Improved NSGA-III using transfer learning and centroid distance for dynamic multi-objective optimization

Haijuan Zhang¹ · Gai-Ge Wang¹

Received: 28 April 2021 / Accepted: 19 October 2021 / Published online: 18 November 2021
© The Author(s) 2021

Abstract

Multi-objective problems in real world are often contradictory and even change over time. As we know, how to find the changing Pareto front quickly and accurately is challenging during the process of solving dynamic multi-objective optimization problems (DMOPs). In addition, most solutions obey different distributions in decision space and the performance of NSGA-III when dealing with DMOPs should be further improved. In this paper, centroid distance is proposed and combined into NSGA-III with transfer learning together for DMOPs, called TC_NSGAIII. Centroid distance-based strategy is regarded as a prediction method to prevent some inappropriate individuals through measuring the distance of the population centroid and reference points. After the distance strategy, transfer learning is used for generating an initial population using the past experience. To verify the effectiveness of our proposed algorithm, NSGAIII, Tr_NSGAIII (NSGA-III combining with transfer learning only), Ce_NSGAIII (NSGA-III combining with centroid distance only), and TC_NSGAIII are compared. Seven state-of-the-art algorithms have been used for comparison on CEC 2015 benchmarks. Besides, transfer learning and centroid distance are regarded as a dynamic strategy, which is incorporated into three static algorithms, and the performance improvement is measured. What's more, twelve benchmark functions from CEC 2015 and eight sets of parameters in each function are used in our experiments. The experimental results show that the performance of algorithms can be greatly improved through the proposed approach.

Keywords Dynamic optimization · Transfer learning · Multi-objective · NSGA-III · Evolutionary algorithms

Introduction

In recent years, dynamic optimization has caught the rising attention of researchers. Compared with static multi-objective optimization, dynamic multi-objective problems (DMOPs) are much closer to real-world application, like 5G ultra-dense networks [1], water quality anomaly detection [2], multi-crane-scheduling problem [3], and cycling energy systems [4]. However, it is worth mentioning that DMOPs contain dynamic characteristics, including changing Pareto front, Pareto set, the number of objectives or other parameters. Therefore, tracking the changing Pareto front quickly and accurately becomes particularly essential for DMOPs.

Many researchers studied the field of dynamic optimization and various methods about solving DMOPs have been put forward. At the same time, new benchmarks, new problems, and new algorithms [5] are gradually emerging. How to design an effective algorithm for coping with DMOPs remains a great challenge for us.

Based on different dynamic characteristics, DMOPs can be divided into four categories [6]: (1) both Pareto front and Pareto set change, (2) Pareto front change but Pareto set doesn't change, (3) Pareto set change but Pareto front doesn't change, and (4) both Pareto front and Pareto set don't change. An algorithm with superior performance should respond to these different changes timely. Also, convergence and diversity should be considered when measuring the performance of dynamic algorithms. In general, the dynamic nature of the environment may lead to moving Pareto front, and the right direction of next environmental change is unknown for us. This undoubtedly brings great challenges and difficulties to the optimization algorithms. Despite this, there are some unknown certain relationships between the current environ-

✉ Gai-Ge Wang
gaigewang@gmail.com

Haijuan Zhang
zhanghaijuan@stu.ouc.edu.cn

¹ Department of Computer Science and Technology, Ocean University of China, Qingdao 266100, China

ment and the next change. Therefore, the past experience can be used as a powerful tool to predict the next change. Many researchers make full use of the information from the past environment and combine it with some static multi-objective algorithms (SMOAs). Of course, DMOPs can be solved through a single SMOA without any strategies as well. But obviously, after one iteration of a single SMOA, the initial population should be reset according to the basic idea. For multi-objective problems with dynamic characteristics, the reconfigured initial population is likely to evolve in the wrong direction, thus failing to obtain the optimal solution to the problem. To lower the probability of evolution in the wrong direction, some timely strategies to cope with change are essential.

These strategies for coping with change come from different perspectives, include predicting the next change direction, reusing the past environmental experience, or some self-adaptive methods. For the prediction method, some specific strategies are employed to predict the next environmental change, so that the population can be evolved towards the right direction, greatly reducing the possibility of reverse evolution. A novel knee point-based prediction approach was proposed by Zou et al. [7]. It maintained the non-dominated solutions around the knee regions and the burden of maintaining diversity and convergence at the same time was reduced for the most part. Memory-based schemes aim at storing the past experience about the non-dominated solutions, and recalling it when the next change occurs. A steady-state evolutionary algorithm for DMOPs was proposed by Jiang et al. [8]. The information from the previous environments was collected, and a part of outdated solutions were reused when facing the new environment.

What's more, a combination of different types of techniques had been proposed. On the basis of prediction methods, the integration of evolutionary algorithms [9, 10] and machine learning becomes possible. Simões et al. [11] incorporated two prediction approaches into memory-based evolutionary algorithms. One used linear regression, and another combined nonlinear regression. The performance of algorithms can be greatly improved through these two methods. Jiang et al. [12] proposed a distribution algorithm based on domain adaptation and nonparametric estimation. Transfer learning, as a branch of machine learning, is considered to be promising for solving DMOPs. The past experience can be used for obtaining the high-quality individuals and the next initial population. Wang et al. [13] presented a novel effective model, which is called regression transfer learning prediction method. Some individuals with better predicted objective values are selected to form the initial population. Jiang et al. [14] successfully applied transfer learning to multi-task optimization and dynamic optimization. For DMOPs, transfer component analysis (TCA) [14] was adopted to construct the prediction model. Besides, various strategies to improve

the performance of the algorithms were combined with transfer learning, including the knee point based [15], individual based [16], manifold based [17], or autoencoding evolutionary search [18]. It is visible that much great progress has been made in the application of transfer learning.

Transfer learning applies the knowledge from other fields or problems to a different but related domain. However, the training time is also one of the criteria to measure the quality of algorithms. Most of the existing methods require a long training time, which suggests that the performance of transfer learning approaches should be further improved. Accordingly, we propose an improved transfer learning method in this paper, which is the centroid distance strategy combined with transfer learning. The centroid distance strategy is employed before generating the initial population through the transfer learning. This strategy define some reference points according to different problems like NSGA-III [19], and then the distance between the population centroid and reference points is calculated as a basis of selecting individuals. Those with smaller distance will be selected for composing the population. In the next step, the population will be put into transfer learning method and the initial population for evolutionary iteration can be generated. The strategy we employed replace the random population with the smallest centroid distance population. This further improves the efficiency of looking for the optimal solutions and can locate the optimal solutions more quickly. Obviously, the training time can be saved a lot.

The approach proposed by this paper aims to improve the performance of NSGA-III in DMOPs. The knowledge about machine learning is adopted for predicting the right direction. The direction of the next change can be predicted according to the environmental information in the last time. The motivation of this paper is mainly divided into two aspects, one is to improve the performance of NSGA-III algorithm in solving DMOPs, and the other is to map these individuals that obey different distributions into a latent space and generate an initial population for evolving.

The contributions of this paper are as follows: first, the integration of centroid distance and transfer learning is proposed. Besides, the centroid distance strategy and transfer learning both are combined with NSGA-III to verify the effectiveness, respectively. Second, the combination we propose is considered as a whole dynamic approach. Three evolutionary algorithms are incorporated with this approach and the percentage improvement of metrics is presented through experiments. Compared with the original algorithm, the algorithms incorporating the strategy have better diversity and convergence.

The rest of this paper is organized as follows: we introduce the preliminary studies and related work in Sect. [The preliminary studies and related work](#). The concrete process of centroid distance and transfer learning is introduced in

Sect. **Proposed methods**. Section **Experiments** shows some details of the experiments and results. At last, in Sect. **Conclusions**, the summary of this work will be drawn and the future direction will be given clearly.

The preliminary studies and related work

In this section, some established definitions about DMOPs are described. We next will introduce the basic knowledge of transfer learning. What's more, the related work about our proposed approach is discussed at the end of this section.

Definitions of DMOPs

A formal DMOP can be expressed mathematically as:

$$\begin{cases} \text{Minimize } F(\mathbf{x}, t) = (f_1(x, t), f_2(x, t), \dots, f_M(x, t))^T \\ \text{s.t. } \mathbf{x} = (x_1, x_2, x_3, \dots, x_n) \in \Omega \end{cases} \quad (1)$$

where $F(\mathbf{x}, t)$ is the objective function, consisting of M objectives. $\mathbf{x}$ means the n -dimensional decision vector and Ω represents the decision space. The mapping of function variables is as follows: $f_i(x, t): \Omega^n \times T \rightarrow R^M$, $\Omega = [L_1, U_1] \times [L_2, U_2] \times \dots \times [L_n, U_n]$, $i = 1, 2, 3, \dots, M$. L_i and U_i are the lower and upper boundaries of the i -th decision variable. Each function changes over time t . Here, a minimization problem is set in this paper, which is considered to be continuous for further research.

Definition 1 (Dynamic Pareto dominance): at time t , a decision variable x_1 dynamically Pareto dominates another decision variable x_2 , denoted by $x_1 \succ x_2$, if and only if

$$\begin{cases} \forall i = 1, \dots, M, f_i(x_1, t) \leq f_i(x_2, t) \\ \exists i = 1, \dots, M, f_i(x_1, t) < f_i(x_2, t) \end{cases} \quad (2)$$

Definition 2 (Dynamic Pareto-optimal Set (DPS)): at time t , x and x^* are decision variables from decision space Ω . If and only if there is no other decision variable $x \in \Omega$ satisfying $x \succ x^*$, the vector x^* can be called as a non-dominated or Pareto-optimal solution. The Dynamic Pareto-optimal Set (DPS) is the set of all Pareto-optimal solutions at time t , which can be expressed as:

$$DPS(t) = \{x^* \in \Omega | \nexists x \in \Omega, x \succ x^*\} \quad (3)$$

Definition 3 (Dynamic Pareto-optimal Front (DPF)): at time t , the set of all corresponding objective vectors of the DPS is defined as the Dynamic Pareto-optimal Front (DPF):

$$DPF(t) = \{F(x^*, t) | x^* \in DPS\} \quad (4)$$

As can be seen from above definitions, DMOPs and static multi-objective problems are quite different. Dynamic features bring a lot of unknown changes to multi-objective problems. A higher standard for dynamic multi-objective evolutionary algorithms (DMOEAs) is put forward, which requires tracking the moving DPS and DPF. At the same time, the Pareto-optimal solutions to be found should be as diverse as possible.

Transfer learning

With the emergence of more and more machine learning application scenarios, transfer learning, as a branch of machine learning, has received more and more attention. In the field of machine learning, transfer learning is mainly to apply knowledge or patterns learned in a certain field or task to different but related fields or problems. Traditional machine learning is often based on the same distribution assumption and requiring a large amount of labeled data. Transfer learning learns from previously labeled data and applies it to unlabeled data to obtain results. Compared with traditional machine learning, transfer learning can not only meet the needs of universal models and individualization through adaptive learning, but also complete the requirements of specific applications through similar knowledge transfer. Besides, the process of transfer learning is not always successful, and negative transfer also occurs. Pan and Yang [20] defined negative transfer as the knowledge learned on the source domain, which has a negative effect on the learning. In practical applications, finding reasonable similarities and choosing or developing a reasonable transfer learning method can avoid the negative transfer.

Transfer learning can also be applied to DMOPs. When the environmental change occurs, transfer learning can be used as a prediction tool to apply the knowledge learned in the source domain to the target domain. The source domain refers to the past experience we get, and the target domain means the unknown changes we may meet. The core idea of transfer learning is to find the similarity between the source domain and the target domain, and use certain measurement criteria to measure the similarity. Through the similarity given by the measurement criteria, we can use certain methods to increase the similarity between the source domain and the target domain, thereby completing transfer learning more successfully. There are various measurement criteria, mainly to measure the difference between two data domains. The maximum mean difference (MMD) is the most frequently used metric in transfer learning.

Definition 4 (Maximum Mean Difference (MMD)): MMD measures the distance between two distributions in

the regenerated Hilbert space, and it is a kernel learning method.

$$\text{MMD}(X, Y) = \left\| \frac{1}{n_1} \sum_{i=1}^{n_1} \phi(\vec{x}_i) - \frac{1}{n_2} \sum_{j=1}^{n_2} \phi(\vec{y}_j) \right\|_{\mathcal{H}}^2 \quad (5)$$

where $\vec{x}_i$ and $\vec{y}_j$ are two variables from source domain X and target domain Y , respectively. n_1 and n_2 are the number of X and Y , respectively. $\phi(\bullet)$ maps the original variable to the Reproducing Kernel Hilbert Space (RKHS). The direct understanding of MMD is to obtain the distance of the mean value of the data of two domains in RKHS. The inner product in RKHS space can be converted into a kernel function, so the final MMD can be directly calculated by the kernel function. This will be discussed in Sect. [Proposed methods](#).

Related works

To date, a great progress has been made in the field of DMOPs. Many DMOEAs and different strategies have emerged from various perspectives. The most DMOEAs can be divided into the following categories [21]: diversity-based approaches, prediction-based approaches, memory-based approaches, and parallel approaches.

The algorithms based on diversity aim at maintaining a good level of diversity during the search process. A modified NSGA-II for DMOPs was proposed by Deb et al. in 2006 [22]. Two versions of DNSGA-II were proposed. The first version of DNSGA-II (DNSGA-II-A) introduced a certain number of random individuals into the population to increase the diversity. Replacing some individuals with mutated solutions was employed for maintaining the diversity in the second version of NSGA-II (DNSGA-II-B). Grefenstette [23] studied the genetic algorithm on dealing with the dynamic environments. To respond to the environmental changes, a random immigrant genetic algorithm (RIGA) was put forward. In RIGA, some new genetic materials were put into the population and other individuals would be replaced randomly. Chen et al. [24] studied the use of additional objective in dynamic environments. The individual diversity was considered as an additional objective in the evolutionary process. A new diversity maintaining evaluation approach and a new diversity measurement were proposed and a useful selection pressure was added into the algorithm.

The core idea of prediction-based methods is easy to follow. Prediction-based approaches can be a complete model or framework, or it can be an effective strategy responding to the unknown changes. In fact, the previous information obtained from the past environment is employed for generating the new population to form a prediction model. When the environment changes, the generated population will be employed for predicting the moving DPF and DPS. A novel prediction-

based reaction mechanism for DMOPs was proposed by Zhang et al. [25]. During the process, three subpopulations were generated according to different approaches, which were a linear prediction model, a sampling strategy, and a shrinking method. The whole population consists of these three subpopulations and was considered to be the initial population that responds to the next change. A multidirectional prediction approach and a multi-model prediction method were introduced by Gong et al. [26, 27] for DMOPs. In the multidirectional prediction, the moving DPF was predicted through the representative individuals generated by the clustering of optimized solutions, and finally the predicted solutions were created. According to the corresponding relations between the type of change and the response strategy, the prediction model corresponding to the type of problem change is selected to generate a predicted solution. Both prediction methods have proved to be superior to other algorithms in performance. In [6], Kalman filter was employed for predicting future changes. The Kalman filter (KF) prediction model made an estimation through feedback control. To avoid other problems caused by the system, a scoring scheme was introduced to obtain a proportion of KF and random re-initialization. The whole prediction process was combined with the structure of Multi-objective Evolutionary Algorithm with Decomposition Based on Differential Evolution (MOEA/D-DE).

When the environment changes slightly comparing with the past environment, storing the useful knowledge from past environments timely is of particular importance. An extra memory is employed to store information about non-dominated solutions, which can be recalled for dealing with new changes. Wang et al. [28] introduced some memory-based strategies, including explicit memory scheme, local search scheme, and hybrid memory scheme. The explicit memory scheme means to store the past solutions and reevaluate them, forming the next population by these solutions. The disadvantages of this mechanism are obvious. The loss of diversity and the vulnerability to local optimality are two major disadvantages. The local search scheme introduced a probability parameter which controls the proportion of local search and a local search operator. Obviously, the performance of local search is greatly affected by parameters, and different parameters may lead to completely opposite performance. The hybrid memory scheme combined explicit memory scheme and local search scheme; the number of memory solutions was controlled by a probability parameter.

The idea of “parallel” is also an important branch for solving DMOPs. Most parallel evolutionary algorithms divided a population into several subpopulations that evolve simultaneously with different processors. As in [29], $M + 1$ multiple subpopulations were divided from a whole population, where M is the number of objectives and a cellular genetic algorithm

was employed for optimizing every subpopulation about each objective. The remaining last subpopulation was used for obtaining the best average of n objectives. Besides, aiming at the dynamic environmental changes, the hyper-mutation operator was utilized for DMOPs. A parallel evolutionary algorithm was proposed in [30], which applied the data decomposition into the algorithm. This parallel procedure was based on an island model and a master process. The master process divided the evolved population into some subpopulations and sent to every worker process. A multi-objective evolutionary algorithm was chosen to obtain the optimal solutions for each subpopulation. Then the found optimal solutions would be sent to the master and all the optimal solutions would form a new population by the master when the change occurs.

Dynamic multi-objective problems also have many practical applications. In addition to the above methods, many algorithms are applied to practical DMOPs, such as scheduling problems [31–33], routing problems [34, 35], and control problems [36]. In [31], the ant colony algorithm (ACO) was applied to the problem of railway junction rescheduling. The dynamic multi-objective railway junction rescheduling problem (DM-RJRP) was formulated in detail and the dynamic modification for a population-based ACO (DM-PACO) which adds an inbuilt memory was proposed. Through the simulated experiments, the performance of proposed algorithm can be verified on dealing with DM-RJRP. The other problems follow the similar process. There are many other practical application scenarios for DMOPs, including the resource management problems and mechanical problems. It can be seen that DMOPs are much closer to the real world, which requires more effective algorithms to deal with them.

In recent years, several researches combined with the machine learning techniques had been put forward in the area of computational intelligence. A novel approach based on transfer learning and genetic programming (GP) was proposed by Iqbal et al. [37] for image classification. The knowledge reconstruction ability of GP was employed to a large extent for obtaining useful information from two images in each class. The extracted knowledge through learning simple image classification problems would be reused to solve complex problems. A novel genetic transfer learning approach for bootstrap problems in evolutionary robotics was presented in [38], which was named as Family Bootstrapping (FB). FB divided the whole evolutionary process into two stages. In the first stage, the initial population was generated with the help of family ancestors which were the solutions evolved for a common source task. A task was selected from a set of related tasks. The evolutionary process started for employing evolution on the selected task in the second stage.

Although a lot of research has been carried out in the dynamic area, there are still problems to be considered. In the prediction-based approach, the data we employed in the

experiments and the data to be predicted may not obey the same distribution. Some researches assume that the solutions predicted by the model and the individuals used for constructing the prediction model meet the condition of independent identical distribution (IID). However, from the point of real world, taking this IID assumption as a precondition does not correspond to the reality. Therefore, it is reasonable to believe that this assumption may cause the failure of some algorithms in solving DMOPs. According to the contents above, a centroid distance and transfer learning-based approach are proposed in this paper. Comparing with other existing dynamic algorithms, the IID assumption is not a precondition in this approach anymore, which is close to the dynamic problems in reality. As one of the efficient many-objective algorithms, NSGA-III can also be used to solve DMOPs, but in practical application, the algorithm performance has yet to be improved. So, the performance of the proposed method will be mainly reflected by the performance improvement of NSGA-III in this paper.

Proposed methods

In this section, the whole process of proposed TC_NSGAIII will be presented in detail. First, the centroid distance method is introduced to get a population closer to reference points. Second, transfer learning is adopted for obtaining a new initial population, and then the obtained population will be put into NSGA-III to get DPFs.

The centroid distance method

This approach is based on the centroid points of individuals. At the beginning of evolution, the initial population was made up of randomly generated individuals. For these randomly generated individuals, some processing on them should be conducted to make them closer to the true DPF. According to the changing characteristics of dynamic problems, it is reasonable to believe that although the environmental changes are different, they are related. The main idea of centroid distance method is to select some populations as the initial population of next evolution by calculating the distance between the centroid of random populations and reference points.

First of all, within the given range of decision variables, a corresponding population is randomly initialized. When an environmental change occurs, the objectives also change to some extent. For the changed objective functions, several reference points are defined corresponding to the objective functions. Then the centroid of the random population will be generated using the following formula:

$$C_T = \frac{1}{|P_T|} \sum_{x \in P_T} x \quad (6)$$

where $|P_T|$ is the cardinality of P_T , and x is a solution of decision space in P_T . After getting the centroid of the random population, the distribution of population can be estimated roughly. Then the Euclidean distance between the centroid point and the reference point can be calculated by Eq. (7).

$$\text{distance} = \sqrt{\sum (C_T - Z_{\min})^2} \quad (7)$$

where $Z_{\min}$ are the reference points defined by the changed objectives. The definition method is based on the reference points setting in NSGA-III. The minimum value of each changing objective is defined as a reference point and here $Z_{\min}$ is a set of reference points. The reason for defining reference points is that to measure the distance between the current population and the ideal points. We will set a distance threshold σ . When the distance is less than σ , that is, the centroid and the reference points are close to a certain degree, this random population can be used as the input population for next transfer learning. Otherwise, the population will be reinitialized until the distance is less than the threshold. In the evolution process, this method is used before the transfer learning strategy. The purpose of this method is to select the population closer to the DPF and put it into the transfer learning when the environment changes, which can improve the efficiency of transfer learning to solve DMOPs. The specific algorithm pseudo code is described in detail in Algorithm 1. Here a simple example is given to illustrate the concrete process of the centroid distance method. For example, when the number of objectives is 2, a random population **RP** containing 200 individuals is firstly obtained, which means **RP** is a 200×2 matrix. Then a set of reference points $Z_{\min} = \{z_1, z_2\}$, where z_1 and z_2 represent the ideal optimal values of two objectives, respectively. When the number of iterations does not reach the maximum, the distance between the population centroid C_T and $Z_{\min}$ will be calculated in a circular way. Suppose the obtained centroid $C_T = (c_1, c_2)$, where c_1 and c_2 mean the center in two objective directions, respectively. For random population **RP**, the distance can be calculated as:

$$\text{distance} = \sqrt{(c_1 - z_1)^2 + (c_2 - z_2)^2}, \quad (8)$$

when the distance is less than our predefined threshold, the random population will be retained to the next stage as the

Fig. 1 The parameter sensitivity analysis

initial population. Otherwise, the random population will be re-initialized.

To explore the influence of distance threshold σ on the generated initial population, a parameter sensitivity analysis is conducted. The different values of σ are set and different populations are retained. In this analysis, we set 50 different values for σ which are evenly distributed across the interval $[0, 10]$. What's more, every individual in the population has ten decision variables and the population size is set as 200. For each generation of the population, the average of each decision variable for the 200 individuals is calculated. Therefore, the tendency of decision variables changing with parameter σ can be obtained as Fig. 1. For easier observation and analysis, five decision variables are randomly selected from 10 decision variables. It can be seen that the overall trend of decision variables is greatly affected by the parameter σ . However, the affected range is small, and the values of the decision variables fluctuate from 0.45 to 0.57. In general, the individuals are not affected much, but the retained population has been filtered to a certain extent, which makes the initial population for evolution closer to the true DPF at the current time.

Algorithm 1 The centroid distance method

Input: random population, RP ;
the reference points, Z_{min} ;
the centroid of population, C_T ;
the distance threshold, σ ;

Output: the input population of transfer learning, P_{IN} ;

- 1: Randomly re-initialize the population RP within the limit;
- 2: Define a set of reference points Z_{min} ;
- 3: **while** the stop condition is not met **do**
- 4: Randomly re-initialize the population RP ;
- 5: Obtain the centroid of population C_T by the equation (6) ;
- 6: Caculate the Euclidean distance between C_T and Z_{min} ;
- 7: **if** the value of distance $\leq \sigma$ **then**
- 8: $P_{IN} = RP$;
- 9: **break**;
- 10: **end if**
- 11: **end while**
- 12: **return** P_{IN} .

Transfer learning method

In transfer learning method, the transfer component analysis (TCA) [14] is mainly used for obtaining the initial population. Firstly proposed by Pan et al. [39], TCA is mainly applied to source and target domains that obey different distributions, and it assumes that there is a feature map φ making the mapped data obey similar distributions. The key is how to find the most suitable mapping to make the distribution the most similar. The TCA uses the MMD definition we introduced in Sect. The preliminary studies and related work. According to Eq. (5), MMD is the difference between the mean values of the source domain and the target domain after mapping. In the process of calculating MMD, TCA introduces kernel function and kernel matrix to find this distance. The kernel matrix used by TCA is:

$$\tilde{\mathbf{K}} = \begin{bmatrix} K_{S,S} & K_{S,T} \\ K_{T,S} & K_{T,T} \end{bmatrix} \quad (9)$$

where $K_{S,T}$ is a kernel function and the value of $K_{S,T}$ is $\varphi(S_i)^T \varphi(T_j)$. φ represents a mapping function of feature. The meaning of $K_{S,S}$, $K_{T,S}$, and $K_{T,T}$ is similar to $K_{S,T}$. What's more, an MMD matrix $\mathbf{L}$ is also employed in TCA. Each element in the matrix $\mathbf{L}$ is evaluated as follows:

$$L_{ij} = \begin{cases} \frac{1}{n_1^2} & x_i, x_j \in D_S \\ \frac{1}{n_2^2} & x_i, x_j \in D_T \\ -\frac{1}{n_1 n_2} & \text{otherwise} \end{cases} \quad (10)$$

Through a series of computational transformations, Eq. (5) can be reduced to:

$$MMD(X, Y) = tr(\tilde{\mathbf{K}}\mathbf{L}) - \lambda tr(\mathbf{K}), \quad (11)$$

where $\mathbf{K}$ is a symmetric matrix and $\mathbf{K}^T = \mathbf{K}$. $tr(\mathbf{K})$ means the trace of matrix $\mathbf{K}$, which calculates the sum of the diagonal elements of a matrix. To simplify the calculation, Pan et al. [39] used the method of dimensionality reduction to construct the results. A lower dimensional matrix $\mathbf{W}$ is used in TCA.

$$\tilde{\mathbf{K}} = (\mathbf{K}\mathbf{K}^{-\frac{1}{2}}\mathbf{W})(\mathbf{W}^T\mathbf{K}^{-\frac{1}{2}}\mathbf{K}) = \mathbf{K}\mathbf{W}\mathbf{W}^T\mathbf{K}. \quad (12)$$

According to the properties of the matrix trace, the final optimization goal of TCA is

$$\begin{aligned} \min_{\mathbf{W}} & tr(\mathbf{W}^T\mathbf{K}\mathbf{L}\mathbf{K}\mathbf{W}) + \mu tr(\mathbf{W}^T\mathbf{W}) \\ s.t. & \mathbf{W}^T\mathbf{K}\mathbf{H}\mathbf{K}\mathbf{W} = \mathbf{I}_{n_1+n_2}. \end{aligned} \quad (13)$$

The following condition in Eq. (13) is to allow the source and target domains to maintain their respective data characteristics. And $\mathbf{H}$ is a center matrix, whose value is $\mathbf{I}_{n_1+n_2} - [1/(n_1 + n_2)]\mathbf{1}\mathbf{1}^T$. $\mathbf{I}_{n_1+n_2}$ is a $(n_1 + n_2) \times (n_1 + n_2)$ identity matrix. $\mathbf{1}$ means a $(n_1 + n_2) \times 1$ matrix with all ones. Pan concluded by seeking the Lagrange duality: the solution to $\mathbf{W}$ is the leading m eigenvalues of $(\mathbf{K}\mathbf{L}\mathbf{K} + \mu\mathbf{I})^{-1}\mathbf{K}\mathbf{H}\mathbf{K}$.

In summary, the general steps of the TCA method are as follows: Firstly, input the data matrix of the source and target domains. Then select some kernel functions to map and obtain the kernel matrix $\tilde{\mathbf{K}}$. The matrix $\mathbf{H}$ and $\mathbf{L}$ should be calculated according to Eqs. (10) and (13). At last, the lower dimensional matrix $\mathbf{W}$ can be generated through calculating the leading m eigenvalues. The process of TCA can be described in Algorithm 2.

Algorithm 2 Transfer Component Analysis (TCA)

Input: the data of source domain, $\mathbf{S}$;
the data of target domain, $\mathbf{T}$;
a kernel function, k ;

Output: the lower dimensional matrix, $\mathbf{W}$;

- 1: Use selected kernel function k to map and obtain the kernel matrix $\tilde{\mathbf{K}}$;
- 2: Calculate the matrix $\mathbf{H}$, and $\mathbf{L}$ according to equations (9) and (12);
- 3: Construct the matrix $\mathbf{W}$ according to the leading m eigenvalues of $(\mathbf{K}\mathbf{L}\mathbf{K} + \mu\mathbf{I})^{-1}\mathbf{K}\mathbf{H}\mathbf{K}$;
- 4: **return** $\mathbf{W}$.

TC_NSGAIII

The overall flow of TC_NSGAIII algorithm is described in this section. Combined with the methods in the previous two sections, we choose NSGA-III as the evolutionary algorithm. The main reason is that few studies focused on improving the performance of NSGA-III in addressing DMOPs. Adding environmental change strategies to the static NSGA-III can improve its ability to solve dynamic problems. The approach presented in this article is a good strategy for dealing with changes. The following describes the overall flow of the algorithm combined with NSGA-III.

At the beginning of the TC_NSGAIII, the population is randomly initialized within the defined range. Then the centroid distance method is used for generating the input population of TCA. After that, according to the objective function before and after the change, two groups of solutions are randomly selected from the input population. The objective values of the two sets of solutions before and after the change are calculated and preserved as the source and target domain of TCA. The matrix $\mathbf{W}$ can be obtained from TCA and the mapping to the latent space can be obtained. For each particle mapped to the latent space, we look for the particle closest to it in the target domain, and use it as an individual in the initial population. The key steps of TC_NSGAIII can be represented by Fig. 2. The Fig. 2 describes the whole mapping process of TC_NSGAIII at time t . The centroid distance method is shown in Fig. 2a. By comparing the distance between the population centroid and the reference points, the population with the minimum distance is retained. The decision space can be mapped into the objective space at time t as depicted in Fig. 2b. Please note that we can get a matrix $\mathbf{W}$ when the input of TCA are the solutions from time t and $t + 1$. The latent space LS can be constructed using the matrix $\mathbf{W}$

Fig. 2 The key steps of TC_NSGAIII

and the mapping function $\varphi(\bullet)$ also can be obtained. Then the mapping particle j in the latent space can be shown clearly in Fig. 2. (d). The final goal is to find a decision variable x in time $t + 1$ such that $\varphi(F_{t+1}(x))$ is closet to j , which means the decision variable x should be an individual of predicted population.

Through above explanations about TC_NSGAIII, the whole process can be clarified clearly. Then the computing complexity analysis will be given. For centroid distance method, the major time is spent on the calculation of Euclidean distance. Since the population centroid and distance should be obtained in each cycle, it takes $O(N \times c)$ time where N and c represent the population size and the number of cycles, respectively. At the stage of transfer learning, the eigenvalue decomposition operation is the most time-consuming, whose time cost is $O(d(m_1 + m_2)^2)$. Here d means the number of nonzero eigenvectors. m_1 and m_2 show the number of solutions selected from source domain and target domain for constructing the latent space. According to [16], the overall complexity of one generation of NSGA-III is $O(N^2M)$, where M is the dimension of objective vectors. Therefore, the whole complexity of TC_NSGAIII can be expressed as $O(Nc + d(m_1 + m_2)^2 + 2N^2M)$.

observed through the comparisons with original algorithms in the second part. The whole experiments are based on MATLAB R2018a, and the running environment is Intel(R) Core (TM) i3-8100 CPU @ 3.60 GHz.

Experimental settings

The proposed strategy is mainly used in the initialization phase of the algorithm. Through a series of selection and mapping methods, an initial population that is more consistent with the current environment can be obtained, and then the optimal solutions after the change can be obtained through evolution. Theoretically, the improved initial population through centroid distance and transfer learning is more adaptable to the changing environment, and then the solutions closer to the real DPF can be obtained. To verify the effectiveness of the proposed method, 12 test functions and eight different sets of dynamic parameters are employed, and three related metrics to measure the convergence and diversity of the algorithm.

Algorithm 3 The main frame of TC_NSGAIII

Input: the objective functions at time t , $F_t(x)$;
the time window, t ;
a kernel function, k ;

Output: the DPF of time window t , DPF_t ;

```

1: Randomly initialize a population  $RP$ ;
2: for  $t = 0 : T$  do
3:    $P_{IN} = \text{Centroid Distance}(RP, \sigma)$ ;
4:   According to the objective function before and after the change, two groups of solutions  $X_s$  and  $X_t$  are randomly selected from  $P_{IN}$ ;
5:   Use NSGA-III to optimize  $F_0(x)$  to get  $DPF_0$ ;
6:   Obtain the matrix  $\mathbf{W}$  through  $\text{TCA}(F_t(X_s), F_{t+1}(X_t), k)$ ;
7:   The set of solutions in latent space  $LS = \emptyset$ ;
8:   for every particle  $i$  in  $DPF_t$  do
9:      $\varphi(i) = \mathbf{W}^T \times k(i)$ ;
10:     $LS = LS \cup \varphi(i)$ ;
11:   end for
12:   for every particle  $j$  in  $LS$  do
13:     The found individual  $p = \text{argmin}(|\varphi(F_{t+1}(p)) - j|)$ ;
14:     The final initial population for evolution  $P_{INI} = P_{INI} \cup p$ ;
15:   end for
16:   Use NSGA-III to optimize the initial population  $P_{INI}$ , and get the  $DPF_t$ ;
17: end for
18: return  $DPF_t$ .
```

Experiments

Our experiments are mainly divided into two parts: the first part shows the effectiveness of TC_NSGAIII through strategies comparisons, and the performance improvement can be

The 12 test functions we use come from CEC 2015 [40]. The CEC 2015 test set contains a total of 12 dynamic questions, including various types of changes. These benchmarks come from some classic dynamic problems and with some variants added. Among them, FDA4, FDA5, and their

Table 1 Parameters of test functions

Function name	Number of objectives	Dimensions of decision variables	DPS change	DPF change
DIMP2	2	10	Yes	No
dMOP3	2	10	Yes	No
DMOP2	2	10	Yes	No
DMOP2_iso	2	10	Yes	Yes
DMOP2_dec	2	10	Yes	Yes
HE2	2	10	No	Yes
HE7	2	10	No	Yes
HE9	2	10	No	Yes
FDA4	3	12	Yes	No
FDA5	3	12	No	Yes
FDA5_iso	3	12	No	Yes
FDA5_dec	3	12	No	Yes

variants are three-objective problems, and the rest are two-objective. There are mainly three types of changes, namely, DPS and DPF change at the same time, DPF changes but DPS does not change, and DPS changes but DPF does not change. The change characteristics of specific problems are shown in Table 1. According to [40], the properties of these benchmarks are different. For example, the DPF changes from convex to concave for dMOP2, HE2, and HE9. The discontinuous DPF can be seen in HE2. For DIMP2, the changing rate of every decision variable is fixed. All in all, these 12 benchmark functions contain different types of dynamic DPF changes, including continuous, discontinuous, concave, convex, and isolated. The CEC 2015 benchmark set is a good standard of measuring whether dynamic algorithms can adapt to various environmental changes well.

In the 12 test benchmarks, we use the time parameter t and define it as $t = (1/n_t) \lfloor \tau_T / \tau_t \rfloor$, where n_t , τ_T , and τ_t mean the change severity, the maximum number of iterations, and change frequency, respectively. As described in Table 2, eight different dynamic parameters are set for conducting experiments. We set different change severity, change frequency and number of iterations to ensure that each function can be changed 20 times. The whole population size is set to 200, which means 200 solutions can be generated within the evolution process. Besides, the parameter σ of centroid distance method is set to 3.6. For transfer learning phase, the Gaussian kernel function is employed. The specific true DPFs of 12 test functions under eight different dynamic parameters can be observed through Fig. 3. Among these DPFs in Fig. 3, the red line depicts the DPF at the initial time step ($t = 0$), and the blue describes the general trend at the time step $t = 1$. It is obvious that the true DPFs of twelve functions are not identical as the time step changes.

Three metrics are mainly used to measure the performance of the dynamic algorithms. These three metrics are based on variants of the classic multi-objective metric inverted gener-

Table 2 The dynamic parameters

Settings	Change severity	Change frequency	Maximum iterations
S1	10	5	100
S2	10	10	200
S3	10	25	500
S4	10	50	1000
S5	1	10	200
S6	1	50	1000
S7	20	10	200
S8	20	50	1000

ational distance (IGD), which is a comprehensive evaluation metric. The overall performance can be measured by calculating the sum of the minimum distances between the real DPF and the set of individuals obtained by the selected algorithms. An algorithm with excellent comprehensive performance not only converges quickly but also has a good distribution. The specific formula of IGD is as follows:

$$\text{IGD}(S, P^*) = \frac{\sum_{x \in P^*} \min_{y \in S} \text{dis}(x, y)}{|P^*|} \quad (14)$$

where P^* is a set of points evenly distributed on the real DPF, and S is the optimal Pareto optimal solution set obtained by the algorithm. What's more, $|P^*|$ shows the number of solutions distributed on the real DPF, while $\text{dis}(x, y)$ calculates the minimum Euclidean distance between the real DPF and obtained solutions. The overall meaning of the formula is to find the average value of the minimum distance from the point set on the real DPF to the obtained population. From the above equation, it can be seen that a smaller value of $\text{dis}(x, y)$ means a better convergence, and then the convergence performance of the algorithm can be displayed concretely. However, a poor distribution represents that most

(a) DIMP2

(b) dMOP3

(c) DMOP2

(d) DMOP2_dec

(e) DMOP2_iso

(f) HE2

(g) HE7

(h) HE9

(i) FDA4

(j) FDA5

(k) FDA5_dec

(l) FDA5_iso

Fig. 3 The true DPFs of 12 benchmarks. The red and blue lines represent the true DPFs at time 0 and 1, respectively

individuals in the population are concentrated in a small area, and then the calculated $\text{dis}(x, y)$ of many individuals will be large. Therefore, the smaller $\text{dis}(x, y)$ obtained, the better the overall performance is. What's more, to measure dynamic algorithms' performance accurately, a variant of IGD was proposed [41]. Based on the definitions of IGD, some concepts about time steps had been added into the metric IGD. The metric MIGD gets the average value of IGD in some time steps over a single run, which is given by:

$$\text{MIGD} = \frac{1}{|T|} \sum_{t \in T} \text{IGD}(S, P^*). \quad (15)$$

In Eq. (15), T represents some discrete time steps within a run. $|T|$ is the cardinality of T . Obviously, it is worth noting that MIGD considers different dynamics in one environment, and the comprehensive performance can be evaluated through observing the final result of MIGD.

In addition, for a more in-depth study of DMOEAs, another metric was adopted. What can be seen obviously from Eq. (15) is that MIGD just takes only one environment into consideration when measuring DMOEAs. However, there is no doubt that more than one environmental change would occur in real-world applications. Therefore, a novel metric DMIGD used for measuring the overall performance under different environments was defined [14], which can be described as follows:

$$\text{DMIGD} = \frac{1}{|E|} \sum_{e \in E} \text{MIGD}(S, P^*, T), \quad (16)$$

where E represents the number of environments in the experiments. According to Table 2, there are eight different configurations (S1–S8) in experiments, which means the value of $|E|$ is 8.

The overall performance of an algorithm can be evaluated through above three metrics. Besides, comparing with the original algorithm, another variant about MIGD is adopted for measuring the performance improvement. Therefore, another metric is proposed in this paper for measuring the degree of improvement. The specific metric IR (Improvement Ratio) is based on the values of MIGD. By comparing with original values of MIGD, the degree of improvement can be expressed as a more intuitive percentage. The metric IR can be formulated as:

$$\text{IR} = \frac{\text{MIGD}_{\text{ori}}(S, P^*, T) - \text{MIGD}_{\text{imp}}(S, P^*, T)}{|\text{MIGD}_{\text{ori}}(S, P^*, T)|} \times 100\%. \quad (17)$$

The metric IR shows the difference between the original algorithm and the improved algorithm. The bigger the value of IR is, the more performance increases.

The overall experiments are mainly divided into two parts: the first part verifies the performance of TC_NSGAIII

through comparing with six dynamic algorithms, and the improvement ratio is calculated through the second experiment, showing the efficiency of our proposed strategy. In addition, the other two algorithms are also combined with our proposed strategy, and the improvement is also obvious.

Our entire experiments are implemented with MATLAB R2018a. In the experiments, we use 12 test functions, which come from CEC 2015 [40]. The specific characteristics of 12 test functions can be shown in Table 1.

Performance comparisons with other algorithms

In this section, the experiments about the performance comparisons are conducted. The metrics MIGD and DMIGD mentioned above are employed, and some algorithms are compared with our proposed algorithm. At first, the MIGD values of four algorithms are compared, which proves the effectiveness of proposed algorithm. These four algorithms include NSGA-III combined both the transfer learning and centroid distance strategies (TC_NSGAIII), NSGA-III incorporated with the transfer learning method only (Tr_NSGAIII), NSGA-III combined the single centroid distance strategy (Ce_NSGAIII), and the original NSGA-III algorithm [19]. In addition to above four algorithms, other three dynamic algorithms are also adopted for comparison, which are Kalman prediction-based MOEA (MOEA/D-KF) [6], multi-objective particle swarm optimization with multi-directional prediction (MDP_MOPSO) [26], and NSGA-II combined with transfer learning (Tr_NSGAII) [14]. The main framework of compared algorithms will be described in detail next.

Because the individual in population is ten-dimensional, it is difficult to describe the individual accurately in a figure. To observe the performance of our proposed algorithm, the DPFs obtained by TC_NSGAIII at time 20 are shown in Fig. 4. On the whole, the DPFs of 12 benchmarks have certain regularity, especially for two-objective functions. Comparing with Fig. 3, the general trend is the same. Obviously, some differences exist between the obtained DPFs and true DPFs, which ensures the diversity of solutions. The convergence of DPFs can also be depicted clearly in Fig. 4. However, it is not accurate to depict the diversity and convergence of solutions on figures alone. Some metrics should be used for measuring the performance of an algorithm more precisely, such as MIGD and its variants.

The MIGD values of four algorithms under eight different dynamic configurations are shown in Table 3. It is pretty obvious that the best results of TC_NSGAIII achieves 74 among 96 total results. The ratio of best results can be up to 78%. In the all results of twelve different benchmarks, TC_NSGAIII performs better than other three algorithms generally. The specific data in Table 3 suggest that the performance of TC_NSGAIII achieves best in dMOP2 and

Table 3 The MIGD values of four comparative algorithms

Functions	Settings	NSGA-III	TC_NSGAIII	Tr_NSGAIII	Ce_NSGAIII	WINNER
DIMP2	S1	3.8637E + 00	2.5230E + 00	2.6405E + 00	3.9843E + 00	TC_NSGAIII
	S2	4.9439E + 00	2.8456E + 00	2.8104E + 00	4.1834E + 00	Tr_NSGAIII
	S3	3.6318E + 00	2.1167E + 00	2.4518E + 00	4.2689E + 00	TC_NSGAIII
	S4	3.9102E + 00	2.3771E + 00	2.6426E + 00	3.7611E + 00	TC_NSGAIII
	S5	3.9982E + 00	2.5202E + 00	2.4879E + 00	4.0133E + 00	Tr_NSGAIII
	S6	3.9014E + 00	2.4372E + 00	2.5824E + 00	4.0594E + 00	TC_NSGAIII
	S7	3.7583E + 00	2.5808E + 00	2.6098E + 00	4.1320E + 00	TC_NSGAIII
	S8	3.7385E + 00	2.6288E + 00	2.7065E + 00	3.8613E + 00	TC_NSGAIII
dMOP2	S1	1.6630E−01	6.0100E−02	3.8270E−01	1.7200E−01	TC_NSGAIII
	S2	1.1810E−01	4.7200E−02	1.1476E + 00	1.2480E−01	TC_NSGAIII
	S3	1.3430E−01	4.6100E−02	8.3000E−01	1.3710E−01	TC_NSGAIII
	S4	1.2740E−01	5.9900E−02	6.9680E−01	1.3210E−01	TC_NSGAIII
	S5	4.7310E−01	2.9250E−01	6.4370E−01	4.0750E−01	TC_NSGAIII
	S6	4.8620E−01	2.3550E−01	7.7480E−01	3.4930E−01	TC_NSGAIII
	S7	1.0880E−01	5.3800E−02	4.1100E−01	1.4400E−01	TC_NSGAIII
	S8	1.0050E−01	4.1000E−02	5.1620E−01	8.5900E−02	TC_NSGAIII
dMOP2_dec	S1	5.0320E−01	3.0260E−01	8.6680E−01	4.2740E−01	TC_NSGAIII
	S2	4.8590E−01	2.0050E−01	9.1510E−01	3.6440E−01	TC_NSGAIII
	S3	3.8500E−01	1.8730E−01	9.0000E−01	4.5560E−01	TC_NSGAIII
	S4	4.1970E−01	1.7030E−01	8.2910E−01	5.5100E−01	TC_NSGAIII
	S5	1.8943E + 00	1.0774E + 00	8.9210E−01	1.7879E + 00	Tr_NSGAIII
	S6	1.7746E + 00	1.0843E + 00	1.0265E + 00	1.7279E + 00	Tr_NSGAIII
	S7	4.1170E−01	2.1760E−01	1.1305E + 00	4.0600E−01	TC_NSGAIII
	S8	3.8840E−01	2.3800E−01	1.2548E + 00	3.7910E−01	TC_NSGAIII
dMOP2_iso	S1	5.2600E−02	2.0000E−03	2.1000E−03	5.2600E−02	TC_NSGAIII
	S2	5.2600E−02	2.2000E−03	2.3000E−03	5.2600E−02	TC_NSGAIII
	S3	5.2700E−02	2.2000E−03	2.1000E−03	5.2700E−02	Tr_NSGAIII
	S4	5.2700E−02	2.1000E−03	2.1000E−03	5.2700E−02	Tr_NSGAIII
	S5	1.2930E−01	1.2590E−01	1.3700E−01	1.2930E−01	TC_NSGAIII
	S6	1.2930E−01	1.2320E−01	1.3320E−01	1.2930E−01	TC_NSGAIII
	S7	5.0800E−02	2.5000E−03	2.7000E−03	5.0800E−02	TC_NSGAIII
	S8	5.0800E−02	1.9000E−03	2.0000E−03	5.0800E−02	TC_NSGAIII
dMOP3	S1	1.2740E−01	3.9500E−02	4.1300E−02	1.2180E−01	TC_NSGAIII
	S2	7.6600E−02	3.8700E−02	3.7500E−02	7.8100E−02	Tr_NSGAIII
	S3	7.6700E−02	3.4200E−02	3.9500E−02	9.0200E−02	TC_NSGAIII
	S4	1.4670E−01	3.9600E−02	4.0700E−02	8.5800E−02	TC_NSGAIII
	S5	7.0240E−01	2.1730E−01	1.9420E−01	6.2590E−01	Tr_NSGAIII
	S6	2.9820E−01	2.3610E−01	3.7303E + 00	2.4440E−01	TC_NSGAIII
	S7	1.0650E−01	3.3900E−02	4.3000E−02	6.8100E−02	TC_NSGAIII
	S8	6.9400E−02	3.4900E−02	3.4100E−02	8.5100E−02	Tr_NSGAIII
FDA4	S1	1.7330E−01	8.2900E−02	8.0600E−02	1.6880E−01	Tr_NSGAIII
	S2	1.4700E−01	7.8100E−02	7.8800E−02	1.5850E−01	TC_NSGAIII
	S3	1.5730E−01	7.7200E−02	8.0400E−02	1.3730E−01	TC_NSGAIII
	S4	1.6500E−01	7.5200E−02	7.5900E−02	1.4730E−01	TC_NSGAIII
	S5	2.8170E−01	7.8200E−02	8.2400E−02	2.9100E−01	TC_NSGAIII
	S6	2.9330E−01	7.7800E−02	8.4300E−02	3.0080E−01	TC_NSGAIII
	S7	1.5280E−01	8.1000E−02	7.6900E−02	1.4530E−01	Tr_NSGAIII

Table 3 continued

Functions	Settings	NSGA-III	TC_NSGAIII	Tr_NSGAIII	Ce_NSGAIII	WINNER
FDA5	S8	1.6110E−01	8.5900E−02	7.5400E−02	1.3850E−01	Tr_NSGAIII
	S1	1.5440E−01	9.7100E−02	9.9400E−02	1.7080E−01	TC_NSGAIII
	S2	1.5440E−01	1.3280E−01	1.5210E−01	3.1150E−01	TC_NSGAIII
	S3	1.5570E−01	1.0710E−01	9.0200E−02	1.8040E−01	Tr_NSGAIII
	S4	1.5780E−01	1.0420E−01	1.3520E−01	1.5500E−01	TC_NSGAIII
	S5	4.8560E−01	3.4740E−01	4.3340E−01	4.9830E−01	TC_NSGAIII
	S6	5.0900E−01	2.9510E−01	3.5370E−01	5.2810E−01	TC_NSGAIII
	S7	1.4210E−01	1.9960E−01	1.5760E−01	1.3670E−01	Ce_NSGAIII
FDA5_dec	S8	1.5530E−01	1.1020E−01	1.1090E−01	1.4530E−01	TC_NSGAIII
	S1	4.2080E−01	3.6520E−01	1.4841E + 00	4.1490E−01	TC_NSGAIII
	S2	3.7460E−01	3.1160E−01	1.6172E + 00	3.3640E−01	TC_NSGAIII
	S3	4.2290E−01	3.6210E−01	1.8043E + 00	2.6680E−01	TC_NSGAIII
	S4	4.1730E−01	3.0100E−01	1.0083E + 00	3.8520E−01	TC_NSGAIII
	S5	6.6060E−01	7.4710E−01	1.2028E + 00	6.1390E−01	Ce_NSGAIII
	S6	6.7580E−01	8.7950E−01	1.3446E + 00	6.3320E−01	Ce_NSGAIII
	S7	2.4600E−01	1.6480E−01	1.1348E + 00	2.9200E−01	TC_NSGAIII
FDA5_iso	S8	2.3120E−01	2.7860E−01	1.3004E + 00	2.9990E−01	NSGAIII
	S1	3.3870E−01	7.8500E−02	8.9330E−01	2.8020E−01	TC_NSGAIII
	S2	4.1550E−01	1.2540E−01	5.7710E−01	2.7490E−01	TC_NSGAIII
	S3	4.0600E−01	7.3500E−02	9.0120E−01	3.0310E−01	TC_NSGAIII
	S4	3.7360E−01	1.6280E−01	4.9740E−01	2.7840E−01	TC_NSGAIII
	S5	6.4900E−01	1.8270E−01	5.4060E−01	2.2720E−01	TC_NSGAIII
	S6	6.9100E−01	2.0860E−01	6.0920E−01	2.2420E−01	TC_NSGAIII
	S7	2.7230E−01	7.6200E−02	7.9710E−01	2.5940E−01	TC_NSGAIII
HE2	S8	2.5870E−01	6.5800E−02	5.7680E−01	2.5960E−01	TC_NSGAIII
	S1	2.6350E−01	1.7580E−01	2.5462E + 00	2.6570E−01	TC_NSGAIII
	S2	1.7940E−01	1.6880E−01	2.5821E + 00	1.6930E−01	TC_NSGAIII
	S3	1.6640E−01	1.4700E−01	2.8485E + 00	1.6590E−01	TC_NSGAIII
	S4	1.7670E−01	1.4420E−01	2.5698E + 00	1.8830E−01	TC_NSGAIII
	S5	3.0640E−01	6.9800E−02	2.6999E + 00	2.1330E−01	TC_NSGAIII
	S6	2.6580E−01	7.0700E−02	2.8734E + 00	1.9070E−01	TC_NSGAIII
	S7	2.4360E−01	1.3510E−01	2.6168E + 00	1.9210E−01	TC_NSGAIII
HE7	S8	1.7080E−01	1.7280E−01	2.1142E + 00	1.7340E−01	NSGAIII
	S1	9.1200E−02	5.4800E−02	4.6600E−02	9.0100E−02	Tr_NSGAIII
	S2	9.5500E−02	4.6000E−02	4.9100E−02	8.7800E−02	TC_NSGAIII
	S3	9.8800E−02	5.0700E−02	5.6800E−02	9.6700E−02	TC_NSGAIII
	S4	9.7300E−02	4.4300E−02	5.0400E−02	9.3800E−02	TC_NSGAIII
	S5	7.3700E−02	4.1000E−02	4.1600E−02	7.5600E−02	TC_NSGAIII
	S6	7.3300E−02	4.3400E−02	4.6400E−02	6.7600E−02	TC_NSGAIII
	S7	9.7300E−02	4.5300E−02	5.6300E−02	9.2600E−02	TC_NSGAIII
HE9	S8	9.1600E−02	4.2200E−02	8.5300E−02	9.2500E−02	TC_NSGAIII
	S1	3.0650E−01	3.1500E−01	3.2520E−01	3.0760E−01	NSGAIII
	S2	3.0880E−01	3.0650E−01	3.7400E−01	2.9300E−01	TC_NSGAIII
	S3	2.9850E−01	3.0460E−01	3.6600E−01	2.9960E−01	NSGAIII
	S4	3.0690E−01	2.7000E−01	2.9690E−01	2.9910E−01	TC_NSGAIII
	S5	2.6070E−01	2.8090E−01	4.5590E−01	2.6360E−01	NSGAIII
	S6	2.6170E−01	2.5820E−01	4.1320E−01	2.6920E−01	TC_NSGAIII
	S7	3.0480E−01	2.9830E−01	3.2650E−01	2.9910E−01	TC_NSGAIII
	S8	3.0560E−01	3.0040E−01	3.9210E−01	3.0130E−01	TC_NSGAIII

Fig. 4 The DPFs of 12 benchmarks obtained by TC_NSGAIII at time 20

FDA5_iso. Even with eight different sets of dynamic parameters, the ratio of the optimal results in the two test functions mentioned reaches 100%. Combined with the dynamic characteristics and DPFs depicted in Table 1 and Fig. 3, our proposed algorithm is suitable for solving DMOPs with changing DPS or DPF. TC_NSGAIII performs equally on the other ten benchmark functions. For each of these 12 test functions, TC_NSGAIII can ensure more than five best results in eight experimental settings. We can see that the performance of TC_NSGAIII is significantly better than Tr_NSGAIII and Ce_NSGAIII. The last column in Table 3 represents the “winner” in this comparison. By comparative analysis, the combination of transfer learning and centroid distance can deal with varying degrees of changes, which is suitable for both two-dimensional and three-dimensional functions. From the perspective of different dynamic parameters, TC_NSGAIII reaches the 11 best results in 12 total functions when the experiments setting is “S4”. “S4” means that the change severity is set to 10 and the change frequency is set to 50. It implies that comparing other change types, TC_NSGAIII performs well when facing relatively severe and infrequent changes. In summary, the data in Table 3 fully reflect the effectiveness and necessity of combining the transfer learning and centroid distance strategy.

The specific experimental data in Table 4 show the DMIGD values of eight different algorithms. Except for the compared algorithms in Table 3, other three dynamic algorithms are employed for comparison. MOEAD-KF proposed by Arrchana et al. [6] used Kalman filter prediction to deal with DMOPs. Kalman filter can be an excellent prediction tool to estimate the process state. In this Kalman filter prediction model, a feedback control was employed for estimation. The Kalman filter was mainly divided into two groups: time update equations which predict the next state and measurement update equations used for information feedback. MDP_MOPSO combined multidirectional prediction with multi-objective particle swarm optimization algorithm, which was proposed by Rong et al. [26]. In multi-objective prediction model, the optimal solution of the problem was clustered into representative individuals. The representative individuals were used for predicting the moving DPF, and then the predicted solutions were generated. Tr_NSGAII incorporated transfer learning into NSGAII for solving DMOPs, which is put forward by Jiang et al. [14]. MOEA/D-SVR, proposed by Cao et al. [42], presented a support vector regression predictor with MOEA/D to generate the initial population in the new environment. A nonlinear mapping was used for mapping the historical solutions into a high-dimensional feature space.

The DMIGD values of eight compared algorithms can be shown in Table 4. It is obvious that the best results of TC_NSGAIII reaches 9 in 12 test functions. The metric DMIGD represents the overall performance of dynamic

algorithms when facing unknown changes. Among the seven compared algorithms, TC_NSGAIII has better performance than other dynamic algorithms over different types of change. Comparing with NSGA-III and Tr_NSGAIII, the DMIGD values of TC_NSGAIII are smaller under 12 functions, which implies that TC_NSGAIII has better stability in tracking moving DPF.

Synthesizing the experimental data in Tables 3 and 4, the effectiveness and stability of TC_NSGAIII can be fully demonstrated. What's more, TC_NSGAIII is competitive through comparing with some chosen dynamic algorithms. It shows good ability to respond to changes in both MIGD and DMIGD metrics, and more accurately tracks the true DPF of the next time step through the predicted population. There is no doubt that TC_NSGAIII is an outstanding dynamic multi-objective algorithm in dealing with different types of DMOPs.

Performance improvement experiments

The experiments about performance improvement are implemented in this section. Three classical evolutionary algorithms are chosen for combining with transfer learning and centroid distance strategies. These evolutionary algorithms are NSGA-III, MOPSO, and NSGA-II, respectively. It is worth mentioning that these three algorithms are not suitable for dynamic optimization. However, some dynamic strategies can be added to these algorithms to solve dynamic problems. It is not difficult to find that MOPSO and NSGA belong to different classes, which can prove the capability of strategies on different algorithms. The reasons why both NSGA-III and NSGA-II are chosen are as follows: first of all, the dynamic version of NSGA-II has been proposed by Deb et al. [22]. However, NSGA-III has not been employed for dynamic optimization. One of the innovations of this paper is to apply the strategies to NSGA-III to solve DMOPs. Second, applying the same dynamic strategy to NSGA-II and NSGA-III and comparing them, a conclusion can be drawn that which algorithm this strategy is more suitable for. It is very helpful to promote the follow-up research of NSGA-III in dynamic optimization.

The experimental data in Tables 5, 6, 7 represent the performance difference between original chosen algorithms and improved algorithms. The IR values in bold mean the positive performance improvements. The observation we can get from Tables 5, 6, 7 is that the overall effective ratio of TC_NSGAIII, TC_MOPSO, and TC_NSGAII is 92.71% (89 cases of total 96 tests), 78.13% (75 cases of total 96 tests), and 67.71% (65 cases of total 96 tests), respectively. By carefully analyzing the data in Tables 5, 6, 7, more precise information can be obtained. For TC_NSGAIII, there are 8 cases whose IR values exceed 100%, and the IR values of 33 tests are between 50 and 100%. For TC_MOPSO, there

Table 4 The DMIGD values of eight different algorithms

Functions	TC_NSGAIII	NSGA-III	Tr_NSGAIII	Ce_NSGAIII	MOEA/D-KF	MDP_MOPSO	Tr_NSGAII	MOEA/D-SVR
DIMP2	2.5037E + 00	3.9682E + 00	2.6165E + 00	4.0329E + 00	2.2936E + 01	7.0764E + 00	3.7953E + 00	1.7831E + 01
dMOP2	1.0450E−01	2.1430E−01	6.7530E−01	1.9410E−01	3.0619E + 00	5.1655E + 00	4.2020E−01	4.1058E + 00
dMOP2_dec	4.3480E−01	7.8290E−01	9.7690E−01	7.6240E−01	9.2188E + 00	9.4826E + 00	6.3030E−01	1.0343E + 00
dMOP2_iso	3.2800E−02	7.1400E−02	3.5400E−02	7.1300E−02	3.1600E−02	3.9500E−02	3.2500E−02	6.4755E + 00
dMOP3	8.4300E−02	2.0050E−01	5.2010E−01	1.7490E−01	8.3600E−02	4.9036E + 00	1.3924E + 00	4.0396E + 00
FDA4	7.9500E−02	1.9140E−01	7.9300E−02	1.8590E−01	1.9130E−01	1.0720E−01	1.9250E−01	2.9821E + 00
FDA5	1.7420E−01	2.3930E−01	1.9160E−01	2.6580E−01	4.9630E−01	2.4400E−01	3.3440E−01	1.6927E + 00
FDA5_dec	4.2620E−01	4.3120E−01	1.3621E + 00	4.0530E−01	5.4760E−01	4.4720E−01	4.5590E−01	3.1968E + 00
FDA5_iso	1.2170E−01	4.2560E−01	6.7410E−01	2.6340E−01	1.4650E−01	1.7990E−01	3.2640E−01	2.8520E + 00
HE2	1.3550E−01	2.2160E−01	2.6064E + 00	1.9480E−01	7.4500E−02	1.4611E + 00	2.0960E−01	3.3632E + 00
HE7	4.6000E−02	8.9800E−02	5.4100E−02	8.7100E−02	2.3650E−01	1.4610E−01	1.9660E−01	6.9010E−01
HE9	2.9240E−01	2.9420E−01	3.6870E−01	2.9160E−01	4.1080E−01	1.4561E + 00	2.9540E−01	1.3827E + 00

are no cases whose IR exceeds 100%, and the IR of 50 cases are greater than 50% and less than 100%. For TC_NSGAII, there are also no IR values exceeding 100%, and only 18 IR values are between 50 and 100%. The results we get from experiments show that the transfer learning and centroid distance strategy can indeed improve the performance of an algorithm when dealing with DMOPs. Through the comparison of the distribution of IR values, it is obvious that the IR values of TC_NSGAIII reach the highest, which means that TC_NSGAIII is more adaptable to changes in the dynamic environments. Although NSGA-II is often used for static multi-objective problems, the experiments demonstrate that our proposed strategy is more suitable for NSGA-III to solve DMOPs. The improved TC_NSGAIII is proved to be competitive through a series of experiments.

Conclusions

A new idea combining transfer learning and centroid distance strategy was proposed in this paper. Our approach was incorporated into the framework of NSGA-III to solve DMOPs. A given DMOP changes with different time steps, and the optimal solutions at different times obey different distributions. Therefore, some classical methods could not be used for dealing with these Non-IID problems. In addition, few studies applied NSGA-III to dynamic optimization, which makes the dynamic version of NSGA-III more valuable for research.

As an outstanding dynamic algorithm, it must be able to track the moving DPF quickly and accurately. In our proposed approach, the centroid distance was first used to find the solutions closer to the reference points in the initial population. And then, the found solutions were put into the transfer learning process. Transfer learning is mainly to map the source data before the change and the target data after the change to the same latent space, and then looks for the individuals closest to the real DPF in the mapped space to form a new initial population. To verify the feasibility and effectiveness of our method, two parts of experiments were conducted in this paper. Through the comparison with some original algorithms and other dynamic algorithms, it is obvious that TC_NSGAIII performs best on 12 CEC 2015 benchmarks, and the proposed technique can improve the performance of NSGA-III to a great extent.

More techniques should be further studied on DMOPs. For future works, we will focus on improving the efficacy of the proposed algorithms and try to propose more techniques on solving DMOPs. Besides, our proposed approach is expected to be integrated into more classical algorithms and applied to solve more DMOPs in real world.

Table 5 The IR values of TC_NSGAIII under 12 test functions

Functions	Algorithms/Metric	S1	S2	S3	S4	S5	S6	S7	S8
DIMP2	NSGA-III	3.8637E + 00	4.9439E + 00	3.6318E + 00	3.9102E + 00	3.9982E + 00	3.9014E + 00	3.7583E + 00	3.7385E + 00
	TC_NSGAIII	2.5230E + 00	2.8456E + 00	2.1167E + 00	2.3771E + 00	2.5202E + 00	2.4372E + 00	2.5808E + 00	2.6288E + 00
	IR	34.70%	42.44%	41.72%	39.21%	36.97%	37.53%	31.33%	29.68%
dMOP2	NSGA-III	1.6630E−01	1.1810E−01	1.3430E−01	1.2740E−01	4.7310E−01	4.8620E−01	1.0880E−01	1.0050E−01
	TC_NSGAIII	6.0100E−02	4.7200E−02	4.6100E−02	5.9900E−02	2.9250E−01	2.3550E−01	5.3800E−02	4.1000E−02
	IR	176.77%	150.19%	191.60%	112.53%	61.72%	106.49%	102.11%	145.35%
dMOP2_dec	NSGA-III	5.0320E−01	4.8590E−01	3.8500E−01	4.1970E−01	1.8943E + 00	1.7746E + 00	4.1170E−01	3.8840E−01
	TC_NSGAIII	3.0260E−01	2.0050E−01	1.8730E−01	1.7030E−01	1.0774E + 00	1.0843E + 00	2.1760E−01	2.3800E−01
	IR	39.87%	58.74%	51.33%	59.44%	43.12%	38.90%	47.15%	38.72%
dMOP2_iso	NSGA-III	5.2600E−02	5.2600E−02	5.2700E−02	5.2700E−02	1.2930E−01	1.2930E−01	5.0800E−02	5.0800E−02
	TC_NSGAIII	2.0000E−03	2.2000E−03	2.2000E−03	2.1000E−03	1.2590E−01	1.2320E−01	2.5000E−03	1.9000E−03
	IR	96.30%	95.74%	95.83%	95.98%	2.64%	4.77%	95.03%	96.17%
dMOP3	NSGA-III	1.2740E−01	7.6600E−02	7.6700E−02	1.4670E−01	7.0240E−01	2.9820E−01	1.0650E−01	6.9400E−02
	TC_NSGAIII	3.9500E−02	3.8700E−02	3.4200E−02	3.9600E−02	2.1730E−01	2.3610E−01	3.3900E−02	3.4900E−02
	IR	69.01%	49.51%	55.49%	72.98%	69.07%	20.83%	68.14%	49.70%
FDA4	NSGA-III	1.7330E−01	1.4700E−01	1.5730E−01	1.6500E−01	2.8170E−01	2.9330E−01	1.5280E−01	1.6110E−01
	TC_NSGAIII	8.2900E−02	7.8100E−02	7.7200E−02	7.5200E−02	7.8200E−02	7.7800E−02	8.1000E−02	8.5900E−02
	IR	52.15%	46.87%	50.92%	54.42%	72.22%	73.47%	46.99%	46.67%
FDA5	NSGA-III	1.5440E−01	1.5440E−01	1.5570E−01	1.5780E−01	4.8560E−01	5.0900E−01	1.4210E−01	1.5530E−01
	TC_NSGAIII	9.7100E−02	1.3280E−01	1.0710E−01	1.0420E−01	3.4740E−01	2.9510E−01	1.0080E−01	1.1020E−01
	IR	37.09%	13.96%	31.21%	33.97%	28.46%	42.03%	29.06%	29.04%
FDA5_dec	NSGA-III	4.2080E−01	3.7460E−01	4.2290E−01	4.1730E−01	6.6060E−01	6.7580E−01	2.4600E−01	2.3120E−01
	TC_NSGAIII	3.6520E−01	3.1160E−01	3.6210E−01	3.0100E−01	7.4710E−01	8.7950E−01	1.6480E−01	2.7860E−01
	IR	13.20%	16.83%	14.39%	27.87%	− 13.09%	− 30.13%	32.99%	− 20.51%
FDA5_iso	NSGA-III	3.3870E−01	4.1550E−01	4.0600E−01	3.7360E−01	6.4900E−01	6.9100E−01	2.7230E−01	2.5870E−01
	TC_NSGAIII	7.8500E−02	1.2540E−01	7.3500E−02	1.6280E−01	1.8270E−01	2.0860E−01	7.6200E−02	6.5800E−02
	IR	76.82%	69.82%	81.90%	56.42%	71.84%	69.81%	72.03%	74.55%
HE2	NSGA-III	2.6350E−01	1.7940E−01	1.6640E−01	1.7670E−01	3.0640E−01	2.6580E−01	2.4360E−01	1.7080E−01
	TC_NSGAIII	1.7580E−01	1.6880E−01	1.4700E−01	1.4420E−01	6.9800E−02	7.0700E−02	1.3510E−01	1.7280E−01
	IR	33.30%	5.92%	11.67%	18.42%	77.22%	73.41%	44.56%	− 1.22%
HE7	NSGA-III	9.1200E−02	9.5500E−02	9.8800E−02	9.7300E−02	7.3700E−02	7.3300E−02	9.7300E−02	9.1600E−02
	TC_NSGAIII	5.4800E−02	4.6000E−02	5.0700E−02	4.4300E−02	4.1000E−02	4.3400E−02	4.5300E−02	4.2200E−02
	IR	39.92%	51.86%	48.70%	54.51%	44.33%	40.78%	53.42%	53.97%
HE9	NSGA-III	3.0650E−01	3.0880E−01	2.9850E−01	3.0690E−01	2.6070E−01	2.6170E−01	3.0480E−01	3.0560E−01
	TC_NSGAIII	3.1500E−01	3.0650E−01	3.0460E−01	2.7000E−01	2.8090E−01	2.5820E−01	2.9830E−01	3.0040E−01
	IR	− 2.78%	0.74%	− 2.02%	12.01%	− 7.77%	1.35%	2.13%	1.69%

Table 6 The IR values of TC_MOPSO under 12 test functions

Functions	Algorithms/Metric	S1	S2	S3	S4	S5	S6	S7	S8
DIMP2	MOPSO	6.8683E + 00	6.3417E + 00	5.7457E + 00	6.1813E + 00	6.4344E + 00	6.8208E + 00	5.6799E + 00	6.0316E + 00
	TC_MOPSO	5.4132E + 00	5.4117E + 00	6.4275E + 00	4.1636E + 00	5.7731E + 00	5.6306E + 00	6.1122E + 00	6.1875E + 00
	IR	21.20%	14.70%	– 11.90%	32.60%	10.30%	17.40%	– 7.60%	– 2.60%
dMOP2	MOPSO	8.2260E–01	4.0460E–01	4.4020E–01	4.4730E–01	4.0210E–01	6.8480E–01	2.9920E–01	2.6030E–01
	TC_MOPSO	1.7630E–01	9.0600E–02	5.4000E–02	9.1700E–02	1.9212E + 01	1.8704E + 01	6.1100E–02	8.1100E–02
	IR	78.56%	77.60%	87.72%	79.51%	– 4677.45%	– 2631.15%	79.57%	68.85%
dMOP2_dec	MOPSO	1.6547E + 00	1.8701E + 00	1.6751E + 00	1.7298E + 00	1.9610E + 00	1.9141E + 00	1.9234E + 00	1.7923E + 00
	TC_MOPSO	1.6117E + 00	1.3992E + 00	3.4470E–01	5.9670E–01	1.8088E + 00	2.0439E + 00	7.4260E–01	3.3740E–01
	IR	2.60%	25.18%	79.42%	65.51%	7.76%	– 6.78%	61.39%	81.17%
dMOP2_iso	MOPSO	5.3600E–02	5.3600E–02	5.3600E–02	5.3600E–02	6.0800E–02	6.0800E–02	5.1300E–02	5.1300E–02
	TC_MOPSO	2.0450E–01	2.0450E–01	2.0450E–01	2.0450E–01	1.3580E–01	1.3580E–01	2.0280E–01	2.0280E–01
	IR	– 281.52%	– 281.34%	– 281.45%	– 281.51%	– 123.37%	– 123.23%	– 294.93%	– 294.95%
dMOP3	MOPSO	4.1960E–01	1.7290E–01	1.9450E–01	1.9260E–01	2.7890E–01	3.9390E–01	2.2550E–01	2.0710E–01
	TC_MOPSO	4.6500E–02	4.7100E–02	5.4100E–02	3.8600E–02	1.3550E–01	1.1420E–01	3.0300E–02	7.2200E–02
	IR	88.90%	72.80%	72.20%	80.00%	51.40%	71.00%	86.60%	65.10%
FDA4	MOPSO	8.6500E–02	8.4400E–02	8.1700E–02	8.5800E–02	8.6400E–02	8.4400E–02	8.6400E–02	8.3100E–02
	TC_MOPSO	6.5100E–02	6.5100E–02	6.5100E–02	6.5300E–02	5.2200E–02	5.2400E–02	6.5200E–02	6.5000E–02
	IR	24.70%	22.90%	20.30%	23.90%	39.60%	37.90%	24.50%	21.80%
FDA5	MOPSO	6.1500E–01	6.1720E–01	6.1560E–01	6.2020E–01	5.1820E–01	5.1990E–01	5.9830E–01	5.9750E–01
	TC_MOPSO	1.5650E–01	1.5650E–01	1.5390E–01	1.5660E–01	1.5190E–01	1.5300E–01	1.4400E–01	1.3540E–01
	IR	74.60%	74.60%	75.00%	74.70%	70.70%	70.60%	75.90%	77.30%
FDA5_dec	MOPSO	4.8320E–01	4.9060E–01	4.8700E–01	4.7420E–01	5.3270E–01	5.4920E–01	5.0320E–01	4.5720E–01
	TC_MOPSO	1.9820E–01	1.9050E–01	1.9440E–01	1.7870E–01	4.4420E–01	5.9230E–01	1.7970E–01	1.6780E–01
	IR	58.97%	61.17%	60.09%	62.32%	16.61%	– 7.84%	64.29%	63.30%
FDA5_iso	MOPSO	6.4330E–01	6.4320E–01	6.4330E–01	6.4330E–01	5.2910E–01	5.2900E–01	6.1980E–01	6.1980E–01
	TC_MOPSO	1.4950E–01	1.4250E–01	1.5100E–01	1.4370E–01	1.3130E–01	1.3980E–01	1.4340E–01	1.3620E–01
	IR	76.80%	77.90%	76.50%	77.70%	75.20%	73.60%	76.90%	78.00%
HE2	MOPSO	2.6460E–01	2.8240E–01	3.0430E–01	2.7870E–01	3.2450E–01	3.4460E–01	2.8550E–01	2.6500E–01
	TC_MOPSO	7.1400E–02	7.4600E–02	7.3700E–02	8.7400E–02	6.0800E–02	6.3800E–02	6.9800E–02	7.5600E–02
	IR	73.00%	73.60%	75.78%	68.65%	81.25%	81.48%	75.54%	71.46%
HE7	MOPSO	7.4600E–02	7.5200E–02	7.3700E–02	7.2900E–02	1.3840E–01	1.3330E–01	7.7400E–02	7.6600E–02
	TC_MOPSO	4.2600E–02	4.3400E–02	4.3400E–02	4.4200E–02	4.0800E–02	4.3000E–02	4.2500E–02	4.2900E–02
	IR	42.90%	42.30%	41.10%	39.30%	70.50%	67.70%	45.10%	44.10%
HE9	MOPSO	2.3890E–01	2.3890E–01	2.3630E–01	2.3790E–01	2.9320E–01	2.8810E–01	2.4520E–01	2.3980E–01
	TC_MOPSO	3.0150E–01	2.9830E–01	2.9340E–01	3.0180E–01	2.5660E–01	2.5830E–01	3.0030E–01	2.9850E–01
	IR	– 26.22%	– 24.86%	– 24.17%	– 26.85%	12.46%	10.35%	– 22.46%	– 24.48%

Table 7 The IR values of TC_NSGAII under 12 test functions

Functions	Algorithms/Metric	S1	S2	S3	S4	S5	S6	S7	S8
DIMP2	NSGA-II	4.1292E + 00	3.7319E + 00	3.4388E + 00	4.5792E + 00	4.7506E + 00	4.3862E + 00	4.1539E + 00	4.0084E + 00
	TC_NSGAII	2.6712E + 00	2.7449E + 00	2.3653E + 00	2.6132E + 00	2.4597E + 00	2.5398E + 00	2.6285E + 00	2.5977E + 00
	IR	35.31%	26.45%	31.22%	42.93%	48.22%	42.10%	36.72%	35.20%
dMOP2	NSGA-II	1.4160E−01	1.2490E−01	1.0500E−01	1.3270E−01	2.6890E−01	7.9910E−01	1.1010E−01	1.0560E−01
	TC_NSGAII	1.2030E−01	1.2860E−01	1.2240E−01	1.3870E−01	3.8680E−01	4.1690E−01	1.0300E−01	9.9200E−02
	IR	15.01%	− 2.99%	− 16.61%	− 4.59%	− 43.83%	47.83%	6.51%	6.07%
dMOP2_dec	NSGA-II	5.3440E−01	5.2810E−01	4.3030E−01	5.1750E−01	1.8459E + 00	2.0647E + 00	3.1990E−01	4.8390E−01
	TC_NSGAII	4.6540E−01	5.3410E−01	4.5810E−01	4.9450E−01	1.7968E + 00	3.5900E−01	4.3570E−01	4.7630E−01
	IR	12.91%	− 1.13%	− 6.46%	4.45%	2.66%	82.61%	− 36.21%	1.58%
dMOP2_iso	NSGA-II	5.2800E−02	5.2800E−02	5.2800E−02	5.2800E−02	1.3210E−01	1.3350E−01	5.0900E−02	5.0900E−02
	TC_NSGAII	5.2700E−02	5.2800E−02	5.2800E−02	5.2800E−02	1.2950E−01	1.3020E−01	5.0900E−02	5.0900E−02
	IR	0.08%	− 0.09%	0.08%	− 0.08%	2.00%	2.46%	− 0.06%	− 0.07%
dMOP3	NSGA-II	1.9910E−01	1.6540E−01	1.4380E−01	1.4260E−01	1.0424E + 00	4.5520E−01	1.6570E−01	1.2680E−01
	TC_NSGAII	7.0000E−02	6.4300E−02	6.4200E−02	6.2800E−02	2.6080E−01	3.0570E−01	4.2900E−02	5.1700E−02
	IR	64.83%	61.14%	55.36%	55.95%	74.98%	32.85%	74.09%	59.21%
FDA4	NSGA-II	1.8400E−01	1.7160E−01	1.6620E−01	1.6570E−01	3.0680E−01	2.7410E−01	1.5140E−01	1.7080E−01
	TC_NSGAII	1.1100E−01	1.0750E−01	1.0950E−01	1.0660E−01	1.0540E−01	1.0480E−01	1.1170E−01	9.9900E−02
	IR	39.66%	37.35%	34.12%	35.66%	65.63%	61.78%	26.26%	41.54%
FDA5	NSGA-II	2.9630E−01	1.9280E−01	1.9740E−01	2.0860E−01	5.1990E−01	5.5920E−01	1.9440E−01	2.6640E−01
	TC_NSGAII	2.7920E−01	1.3670E−01	1.4290E−01	1.9330E−01	3.5040E−01	3.3700E−01	1.2950E−01	1.4810E−01
	IR	5.77%	29.12%	27.61%	7.35%	32.60%	39.74%	33.40%	44.43%
FDA5_dec	NSGA-II	3.6610E−01	4.1790E−01	4.1570E−01	2.4330E−01	6.8910E−01	6.6180E−01	2.7130E−01	3.0340E−01
	TC_NSGAII	4.3320E−01	3.4460E−01	3.8570E−01	4.5340E−01	6.5480E−01	6.9850E−01	2.9530E−01	2.4080E−01
	IR	− 18.32%	17.52%	7.21%	− 86.35%	4.97%	− 5.54%	− 8.85%	20.62%
FDA5_iso	NSGA-II	2.8810E−01	2.6010E−01	2.6220E−01	2.4990E−01	2.2810E−01	2.2010E−01	2.5820E−01	2.5450E−01
	TC_NSGAII	1.4590E−01	1.8870E−01	2.1660E−01	1.6570E−01	1.2160E−01	1.4700E−01	1.4010E−01	1.7180E−01
	IR	49.37%	27.45%	17.39%	33.67%	46.72%	33.23%	45.74%	32.48%
HE2	NSGA-II	2.4230E−01	1.6040E−01	1.6120E−01	3.6060E−01	2.1240E−01	2.6490E−01	2.3260E−01	1.6830E−01
	TC_NSGAII	2.7530E−01	2.7400E−01	2.7530E−01	2.9550E−01	1.3790E−01	1.3280E−01	2.8100E−01	2.7360E−01
	IR	− 13.60%	− 70.85%	− 70.80%	18.05%	35.09%	49.89%	− 20.79%	− 62.53%
HE7	NSGA-II	2.6900E−01	2.7450E−01	2.7110E−01	2.7100E−01	2.4790E−01	2.5170E−01	2.6360E−01	2.6420E−01
	TC_NSGAII	8.0600E−02	6.7000E−02	7.4500E−02	6.0900E−02	5.1400E−02	4.3500E−02	6.3400E−02	6.6400E−02
	IR	70.05%	75.59%	72.52%	77.53%	79.26%	82.70%	75.97%	74.85%
HE9	NSGA-II	2.8900E−01	2.8420E−01	2.8800E−01	2.8540E−01	2.4990E−01	2.5340E−01	2.8470E−01	2.7640E−01
	TC_NSGAII	4.6510E−01	4.5440E−01	4.2360E−01	4.4620E−01	3.4550E−01	3.4670E−01	4.0260E−01	4.3550E−01
	IR	− 60.93%	− 59.90%	− 47.10%	− 56.37%	− 38.23%	− 36.79%	− 41.41%	− 57.55%

Funding The authors confirm that there is no source of funding for this study.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

Human participants and/or animals None.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Luna F, Zapata-Cano PH, González-Macías JC, Valenzuela-Valdés JF (2020) Approaching the cell switch-off problem in 5G ultra-dense networks with dynamic multi-objective optimization. *Futur Gener Comput Syst* 110:876–891
- Ribeiro VHA, Moritz S, Rehbach F, Reynoso-Meza G (2020) A novel dynamic multi-criteria ensemble selection mechanism applied to drinking water quality anomaly detection. *Sci Total Environ* 749:142368
- Li J, Xu A, Zang X (2020) Simulation-based solution for a dynamic multi-crane-scheduling problem in a steelmaking shop. *Int J Prod Res* 58(22):6970–6984
- Kim R, Lima FV (2020) A Tchebycheff-based multi-objective combined with a PSO–SQP dynamic real-time optimization framework for cycling energy systems. *Chem Eng Res Des* 156:180–194
- Feng Y, Wang G-G, Dong J, Wang L (2018) Opposition-based learning monarch butterfly optimization with Gaussian perturbation for large-scale 0–1 knapsack problem. *Comput Elect Eng Jpn* 67:454–468
- Muruganantham A, Tan KC, Vadakkepat P (2015) Evolutionary dynamic multiobjective optimization via Kalman filter prediction. *IEEE Trans Cybern* 46(12):2862–2873
- Zou F, Yen GG, Tang L (2020) A knee-guided prediction approach for dynamic multi-objective optimization. *Inf Sci* 509:193–209
- Jiang S, Yang S (2016) A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization. *IEEE Trans Evol Comput* 21(1):65–82
- Wang G-G, Tan Y (2017) Improving metaheuristic algorithms with information feedback models. *IEEE Trans Cybern* 49(2):542–555
- Gao D, Wang G-G, Pedrycz W (2020) Solving fuzzy job-shop scheduling problem using DE algorithm improved by a selection mechanism. *IEEE Trans Fuzzy Syst* 28(12):3265–3275
- Simões A, Costa E (2014) Prediction in evolutionary algorithms for dynamic environments. *Soft Comput* 18(8):1471–1497
- Jiang M, Qiu L, Huang Z, Yen GG (2018) Dynamic multi-objective estimation of distribution algorithm based on domain adaptation and nonparametric estimation. *Inf Sci* 435:203–223
- Zhenzhong W, Jiang M, Xing G, Liang F, Weizhen H, Tan KC (2019) Evolutionary dynamic multi-objective optimization via regression transfer learning. In: 2019 IEEE Symposium Series on Computational Intelligence (SSCI 2019). Xiamen, December 6–9, IEEE, 2019, 2375–2381
- Jiang M, Huang Z, Qiu L, Huang W, Yen GG (2017) Transfer learning-based dynamic multiobjective optimization algorithms. *IEEE Trans Evol Comput* 22(4):501–514
- Jiang M, Zhenzhong W, Haokai H (2021) Knee point based imbalanced transfer learning for dynamic multi-objective optimization. In: IEEE Transactions on Evolutionary Computation, to be published. [online]. <https://doi.org/10.1109/TEVC.2020.3004027>. Available at <https://ieeexplore.ieee.org/abstract/document/9122031>
- Jiang M, Wang Z, Guo S, Gao X, Tan KC (2021) Individual-based transfer learning for dynamic multiobjective optimization. In: IEEE Transactions on Cybernetics, to be published. [online]. <https://doi.org/10.1109/TCYB.2020.3017049>. Available at <https://ieeexplore.ieee.org/abstract/document/9199822>
- Jiang M, Wang Z, Qiu L, Guo S, Gao X, Tan KC (2021) A fast dynamic evolutionary multiobjective algorithm via manifold transfer learning. In: IEEE Transactions on Cybernetics, to be published. [online]. <https://doi.org/10.1109/TCYB.2020.2989465>. Available at <https://ieeexplore.ieee.org/abstract/document/9097186>
- Feng L, Zhou W, Liu W, Ong Y-S, Tan KC (2021) Solving dynamic multiobjective problem via autoencoding evolutionary search. In: IEEE Transactions on Cybernetics, to be published. [online]. <https://doi.org/10.1109/TCYB.2020.3017017>. Available at <https://ieeexplore.ieee.org/abstract/document/9210737>
- Deb K, Jain H (2013) An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. *IEEE Trans Evol Comput* 18(4):577–601
- Pan SJ, Yang Q (2009) A survey on transfer learning. *IEEE Trans Knowl Data Eng* 22(10):1345–1359
- Azzouz R, Bechikh S, Said LB (2017) Dynamic multi-objective optimization using evolutionary algorithms: a survey. *Recent Adv Evoluti Multiobject Optim* 2017:31–70 ((Springer))
- Deb K, Karthik S (2007) Dynamic multi-objective optimization and decision-making using modified NSGA-II: a case study on hydro-thermal power scheduling. *Int Conf Evolut Multicrit Optim* 2007:803–817 ((Springer))
- Grefenstette JJ (1992) Genetic algorithms for changing environments. *Ppsn. Citeseer*. 137–144
- Chen H, Li M, Chen X (2009) Using diversity as an additional-objective in dynamic multi-objective optimization algorithms. In: 2009 Second International Symposium on Electronic Commerce and Security (ISECS 2009), Nanchang, March 22–24, 2009. IEEE, 484–487
- Zhang Q, Yang S, Jiang S, Wang R, Li X (2019) Novel prediction strategies for dynamic multiobjective optimization. *IEEE Trans Evol Comput* 24(2):260–274
- Rong M, Gong D, Zhang Y, Jin Y, Pedrycz W (2018) Multidirectional prediction approach for dynamic multiobjective optimization problems. *IEEE Trans Cybern* 49(9):3362–3374
- Rong M, Gong D, Pedrycz W, Wang L (2019) A multimodel prediction method for dynamic multiobjective evolutionary optimization. *IEEE Trans Evol Comput* 24(2):290–304
- Wang Y, Li B (2009) Investigation of memory-based multi-objective optimization evolutionary algorithm in dynamic environment. In: 2009 IEEE Congress on Evolutionary Computation (CEC 2009), Trondheim, March 18–21, IEEE, 630–637

29. Zheng B (2007) A new dynamic multi-objective optimization evolutionary algorithm. In: Third International Conference on Natural Computation (ICNC 2007). IEEE, 565–570
30. Camara M, Ortega J, Toro FJ (2007) Parallel processing for multi-objective optimization in dynamic environments. In: 2007 IEEE International Parallel and Distributed Processing Symposium, Long Beach, March 26–30, IEEE, 1–8
31. Eaton J, Yang S, Gongora M (2017) Ant colony optimization for simulated dynamic multi-objective railway junction rescheduling. *IEEE Trans Intell Transp Syst* 18(11):2980–2992
32. Wang D-J, Liu F, Jin Y (2017) A multi-objective evolutionary algorithm guided by directed search for dynamic scheduling. *Comput Oper Res Lett* 79:279–290
33. Nguyen S, Zhang M, Johnston M, Tan KC (2013) Automatic design of scheduling policies for dynamic multi-objective job shop scheduling via cooperative coevolution genetic programming. *IEEE Trans Evol Comput* 18(2):193–208
34. Guo Y-N, Cheng J, Luo S, Gong D, Xue Y (2017) Robust dynamic multi-objective vehicle routing optimization method. *IEEE/ACM Trans Comput Biol* 15(6):1891–1903
35. Bozorgi-Amiri A, Khorsi M (2016) A dynamic multi-objective location–routing model for relief logistic planning under uncertainty on demand, travel time, and cost parameters. *Int J Adv Manufact Technol* 85(5–8):1633–1648
36. Qiao J, Zhang W (2018) Applications. Dynamic multi-objective optimization control for wastewater treatment process. *Neural Computing* 29(11):1261–1271
37. Iqbal M, Xue B, Al-Sahaf H, Zhang M (2017) Cross-domain reuse of extracted knowledge in genetic programming for image classification. *IEEE Trans Evol Comput* 21(4):569–587
38. Moshaiov A, Tal A (2014) Family bootstrapping: a genetic transfer learning approach for onsetting the evolution for a set of related robotic tasks. In: 2014 IEEE Congress on Evolutionary Computation (CEC 2014), Beijing, July 6–11, 2014. IEEE, 2801–2808
39. Pan SJ, Tsang IW, Kwok JT, Yang Q (2010) Domain adaptation via transfer component analysis. *IEEE Trans Neural Netw* 22(2):199–210
40. Helbig M, Engelbrecht A (2015) Benchmark functions for CEC 2015 special session and competition on dynamic multi-objective optimization. University of Pretoria, Pretoria
41. Zhou A, Jin Y, Zhang Q (2013) A population prediction strategy for evolutionary dynamic multiobjective optimization. *IEEE Trans Cybern* 44(1):40–53
42. Cao L, Xu L, Goodman ED, Bao C, Zhu S (2019) Evolutionary dynamic multiobjective optimization assisted by a support vector regression predictor. *IEEE Trans Evol Comput* 24(2):305–319

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.