, the randomly injected solutions of DNGSA-II cannot provide sufficient selection pressure towards the PF. As for MOEA/D-KF, due to the large prediction errors, the effect of those predicted solutions of the Kalman filter is

TABLE I: Performance Comparisons of DTAEA and the Other Algorithms on MIGD Metric

	τ_t	NSGA-II MIGD	R	DNSGA-II MIGD	R	MOEA/D MIGD	R	MOEA/D-KF MIGD	R	DTAEA MIGD	R
F1	25	1.40E-2(9.25E-3)†	3.3	1.32E-2(6.51E-3)†	3.3	1.53E-3(2.18E-4)†	3.1	2.05E-2(1.81E-2)†	4.2	**5.50E-4(1.99E-4)**	**1.1**
	50	3.36E-2(3.33E-2)†	3.7	3.38E-2(2.07E-2)†	3.8	8.53E-4(9.43E-4)†	2.9	5.22E-3(6.23E-3)†	3.5	**3.88E-4(1.03E-5)**	**1**
	100	1.99E-2(5.68E-2)†	3.9	2.84E-2(6.27E-2)†	4.1	7.36E-4(4.29E-4)†	2.7	9.78E-4(6.03E-4)†	3.3	**3.64E-4(2.49E-6)**	**1**
	200	1.71E-2(5.38E-2)†	4.3	7.83E-3(3.45E-2)†	4.1	6.47E-4(8.72E-4)†	2.6	6.75E-4(4.58E-4)†	3	**3.55E-4(1.41E-6)**	**1**
F2	25	1.92E-3(1.27E-4)†	3.2	2.02E-3(2.12E-4)†	3.6	2.19E-3(1.17E-4)†	3.3	2.26E-3(6.71E-5)†	3.8	**1.26E-3(4.58E-6)**	**1.1**
	50	2.20E-3(9.07E-5)†	3.6	2.24E-3(1.47E-4)†	3.7	2.08E-3(5.81E-5)†	3.2	2.10E-3(6.03E-5)†	3.5	**1.25E-3(3.26E-6)**	**1**
	100	2.45E-3(1.37E-4)†	3.8	2.47E-3(1.41E-4)†	3.9	2.01E-3(2.89E-5)†	3.1	1.99E-3(3.51E-5)†	3.1	**1.22E-3(1.30E-6)**	**1**
	200	2.46E-3(1.61E-4)†	3.8	2.50E-3(1.48E-4)†	3.9	1.93E-3(4.26E-5)†	3.1	1.92E-3(3.41E-5)†	3.1	**1.20E-3(2.20E-6)**	**1**
F3	25	2.15E-2(1.78E-2)†	3.1	2.62E-2(1.82E-2)†	3.1	4.68E-3(1.25E-2)†	3.5	4.90E-2(6.92E-2)†	4.1	**2.06E-3(1.34E-3)**	**1.1**
	50	9.09E-2(3.82E-2)†	3.7	8.88E-2(4.13E-2)†	3.9	2.96E-3(5.33E-4)†	2.9	1.02E-2(5.72E-3)†	3.4	**1.39E-3(1.06E-4)**	**1.1**
	100	1.61E-1(1.08E-1)†	4	1.35E-1(1.22E-1)†	4	2.22E-3(1.31E-4)†	2.7	2.57E-3(9.73E-4)†	3.2	**1.24E-3(9.04E-6)**	**1**
	200	1.82E-1(2.01E-1)†	4.2	1.91E-1(1.47E-1)†	4.2	2.09E-3(4.00E-3)†	2.6	2.16E-3(9.26E-4)†	2.9	**1.22E-3(4.17E-6)**	**1**
F4	25	3.06E-3(5.08E-4)†	2.7	3.66E-3(5.08E-4)†	3.4	4.19E-3(3.56E-4)†	4.1	4.02E-3(2.12E-4)†	3.8	**1.32E-3(7.80E-5)**	**1.1**
	50	2.93E-3(3.78E-4)†	3.1	3.19E-3(3.71E-4)†	3.4	3.74E-3(2.07E-4)†	3.9	3.62E-3(3.94E-4)†	3.6	**1.24E-3(4.05E-6)**	**1**
	100	3.19E-3(3.12E-4)†	3.4	3.16E-3(4.91E-4)†	3.4	3.38E-3(6.86E-4)†	3.6	3.15E-3(3.86E-4)†	3.6	**1.22E-3(1.50E-6)**	**1**
	200	2.79E-3(1.82E-4)†	3.3	2.79E-3(3.63E-4)†	3.3	2.73E-3(4.51E-4)†	3.6	2.78E-3(5.72E-4)†	3.8	**1.20E-3(1.35E-6)**	**1**
F5	25	7.93E-3(3.37E-3)†	4.2	3.69E-3(2.93E-4)†	4	2.61E-3(2.62E-4)	2.6	**2.56E-3(1.78E-4)‡**	2.5	2.61E-3(9.71E-5)	**1.8**
	50	3.36E-3(2.45E-4)†	3.8	3.69E-3(2.36E-4)†	4.5	2.31E-3(1.25E-4)†	2.6	2.24E-3(1.08E-4)†	2.5	**1.91E-3(6.69E-5)**	**1.7**
	100	2.02E-3(1.80E-4)†	3.2	2.24E-3(1.23E-4)†	4	2.13E-3(9.99E-5)†	3.5	1.99E-3(4.51E-5)†	2.9	**1.45E-3(2.98E-5)**	**1.3**
	200	1.91E-3(1.06E-4)†	2.9	2.20E-3(1.23E-4)†	4.3	2.02E-3(7.70E-5)†	3.4	1.93E-3(4.98E-5)†	3.1	**1.36E-3(6.77E-6)**	**1.2**
F6	25	8.10E-3(1.09E-3)†	4.3	5.19E-3(7.90E-4)†	3.7	3.12E-3(3.53E-4)†	2.7	**2.93E-3(1.86E-4)**	2.5	2.98E-3(1.76E-4)	**1.7**
	50	5.27E-3(5.91E-4)†	4.5	5.60E-3(4.67E-4)†	4.1	2.68E-3(7.39E-5)†	2.6	2.71E-3(1.72E-4)†	2.5	**2.08E-3(5.08E-5)**	**1.3**
	100	3.56E-3(3.48E-4)†	4.2	3.29E-3(1.11E-4)†	4.2	2.36E-3(1.95E-4)†	2.8	2.32E-3(1.08E-4)†	2.8	**1.56E-3(2.25E-5)**	**1.1**
	200	3.90E-3(3.72E-4)†	4.4	3.10E-3(1.46E-4)†	4	2.24E-3(9.08E-5)†	2.6	2.16E-3(1.06E-4)†	2.8	**1.46E-3(2.09E-5)**	**1.2**

R denotes the global rank assigned to each algorithm by averaging the ranks obtained at all time steps. Wilcoxon's rank sum test at a 0.05 significance level is performed between DTAEA and each of NSGA-II, DNSGA-II, MOEA/D and MOEA/D-KF. † and ‡ denote the performance of the corresponding algorithm is significantly worse than and better than that of DTAEA, respectively. The best median value is highlighted in boldface with gray background.

TABLE II: Performance Comparisons of DTAEA and the Other Algorithms on MHV Metric

	τ_t	NSGA-II MHV	R	DNSGA-II MHV	R	MOEA/D MHV	R	MOEA/D-KF MHV	R	DTAEA MHV	R
F1	25	74.8%(9.58E-2)†	3.6	72.3%(8.08E-2)†	3.7	97.3%(1.09E-2)†	2.7	74.6%(4.33E-2)†	3.9	**99.7%(6.03E-3)**	**1.1**
	50	60.1%(1.16E-1)†	4.1	61.2%(1.02E-1)†	4.1	99.7%(3.40E-2)†	2.6	78.0%(1.22E-1)†	3.2	**100.0%(8.39E-5)**	**1**
	100	74.9%(2.10E-1)†	4	69.3%(2.43E-1)†	4.1	99.9%(1.65E-2)†	2.7	99.0%(2.07E-2)†	3.2	**100.0%(1.84E-5)**	**1**
	200	76.0%(3.43E-1)†	4.1	83.3%(3.58E-1)†	3.9	100.0%(4.65E-2)†	2.8	100.0%(2.15E-2)†	3.2	**100.0%(2.69E-5)**	**1**
F2	25	92.4%(3.47E-3)†	4.1	92.4%(6.01E-3)†	4	93.5%(2.91E-3)†	2.6	93.4%(1.36E-3)†	3.1	**94.4%(6.26E-5)**	**1**
	50	91.9%(4.14E-3)†	4.2	91.7%(5.28E-3)†	4.1	93.8%(1.06E-3)†	2.6	93.7%(1.36E-3)†	3.1	**94.4%(3.09E-5)**	**1**
	100	91.0%(5.33E-3)†	4.3	90.6%(8.46E-3)†	4.4	93.9%(4.55E-4)†	2.5	93.9%(6.22E-4)†	2.8	**94.5%(8.88E-6)**	**1**
	200	90.8%(6.80E-3)†	4.3	90.4%(7.10E-3)†	4.4	94.0%(5.10E-4)†	2.6	94.0%(5.09E-4)†	2.7	**94.5%(4.03E-6)**	**1**
F3	25	67.1%(7.75E-2)†	3.4	69.3%(7.98E-2)†	3.3	80.6%(6.04E-1)†	3.4	59.9%(5.09E-2)†	3.8	**90.6%(9.47E-2)**	**1.1**
	50	49.9%(2.21E-2)†	4	49.3%(3.46E-2)†	4.1	89.8%(2.96E-2)†	2.5	67.2%(5.19E-2)†	3.3	**94.0%(3.66E-3)**	**1.1**
	100	45.9%(6.49E-2)†	4.3	44.8%(7.63E-2)†	4.3	93.2%(8.02E-3)†	2.4	91.3%(6.33E-2)†	3	**94.4%(3.32E-4)**	**1**
	200	44.0%(7.13E-2)†	4.2	44.6%(7.75E-2)†	4.2	93.6%(7.25E-2)†	2.7	93.3%(5.40E-2)†	3	**94.4%(1.26E-4)**	**1**
F4	25	91.2%(4.96E-3)†	3.3	88.8%(2.09E-2)†	3.9	90.8%(1.67E-2)†	3.4	91.7%(8.38E-3)†	3.3	**94.4%(1.60E-4)**	**1**
	50	88.8%(2.04E-2)†	3.6	87.6%(2.81E-2)†	3.7	91.9%(9.97E-3)†	3.4	92.1%(1.41E-2)†	3.3	**94.4%(1.16E-5)**	**1**
	100	86.6%(2.83E-2)†	3.6	86.7%(3.45E-2)†	3.5	92.4%(1.31E-2)†	3.4	92.6%(8.25E-3)†	3.5	**94.5%(3.26E-6)**	**1**
	200	88.8%(1.32E-2)†	4	88.5%(2.65E-2)†	3.9	92.9%(1.13E-2)†	3	92.9%(1.26E-2)†	3.2	**94.5%(4.19E-6)**	**1**
F5	25	58.6%(1.38E-1)†	4.5	87.0%(2.36E-2)†	4.1	93.3%(6.10E-3)†	2.2	**93.3%(2.82E-3)‡**	2.3	89.4%(6.64E-3)	**2**
	50	86.3%(1.60E-2)†	4.3	85.9%(1.60E-2)†	4.3	94.0%(2.02E-3)†	2.1	**94.1%(3.35E-3)‡**	**1.9**	93.7%(1.25E-3)	2.4
	100	93.4%(5.24E-3)†	3.8	93.5%(4.20E-3)†	4.3	94.3%(1.15E-3)†	2.9	94.4%(1.95E-3)†	2.3	**94.6%(3.62E-4)**	**1.7**
	200	94.0%(3.40E-3)†	3.7	93.3%(4.36E-3)†	4.5	94.5%(6.96E-4)†	2.9	94.5%(2.20E-3)†	2.3	**94.8%(1.84E-4)**	**1.6**
F6	25	60.1%(6.41E-2)†	4.7	75.8%(6.36E-2)†	3.8	93.2%(1.72E-2)†	2.2	**93.5%(2.77E-3)‡**	**2.1**	89.1%(9.72E-3)	2.3
	50	78.9%(3.83E-2)†	4.6	73.6%(2.76E-2)†	4.3	94.0%(1.71E-3)†	1.9	**93.9%(1.73E-3)‡**	**1.8**	93.7%(1.45E-3)	2.4
	100	88.0%(1.36E-2)†	4.8	91.8%(4.13E-3)†	4	94.3%(1.54E-3)†	2.3	94.2%(2.15E-3)†	2.4	**94.5%(3.50E-4)**	**1.5**
	200	87.9%(1.91E-2)†	4.7	92.1%(5.84E-3)†	4.1	94.5%(5.71E-4)†	2.5	94.5%(1.35E-3)†	2.3	**94.7%(3.39E-4)**	**1.4**

R denotes the global rank assigned to each algorithm by averaging the ranks obtained at all time steps. Wilcoxon's rank sum test at a 0.05 significance level is performed between DTAEA and each of NSGA-II, DNSGA-II, MOEA/D and MOEA/D-KF. † and ‡ denote the performance of the corresponding algorithm is significantly worse than and better than that of DTAEA, respectively. The best median value is highlighted in boldface with gray background.

almost the same as the random solutions. In contrast, DTAEA uses the non-dominated solutions of the elite population to construct the CA; in the meanwhile, it uses the remaining ones, which have a well spread in the objective space, to construct the DA. By these means, the population is able to have a well balance between convergence and diversity in the changed environment. In Fig. 4 of the supplementary file, we keep track of the population distribution for several iterations after the number of objectives decreases from four to three. From this figure, we can clearly see that the population spread after decreasing the number of objectives is acceptable in the first place, but having some duplicate solutions. After several iterations, DTAEA achieves a significantly better performance on both convergence and diversity than the other comparative algorithms.

2) Effects of the Restricted Mating Selection Mechanism: As discussed in Section III-C, the CA and the DA have different characteristics. In particular, the prior one concerns more about convergence whereas the latter one concerns more about the diversity. Their complementary effect, which finally contributes to the balance between convergence and diversity, is implemented by the restricted mating selection mechanism between them. In particular, one of the mating parents is constantly selected from the CA while the selection of the other one depends on the diversity information of the CA. In order to validate the effect of this restricted mating selection mechanism, we choose F2 as the instance where $\tau_t = 50$ and plot the trajectories of proportion of second mating parent selected from the CA and DA respectively in Fig. 6. From this figure, we notice that the proportion for selecting the second mating parent from the DA is relatively high at the beginning when the number of objectives is changed. But with the progress of evolution, both mating parents are selected from the CA when its diversity becomes better.
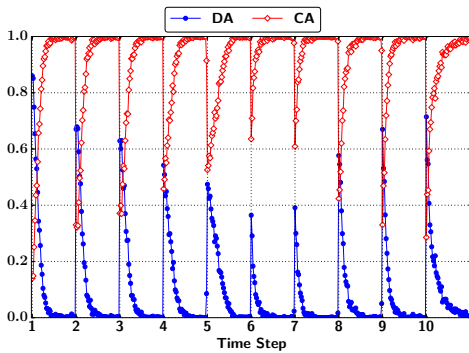

Fig. 6: Proportion of the second mating parent selected from the CA and DA respectively.

3) Effects of the Update Mechanisms: As discussed in Section III-B, the update mechanisms are used to maintain the complementary effects between the CA and the DA. The CA keeps a continuously strong selection pressure for the population convergence; while the DA maintains a set of well diversified solutions. In order to validate the importance of the three different components of DTAEA, we developed three DTAEA variants as follows:

- DTAEA-$v1$: this variant modifies DTAEA by removing the restricted mating selection mechanism proposed in Section III-C. In particular, now the mating parents are respectively selected from the CA and the DA without considering the population distribution.
- DTAEA-$v2$: this variant modifies DTAEA by removing the reconstruction mechanism proposed in Section III-A. In other words, it does not make any response to the changing environment.
- DTAEA-$v3$: this variant merely uses the update mechanisms to maintain the CA and the DA whereas it does not make any response to the changing environment. In addition, it does not use the restricted mating selection mechanism as DTAEA-$v1$.

We conduct the experiments on F1 to F6 according to the same experimental settings introduced in Section IV. Table III and Table IV of the supplementary file give the median and IQR values according to the MIGD and MHV metrics. From these results we can see that the original DTAEA, consisted of all three components, has shown clearly better performance than the other three variants. More specifically, as shown in Table III and Table IV of the supplementary file, the performance of DTAEA-$v3$ is the worst among all three variants. This observation is reasonable as DTAEA-$v3$ neither responds to the changing environment nor takes advantages of the complementary effect of the CA and the DA for offspring generation. The performance of DTAEA-$v2$ is slightly better than DTAEA-$v3$. Therefore, we can see that even without any response to the changing environment, the restricted mating selection mechanism can also provide some guidance to the search process. As for DTAEA-$v1$, we can see that the performance can be significantly improved by using the reconstruction mechanism proposed in Section III-A to respond to the changing environment. This superiority is more obvious when the frequency of change is high. By comparing the results between DTAEA-$v1$ and the original DTAEA, we can clearly see the importance of taking advantages of the complementary effect of the CA and the DA for offspring generation. All in all, we can conclude that all three components of DTAEA are of significant importance for handling the DMOP with a changing number of objectives.

4) Performance Comparisons on a Different Changing Sequence: In Section V-A and Section V-B of our paper, the experiments only consider the scenarios where the number of objectives increases or decreases by one at each time step. A natural question is: how is the performance of our proposed algorithm under the circumstance where the number of objectives changes in a different sequence? Without loss of generality, this further experiment considers the time varying number of objectives $m(t)$ as follows:

$$m(t) = \begin{cases} 3, & t = 1 \\ m(t-1) + 2, & t \in [2, 3] \\ m(t-1) - 2, & t \in [4, 5] \\ m(t-1) - 1, & t = 6 \end{cases} \tag{13}$$

where $t \in \{1, \cdots, 6\}$ is a discrete time. Here we also consider four different frequencies of change, i.e., τ_t is as 25, 50,

100 and 200, respectively. From the empirical results shown in Table V and Table VI of the supplementary file, we can clearly see that our proposed DTAEA is the best optimizer on almost all comparisons (92 out of 96 for MIGD and 88 out 96 for MHV). Similar to the observations from the previous sections, the performance of DTAEA might not be stable under a high frequency of change; while its performance becomes constantly competitive with the increase of τ_t.

VI. CONCLUSIONS

Different from the current studies on DMO, which mainly focus on problems with time-dependent objective functions, this paper considers the DMOP with a time varying number of objectives. This type of dynamics brings the expansion or contraction of the PF or PS manifold when increasing or decreasing the number of objectives. In this case, the existing dynamic handling techniques are not able to handle this scenario. To address this challenge, this paper has proposed a dynamic two-archive EA (denoted as DTAEA) for handling the DMOP with a changing number of objectives. More specifically, DTAEA simultaneously maintains two co-evolving populations, i.e., the CA and the DA, during the evolution process. In particular, they have complementary effects: the CA concerns more about the convergence while the DA concerns more about the diversity. The CA and the DA are separately reconstructed whenever the environment changes. In the meanwhile, they are maintained in different manners. By using a restricted mating selection mechanism, DTAEA takes advantages of the complementary effects of the CA and the DA to strike the balance between convergence and diversity all the time. Comprehensive experiments on a set of dynamic multi-objective benchmark problems fully demonstrate the superiority of DTAEA for handling the DMOP with a dynamically changing number of objectives. In particular, it is able to effectively and efficiently track the expansion or contraction of the PF or PS manifold whenever the environment changes. In the experiments, we also noticed that some state-of-the-art dynamic EMO algorithms even showed inferior performance than their stationary counterparts.

This paper is the first attempt towards a systematic investigation of the methods for handling the DMOP with a changing number of objectives. Much more attention and effort should be required on this topic. In future, it is interesting to investigate the dynamic environments where the changes are hard to detect and do not vary regularly. Constrained optimization problems, which are ubiquitous in the real-life application scenarios, have not been considered in this work. It is worth considering dynamic constraints in the dynamic environment as well and thus requires effective constraint handling techniques accordingly [54]. Furthermore, this work focuses on designing an effective environmental selection operator to adapt to the changing environment. It is also interesting to investigate the reproduction operation which can adapt the search behavior autonomously [69]–[74] according to the current landscape. In addition, our recently developed efficient non-domination level update method [75] can be considered to accelerate the process of identifying the current non-domination level structure. The benchmark problems used in this paper are developed from the classic DTLZ benchmark suite. It is interesting to develop more challenging benchmark problems with additional characteristics, e.g., from the WFG [76] and the CEC 2009 competition [77] benchmark suites. Last but not the least, new performance metrics are also needed to better facilitate the analysis and comparisons of dynamic EMO algorithms.
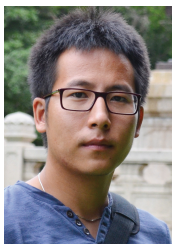
REFERENCES

[1] J. H. Holland, *Adaptation in Natural and Artificial Systems*. MIT Press, 1975.

[2] S. Yang and X. Yao, *Evolutionary Computation for Dynamic Optimization Problems*. Springer, 2013.

[3] K. Deb, U. B. R. N., and S. Karthik, "Dynamic multi-objective optimization and decision-making using modified NSGA-II: A case study on hydro-thermal power scheduling," in *EMO'07: Proc. of the 4th International Conference on Evolutionary Multi-Criterion Optimization*, 2007, pp. 803–817.

[4] B. Zheng, "A new dynamic multi-objective optimization evolutionary algorithm," in *ICNC'07: Proc. of the 2007 International Conference on Computing, Networking and Communications*, 2007, pp. 565–570.

[5] R. Liu, W. Zhang, L. Jiao, F. Liu, and J. Ma, "A sphere-dominance based preference immune-inspired algorithm for dynamic multi-objective optimization," in *GECCO'10: Proc. of the 2010 Genetic and Evolutionary Computation Conference*, 2010, pp. 423–430.

[6] V. Aragon, S. Esquivel, and C. A. C. Coello, "Evolutionary multiobjective optimization in non-stationary environments," *Journal of Computer Science and Technology*, vol. 5, no. 3, pp. 133–143, 2005.

[7] C. R. B. Azevedo and A. F. R. Araújo, "Generalized immigration schemes for dynamic evolutionary multiobjective optimization," in *CEC'11: Proc. of the 2011 IEEE Congress on Evolutionary Computation*, 2011, pp. 2033–2040.

[8] R. Azzouz, S. Bechikh, and L. B. Said, "A multiple reference point-based evolutionary algorithm for dynamic multi-objective optimization with undetectable changes," in *CEC'14: Proc. of the 2014 IEEE Congress on Evolutionary Computation*, 2014, pp. 3168–3175.

[9] K. Li, S. Kwong, J. Cao, M. Li, J. Zheng, and R. Shen, "Achieving balance between proximity and diversity in multi-objective evolutionary algorithm," *Inf. Sci.*, vol. 182, no. 1, pp. 220–242, 2012.

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

[11] Y. Wang and B. Li, "Investigation of memory-based multi-objective optimization evolutionary algorithm in dynamic environment," in *CEC'09: Proc. of the 2009 IEEE Congress on Evolutionary Computation*, 2009, pp. 630–637.

[12] Z. Peng, J. Zheng, and J. Zou, "A population diversity maintaining strategy based on dynamic environment evolutionary model for dynamic multiobjective optimization," in *CEC'14: Proc. of the 2014 IEEE Congress on Evolutionary Computation*, 2014, pp. 274–281.

[13] I. Hatzakis and D. Wallace, "Dynamic multi-objective optimization with evolutionary algorithms: a forward-looking approach," in *Genetic and Evolutionary Computation Conference, GECCO 2006, Proceedings, Seattle, Washington, USA, July 8-12, 2006*, 2006, pp. 1201–1208.

[14] ——, "Topology of anticipatory populations for evolutionary dynamic multi-objective optimization," in *AIAA/ISSMO'06: Proc. of the 11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2006, pp. 1–10.

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

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

[17] A. Zhou, Y. Jin, Q. Zhang, B. Sendhoff, and E. P. K. Tsang, "Prediction-based population re-initialization for evolutionary dynamic multi-objective optimization," in *EMO'07: Proc. of the 4th International Conference Evolutionary Multi-Criterion Optimization*, 2006, pp. 832–846.

[18] Z. Peng, J. Zheng, J. Zou, and M. Liu, "Novel prediction and memory strategies for dynamic multiobjective optimization," *Soft Comput.*, vol. 19, no. 9, pp. 2633–2653, 2015.

[19] Y. Wu, Y. Jin, and X. Liu, "A directed search strategy for evolutionary dynamic multiobjective optimization," *Soft Comput.*, vol. 19, no. 11, pp. 3221–3235, 2015.

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

[21] A. Muruganantham, K. C. Tan, and P. Vadakkepat, "Evolutionary dynamic multiobjective optimization via kalman filter prediction," *IEEE Trans. Cybernetics*, 2015, accepted for publication.

[22] P. Kruchten, *The Rational Unified Process: An Introduction, Second Edition*, 2nd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2000.

[23] X. Shen, L. L. Minku, R. Bahsoon, and X. Yao, "Dynamic software project scheduling through a proactive-rescheduling method," *IEEE Trans. Software Eng.*, vol. 42, no. 7, pp. 658–686, 2016.

[24] X. Shen and X. Yao, "Mathematical modeling and multi-objective evolutionary algorithms applied to dynamic flexible job shop scheduling problems," *Inf. Sci.*, vol. 298, pp. 198–224, 2015.

[25] J. Xiao, L. J. Osterweil, Q. Wang, and M. Li, "Dynamic resource scheduling in disruption-prone software development environments," in *FASE'10: Proc. of the 13th International Conference on Fundamental Approaches to Software Engineering*, 2010, pp. 107–122.

[26] J. Zhuo and C. Chakrabarti, "An efficient dynamic task scheduling algorithm for battery powered DVS systems," in *ASP-DAC'05: Proc. of the 2005 Conference on Asia South Pacific Design Automation*, 2005, pp. 846–849.

[27] R. Chen, P. R. Lewis, and X. Yao, "Temperature management for heterogeneous multi-core fpgas using adaptive evolutionary multi-objective approaches," in *ICES'14: Proc. of the 2014 IEEE International Conference on Evolvable Systems*, 2014, pp. 101–108.

[28] S. Sharifi, A. K. Coskun, and T. S. Rosing, "Hybrid dynamic energy and thermal management in heterogeneous embedded multiprocessor socs," in *ASP-DAC'10: Proc. of the 15th Asia South Pacific Design Automation Conference*, 2010, pp. 873–878.

[29] J. Luo and N. K. Jha, "Static and dynamic variable voltage scheduling algorithms for real-time heterogeneous distributed embedded systems," in *ASP-DAC'02: Proc. of the 7th Asia and South Pacific and the 15th International Conference on VLSI Design*, 2002, p. 719.

[30] H. M. Fard, R. Prodan, and T. Fahringer, "A truthful dynamic workflow scheduling mechanism for commercial multicloud environments," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1203–1212, 2013.

[31] Z. Zhan, X. F. Liu, Y. Gong, J. Zhang, H. S. Chung, and Y. Li, "Cloud computing resource scheduling and a survey of its evolutionary approaches," *ACM Comput. Surv.*, vol. 47, no. 4, p. 63, 2015.

[32] Z. Chen, K. Du, Z. Zhan, and J. Zhang, "Deadline constrained cloud computing resources scheduling for cost optimization based on dynamic objective genetic algorithm," in *CEC'15: Proc. of the 2015 IEEE Congress on Evolutionary Computation*, 2015, pp. 708–714.

[33] S. U. Guan, Q. Chen, and W. Mo, "Evolving dynamic multi-objective optimization problems with objective replacement," *Artif. Intell. Rev.*, vol. 23, no. 3, pp. 267–293, 2005.

[34] L. Huang, I. H. Suh, and A. Abraham, "Dynamic multi-objective optimization based on membrane computing for control of time-varying unstable plants," *Inf. Sci.*, vol. 181, no. 11, pp. 2370–2391, 2011.

[35] K. Praditwong and X. Yao, "A new multi-objective evolutionary optimisation algorithm: The two-archive algorithm," in *CIS'06: Proc. of the 2006 Computational Intelligence and Security*, 2006, pp. 95–104.

[36] H. Wang, L. Jiao, and X. Yao, "Two_arch2: An improved two-archive algorithm for many-objective optimization," *IEEE Trans. Evolutionary Computation*, vol. 19, no. 4, pp. 524–541, 2015.

[37] K. Li, S. Kwong, and K. Deb, "A dual-population paradigm for evolutionary multiobjective optimization," *Inf. Sci.*, vol. 309, pp. 50–72, 2015.

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

[39] O. Schütze, C. A. C. Coello, S. Mostaghim, E.-G. Talbi, and M. Dellnitz, "Hybridizing evolutionary strategies with continuation methods for solving multi-objective problems," *Journal of Engineering Optimization*, vol. 40, no. 5, pp. 383–402, 2008.

[40] H. Zhang, A. Zhou, S. Song, Q. Zhang, X. Gao, and J. Zhang, "A self-organizing multiobjective evolutionary algorithm," *IEEE Trans. Evolutionary Computation*, 2016.

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

[42] A. J. Nebro, J. J. Durillo, F. Luna, B. Dorronsoro, and E. Alba, "Mocell: A cellular genetic algorithm for multiobjective optimization," *Int. J. Intell. Syst.*, vol. 24, no. 7, pp. 726–746, 2009.

[43] X. Ma, F. Liu, Y. Qi, X. Wang, L. Li, L. Jiao, M. Yin, and M. Gong, "A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables," *IEEE Trans. Evolutionary Computation*, vol. 20, no. 2, pp. 275–298, 2016.

[44] M. Farina, K. Deb, and P. Amato, "Dynamic multiobjective optimization problems: test cases, approximations, and applications," *IEEE Trans. Evolutionary Computation*, vol. 8, no. 5, pp. 425–442, 2004.

[45] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization*, ser. Advanced Information and Knowledge Processing, A. Abraham, L. Jain, and R. Goldberg, Eds. Springer London, 2005, pp. 105–145.

[46] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II," *IEEE Trans. Evolutionary Computation*, vol. 13, no. 2, pp. 284–302, 2009.

[47] H. Ishibuchi, N. Akedo, and Y. Nojima, "Behavior of multiobjective evolutionary algorithms on many-objective knapsack problems," *IEEE Trans. Evolutionary Computation*, vol. 19, no. 2, pp. 264–283, 2015.

[48] B. Li, J. Li, K. Tang, and X. Yao, "Many-objective evolutionary algorithms: A survey," *ACM Comput. Surv.*, vol. 48, no. 1, p. 13, 2015.

[49] K. V. Price, "Differential evolution," in *Handbook of Optimization - From Classical to Modern Approach*, ser. Intelligent Systems Reference Library. Springer, 2013, vol. 38, pp. 187–214.

[50] K. Praditwong and X. Yao, "A new multi-objective evolutionary optimisation algorithm: The two-archive algorithm," in *CIS'06: Proc. of the 2006 International Conference Computational Intelligence and Security*, 2006, pp. 95–104.

[51] B. Li, J. Li, K. Tang, and X. Yao, "An improved two archive algorithm for many-objective optimization," in *CEC'14: Proc. of the 2014 IEEE Congress on Evolutionary Computation*, 2014, pp. 2869–2876.

[52] M. D. McKay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, pp. 239–245, 1979.

[53] K. Deb and M. Goyal, "A combined genetic adaptive search (GeneAS) for engineering design," *Computer Science and Informatics*, vol. 26, pp. 30–45, 1996.

[54] K. Li, K. Deb, Q. Zhang, and S. Kwong, "An evolutionary many-objective optimization algorithm based on dominance and decomposition," *IEEE Trans. Evolutionary Computation*, vol. 19, no. 5, pp. 694–716, 2015.

[55] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[56] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, 1994.

[57] H. Liu, F. Gu, and Q. Zhang, "Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems," *IEEE Trans. Evolutionary Computation*, vol. 18, no. 3, pp. 450–455, 2014.

[58] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints," *IEEE Trans. Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.

[59] Z. Wang, Q. Zhang, A. Zhou, M. Gong, and L. Jiao, "Adaptive replacement strategies for MOEA/D," *IEEE Trans. Cybernetics*, vol. 46, no. 2, pp. 474–486, 2016.

[60] K. Li, Q. Zhang, S. Kwong, M. Li, and R. Wang, "Stable matching-based selection in evolutionary multiobjective optimization," *IEEE Trans. Evolutionary Computation*, vol. 18, no. 6, pp. 909–923, 2014.

[61] K. Li, S. Kwong, Q. Zhang, and K. Deb, "Interrelationship-based selection for decomposition multiobjective optimization," *IEEE Trans. Cybernetics*, vol. 45, no. 10, pp. 2076–2088, 2015.

[62] M. Wu, K. Li, S. Kwong, Y. Zhou, and Q. Zhang, "Matching-based selection with incomplete lists for decomposition multi-objective optimization," *IEEE Trans. on Evolutionary Computation*, 2017, accepted for publication.

[63] S. Jiang and S. Yang, "Evolutionary dynamic multi-objective optimization: benchmarks and algorithm comparisons," *IEEE Trans. Cybernetics*, 2016, accepted for publication.

[64] P. A. N. Bosman and D. Thierens, "The balance between proximity and diversity in multiobjective evolutionary algorithms," *IEEE Trans. Evolutionary Computation*, vol. 7, no. 2, pp. 174–188, 2003.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TEVC.2017.2669638, IEEE Transactions on Evolutionary Computation

IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, VOL. XX, NO.X, XXXX   15

[65] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[66] V. R. Khare, X. Yao, and K. Deb, "Performance scaling of multi-objective evolutionary algorithms," in *EMO'03: Proc. of the 2nd International Conference on Evolutionary Multi-Criterion Optimization*, 2003, pp. 376–390.

[67] R. Shang, L. Jiao, Y. Ren, J. Wang, and Y. Li, "Immune clonal coevolutionary algorithm for dynamic multiobjective optimization," *Natural Computing*, vol. 13, no. 3, pp. 421–445, 2014.

[68] S. Jiang and S. Yang, "A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization," *IEEE Trans. Evolutionary Computation*, 2016, accepted for publication.

[69] K. Li, Á. Fialho, S. Kwong, and Q. Zhang, "Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evolutionary Computation*, vol. 18, no. 1, pp. 114–130, 2014.

[70] K. Li, Á. Fialho, and S. Kwong, "Multi-objective differential evolution with adaptive control of parameters and operators," in *LION 5: Proc. of the 5th International Conference on Learning and Intelligent Optimization*, 2011, pp. 473–487.

[71] K. Li, S. Kwong, R. Wang, J. Cao, and I. J. Rudas, "Multi-objective differential evolution with self-navigation," in *SMC'12: Proc. of the 2012 IEEE International Conference on Systems, Man, and Cybernetics*, 2012, pp. 508–513.

[72] K. Li, R. Wang, S. Kwong, and J. Cao, "Evolving extreme learning machine paradigm with adaptive operator selection and parameter control," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 21, pp. 143–154, 2013.

[73] K. Li and S. Kwong, "A general framework for evolutionary multiobjective optimization via manifold learning," *Neurocomputing*, vol. 146, pp. 65–74, 2014.

[74] K. Li, S. Kwong, R. Wang, K. Tang, and K. Man, "Learning paradigm based on jumping genes: A general framework for enhancing exploration in evolutionary multiobjective optimization," *Inf. Sci.*, vol. 226, pp. 1–22, 2013.

[75] K. Li, K. Deb, Q. Zhang, and Q. Zhang, "Efficient non-domination level update method for steady-state evolutionary multiobjective optimization," *IEEE Trans. Cybernetics*, 2016, accepted for publication.

[76] S. Huband, P. Hingston, L. Barone, and R. L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Trans. Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.

[77] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and 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*, vol. 264, 2008.

Xin Yao (M91-SM96-F03) is a Chair Professor of Computer Science at the Southern University of Science and Technology, Shenzhen, China, and a Professor of Computer Science at the University of Birmingham, UK. He is an IEEE Fellow, and a Distinguished Lecturer of IEEE Computational Intelligence Society (CIS). His major research interests include evolutionary computation, ensemble learning, and their applications in software engineering. He has been working on multi-objective optimisation since 2003, when he published a well-cited EMO'03 paper on many objective optimisation. His research won the 2001 IEEE Donald G. Fink Prize Paper Award, 2010, 2016, and 2017 IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION Outstanding Paper Awards, 2010 BT Gordon Radley Award for Best Author of Innovation (Finalist), 2011 IEEE TRANSACTIONS ON NEURAL NETWORKS Outstanding Paper Award, and many other best paper awards. He received the prestigious Royal Society Wolfson Research Merit Award in 2012 and the IEEE CIS Evolutionary Computation Pioneer Award in 2013. He was the the President (2014-15) of IEEE CIS, and the Editor-in-Chief (2003-08) of IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION.
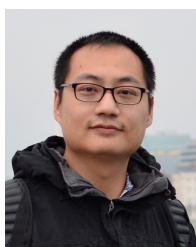
Ke Li (SM'10-M'17) received the B.Sc. and M.Sc. in computer science and technology from Xiangtan University, China, in 2007 and 2010, respectively, and the PhD in computer science from City University of Hong Kong, Hong Kong, in 2014.

He was a postdoctoral research associate at Michigan State University and a research fellow at University of Birmingham. Now he is a Lecturer (Assistant Professor) in Data Analytics at University of Exeter. His current research interests include the evolutionary multi-objective optimization, large scale optimization, statistical machine learning and applications in software engineering and industrial design.

He has served as the regular reviewer and has published several research papers in renowned journals such as IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and IEEE TRANSACTIONS ON CYBERNETICS.

Renzhi Chen received his bachelor's degree in computer science from National University of Defence Technology, Hunan, China in 2012. He is currently pursuing the Ph.D. degree in computer science at University of Birmingham, UK.

His current research interest is evolutionary algorithms for solving dynamic multi-objective optimization problems.