Dynamic Multi-Objective Evolutionary Algorithm Based on Learning-Evolved Knowledge

1st Biao Xu, Chenbo Meng, Gejie Rang, Wenji Li
dept. College of Engineering
Shantou University
Shantou, China
xubiao@stu.edu.cn

2nd Given Name Surname
dept. Institute of Advanced Study
University of Electronic Science and Technology of China (Shenzhen)
Shenzhen, China
zhun_fan@126.com

***Abstract*—Dynamic multi-objective optimization problems (DMOPs) pose significant challenges for traditional evolutionary algorithms due to the continuous evolution of their Pareto-optimal sets (PSs) and Pareto frontiers (PFs). Learning-based approaches demonstrate great potential for rapidly tracking changing Pareto optimal solution sets while maintaining population diversity. However, existing methods for learning evolutionary knowledge fail to adequately account for problem-specific variations. Directly applying historical evolutionary knowledge to new problems may yield suboptimal results when problems undergo significant changes. To overcome this limitation, this study proposes a novel dynamic multi-objective evolutionary optimization algorithm comprising a transfer learning model and a multi-layer perceptron (MLP) model. First, individuals are improved based on a selection mechanism. If the transfer learning model is selected, historical evolutionary knowledge is transferred to the new environment, mitigating knowledge obsolescence due to problem differences and creating more promising individuals. If the multilayer perceptron model is chosen, it enhances individual quality by learning evolutionary process knowledge specific to the current problem. Population iteration is then completed through differential evolution operators and environmental selection. Our proposed algorithm underwent rigorous testing using benchmark functions, with results demonstrating superior performance compared to existing state-of-the-art algorithms.**

***Index Terms*—Dynamic multi-objective; Evolutionary algorithms; Transfer learning; Multilayer perceptron**

I. Introduction

Multi-objective optimization problems (MOPs) involve multiple conflicting objectives. Dynamic multi-objective optimization problems (DMOPs), however, not only feature conflicting objectives but also objective functions that change dynamically over time [1]. In the real world, problems such as path planning [2], securities portfolio selection [3], software project task allocation [4], and financial asset allocation [3] all fall under the category of dynamic multi-objective optimization. Due to their significant practical importance, these problems have attracted considerable attention from researchers.

Traditional multi-objective evolutionary algorithms (MOEAs) face significant challenges when addressing dynamic multi-objective optimization problems. Due to the dynamic nature of the problem, the Pareto optimal solution set evolves accordingly. However, after convergence, traditional MOEAs often exhibit reduced diversity and weakened exploration capabilities, making it difficult to rapidly adapt to new environmental changes. Therefore, it is necessary to introduce a change-responsive mechanism to swiftly track the evolving Pareto optimal solution set [5].

In dynamic multi-objective optimization problems, historical problems play a crucial role when addressing new challenges due to the inherent interconnectivity among static subproblems. The key to dynamic response mechanisms lies in effectively leveraging historical information to generate high-quality initial populations. Over the past decade, researchers have proposed numerous learning-based prediction methods. These approaches utilize the PS of historical problems as training data to identify underlying patterns of change. They then predict the Pareto-optimal set (PS) of new problems based on these patterns, employing the predictions as the initial population. Although prediction-based methods have achieved success in many problems, they still face the following challenges:

1. A single prediction method cannot solve all problems. Prediction methods rely on predictive models, and due to the diverse nature of problems, a single model is unlikely to be universally applicable.2. The quality of historical solutions does not support prediction. In prediction-based methods, the PS used as model data is obtained through algorithms rather than actual PS, potentially introducing errors that may result in the final predicted population failing to meet requirements.3. Predictive methods require additional time investment. The prediction process often incurs higher time costs due to the need for model training.

Due to the inherent correlation between adjacent problems, learning evolutionary knowledge from historical problems and applying it to solve new problems can accelerate population convergence while maintaining diversity. However, existing methods based on learning evolutionary knowledge directly apply historical evolutionary insights to new problem-solving processes without accounting for problem-specific differences. While adjacent problems share similarities, they are not identical, leading to variations in population evolution. Directly applying learned historical evolutionary knowledge to new problems may introduce bias. This is particularly true for problems with significant variations, where adjacent problems exhibit weak similarity. Past evolutionary knowledge

may prove insufficient for current problems and could even misguide the evolutionary direction.

To address the aforementioned issues, this chapter proposes a Learning-Evolved Knowledge-based Dynamic Multi-Objective Evolutionary Algorithm (LEK). The algorithm comprises a transfer learning model and a multi-layer perceptron model. Considering that adjacent problems in dynamic multi-objective optimization share similar evolutionary trajectories, the transfer learning model transfers historical evolutionary knowledge to the solution process in new environments, thereby mitigating knowledge obsolescence caused by problem variations. Recognizing that evolutionary trajectories within populations may also exhibit variations, the multilayer perceptron model learns evolutionary knowledge specific to the current problem. During solution refinement, this model identifies population evolution directions, guiding subsequent evolutionary steps—an aspect overlooked by existing dynamic multi-objective algorithms. By fully leveraging this knowledge, the proposed method significantly enhances the quality of offspring populations and accelerates convergence speed.

The main contributions of this study are as follows.

The remainder of this paper is organized as follows. Section II reviews related work. Section III details the algorithm's specific implementation. Section IV presents the experimental setup and analysis of results. Finally Section V summarizes our contributions and proposes future research directions.

II. Related work

A. Dynamic Multi-Objective Optimization Problem

Without loss of generality, an unconstrained dynamic multi-objective optimization problem can be expressed in the following form:

$$\min_{x\in\Omega} F(x,t) = (f_1(x,t), f_2(x,t), \ldots, f_m(x,t))^T \quad (1)$$

where x is an n-dimensional decision vector, m represents the number of objectives, and t is a time parameter, Ω represents the decision space, and F denotes the objective vector composed of m time-varying objective functions.Typically, the mathematical definition of t is as follows:

$$t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \quad (2)$$

where τ_t is the frequency of environmental change, i.e., the time interval during which the environment remains unchanged. n_t is the intensity of environmental change. τ is the iteration counter.

B. Dynamic Multi-Objective Evolutionary Optimization

Typically, a complete dynamic multi-objective evolutionary algorithm primarily consists of three components: environmental change detection, change response strategies, and static multi-objective evolutionary algorithms [6]. Currently, most research focuses on designing change response strategies, which can be broadly categorized into three types: diversity-based methods, learning-based prediction methods, and memory-based methods.

When addressing dynamic optimization problems, enhancing the effectiveness of multi-objective optimization methods hinges on maintaining population diversity. Under diversity-based response strategies, a certain number of random or mutated individuals are introduced to better adapt to evolving optimization objectives. These strategies demonstrate exceptional diversity preservation capabilities, ensuring populations sustain their richness throughout the evolutionary process [7] [8]. Deb et al. [9] proposed two diversity-based dynamic NSGA-II variants: DNSGAII-A and DNSGAII-B. Upon detecting environmental changes, DNSGAII-A randomly generates individuals to replace those in the original population, while DNSGAII-B introduces solutions with high mutation rates to enhance diversity.

Prediction-based methods accelerate population convergence by leveraging historical optimal solutions to learn predictions for new time steps, enabling rapid tracking of evolving Pareto optimal sets. Many predictive learning methods employ simple linear models. For instance, Xu et al. [10] proposed a co-evolutionary algorithm that divides decision variables into time-dependent and time-independent categories. It then uses differential prediction and Cauchy mutation to forecast the initial population at the next time step, thereby accelerating response to the problem. Rong et al. [11] proposed a multi-directional prediction strategy that clusters the population into several groups of representative individuals, with the number of clusters adjusted according to the intensity of environmental change. Different clusters employ distinct linear prediction methods.

Memory-based methods store optimal solutions to historical problems, enabling efficient tracking of evolving optimal solutions through reuse of past optimal outcomes. The memory strategy demonstrates significant effectiveness in preserving population diversity due to its capability to store historical optimal solutions. Azzouz et al. [12] proposed an adaptive hybrid population management strategy that integrates memory-based local search with random strategies, effectively addressing certain DMOPs with highly dynamic characteristics. Subsequently, Chen et al. [13] designed a dynamic dual-file evolutionary algorithm. The two files focus on convergence and diversity respectively, achieving complementarity through a mating selection mechanism to enhance overall algorithm performance.

In recent years, DMOEAs have garnered significant attention in learning-based evolutionary knowledge methods. Yu et al. [14] proposed a framework based on historical evolutionary learning to assist static optimizers in fully leveraging historical evolutionary directions and the distribution of Pareto optimal solutions. Specifically, two novel models were designed: offspring generation and environment selection. To enhance offspring contribution, a direction-oriented model was established based on historical evolutionary trend directions. Additionally, to improve the robustness of static optimizers and mitigate interference from misleading solutions, a manifold-corrected

model was proposed. This model generates promising individuals by learning the distribution of historical Pareto optimal solutions. Zhao et al. [15] introduced a knowledge learning strategy for change response in dynamic multi-objective optimization. In the proposed strategy, responses to changes are achieved by learning from historical search processes. They introduce a method to extract knowledge from past search experiences. The extracted knowledge accelerates population convergence and increases population diversity.

The two aforementioned methods directly apply learned or extracted knowledge to the current problem, aiming to generate more promising offspring. However, since adjacent problems in dynamic multi-objective optimization cannot be entirely identical and exhibit certain differences, past knowledge may not be fully applicable to the current problem. For problems with significant variability, directly applying past knowledge to the existing problem may even misguide the evolutionary direction. To address these issues, this paper proposes a novel dynamic multi-objective evolutionary algorithm based on learning and evolving knowledge.

III. The Proposed Algorithm

This section proposes a dynamic multi-objective evolutionary algorithm based on learning evolutionary knowledge (LEK). Its overall framework pseudocode is presented in Algorithm 1.

Algorithm 1 Framework of LEK

Require: N (Population size)
Ensure: A series of approximations PS

$t = 0$
Randomly initialize a population p^t
while Does not satisfy the termination conditions **do**
  if Detected environmental changes **then**
    $PS_g \leftarrow PS_u \cup PS^t$
    $t = t + 1$
    $g = 0$
    $P_g^t \leftarrow$ Change Response Mechanism
  end if
  for $i = 1$ **to** n **do**
    $R = \text{rand}(i)$
    if $R < q_i$ **then**
      $P_g^t \leftarrow$ Transfer learning model $(P_g^t, P_{g-1}^t, P_g^{t-1}, P_{g-1}^{t-1}, PS^{t-1})$
    end if
    if $0.5 < R < m_i$ **then**
      $P_g^t \leftarrow$ Multi-Layer Perceptron Model (P_g^t)
    end if
    $\text{off}_g^t(i) \leftarrow$ Differential Evolution Operator
    $g = g + 1$
    $P_g^t \leftarrow$ Environmental Selection $(P_{g-1}^t, \text{off}_{g-1}^t(i))$
  end for
end while
return PS

A. Framework of LEK

In the proposed algorithm, the MOEA/D-based framework activates a change response mechanism to generate an initial population whenever environmental changes are detected. Since the innovation of this chapter's algorithm focuses on the static optimization phase, any existing method can be adopted for the change response mechanism. If no environmental change is detected, the population is optimized using learned evolutionary knowledge. For each individual in the population, a random number R is first generated. If R is less than q_i, the individual is improved using the transfer learning model. by leveraging evolutionary knowledge from historical problems to enhance its potential. If R is greater than 0.5 and less than m_i, the multi-layer perceptron model is applied to improve the individual's quality by learning evolutionary experience from the current problem. Finally, the population iteration is completed through differential evolution operators and environmental selection. It is worth noting that the initial values of q_i and m_i are hyperparameters, and their magnitudes will change based on a feedback mechanism, with specific details provided later in the text.

B. Subspace Distribution Alignment Based between Infinite Subspaces

Subspace Distribution Alignment between Infinite Subspaces (SDA-IS) is a domain-adaptive method capable of uncovering correlations between search spaces in historical and current environments. It constructs a geodesic flow path from the source subspace (historical search space) to a subspace of the target (current search space), employing a kernel trick to integrate over an infinite number of subspaces along this path. Subsequently, the distribution of subspaces is aligned along each segment of the geodesic flow kernel.

Assume $D_S = \{X_{S_1}, ..., X_{S_N}\} \subseteq R_S$ is the source data, and the target data is represented as $D_T = \{X_{T_1}, .., X_{T_N}\} \subseteq R_T$. First, Principal Component Analysis (PCA) is employed to extract key features from the source and target data, forming corresponding subspaces denoted as $S_S \in R^{n \times d}$ and $S_T \in R^{n \times d}$, respectively. Here, n represents the number of decision variables, and d denotes the dimension of the subspace. Thus, the mapping matrix M_s can be expressed as follows:

$$M_S = [S_S U_1 R_S U_2] \begin{bmatrix} 12 \\ 23 \end{bmatrix} A_{TS} [S_S U_1 R_S U_2]^T \qquad (3)$$

C. Historical Evolution and Knowledge Transfer

The process of historical evolutionary knowledge transfer is illustrated in Fig. 1. Our proposed algorithm is built upon the MOEA/D framework, which serves as a static optimizer establishing one-to-one correspondence between weight vectors and individuals. Assuming $P_g^{t-1} = (X_{g_1}^{t-1}, X_{g_2}^{t-1}, \ldots, X_{g_N}^{t-1})$ represents the population at generation g of time $t-1$, where N denotes the population size and $X_{g_i}^{t-1}$ represents the individual associated with the i-th weight vector, and $PS^{t-1} = (PS_1^{t-1}, PS_2^{t-1}, \ldots, PS_N^{t-1})$ denotes the final optimized population at time $t-1$, then $V_{g_i}^{t-1} = PS_i^{t-1} - X_{g_i}^{t-1}$ represents

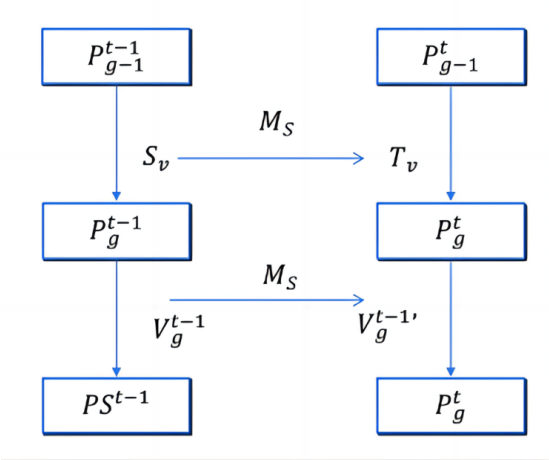

Fig. 1. Transfer Learning Process Diagram

the evolutionary direction of the individual associated with the i-th weight vector at generation g of time $t-1$. Consequently, $V_g^{t-1} = (V_{g_1}^{t-1}, V_{g_2}^{t-1}, \ldots, V_{g_N}^{t-1})$ represents the evolutionary direction of the population at generation g of time $t-1$, which can be regarded as historical knowledge.

To make the historical evolutionary knowledge more applicable to the new environment, we employ the SDA-IS method for domain adaptation. Assuming the evolutionary generation in the new environment is g, we consider the evolutionary direction from generation $g-1$ to generation g at time $t-1$ as the source domain data, i.e., $S_v = P_g^{t-1} - P_{g-1}^{t-1}$, and the evolutionary direction from generation $g-1$ to generation g at time t as the target domain data, i.e., $T_v = P_g^t - P_{g-1}^t$. Based on S_v and T_v, we construct a mapping matrix M_s. Subsequently, the historical knowledge V_g^{t-1} is transferred to the new environment through the mapping matrix M_s, denoted as $V_g^{t-1'}$.

$$V_g^{t-1'} = V_g^{t-1} \cdot M_s \tag{4}$$

Finally, individuals in the current population have the potential to generate more promising offspring through the following formula:

$$X_{g_i}^t = X_{g_i}^t + V_{g_i}^{t-1'} \quad (i = 1, 2, \ldots, N) \tag{5}$$

It is worth noting that, as can be seen from the overall framework presented in the previous subsection, for each individual $X_{g_i}^t$ in the current population, whether to utilize the transferred historical knowledge is highly dependent on the parameter q_i. Since the effectiveness of historical evolutionary knowledge may vary significantly for individuals associated with different weight vectors and at different evolutionary stages, we propose an adaptive adjustment strategy to dynamically modify the value of q_i, thereby controlling the probability of each individual using the transfer learning model.Specifically, after the individual $X_{g_i}^t$ associated with the i-th weight vector employs the transfer learning model, if the distance between the offspring $Off_{g_i}^t$ generated by the differential operator and the individual $X_{(g-1)i}^t$ from generation $g-1$ is greater than the distance between $X_{(g-1)i}^t$ and $X_{(g-2)i}^t$, it indicates that the historical evolutionary knowledge is beneficial for the evolution of the current individual. In this case, q_i is increased by 0.05. Conversely, q_i is decreased by 0.05.

D. Multi-Layer Perceptron

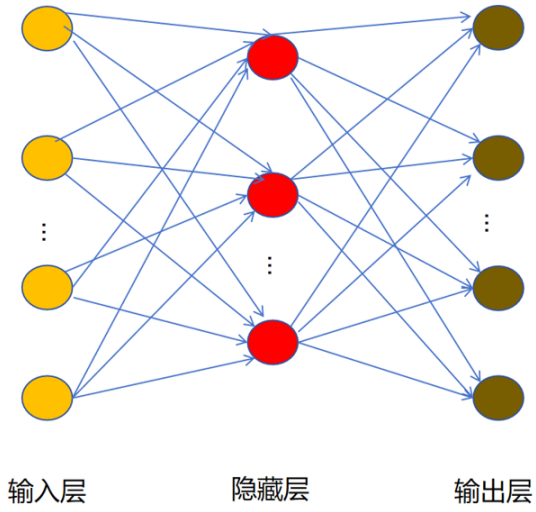

Fig. 2. Transfer Learning Process Diagram

Artificial neural networks are a well-known machine learning method for data segmentation, capable of simulating the neural systems of the human brain. Structurally, a neural network consists of numerous neurons organized in layers. One of the simplest neural networks is the feedforward neural network, also known as a multi-layer perceptron (MLP), where each layer of neurons receives input from the preceding layer. Mathematically, each neuron sums its inputs to compute a weighted linear combination, then further modifies this result using a nonlinear activation function to produce an output. As shown in Fig. 2, an MLP features an input layer and an output layer to connect with the external world.

Furthermore, a Multilayer Perceptron (MLP) typically contains one or more hidden layers of neurons to extract important features from the input data. The output value of each node in a layer is calculated as follows:

$$h_l = \varphi\left(\sum_{i=1}^{m} W_i^l I_i + \beta_i\right) \quad l = (1, 2, \ldots, j) \tag{6}$$

where I_i is the input vector, W_i^l is the connection weight between I_i and node l, m is the number of input vectors, j is the number of nodes in the layer, β_i is the bias of the i-th node, and φ is an activation function, such as the standard logistic sigmoid function:

$$\varphi(x) = \frac{1}{1 + e^{-x}} \tag{7}$$

The MLP operation can be divided into two phases: training and inference. During the training phase, a set of training

samples is used to iteratively update the parameters (i.e., weights and biases) based on gradient descent via backpropagation. In the inference phase, a trained MLP model produces outputs according to the input values. The training of an MLP is accomplished by modifying the weights and biases over successive iterations to minimize the error. The objective of training is to minimize the Mean Squared Error (MSE):

$$L_{MSE} = \frac{1}{M}\sum_{i=1}^{M}(y_i^* - y_i)^2 \tag{8}$$

where y_i^* and y_i are the target output and predicted output of the i-th training iteration, respectively, and M is the total number of training iterations.

Algorithm 2 Learning Evolutionary Process Knowledge

Require: Interval generations interval, current generation g, initial population P_0^t at time t
Ensure: Model-accelerated population P_g^t

```
g ← 0
X_train ← P_0^t
while environment change not detected do
    if g ≥ interval then
        if mod(g, interval) == 0 then
            Y_train ← P_g^t
            net ← MLP-train(X_train, Y_train)
            X_train ← Y_train
        end if
        P_g^t ← MLP-prediction(P_g^t, net)
    end if
    g ← g + 1
end while
```

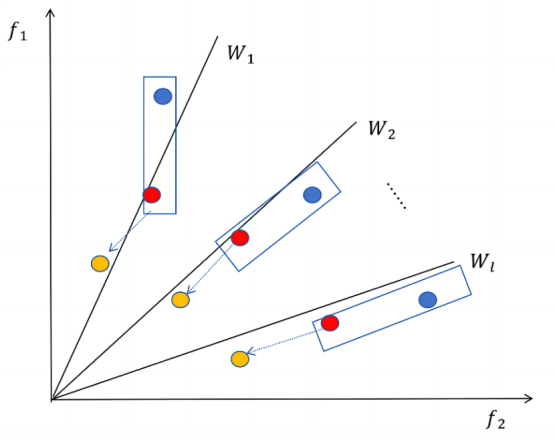

Fig. 3. Transfer Learning Process Diagram

E. Learning about the evolutionary process

In solving the current problem, similarly, since we employ MOEA/D as the static optimizer, let $X_{g_i}^t$ represent the individual associated with the i-th weight vector in the population at generation g of time t, and $X_{(g-\text{interval})_i}^t$ denote the individual from interval generations prior to $X_{g_i}^t$ (where interval is a hyperparameter). Since both are associated with the same weight vector, they maintain a corresponding relationship. The difference $X_{g_i}^t - X_{(g-\text{interval})_i}^t$ can be viewed as the evolutionary direction of the i-th individual over interval generations, with $X_{g_i}^t$ being the evolutionary target of $X_{(g-\text{interval})_i}^t$.

During MLP training, we use $X_{(g-\text{interval})_i}^t$ as input and $X_{g_i}^t$ as the label, resulting in N training data samples $(X_{(g-\text{interval})_i}^t, X_{g_i}^t)$ for $i = 1, \ldots, N$, where N represents the population size. The trained MLP can extract the population's advancement direction and learn the evolutionary process knowledge. After training is completed, when $X_{g_i}^t$ is input to the MLP, the output represents an individual with higher quality and greater potential than $X_{g_i}^t$.

Fig. 3 illustrates the acquisition of training data and the prediction process of the MLP model, where blue and red circles represent individuals from generation $g - \text{interval}$ and generation g, respectively, and yellow circles indicate the model's output individuals. Algorithm 2 provides the pseudo-code for this process. As shown in the pseudo-code, to conserve computational resources and reduce time consumption, the MLP model is trained only once every interval generations.

It is worth mentioning that in Algorithm 1, the value of m_i directly determines whether individual $X_{g_i}^t$ uses the MLP model. Similar to the transfer learning model, the initial value of m_i is a hyperparameter, and it employs the same feedback mechanism as q_i for dynamic adjustment.

IV. EXPERIMENTAL STUDY

A. Test problems and performance indicators

In this study, DF benchmark components were selected as test problems for the algorithm. These components encompass diverse attributes and effectively represent various real-world scenarios. Regarding performance metrics, we adopted two measures:

1) The Inverted Generational Distance (IGD) [16] is widely used to evaluate the convergence and diversity of solutions obtained by algorithms. At time t, it is defined as follows:

$$\text{IGD}(PF_t^*, PF_t) = \frac{\sum_{p \in PF_t^*} d(p, PF_t)}{|PF_t^*|} \tag{9}$$

where PF_t^* represents a set of points uniformly sampled from the true Pareto front, PF_t denotes the approximate Pareto front obtained by the test algorithm, and $d(p, PF_t)$ represents the minimum distance between point p and all points in PF_t.
To validate the performance of dynamic multi-objective algorithms, the IGD metric is extended to propose the Mean Inverted Generational Distance (MIGD) performance metric:

$$\text{MIGD} = \frac{\sum_{t \in T} \text{IGD}(PF_t^*, PF_t)}{|T|} \tag{10}$$

where T is a set of discrete instances, and $|T|$ represents the number of changes in a single run. In this paper, we uniformly sample 1000 points from PF_t^* to calculate the IGD values.

2) Hypervolume (HV) evaluates the comprehensive performance of an algorithm by calculating the volume of the space enclosed by the obtained solution set and a reference point in the objective space. It is defined as:

$$\text{HV} = \bigcup_{i=1}^{|P^t|} \text{volume}(u_i) \quad u_i \in P^t \tag{11}$$

where $|P^t|$ is the cardinality of the Pareto front obtained at time t, and volume(u_i) represents the volume of the hypercube formed by the i-th point u_i and the reference point z_{ref}^t. In our experiments, z_{ref}^t is set to $(z_1^t \times 1.1, z_2^t \times 1.1, \ldots, z_m^t \times 1.1)$, where z_j^t is the maximum value of the j-th objective in the true Pareto front at time t, and m is the number of objectives.

Similarly, the Mean Hypervolume (MHV) is defined as:

$$\text{MHV} = \frac{\sum_{t \in T} \text{HV}}{|T|} \tag{12}$$

B. Comparison algorithms and parameter settings

To determine how the LEK algorithm proposed in this paper compares with other algorithms, we selected HEL, KL, DNSGAII-B, and PPS as benchmarking algorithms. Among these, HEL and KL are two existing dynamic multi-objective optimization algorithms based on learning evolutionary knowledge, while DNSGAII-B and PPS represent classical diversity-based and prediction-based methods, respectively. To ensure a fair comparison, the testing environment and relevant parameter settings are as follows:

1) For bi-objective and tri-objective problems, the population size N is set to 100 and 105, respectively, and the dimension of decision variables n is set to 10.
2) For each problem, three sets of dynamic parameters are configured: (10, 5), (10, 10), and (10, 20). The number of environmental changes $T = 100$. To minimize the impact of static multi-objective algorithms on overall performance, 50 iterations are executed before the first environmental change occurs. Each algorithm is independently run 20 times for each problem instance.
3) To eliminate the influence of static optimizers on algorithm performance, with the exception of DNSGAII-B, all algorithms employ MOEA/D as the static optimizer. The differential evolution (DE) operator is selected as the crossover operator with parameters $CR = 1$ and $F = 0.5$. The polynomial mutation operator is used with parameters $proM = 0.1$ and $disM = 20$. The dynamic response strategies of LEK, HEL, and KL remain consistent with DNSGAII-B, where 80% of individuals are preserved for the initial population in the new environment upon detecting environmental changes, while the remaining 20% are generated through high mutation. For LEK, the generation interval interval $= 3$, and the initial values of q_i and m_i are set to 0.3 and 0.8, respectively. The multilayer perceptron is configured with one hidden layer, and other parameters of comparative algorithms remain consistent with their original publications.

C. Parameter sensitivity analysis

The following two tables present the experimental results of our proposed algorithm LEK and four comparison algorithms on the performance metrics $MIGD$ and MHV. For clarity, the best values are highlighted. The symbols "+", "=", and "-" in the tables indicate that LEK significantly outperforms, equals, or underperforms the comparison algorithms, respectively. The Wilcoxon signed-rank test at the 0.05 significance level was conducted to demonstrate significant differences between the results.

Table I clearly demonstrates that LEK exhibits significant advantages over other algorithms in the $MIGD$ metric. Across 42 test problem configurations, LEK achieved the best values in 37 cases. This sufficiently illustrates that our proposed algorithm enables rapid population convergence while maintaining diversity. Through the transfer learning model and the multilayer perceptron model, LEK not only learns evolutionary knowledge from historical problems but also acquires knowledge about the evolutionary processes of current problems. Via a feedback mechanism, the algorithm fully leverages both types of knowledge. This substantially enhances the quality of the offspring population, enabling the population to approach the true PS of the problem more rapidly. Notably, compared to the other two algorithms—HEL and KL—that also learn evolutionary knowledge, LEK demonstrates substantial advantages across various dynamic environment settings. This further validates the effectiveness of our proposed dual-model approach in acquiring evolutionary knowledge. However, it is also noteworthy that KT-MOEAD outperformed our algorithms in terms of $MIGD$ on DF2 when the dynamic environment was set to (10, 20). This may be attributed to the relatively simple variation pattern in the DF2 problem, where prediction-based algorithms could accurately forecast the PS at new time steps, granting them a competitive edge. Similarly, under the dynamic environment (10, 20), DNSGAII-B performs well on DF4. This may stem from DF4's PS length and position varying over time, with PF segment length and curvature also being time-dependent. In later evolutionary stages, historical evolutionary knowledge exerts limited influence on the current problem, causing the algorithm to exhibit diminished performance as generations increase. On the DF11 problem, HEL outperformed our algorithms across all three dynamic environments. This demonstrates that HEL's directional guidance model and popular repair model effectively ensure population convergence and diversity on this problem, yielding strong performance. Beyond this, our algorithms achieved the best results on all other problems. This indicates that regardless of whether the problem is two-objective or three-objective, and regardless of the type of

TABLE I
MEAN AND STANDARD DEVIATION OF LEK AND THE COMPARISON ALGORITHM $MIGD$

Problems	(η_t, τ_t)	KT-MOEAD	DNSGAII-B	KL	HEL	LEK
DF1	(10,5)	0.1198(1.08E-02)+	0.2731(1.51E-02)+	0.1786(2.62E-02)+	0.3225(2.72E-02)+	**0.0443(3.71E-03)**
	(10,10)	0.0698(6.57E-03)+	0.0920(6.17E-03)+	0.1988(3.46E-02)+	0.1487(1.27E-02)+	**0.0137(6.12E-04)**
	(10,20)	0.0301(1.74E-03)+	0.0259(1.27E-03)+	0.1792(2.85E-02)+	0.0712(2.66E-03)+	**0.0071(1.32E-04)**
DF2	(10,5)	0.1267(8.45E-03)+	0.2556(1.34E-02)+	0.1978(2.20E-02)+	0.2382(1.14E-02)+	**0.1176(4.79E-03)**
	(10,10)	0.0703(5.83E-03)=	0.1642(6.06E-03)+	0.1819(2.74E-02)+	0.1592(7.62E-03)+	**0.0670(3.52E-03)**
	(10,20)	**0.0303(3.01E-03)**-	0.0956(5.16E-03)+	0.2008(2.14E-02)+	0.0984(3.33E-03)+	0.0330(2.64E-03)
DF3	(10,5)	0.2230(1.50E-02)+	0.4448(2.76E-02)+	0.4993(6.83E-02)+	0.3839(3.41E-02)+	**0.0362(1.49E-03)**
	(10,10)	0.1420(9.32E-03)+	0.3519(1.35E-02)+	0.5412(8.04E-02)+	0.2873(3.31E-02)+	**0.0162(1.21E-03)**
	(10,20)	0.0615(5.50E-03)+	0.2952(9.37E-03)+	0.5168(6.32E-02)+	0.2164(2.46E-02)+	**0.0092(2.69E-04)**
DF4	(10,5)	0.3016(1.14E-02)+	0.1900(1.40E-02)+	8.0913(8.60E-01)+	0.2529(1.22E-02)+	**0.1242(2.89E-03)**
	(10,10)	0.1561(8.64E-03)+	0.1321(6.34E-03)+	8.5332(9.80E-01)+	0.1663(8.37E-03)+	**0.1161(8.65E-04)**
	(10,20)	0.1152(1.45E-02)+	**0.0951(1.80E-03)**-	8.1889(1.09E+00)+	0.1163(3.54E-03)+	0.1131(2.23E-04)
DF5	(10,5)	0.1712(1.76E-02)+	0.3311(3.80E-02)+	1.4679(3.98E-01)+	0.2579(2.07E-02)+	**0.0326(1.62E-03)**
	(10,10)	0.0776(8.91E-03)+	0.0947(5.94E-03)+	1.3669(3.70E-01)+	0.1389(7.80E-03)+	**0.0113(4.09E-04)**
	(10,20)	0.0249(1.48E-03)	0.0263(1.54E-03)+	1.5487(2.08E-01)+	0.0758(4.62E-03)+	**0.0064(1.07E-04)**
DF6	(10,5)	1.9188(3.28E-01)+	8.3997(2.45E-01)+	8.6137(5.95E+00)+	2.3053(5.34E-01)+	**1.1126(1.51E-01)**
	(10,10)	1.1190(2.73E-01)+	3.7900(3.43E-01)+	8.5357(5.26E+00)+	1.2094(3.27E-01)+	**0.3379(8.40E-02)**
	(10,20)	0.6050(1.76E-01)+	1.4423(2.04E-01)+	7.4075(5.46E+00)+	0.8853(2.90E-01)+	**0.1866(6.71E-02)**
DF7	(10,5)	0.4879(6.68E-02)+	1.2785(7.66E-02)+	0.7626(5.54E-02)+	0.6969(1.92E-02)+	**0.2207(2.25E-02)**
	(10,10)	0.4437(6.76E-02)+	1.2229(9.43E-02)+	0.7496(8.09E-02)+	0.5727(4.42E-02)+	**0.1664(7.38E-03)**
	(10,20)	0.3255(5.06E-02)+	0.7830(1.08E-01)+	0.7731(7.09E-02)+	0.3901(4.97E-02)+	**0.1554(3.65E-03)**
DF8	(10,5)	0.2301(2.12E-02)+	0.0805(1.18E-03)+	0.7133(5.22E-02)+	0.1322(6.15E-03)+	**0.0249(1.52E-03)**
	(10,10)	0.2064(2.06E-02)+	0.0776(9.08E-04)+	0.7238(4.91E-02)+	0.1017(8.55E-03)+	**0.0193(8.77E-04)**
	(10,20)	0.1677(1.47E-02)+	0.0716(7.56E-04)+	0.7128(5.05E-02)+	0.0661(6.02E-03)+	**0.0167(7.45E-04)**
DF9	(10,5)	0.4166(2.93E-02)+	1.3024(7.83E-02)+	1.5967(3.95E-01)+	0.5246(7.40E-02)+	**0.2823(4.82E-02)**
	(10,10)	0.2694(2.26E-02)+	0.9049(6.50E-02)+	1.6515(4.17E-01)+	0.3577(3.29E-02)+	**0.0691(4.47E-03)**
	(10,20)	0.1855(1.72E-02)+	0.5443(2.79E-02)+	1.5614(3.61E-01)+	0.2477(1.22E-02)+	**0.0288(1.44E-03)**
DF10	(10,5)	0.4000(2.54E-02)+	0.1597(2.40E-03)+	0.6940(1.19E-01)+	0.1712(6.89E-03)+	**0.1496(4.82E-03)**
	(10,10)	0.3653(1.94E-02)+	0.1537(2.27E-03)+	0.6767(1.40E-01)+	0.1628(6.93E-03)+	**0.1098(1.95E-03)**
	(10,20)	0.3507(1.02E-02)+	0.1562(1.61E-03)+	0.7087(1.39E-01)+	0.1559(5.87E-03)+	**0.0997(1.30E-03)**
DF11	(10,5)	0.6241(1.26E-02)+	0.2133(9.63E-04)+	0.7181(5.61E-02)+	**0.1271(4.55E-03)**-	0.1941(4.34E-04)
	(10,10)	0.5586(1.19E-02)+	0.1949(6.49E-04)+	0.6818(5.47E-02)+	**0.1076(3.86E-03)**-	0.1822(1.61E-04)
	(10,20)	0.5096(6.58E-03)+	0.1996(4.77E-04)+	0.7171(4.22E-02)+	**0.0917(1.84E-03)**-	0.1839(9.71E-05)
DF12	(10,5)	0.3547(1.37E-02)+	0.2488(2.67E-02)+	1.2815(3.15E-01)+	0.3920(2.27E-02)+	**0.1151(4.33E-03)**
	(10,10)	0.2915(1.19E-02)+	0.1501(3.59E-02)+	1.2128(2.74E-01)+	0.3399(1.28E-02)+	**0.0920(9.52E-04)**
	(10,20)	0.2357(1.31E-02)+	0.1425(6.30E-02)+	1.2264(2.37E-01)+	0.3135(8.12E-02)+	**0.0823(7.17E-04)**
DF13	(10,5)	1.0276(8.22E-03)+	2.1437(9.75E-02)+	1.5441(1.14E-01)+	0.4305(1.45E-02)+	**0.3032(6.36E-03)**
	(10,10)	0.9637(8.55E-03)+	1.5821(6.05E-02)+	1.5541(1.49E-01)+	0.3470(8.31E-03)+	**0.2649(2.90E-03)**
	(10,20)	0.9177(1.03E-02)+	1.7027(9.26E-02)+	1.5593(1.40E-01)+	0.3080(6.67E-03)+	**0.2636(2.04E-03)**
DF14	(10,5)	0.4608(4.67E-02)+	1.2100(1.30E-01)+	0.9421(3.19E-01)+	0.1708(1.09E-02)+	**0.0753(1.79E-03)**
	(10,10)	0.4271(3.28E-02)+	1.1433(1.28E-01)+	0.8244(2.15E-01)+	0.1092(7.35E-03)+	**0.0578(7.69E-04)**
	(10,20)	0.3700(1.14E-02)+	0.9480(9.03E-02)+	0.8620(2.85E-01)+	0.0795(4.26E-03)+	**0.0541(3.28E-04)**
	+/=/−	40/1/1	41/0/1	42/0/0	39/0/3	

problem variation, LEK possesses strong competitiveness by fully learning evolutionary knowledge.

Table II presents the experimental results of various algorithms on the MHV performance metric. Although LEK performs less effectively on MHV than on MIGD, it still achieves the best results on over half of the problems, outperforming all other comparison algorithms. The primary reason for this phenomenon may be that our algorithm does not sufficiently cover the boundaries of PF edges, failing to adequately address the impact of special points such as boundary points on population evolution. This is an area where the algorithm aims to improve in the future.

TABLE II
MEAN AND STANDARD DEVIATION OF LEK AND THE COMPARISON ALGORITHM MHV

Problems	(η_t, τ_t)	KT-MOEAD	DNSGAII-B	KL	HEL	LEK
DF1	(10,5)	0.4809(1.08E-02)+	0.3542(1.51E-02)+	0.3930(2.62E-02)+	0.2970(2.72E-02)+	**0.5826(3.71E-03)**
	(10,10)	0.5510(6.57E-03)+	0.5156(6.17E-03)+	0.3617(3.46E-02)+	0.4481(1.27E-02)+	**0.6346(6.12E-04)**
	(10,20)	0.6102(1.74E-03)+	0.6154(1.27E-03)+	0.4000(2.85E-02)+	0.5410(2.66E-03)+	**0.6474(1.32E-04)**
DF2	(10,5)	0.6607(8.45E-03)+	0.5160(1.34E-02)+	0.5145(2.20E-02)+	0.5595(1.14E-02)+	**0.6997(4.79E-03)**
	(10,10)	0.7435(5.83E-03)+	0.6410(6.06E-03)+	0.5377(2.74E-02)+	0.6545(7.62E-03)+	**0.7703(3.52E-03)**
	(10,20)	0.8145(3.01E-03)+	0.7458(5.16E-03)+	0.5059(2.14E-02)+	0.7376(3.33E-03)+	**0.8195(2.64E-03)**
DF3	(10,5)	0.3540(1.14E-02)+	0.2366(1.24E-02)+	0.0978(9.78E-02)+	0.2725(1.49E-02)+	**0.5556(2.49E-03)**
	(10,10)	0.4412(1.03E-02)+	0.2984(7.79E-03)+	0.0723(7.23E-02)+	0.3400(2.07E-02)+	**0.5920(1.81E-03)**
	(10,20)	0.5294(7.46E-03)+	0.3440(6.20E-03)+	0.0729(7.29E-02)+	0.3961(1.64E-02)+	**0.6055(4.80E-04)**
DF4	(10,5)	4.8618(4.76E-02)-	**5.2875(4.79E-02)**-	0.0060(6.02E-03)+	0.7355(6.17E-03)+	0.8350(1.68E-03)
	(10,10)	5.3915(2.57E-02)-	**5.5346(1.95E-02)**-	0.0039(3.93E-03)+	0.7864(4.05E-03)+	0.8598(6.25E-04)
	(10,20)	5.6731(8.00E-03)-	**5.7031(6.30E-03)**-	0.0049(4.93E-03)+	0.8193(2.10E-03)+	0.8688(2.19E-04)
DF5	(10,5)	0.4800(1.45E-02)+	0.2980(2.18E-02)+	0.0182(1.82E-02)+	0.3698(1.38E-02)+	**0.6491(2.74E-03)**
	(10,10)	0.5886(7.64E-03)+	0.5457(8.76E-03)+	0.0168(1.68E-02)+	0.4973(9.65E-03)+	**0.6881(7.23E-04)**
	(10,20)	0.6639(2.29E-03)+	0.6623(2.59E-03)+	0.0192(1.92E-02)+	0.5860(5.68E-03)+	**0.6972(1.96E-04)**
DF6	(10,5)	0.3125(3.78E-02)=	0.0961(1.21E-02)+	0.0258(2.58E-02)+	0.2904(8.74E-02)+	**0.3498(7.86E-02)**
	(10,10)	0.4708(3.94E-02)+	0.2390(2.86E-02)+	0.0091(9.09E-03)+	0.4583(1.28E-01)+	**0.7043(4.82E-02)**
	(10,20)	0.6022(1.77E-02)+	0.4590(1.71E-02)+	0.0279(2.79E-02)+	0.5610(1.29E-01)+	**0.7822(4.00E-02)**
DF7	(10,5)	**1.6390(5.03E-02)**-	1.4790(5.34E-02)-	0.2471(2.47E-01)+	0.5972(6.05E-03)+	0.8605(4.91E-03)
	(10,10)	**1.8052(3.44E-02)**-	1.5244(4.24E-02)-	0.2316(2.32E-01)+	0.6457(1.94E-02)+	0.8845(2.82E-03)
	(10,20)	**1.9853(2.35E-02)**-	1.6849(3.94E-02)-	0.2339(2.34E-01)+	0.7299(1.97E-02)+	0.8934(1.90E-03)
DF8	(10,5)	0.5893(2.21E-02)+	0.6981(1.39E-03)+	0.0973(9.73E-02)+	0.6760(2.05E-03)+	**0.7433(1.79E-04)**
	(10,10)	0.6195(1.86E-02)+	0.7010(1.24E-03)+	0.0933(9.33E-02)+	0.6903(2.74E-03)+	**0.7462(1.13E-04)**
	(10,20)	0.6538(1.24E-02)+	0.7058(1.36E-03)+	0.1011(1.01E-01)+	0.7068(1.93E-03)+	**0.7476(7.74E-05)**
DF9	(10,5)	0.2849(1.35E-02)+	0.1066(1.08E-02)+	0.0076(7.57E-03)+	0.3260(2.84E-02)+	**0.3691(2.29E-02)**
	(10,10)	0.4002(1.53E-02)+	0.2526(9.41E-03)+	0.0047(4.72E-03)+	0.4112(1.49E-02)+	**0.5681(5.16E-03)**
	(10,20)	0.4913(1.12E-02)+	0.3675(8.50E-03)+	0.0068(6.79E-03)+	0.5058(1.14E-02)+	**0.6319(2.20E-03)**
DF10	(10,5)	0.6690(2.37E-02)-	**0.9420(5.99E-03)**-	0.2780(2.78E-01)+	0.8762(1.86E-02)-	0.6399(4.87E-03)
	(10,10)	0.7290(1.54E-02)=	**0.9487(2.05E-03)**-	0.2638(2.64E-01)+	0.9098(9.45E-03)-	0.7271(3.88E-03)
	(10,20)	0.7828(1.15E-02)-	**0.9519(1.76E-03)**-	0.2495(2.50E-01)+	0.9298(5.23E-03)-	0.7599(9.65E-04)
DF11	(10,5)	0.5120(1.94E-02)+	**1.1643(2.46E-03)**-	0.8200(8.20E-01)-	0.5452(7.72E-03)+	0.5539(1.09E-03)
	(10,10)	0.6043(1.81E-02)-	**1.1933(1.95E-03)**-	0.9558(9.56E-01)-	0.5736(6.87E-03)-	0.5667(5.51E-04)
	(10,20)	0.6795(1.13E-02)+	**1.2159(1.95E-03)**-	0.8682(8.68E-01)-	0.6004(3.72E-03)-	0.5722(3.11E-04)
DF12	(10,5)	1.0051(3.14E-02)-	**1.3132(1.15E-02)**-	0.0780(7.80E-02)+	1.1827(1.40E-02)-	0.8389(3.57E-03)
	(10,10)	1.1284(3.28E-02)-	**1.3363(1.25E-02)**-	0.0942(9.42E-02)+	1.2065(3.93E-03)-	0.8635(9.82E-04)
	(10,20)	1.2358(2.28E-02)-	**1.3856(7.71E-03)**-	0.0723(7.23E-02)+	1.2150(1.91E-03)-	0.8734(1.06E-03)
DF13	(10,5)	**1.8441(6.43E-02)**-	1.1691(6.39E-02)-	0.1964(1.95E-01)+	0.3862(3.07E-02)+	0.6062(1.51E-02)
	(10,10)	**2.1836(7.10E-02)**-	1.5380(1.11E-01)-	0.0978(9.78E-02)+	0.5201(2.66E-02)+	0.7309(7.43E-03)
	(10,20)	**2.4649(2.92E-02)**-	1.6942(9.23E-02)-	0.1199(1.20E-01)+	0.6427(2.04E-02)+	0.7675(3.26E-03)
DF14	(10,5)	0.2275(8.15E-03)+	0.0652(7.15E-03)+	0.0675(5.75E-02)+	0.5948(2.40E-02)+	**0.8628(5.89E-03)**
	(10,10)	0.2473(9.22E-03)+	0.0658(2.20E-03)+	0.0512(5.12E-02)+	0.7491(2.31E-02)+	**0.9385(1.99E-03)**
	(10,20)	0.2714(9.31E-03)+	0.0122(4.46E-03)+	0.0693(6.93E-02)+	0.8601(1.72E-02)+	**0.9546(1.11E-03)**
	+/=/−	26/2/14	24/0/18	39/0/3	34/0/8	

Overall, whether in terms of MIGD or MHV performance metrics, LEK demonstrates significant advantages over other algorithms. This fully reflects the algorithm's outstanding performance and highlights its ability to adapt to diverse dynamic environments with exceptional stability.

D. Ablation study

For our proposed algorithm LEK, the most significant innovation lies in the transfer learning (DA) model and the multi-layer perceptron (MLP) model. These two models respectively learn historical evolutionary knowledge and process evolutionary knowledge to guide population evolution,

TABLE III
MEAN AND STANDARD DEVIATION OF LEK AND VARIANTS ON $MIGD$

Problems	(η_t, τ_t)	DA	MLP	LEK
DF1	(10,5)	0.0580(5.42E-03)$^+$	0.0996(1.58E-02)$^+$	**0.0443(3.71E-03)**
	(10,10)	0.0321(2.47E-03)$^+$	0.0412(5.99E-03)$^+$	**0.0137(6.12E-04)**
	(10,20)	0.0233(1.34E-03)$^+$	0.0161(1.22E-03)$^+$	**0.0071(1.32E-04)**
DF2	(10,5)	0.1359(6.60E-03)$^+$	0.1477(8.98E-03)$^+$	**0.1176(4.79E-03)**
	(10,10)	0.0975(5.83E-03)$^+$	0.1069(6.10E-03)$^+$	**0.0670(3.52E-03)**
	(10,20)	0.0757(5.46E-03)$^+$	0.0693(3.80E-03)$^+$	**0.0330(2.64E-03)**
DF3	(10,5)	0.0472(1.60E-03)$^+$	0.0628(2.59E-02)$^+$	**0.0362(1.49E-03)**
	(10,10)	0.0293(8.66E-04)$^+$	0.0457(2.86E-02)$^+$	**0.0162(1.21E-03)**
	(10,20)	0.0225(7.68E-04)$^+$	0.0185(1.12E-02)$^+$	**0.0092(2.69E-04)**
DF4	(10,5)	0.1338(2.77E-03)$^+$	0.1304(3.59E-03)$^+$	**0.1242(2.89E-03)**
	(10,10)	0.1260(2.10E-03)$^+$	0.1199(2.01E-03)$^+$	**0.1161(8.65E-04)**
	(10,20)	0.1202(9.60E-04)$^+$	0.1153(9.95E-04)$^+$	**0.1131(2.23E-04)**
DF5	(10,5)	0.0464(2.13E-03)$^+$	0.1187(1.17E-02)$^+$	**0.0326(1.62E-03)**
	(10,10)	0.0265(1.27E-03)$^+$	0.0442(3.53E-03)$^+$	**0.0113(4.09E-04)**
	(10,20)	0.0190(9.53E-04)$^+$	0.0150(8.81E-04)$^+$	**0.0064(1.07E-04)**
DF6	(10,5)	1.2059(1.71E-01)$^+$	1.5595(1.21E-01)$^+$	**1.1126(1.51E-01)**
	(10,10)	0.6177(1.28E-01)$^+$	1.0587(1.47E-01)$^+$	**0.3379(8.40E-02)**
	(10,20)	0.3714(9.05E-02)$^+$	0.6075(1.33E-01)$^+$	**0.1866(6.71E-02)**
DF7	(10,5)	0.1947(1.13E-02)$^+$	0.3771(6.58E-02)$^+$	**0.2207(2.25E-02)**
	(10,10)	0.1696(7.87E-03)$^+$	0.2799(5.53E-02)$^+$	**0.1664(7.38E-03)**
	(10,20)	0.1579(3.72E-03)$^+$	0.2019(2.60E-02)$^+$	**0.1554(3.65E-03)**
DF8	(10,5)	0.0269(1.41E-03)$^+$	0.0240(1.69E-03)$^+$	**0.0249(1.52E-03)**
	(10,10)	0.0214(9.12E-04)$^+$	0.0194(8.49E-04)$^+$	**0.0193(8.77E-04)**
	(10,20)	0.0183(1.12E-03)$^+$	0.0169(6.59E-04)$^+$	**0.0167(7.45E-04)**
DF9	(10,5)	0.3734(8.51E-02)$^+$	0.5177(8.87E-02)$^+$	**0.2823(4.82E-02)**
	(10,10)	0.1461(1.13E-02)$^+$	0.2105(4.95E-02)$^+$	**0.0691(4.47E-03)**
	(10,20)	0.0807(4.30E-03)$^+$	0.0812(6.10E-03)$^+$	**0.0288(1.44E-03)**
DF10	(10,5)	0.1647(6.20E-03)$^+$	0.1862(1.51E-02)$^+$	**0.1496(4.82E-03)**
	(10,10)	0.1246(1.87E-03)$^+$	0.1508(9.97E-03)$^+$	**0.1098(1.95E-03)**
	(10,20)	0.1147(1.16E-03)$^+$	0.1206(6.18E-03)$^+$	**0.0997(1.30E-03)**
DF11	(10,5)	0.1958(3.28E-04)$^+$	0.1964(2.06E-03)$^+$	**0.1941(4.34E-04)**
	(10,10)	0.1903(2.20E-04)$^+$	0.1916(1.20E-03)$^+$	**0.1882(1.61E-04)**
	(10,20)	0.1881(1.45E-04)$^+$	0.1888(1.35E-03)$^+$	**0.1859(9.71E-05)**
DF12	(10,5)	0.1131(4.55E-03)$^-$	0.1291(1.21E-02)$^+$	**0.1151(4.33E-03)**
	(10,10)	0.0929(1.46E-03)$^+$	0.0993(2.76E-03)$^+$	**0.0920(9.52E-04)**
	(10,20)	0.0866(1.23E-03)$^+$	0.0846(1.70E-03)$^+$	**0.0823(7.17E-04)**
DF13	(10,5)	0.3175(6.67E-03)$^+$	0.3407(6.62E-03)$^+$	**0.3032(6.36E-03)**
	(10,10)	0.2738(3.49E-03)$^+$	0.2971(3.71E-03)$^+$	**0.2649(2.90E-03)**
	(10,20)	0.2682(2.64E-03)$^+$	0.2821(2.58E-03)$^+$	**0.2636(2.04E-03)**
DF14	(10,5)	0.0837(2.33E-03)$^+$	0.1145(7.70E-03)$^+$	**0.0753(1.79E-03)**
	(10,10)	0.0645(6.68E-04)$^+$	0.0819(4.07E-03)$^+$	**0.0578(7.69E-04)**
	(10,20)	0.0588(5.05E-04)$^+$	0.0627(1.38E-03)$^+$	**0.0541(3.28E-04)**
	+/-/	38/4/0	39/3/0	

enabling the algorithm to generate high-quality offspring with greater probability and accelerate population convergence. The comparative experiments above demonstrate that combining these two models significantly enhances the algorithm's performance. However, the positive contribution of each model individually to the algorithm remains unverified. To investigate the role of these models in the algorithm, we conducted ablation experiments to validate their effects.

In the ablation experiments, we set up two variants, denoted as DA and MLP respectively. Specifically, DA denotes using only the transfer learning model, while MLP denotes using only the MLP model. Notably, for both variants, qm_i controls

the probability of model usage (each variant employs its corresponding model). When the random number is less than qm_i, it indicates that after generating individuals using the model, offspring are produced via the DE operator. Conversely, when the random number is greater than or equal to qm_i, the model is not used. The value of qm_i is dynamically adjusted using the same feedback mechanism as q_i and m_i, with an initial value set to 0.8.

Table III presents the comparison results of the two variants and the original algorithm on $MIGD$. The symbols "+", "=", and "-" in the table indicate that LEK's performance is significantly better than, equal to, or worse than the variants, respectively. Out of a total of 42 problems, LEK achieved the best value on 40 problems, outperforming both the DA and MLP variants. This fully demonstrates that both models have a positive effect on the algorithm. By integrating these two models, the algorithm not only learns historical evolutionary knowledge and transfers it to solving current problems but also acquires knowledge specific to the evolution of the current problem. These two types of knowledge yield different effects for problems with varying mutation types or in different dynamic environments. Through the feedback mechanism, both types of knowledge are fully leveraged, granting the LEK algorithm a significant advantage over using either the DA or MLP model alone.

V. Conclusions

This chapter proposes a dynamic multi-objective evolutionary algorithm based on learning evolutionary knowledge for dynamic multi-objective optimization problems. The method employs transfer learning models and multilayer perceptron models to learn historical evolutionary knowledge and evolutionary process knowledge of the current problem, respectively.

Specifically, this algorithm adopts MOEAD as its foundational framework. Since individuals sharing the same weight vector exhibit relatedness, the transfer learning model employs domain adaptation to migrate the evolutionary direction of the population from a previous historical problem to the current one, enhancing the adaptability of historical knowledge to the current problem. In the MLP model application, data sample pairs composed of parent and offspring individuals are used for training. This enables the MLP to extract evolutionary trends within the current problem's population, which are then applied to subsequent evolutionary steps. This approach improves offspring quality and accelerates population convergence. Finally, the newly proposed LEK algorithm was experimentally compared with four other algorithms in the DF testing component, demonstrating its strong competitiveness. Ablation experiments were conducted to confirm that each model positively contributes to the algorithm.

Existing learning-based evolutionary knowledge methods fail to adequately account for problem-specific variations, often directly applying past evolutionary knowledge to new problems—a practice that may misguide evolutionary trajectories. Specifically, extracting effective knowledge from evolutionary processes to accelerate population convergence while preserving diversity could represent a novel approach for dynamic multi-objective optimization problems, enabling rapid adaptation to environmental changes.

References

[1] K. Zhang, C. Shen, X. Liu, and G. G. Yen, "Multiobjective evolution strategy for dynamic multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 5, pp. 974–988, 2020.

[2] Z. Wang, G. Li, and J. Ren, "Dynamic path planning for unmanned surface vehicle in complex offshore areas based on hybrid algorithm," *Computer Communications*, vol. 166, no. 17, pp. 49–56, 2021.

[3] D. Gong, B. Xu, Y. Zhang, Y. Guo, and S. Yang, "A similarity-based cooperative co-evolutionary algorithm for dynamic interval multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 142–156, 2020.

[4] L. T. Bui, Z. Michalewicz, E. Parkinson, and M. B. Abello, "Adaptation in dynamic environments: A case study in mission planning," *IEEE Transactions on Evolutionary Computation*, vol. 16, no. 2, pp. 190–209, 2012.

[5] Z. Liang, T. Wu, X. Ma, Z. Zhu, and S. Yang, "A dynamic multiobjective evolutionary algorithm based on decision variable classification," *IEEE Transactions on Cybernetics*, vol. 52, no. 3, pp. 1602–1615, 2022.

[6] S. Jiang and S. Yang, "Evolutionary dynamic multiobjective optimization: Benchmarks and algorithm comparisons," *IEEE Transactions on Cybernetics*, vol. 47, no. 1, pp. 198–211, 2017.

[7] A. Ghosh, S. Tsutsui, and H. Tanaka, "Function optimization in nonstationary environment using steady state genetic algorithms with aging of individuals," in *IEEE International Conference on Evolutionary Computation*, pp. 666–671, IEEE, 1998.

[8] S. Yang, "Genetic algorithms with elitism-based immigrants for changing optimization problems," in *Applications of Evolutionary Computing*, pp. 627–636, Springer, 2007.

[9] K. Deb, U. B. Rao, and S. Karthik, "Dynamic multi-objective optimization and decision-making using modified nsga-ii: A case study on hydro-thermal power scheduling," *Heidelberg, Germany: Springer*, 2007.

[10] B. Xu, Y. Zhang, D. Gong, Y. Guo, and M. Rong, "Environment sensitivity-based cooperative co-evolutionary algorithms for dynamic multi-objective optimization," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 15, no. 6, pp. 1877–1890, 2018.

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

[12] R. Azzouz, S. Bechikh, and L. B. Said, "A dynamic multi-objective evolutionary algorithm using a change severity-based adaptive population management strategy," *Soft Computing*, vol. 21, no. 4, pp. 1–22, 2015.

[13] R. Chen, K. Li, and X. Yao, "Dynamic multiobjectives optimization with a changing number of objectives," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 157–171, 2018.

[14] K. Yu *et al.*, "A framework based on historical evolution learning for dynamic multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, 2023. TEVC.2023.3290485.

[15] Q. Zhao, B. Yan, Y. Shi, and M. Middendorf, "Evolutionary dynamic multiobjective optimization via learning from historical search process," *IEEE Transactions on Cybernetics*, vol. 52, no. 7, pp. 6119–6130, 2022.

[16] Y. Guo, G. Chen, M. Jiang, D. Gong, and J. Liang, "A knowledge guided transfer strategy for evolutionary dynamic multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 6, pp. 1750–1764, 2023.