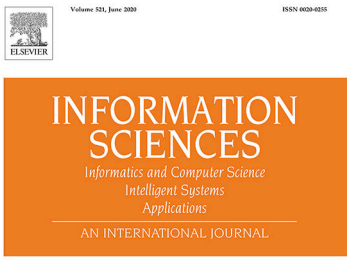

A Reinforcement Learning Approach for Dynamic Multi-objective Optimization

Fei Zou, Gary G. Yen, Lixin Tang, Chunfeng Wang

PII: S0020-0255(20)30867-7
DOI: https://doi.org/10.1016/j.ins.2020.08.101
Reference: INS 15816

To appear in: *Information Sciences*

Received Date: 15 October 2019
Revised Date: 14 May 2020
Accepted Date: 25 August 2020

Please cite this article as: F. Zou, G.G. Yen, L. Tang, C. Wang, A Reinforcement Learning Approach for Dynamic Multi-objective Optimization, *Information Sciences* (2020), doi: https://doi.org/10.1016/j.ins.2020.08.101

A Reinforcement Learning Approach for Dynamic Multi-objective Optimization

Fei Zou[a], Gary G. Yen[b,*], Lixin Tang[c], Chunfeng Wang[d]

[a]*Key Laboratory of Data Analytics and Optimization for Smart Industry, Ministry of Education, Liaoning Engineering Laboratory of Operation Analytics and Optimization for Smart Industry, Northeastern University, Shenyang 110819, China*
[b]*School of Electrical and Computer Engineering, Oklahoma State University, Stillwater, OK 74078, USA*
[c]*Liaoning Key Laboratory of Manufacturing System and Logistics, Institute of Industrial and Systems Engineering, Northeastern University, Shenyang 110819, China*
[d]*College of Mathematics and Information Science, Henan Normal University, Xinxiang, 453007, China*

Abstract

Dynamic Multi-objective Optimization Problem (DMOP) is emerging in recent years as a major real-world optimization problem receiving considerable attention. Tracking the movement of Pareto front efficiently and effectively over time has been a central issue in solving DMOPs. In this paper, a reinforcement learning-based dynamic multi-objective evolutionary algorithm, called RL-DMOEA, which seamlessly integrates reinforcement learning framework and three change response mechanisms, is proposed for solving DMOPs. The proposed algorithm relocates the individuals based on the severity degree of environmental changes, which is estimated through the corresponding changes in the objective space of their decision variables. When identifying different severity degree of environmental changes, the proposed RL-DMOEA approach can learn better evolutionary behaviors from environment information, based on which apply the appropriate response mechanisms. Specifically, these change response mechanisms including the knee-based prediction, center-based prediction and indicator-based local search, are devised to promote both convergence and diversity of the algorithm under different severity of environmental changes.

*Corresponding author
Email address: gyen@okstate.edu (Gary G. Yen)

To verify this idea, the proposed RL-DMOEA is evaluated on CEC 2015 test problems involving various problem characteristics. Empirical studies on chosen state-of-the-art designs validate that the proposed RL-DMOEA is effective in addressing the DMOPs.

Keywords: Dynamic multi-objective optimization, severity degree, reinforcement learning, prediction

1. Introduction

Many dynamic multi-objective optimization problems (DMOPs) are derived from real-world problems, involving multiple, conflicting time-dependent objectives or constraints [11]. Such scenarios arise from practical disciplines in fault tolerant control, priority scheduling and vehicle routing [9]. They pose a challenge to address DMOPs since their objective functions, constraints and parameters will vary over time. Owing to its inherent dynamism nature, the goal of solving DMOPs is to facilitate the tracking capability of the algorithm after detecting the environmental changes. Therefore, the research has focused on the active design of effectively solving DMOPs in the past few years.

In recent decades, many researchers have recognized that a variety of multi-objective evolutionary algorithms (MOEAs) are efficient tools to solve DMOPs. These methods including diversity enhancement strategies [20], prediction mechanisms [45], and memory methods [4],roughly constitute the techniques for handling dynamisms of DMOPs. In particular, the prediction methods, which have shown competitive performances, aims to predict the changing Pareto optimal set (POS) or Pareto optimal front (POF) through built prediction models based on historical and existing information. These prediction-based models, either utilizing machine learning technologies (e.g., autoregressive models [44], the transfer learning model [14], and the Kalman Filter-based model [23]) or capturing the historic movement of the POS center [24] to relocate the individuals in the population, are considered state-of-the-art solutions.

As demonstrated in the literature, these prediction methods mainly reinitial-

ize the population based on learned historical information, so that the algorithm can respond to these changes in advance after detecting the environmental changes. However, the prediction strategy has been plagued by various deficiencies in solving the DMOPs. Most existing prediction-based methods only apply the predicted response mechanisms at the instance of changes without considering the suitability of these mechanisms for these environmental changes, leading to the waste of valuable environmental information. If a dynamic multi-objective evolutionary algorithm (DMOEA) could observe the severity of detected changes, it can effectively adapt to the changing environment in time and guide the population to move towards the POF during whole evolution.

Meanwhile, it is critical to develop an algorithm framework which can determine the reasonable response mechanism to achieve a dynamism adjustment based on the environmental feedback. However, most current approaches lack learning or feedback mechanisms which can be assessed through environment information to guide the search directions. The information interacted with the dynamic environment is very valuable, helping to ensure the correct moving direction after detecting the environmental changes. In particular, different environmental conditions may require different search operations to track the moving POF more effectively. Therefore, it is expected that a more ideal DMOEA can address a variety of challenges in solving DMOPs.

To address the above issue, the interaction between machine learning and MOEAs has received considerable attention in evolutionary computation community. Among these machine learning algorithms, reinforcement learning (RL) is considered as a classic representative due to its sequential decision making characteristics under the stochastic environment [41]. Especially, RL algorithm requires the agent to find an optimal strategy which optimizes multiple objectives and achieves a trade-off among the conflicting objectives [36].

In the literature, RL techniques have been used in evolutionary computation to enhance the algorithm performance and to solve the real-world problems. Hussein et al. [27] utilized a reinforcement learning-based memetic particle swarm optimization (RLMPSO) approach during whole search process. Liao

et al. [19] proposed a novel algorithm, named multi-objective optimization by reinforcement learning (MORL), to solve the real-world application in the power system. These researchers believe that reinforcement learning techniques can facilitate the evolutionary process of the algorithm by means of using previous information and Markov decision process (MDP).

However, RL still leaves much room for a better design of prediction-based algorithms combined with dynamic environment characteristics when dealing with DMOPs. As expected, integrating reinforcement learning methods into D-MOEAs is still considered in its infancy. The distribution of POF in DMOPs at different time is mutually related to the dynamic environments, whose severity of changes is not exactly the same. Obviously, the reinforcement learning algorithms which exploit environmental information could contribute significantly to the DMOEA.

The motivation of this study is that RL is an effective method to learn the optimal behavior by interactions between an agent and dynamic environments, which is therefore suitable to address DMOPs with dynamic environment characteristics. First, the RL method is equipped with the ability to learn the environmental characteristics and then utilize the environment information to guide the population evolution in time-varying environments. Second, the RL method has the ability to alleviate the probability of inaccurate prediction, thereby enhancing the algorithm's tracking ability through the reasonable change response mechanisms. It estimates the severity degree of the environmental changes (e.g., slight-severity, medium-severity and high-severity changes) and applies the appropriate response mechanism to adapt to the environmental changes, which is particularly useful in DMOPs in terms of searching optimal solutions.

Some researchers utilize a multi-objective two-archive memetic algorithm based on Q-learning (MOTAMAQ) to solve the dynamic software project scheduling problems [31]. In spite of adopting a similar Q-learning framework, specific definitions of dynamic environments and individual representations are fairly different. Our proposed RL-DMOEA perceives severity degree of environmental changes which are estimated within the objective space of the continuous

decision variables. Afterwards, three distinct, yet complement, prediction-based mechanisms which relocate the individuals are ensembled based on Q-learning framework according to the location correlations of optimal solutions at different time. On the other hand, MOTAMAQ employs discrete binary encoding and its environmental changes are mainly affected by the number of available tasks and available employees. MOTAMAQ exploits the appropriate global and local search operators of the memetic algorithm to generate non-dominated solutions. Due to its own characteristics, MOTAMAQ may not be appropriate for implementing the prediction-based strategies.

Motivated by the dynamism nature of DMOPs, a computationally efficient RL framework is proposed, in which the knee-based prediction, center-based prediction and indicator-based local search are employed due to following reasons. As demonstrated in [7], the global knee solution contains valuable information for guiding the predicted evolutionary direction, which reflects the optimal solution accurately. It should be noted that it is effective to track the movement of POF via evolution direction of the global knee solution, because higher severity of change demands greater exploration capability. Therefore, the knee-based prediction mechanism, which is computed through computationally efficient minimum Manhattan distance (MMD) approach [7], predicts the new locations to respond to *high-severity* environmental changes quickly. Referring to [45], it is worth noting that the population centers provide efficient historical optimal information at different time instances, guiding the promising evolutionary direction. The center-based prediction is adopted to relocate individuals when detecting the *medium-severity* changes from the empirical perspective. The indicator-based local search mechanism [46], on the other hand, shows great promise in facilitating convergence when confronting with the *slight-severity* environmental changes. They collectively generate high quality solutions in the close neighborhood of POS and provide a faster convergence, which makes the algorithm well suited for responding to varying degrees of the environmental changes. The proposed algorithm, termed RL-DMOEA, is evaluated on CEC 2015 benchmark problems to verify its effective performance in solving DMOPs.

Intuitively, the main contributions of this paper are given as follows:

(1) We devise a reinforcement learning-based framework (in short for RL-DMOEA), which predicts and relocates the POS more adaptively by incorporating RL-based Q-learning into the evolutionary process.

(2) Based on different severity degree of environmental changes, the knee-based prediction, the center-based prediction, and the indicator-based local search prediction methods are synergistically integrated to predict the location of non-dominated solutions in the new environment.

(3) According to the dynamism correlations of decision space at different time, our proposed algorithm can learn valuable information from the dynamic environment, based on which to determine the appropriate prediction-based strategy to adapt to the environmental changes.

The remainder of the paper is structured as follows. We describe some related approaches and preliminaries in Section 2. The technical details of proposed RL-DMOEA are presented step by step in Section 3. Section 4 overviews the benchmark problems, performance metrics adopted and shows the empirical results and discussions. We summarize the paper and discuss the future research direction in Section 5.

2. Preliminaries and related works

Firstly, we introduce the concepts of DMOPs investigated in this section. Next, a detailed description of the related works for DMOPs is briefly introduced. Then, an introduction of Q-learning is given, which is regarded as the most well-known reinforcement learning algorithm.

2.1. Concepts of Dynamic Multi-Objective Optimization

The characteristic of DMOPs is that their objective functions change over time. Generally, a typical DMOP is formulated as follows [14].

$$\begin{aligned} &Minimize \qquad f(x,t) = [f_1(x,t), f_2(x,t), ..., f_M(x,t)] \\ &subject \quad to \quad x \in \Omega \end{aligned} \tag{1}$$

In the definition, $x = (x_1, x_2, ..., x_n)$ represents a decision vector, and Ω indicates the feasible decision space. $f_i(x,t) : \Omega \to R(i = 1, 2, ..., M)$. $\Omega = [L_1, U_1] \times [L_2, U_2] \times \cdots \times [L_n, U_n]$. The function $f(x,t)$ denotes the objective vector with M objectives, which changes over time t. L_i and U_i denote the lower and upper bounds of the ith decision vector, respectively.

Definition 1. Dynamic Pareto domination: Assume that x_1 and x_2 are two decision vectors, and x_1 dominates x_2 ($x_1 \prec_t x_2$) at time t implies the following relationship:

$$\begin{aligned} f_i(x_1, t) &\leq f_i(x_2, t) \qquad \forall i = 1, ..., M \\ f_j(x_1, t) &< f_j(x_2, t) \qquad \exists j = 1, ..., M \end{aligned} \tag{2}$$

Definition 2. Dynamic Pareto optimal set: Two decision vectors x^* and x are given. In the decision space, x^* is said to be non-dominated if there is no other solution x dominate x^* at time t. The Dynamic Pareto optimal set (DPOS) is formed by all the Pareto optimal solutions at time t.

$$DPOS = \{x^* \mid \neg \exists x \in \Omega : x \prec_t x^*\} \tag{3}$$

Definition 3. Dynamic Pareto optimal front: At time t, the Dynamic Pareto optimal front (DPOF) is composed by the corresponding objective vectors of the DPOS.

$$DPOF = \{f(x^*, t) = (f_1(x^*, t), f_2(x^*, t), ..., f_M(x^*, t)) \mid x^* \in DPOS\} \tag{4}$$

2.2. Related Works

Literatures related to DMOPs field are thoroughly reviewed and numerous representative approaches are mainly divided into three categories, which are diversity-based mechanisms, memory-based mechanisms, and prediction-based mechanisms.

The diversity-based mechanism aims to maintain or preserve the population diversity using a kind of technique when the dynamic changes are detected. Liu et al. devised the hypermutation strategies to tackle DMOPs [22]. Coello Coello et al. [1] suggested a novel diversity enhancement mechanism that

promotes and adapts to the environmental changes. Woldesenbet and Yen [39] introduced a novel evolutionary algorithm. This algorithm relocates the solutions according to the average sensitivities of their decision variables in changing environment to solve DMOPs. Kwong et al. [18] suggested a minimum spanning tree algorithm to address the difficulties of DMOPs. Goh and Tan [12] proposed a multi-population strategy for maintaining the diversity of population when solving DMOPs. Deb and Karthik [10] introduced two versions of DNSGA-II when handling dynamic optimization problems. Camara et al. [5] utilized some Pareto optimal solutions and crowding mechanisms to effectively handle dynamic changes in DMOPs.

The memory mechanism records the past information and reuses the historical information in order to enhance the performance of DMOEAs when dynamic changes are detected. In [8], the authors proposed a PSO algorithm with an external repository, and the exploratory capabilities of the algorithm are enriched. Wang and Li [37] devised a memory-based algorithm to address DMOPs. In this method, the restart, explicit memory, local search memory and hybrid memory are incorporated to improve the searching ability of their proposed algorithm. Peng et al. [24] utilized an evolutionary environment model which helps to adapt to the changing environment to solve DMOPs. Azzouz et al. [2] presented a memory management strategy to address environment dynamicity effectively. The authors used memory, local search and random strategies when the environmental changes are detected.

The prediction-based mechanisms have drawn much attention to handle DMOPs in recent decades. The strategy is combined with a prediction model and predicts the population based on better expected knowledge. Zhou et al. [44] presented the population prediction strategy (PPS) which uses previous center points and the previous manifolds to predict the next center point and next manifold, respectively. Wu et al. [40] proposed a special directed search strategy (DSS). This method uses predicted direction of movement to reinitialize new population for solving DMOPs. Koo et al. [16] devised the method which predicts the moving direction and magnitude of POS when solving DMOPs.

Jiang et al. [14] developed a Tr-DMOEA algorithm, which adopts the transfer component analysis method to relocate the individuals in population and speed up evolutionary process. Muruganantham et al. [23] proposed an algorithm through utilizing a Kalman filter (KF) technique to estimate the moving direction in state of the system when solving DMOPs. Ruan et al. [25] developed a hybrid algorithm to predict the distribution of the Pareto front accurately. The prediction-based method can predict the individuals which are close to the new POF by observing the movement direction of the center points.

Recently, some works exploit new research directions in evolutionary computation. Liu et al. [21] developed a novel algorithm which uses the co-evolutionary optimizer to address DMOPs. Jiang and Yang [15] presented a novel method to handle DMOPs. This method integrates the steady-state tracking ability and preserves a good diversity. Shang et al. [30] devised a quantum immune clone co-evolutionary algorithm (QICCA) which adopts entire cloning to solve DMOPs. Chen et al. [6] attempted to tackle the DMOPs which have a changing number of objectives. They designed two archives which maintain two co-evolving populations to overcome the difficulty.

2.3. Q-learning

In a RL algorithm, an agent aims to learn feedback by interacting with a dynamic environment [35]. In other words, RL focuses on how the agent determines the action in the environment to achieve maximum cumulative reward [34]. Generally, the researchers usually formulate this kind of sequential decision process as a Markov decision process (MDP) [38].

Q-learning is one of breakthroughs in RL and derives from machine learning community. Known as a model-free reinforcement learning method, Q-learning is an off-policy temporal-difference (TD) algorithm to estimate the accumulative rewards of performing an action in a given state [33]. Its key components consist of an agent, environment, states, actions, and rewards. The main purpose of combining Q-learning algorithm with MOEAs is to identify the optimal policy for maximizing the overall rewards [29]. $S = [s_1, s_2, ..., s_n]$ denotes a set of states

for the learning agent, and $A = [a_1, a_2, ..., a_n]$ denotes a set of actions performed by the agent, r_{t+1} is the immediate reward for executing action a. Afterwards, the learned action-value function of Q-learning at time t is given as follows:

$$Q(s_t, a_t) = (1 - \alpha)Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a \in A} Q(s_{t+1}, a)] \tag{5}$$

where the discount rate γ is to control the convergence. In addition, the learned action value function Q prompts the early convergence proof [26]. As reported in [35], the procedures of Q-learning are shown in Algorithm 1.

Algorithm 1 Procedure of Q-learning

1: Initialize $Q(s_t, a_t)$ arbitrarily and parameters
2: Loop for each episode
3: Initialize s_t
4: Loop for each step of episode
5: Choose a_t from s_t using policy derived from Q-learning
6: Take action a_t and observe r_{t+1} and s_{t+1}
7: Adjust Q value $Q(s_t, a_t) = (1 - \alpha)Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a \in A} Q(s_{t+1}, a)]$
8: $s_t = s_{t+1}$
9: Until s_t is terminal

3. The RL-DMOEA

The proposed reinforcement learning-based dynamic multi-objective evolutionary algorithm (in short for RL-DMOEA) is presented in this section. Firstly, the general framework of RL-DMOEA is outlined. Then the main innovative component, the Q-learning algorithm to implement RL framework is illustrated in details. Afterwards, three change response mechanisms of dealing with environment changes are depicted. At last, we analyze the computational complexity of proposed RL-DMOEA.

3.1. General framework of RL-DMOEA

The major components of overall RL-DMOEA framework are illustrated in Algorithm 2. As depicted in Fig. 1, the agent learns information from dynamic environment changes, and then determines which action to be taken in the new environment. The actions involve the knee-based prediction, the center-based prediction and indicator-based local search methods in RL-DMOEA approach. The environment for performing Q-learning is considered as the entire searching space in the given DMOP. The RL-DMOEA approach follows the decisions of the agent. After generating non-dominated solutions, an agent evaluates reward of the selected action and updates the Q values accordingly. Figure 2 is used to illustrate the proposed algorithm clearly.

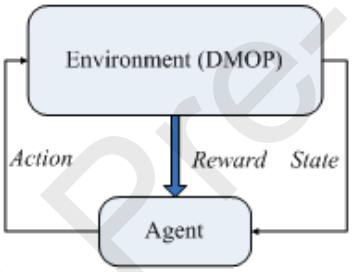

Fig. 1. Major components in the proposed RL framework.

To be specific, the overall RL-DMOEA calls the following steps. RL-DMOEA initializes the population and parameters. In the main loop, RL-DMOEA detects environment changes and evolves the population. If no change is detected, the stationary MOP is optimized with the MOEA. When detecting the environmental change, a change severity detection operation is adopted to estimate the severity of changes which is handled by Q-learning procedure. After that, the selection policy operation is applied to determine which prediction strategy should be used to guide the population evolution. Positive rewards are updated according to HV value which is computed through the non-dominated solutions. Then, the next state is observed and Q-table is updated. The overall framework of proposed RL-DMOEA is presented in Algorithm 2.

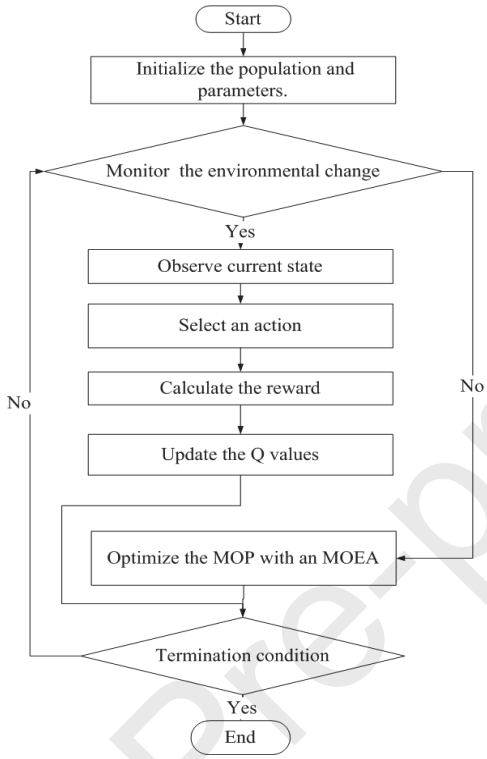

Fig. 2. Diagram of the proposed RL-DMOEA algorithm.

3.2. Change severity detection operation

This subsection introduces a change severity detection strategy, which can help the agent learn the severity of environment change and make an appropriate decision about selecting response mechanisms to be taken in the new environment. The POFs usually change over time. Therefore, an agent should be able to detect and observe the environmental changes in time. To detect environment changes effectively, the change severity detection is designed. This operation identifies the severity degree of the environmental change by computing the deviation of the detectors at different time instances. The definition of the state (severity of environmental change) is the deviation between the objective values at time step t-1 and time step t. Based on the archive m_k, the

Algorithm 2 Framework of RL-DMOEA

Input: the maximum generation number, $Gmax$; generation index, $g = 0$; learning rate α;discount factor γ;

Output: approximated POS;

Initialize sets of Q values $Q(s, a)$ as empty sets for each (s, a) pair;
Initialize average reward vector $R(s, a)$ as zero for each (s, a) pair;
Randomly generate an initial population P;
while stopping criteria are not satisfied ($g < Gmax$) **do**
  if change is detected **then**
    Observe current state s according to the change severity detection operator;
    Select an action a from (a_1, a_2, a_3) through the selection policy;
    Calculate the reward (HV value of the non-dominated solutions), update $R(s, a)$ and observe next state $s^{'}$;
    Update the Q values $Q(s, a)$ according to update of the Q-value;
  else
    Optimize the MOP with NSGA-II-DE;
  end if
  $g = g + 1$;
end while

objective values of all detectors are stored. To implement this method, we check the environmental changes as follows:

$$\varepsilon_t = \frac{\sum_{j \in m_k} |\frac{f_m(x_{j,t}) - f_m(x_{j,t-1})}{R(t) - U(t)}|}{|m_k|} \quad m = 1, 2, ..., M \tag{6}$$

where ε_t is a pre-defined threshold. $R(t)$ and $U(t)$ represent the nadir point and utopia point varying over time, respectively. The objective value at time t, denoted as $f_m(x_{j,t})$, is compared with that at time $t-1$, denoted as $f_m(x_{j,t-1})$, to compute the severity degree.

3.3. Actions of RL-DMOEA

When the environment changes are detected, the change response mechanism should equip with a good tracking ability in the dynamically changing environment. The severity degree of the environment change should be identified, and then the agent applies suitable corresponding response strategies. The knee-based prediction, center-based prediction and indicator-based local search mechanisms are regarded as different actions.

3.3.1. Knee-based prediction mechanism (KBP)

In the dynamic environment, it has been an active research direction to track the POF, the distribution of which has been confirmed to show inherent laws of DMOPs [3]. Meanwhile, the knee solution [42] could reflect the optimal solution accurately, showing a unique significance in addressing multi-objective optimization problems (MOPs). Thus, we can utilize knee information to track the movement of the POF when a high-severity environmental change is detected.

The knee-based prediction (KBP) can predict the new locations of non-dominated solutions through searching for the knee solution. In order to reduce the impact of inaccurate prediction, the updated population which consist of some existing solutions and random solutions, is generated by KBP mechanism. Referring to [7], the Minimum Manhattan Distance (MMD) method is adopted, which determines the global knee point discriminately. Identifying the

knee point is important to improve the searching capability of RL-DMOEA. Compared to other mechanisms, the MMD knee selection mechanism shows its superiority in searching for the global knee solution, especially in exploiting geometrical knowledge and performance measure. Based on these reasons mentioned above, we choose MMD knee selection as a basic construction for the proposed knee-based prediction design. Algorithm 3 shows the detailed steps which selects MMD knee solution from a given non-dominated set.

Algorithm 3 MMD knee selection

Input: the non-dominated solutions set A;

Output: MMD knee solution set, Q;

1: **for** $m = 1$ to M **do**
2: Identify minimum value f_m^{Min} in objective m from A;
3: Identify maximum value f_m^{Max} in objective m from A;
4: **for** $i = 1$ to $|A|$ **do**
5: $Dist_i = Dist_i + \frac{f_m^i - f_m^{Min}}{f_m^{Max} - f_m^{Min}}$
6: **end for**
7: **end for**
8: Select the MMD knee solution with minimum value of $Dist_i$;
9: Copy the the MMD knee solution into set Q

If the environmental changes are detected, it is critical to generate the individuals based on the correct direction of movement. Therefore, as shown in Fig. 3, the global knee is obtained to compute the moving direction. K_t represents the knee solution calculated through MMD method at time t. The movement step-size D_t can be computed as follows:

$$D_t = \|K_t - K_{t-1}\| \tag{7}$$

where D_t refers to the Euclidean distance which is computed between the global knees K_t and K_{t-1}. In this way, the direction of movement is calculated through the global knee solutions.

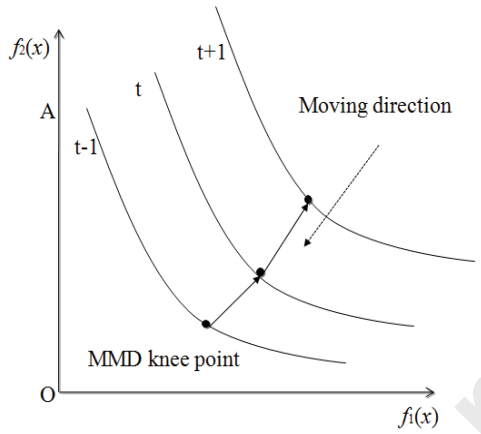

Fig. 3. Prediction method by MMD knee point.

The new individuals can be reinitialized based on the direction of movement and current location. The new location of individuals in the search space is predicted in the following way [15]:

$$x_{t+1} = x_t + D_t + \varepsilon_t \tag{8}$$

where $\varepsilon_t \sim N(0, d)$ is a Gaussian perturbation which has a mean value of zero and standard deviation of d. In the new environment, the algorithm utilizes the global knee solution to reinitialize the population. The details are presented in Algorithm 4.

The knee-based prediction strategy contributes to accelerating the convergence speed of the algorithm. To effectively address DMOPs, maintaining diversity should not be ignored. Since random solutions help to preserve the diversity, we adopt the random strategy to reinitialize a portion of individuals in the new population. The detailed procedure is shown in Algorithm 5. The Pareto optimal front predicted by the knee-based strategy, and random solutions constitute the population in the new environment, which maintains the well-distributed and well-converged POS. Afterwards, the population is updated with NSGA-II-DE [17].

It is worth noting that, although the knee-guided prediction approach [48] in

KPEA shares a similar calculation as Equation (8), they are different in essence. The knee-guided prediction approach uses the knee solution to predict the moving direction, and then update the boundary solutions and global knee solution accordingly in the new environment. KPEA aims to search the neighborhoods surrounding the knee solution, because it solely concentrates on searching for the updated knee solution when detecting environmental changes. In contrast, although the knee-based prediction mechanism in RL-DMOEA calculates the moving direction via the knee solution as well, it updates the POS at time t by using the obtained moving direction and the POS at time $t-1$. In addition, in order to enhance the diversity, these updated POS and some random solutions are incorporated into the population, which in turn facilitate to reinitialize the population.

Algorithm 4 Knee-based prediction

Input: the POS_t at time t;the size of population NP;previous POS_{t-1} at time $t-1$;

Output: approximated POS;

1: Calculate the knee solution K_t of POS_t and K_{t-1} of POS_{t-1} according to Minimum Manhattan Distance (MMD) method (Algorithm 2);
2: Generate new individuals using POS_t using Equation (7) and (8) , then check the boundary;
3: Copy them into P_{knee} and identify N_{knee} (the size of P_{knee});
4: Generate P_{random} with the size $N_r = NP - N_{knee}$ using Algorithm 5;
5: Get the new population $P_g = P_{knee} + P_{random}$;
6: Optimize P_g with NSGA-II-DE ;

Algorithm 5 Random strategy

Input: the size of population N_r;

Output: random generated population P_{random};

1: Randomly generate N_r individuals and check the boundary constraints;
2: Copy them into P_{random} and get the random generated population;

3.3.2. Indicator-based Local search mechanism (ILS)

The indicator-based local search mechanism (ILS) is believed to generate high quality solutions around POS and accelerate the convergence speed. When slight-severity environmental changes are detected, this kind of local search process could lead the population towards new promising search directions [46]. Thus, we propose this mechanism which is able to search for promising solutions in the neighborhood of the updated population.

In addition, the method adopts the quality indicator $I_{\varepsilon+}$ as a selection principle for fitness assignment in [46]. During the selection process, it calculates the quality of comparison of two Pareto sets with respect to each other [46]. Generally, the indicator is stated in the following way:

$$I_{\varepsilon+}(A,B) = min(\forall x_2 \in B, \exists x_1 \in A : f_i(x_1) - \varepsilon \leq f_i(x_2))$$
$$i = 1, ..., M \tag{9}$$
$$Fit_(x) = I(P\backslash\{x\}, x) = \sum_{z \in P\backslash\{x\}} -e^{-I_{\varepsilon+}(z,x)/w}$$

where P is the population, and w is usually set to 0.05.

To construct the new population, the obtained better quality neighborhood solutions are introduced into the new population. During evolutionary iterations, an individual which has smallest fitness is eliminated. When detecting the slight environment changes, some individuals are generated by the indicator-based local search which can inadvertently promote the convergence [32]. Some random individuals are generated to maintain the population diversity. Finally, the population is updated using NSGA-II-DE [17].

3.3.3. Center-based prediction mechanism (CBP)

For medium-severity environmental changes, a center-based prediction (CBP) mechanism is proposed to redistribute individuals to the regions close to the POF. The center-based prediction method [45] aims to predict the new population based on previous information. Traditional prediction-based approaches [45] execute the process by using Equation (10), maybe leading to prediction

Algorithm 6 Indicator-based local search

Input: the POS_t at time t, individual x, fitness indicator $Fit(x)$, Num(number of neighborhood trials), index $k = 0$;

Output: approximated POS;

Evaluate the fitness values of each individual x in POS_t;
$Fit(x) = I(POS_t \backslash \{x\}, x)$;
for each $x \in POS_t$ **do**
  while $(k < Num)$ **do**
    x_1=mutation(x);
    Compute fitness of x_1,$Fit(x_1) = I(POS_t \backslash \{x\}, x_1)$;
    if $(Fit(x_1) > Fit(x))$ **then**
      $x = x_1$
      Update the fitness values of individuals;
      Break;
    else
      $k = k + 1$
    end if
  end while
  $k = 0$
end for
Copy POS_t into P_{local} and identify N_{local} (the size of P_{local});
Generate P_{random} with the size $N_r = NP - N_{local}$ using Algorithm 5;
Get the new population $P_g = P_{local} + P_{random}$;
Optimize P_g with NSGA-II-DE ;

errors. To address this issue, CBP makes the predicted solutions and random solutions constitute the population in the new environment, in order to reduce the impact of inaccurate prediction.

$$C_t = \frac{1}{|POS_t|} \sum_{x_t \in POS_t} x_t \tag{10}$$

The prediction mechanism used in this paper is defined as Equation (11):

$$x_{t+1} = x_t + \parallel C_t - C_{t-1} \parallel + \varepsilon_t \tag{11}$$

where C_t and C_{t-1} are the center points at t and t-1 time steps, respectively. x_{t+1} represent the predicted individual at t+1 time step. The detailed process is illustrated in Algorithm 7.

Algorithm 7 Center-based prediction

Input: the POS_t at time t;the size of population NP;previous POS_{t-1} at time $t-1$;

Output: approximated POS;

1: Calculate the knee solution C_t of POS_t and C_{t-1} of POS_{t-1} according to Equation (10);

2: Generate new individuals using POS_t using Equation (11) , then check the boundary;

3: Copy them into P_{center} and identify N_{center} (the size of P_{center});

4: Generate P_{random} with the size $N_r = NP - N_{center}$ using Algorithm 5;

5: Get the new population $P_g = P_{center} + P_{random}$;

6: Optimize P_g with NSGA-II-DE ;

3.4. State-action table

From the empirical perspective, we use the following state-action table (Table 1) to reflect the suitability of the change response strategies. Q-learning utilizes this Q-table to determine which action to be applied for the given state explained in subsection 3.2. The Q values are updated as described in subsections 3.5, 3.6, and 3.7. The RL-DMOEA approach follows the agent's

instructions. When detecting environmental changes, signals are sent to the agent. Then, the agent selects the appropriate action based on the severity of environment changes and provides the algorithm with learned information. An agent evaluates the rewards of the selected operation and updates the corresponding Q value. Meanwhile, RL-DMOEA executes the action from the agent and relocates the individuals to areas near the new POF.

These actions are introduced in subsection 3.3. We classify the environmental changes into three severity levels, including slight (s_1), medium (s_2) and high (s_3)-severity environmental changes. The proposed RL-DMOEA could retrieve the best action from the Q-table in the current state, which has the maximum Q-value.

Table 1: State-action Table(Q-Table)

State	*Action*		
	$a_1(KBP)$	$a_2(ILS)$	$a_3(CBP)$
$s_1(0 \leq \varepsilon_t \leq 0.001)$	$Q(s_1, a_1)$	$Q(s_1, a_2)$	$Q(s_1, a_3)$
$s_2(0.001 \leq \varepsilon_t \leq 0.003)$	$Q(s_2, a_1)$	$Q(s_2, a_2)$	$Q(s_2, a_3)$
$s_3(\varepsilon_t > 0.003)$	$Q(s_3, a_1)$	$Q(s_3, a_2)$	$Q(s_3, a_3)$

3.5. Action selection policy

For the state s_t, an agent uses a selection policy π to select an action. For each candidate action a_i, the selection probability $Pr(s_t, a_i)$ is computed by the Equation (12) in the Q-table. The softmax function utilizes the selection probability, which is identified through using a Boltzmann distribution [35] ranking the value-function estimates: where τ refers to a positive parameter. We define τ as the $maxQ(s_t, a_i)$. K is the number of candidate actions.

$$\pi(a_i|s_t) = P_r\{a_i = a|s_t = s\} = \frac{e^{\frac{Q(s_t,a_i)}{\tau}}}{\sum_{i=1}^{K} e^{\frac{Q(s_t,a_i)}{\tau}}} \tag{12}$$

$$\tau = \max Q(s_t, a_i)$$

3.6. Reward of an action

After an action a is performed, a reward value r is used to evaluate the performance of a and update the Q(s,a) value. The hypervolume (HV) is a commonly used performance metric in DMOPs [14]. In this paper, HV is regarded as the reward r for executing the action a. The reference point is computed through the worst value for each objective in the current state s.

3.7. Update of the Q-value

The Q -value of $Q(s_t, a_t)$ is computed as follows:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \quad (13)$$

where the learning rate α is to control the learning speed.

3.8. Computational complexity

This subsection analyzed the computational complexity of the proposed RL-DMOEA. Here, M is the number of objectives and K refers to the number of non-dominated solutions in POF. N represents the population size. Considering the computational complexity, the overall complexity of the proposed RL-DMOEA is mainly affected by the three change response mechanisms and NSGA-II-DE. The time complexity of knee-based prediction includes two parts, MMD knee selection and prediction. The MMD knee selection takes $O(MK)$ computational complexity. And the computational complexity of prediction is $O(N)$. The indicator-based local search requires $O(MK^2)$ computational complexity. The center-based prediction, on the other hand, consumes $O(N)$ computational complexity. In addition, the theoretical time complexity of DE-NSGA-II is $O(MN^2)$. Obviously, $O(MN^2) > O(MK^2)$ and $O(MN^2) > O(N)$. Therefore, the proposed RL-DMOEA takes $O(MN^2)$ computational complexity.

4. Experimental study

4.1. Test instances

To solve DMOPs, the proposed RL-DMOEA is tested on twelve IEEE CEC 2015 [13] test problems, including four FDA instances, one DIMP instance, three

HE instances and four DMOP instances. The classification and properties of the benchmark function are depicted in Table 2. In the table, Type I test functions mean that POS changes, but POF remains fixed. Type II test functions illustrate that POF and POS both change. Type III test functions illustrate that POF changes, but POS remains unchanged. To be specific, FDA4, FDA5 and its variants refer to three-objective test functions, while the remaining test functions are two-objective ones. In the definition, t refers to $t = (1/n_t)\lfloor(\tau/\tau_t)\rfloor$. In addition, τ_t, n_t and τ means the frequency of change, the severity of change, and maximum number of iterations respectively.

Table 2: Summary of Benchmark Functions

Type	Function
TYPE I	FDA4, DIMP2, DMOP3
TYPE II	FDA5, FDA5$_{iso}$, FDA5$_{dec}$, DMOP2, DMOP2$_{iso}$, DMOP2$_{dec}$
TYPE III	HE2, HE7, HE9

4.2. Performance indicators

The commonly used performance metrics [47] are adopted in the experiment study, such as a variant of modified inverted generational distance (VMIGD), a variant of modified hypervolume (VMHV), and Schott's spacing metric (SP). These performance indicators are considered to measure the convergence and diversity.

4.2.1. A variant of modified inverted generational distance (VMIGD)

The inverted generational distance (IGD) [43] is adopted to evaluate the performance of the algorithm, including the both convergence and distribution. POF_t is a uniformly distributed true POF and POF_t^* is the POF obtained by the underlying algorithm at time t. The definition of IGD is shown as Equation

(14),

$$IGD(POF_t, POF_t^*) = \frac{\sum_{v \in POF_t} \min_{u \in POF_t^*} \parallel v - u \parallel}{|POF_t|} \tag{14}$$

MIGD metric, given by Equation (15), is the average IGD values over some time steps for a given run [14]. T represents a set of discrete time instances over a run and $|T|$ is the cardinality of T.

$$MIGD = \frac{1}{|T|} \sum_{t \in T} IGD(POF_t, POF_t^*) \tag{15}$$

VMIGD evaluates those algorithms under different (τ_t, n_t) configurations, and the definition of the VMIGD is as follows:

$$VMIGD = \frac{1}{|E|} \sum_{C \in E} MIGD(POF_t, POF_t^*, C) \tag{16}$$

where $|E|$ is the number of different (τ_t, n_t) configurations. There are three configurations and $|E|$ is set as three. The smaller VMIGD value indicates a better performance of the algorithm.

4.2.2. A variant of modified hypervolume (VMHV)

The hypervolume (HV) [14] measures the volume enclosed by the obtained POF. Let POF_t^* be obtained POF through the algorithm at time t. MHV takes the average of HV values in some time steps for a given run [14].

$$MHV = \frac{1}{|T|} \sum_{t \in T} HV(POF_t^*) \tag{17}$$

In the definition, HV(POF_t^*) means the hypervolume of POF_t^*. T represents a set of discrete time instances over a run, and $|T|$ is the cardinality of T. Under each environment, the reference point is $(z_1+0.2, z_2+0.2, ..., z_m+0.2)$ according to the empirical experiments, where z_j is the maximum value of the jth objective of the true POF and m is the number of objectives.

$$VMHV = \frac{1}{|E|} \sum_{C \in E} MHV(POF_t^*, C) \tag{18}$$

VMHV is similar to VMIGD. The larger VMHV value indicates a better performance of the algorithm.

4.2.3. Schott's spacing metric (SP)

The performance indicator is proposed in [28] to evaluate the distribution of obtained POF. The definition of SP is shown as follows:

$$SP = \sqrt{\frac{1}{|POF_t^*| - 1} \sum_{i=1}^{POF_t^*} (D_i - \overline{D})^2} \tag{19}$$

where D_i represents the Euclidean distance between the ith member in POF_t^* and its nearest member in POF_t^*. $\overline{D}$ refers to the average value of D_i.

4.3. Parameter settings

The parameter settings of chosen DMOEAs in the experiments are presented in this subsection. Five representative DMOEAs, including Tr-NSGA-II [14], DMS [25], PPS [44], MOEA/D-KF [23] and KPEA [48], are compared with RL-DMOEA. Because of their popularity and promising performance, they are chosen in testing different environments.

1) In all the experiments, the population size for all competing algorithms is set to 100.

2) The dimensions of decision variables are set according to [13].

3) In the parameter settings for RL-DMOEA, the learning rate α is 0.9 and the discount rate γ is set to 0.6. Moreover, the DE-based mutation and crossover operators are employed. The crossover probability is 0.8 and the mutation probability is 0.5.

4) The algorithm-specific parameters for Tr-NSGA-II, DMS, PPS, MOEA/D-KF and KPEA are set according to their original publications [14], [25], [44], [23] and [48], respectively.

5) The severity n_t is set to 10 and the frequency τ_t is set to 5,10 and 20.

6) All the algorithms run 20 independent times on benchmark problems. In addition, we set the total number of generations to $20\tau_t + 50$. This means that there are entire 20 changes for all the test problems and 50 generations are executed before the first environmental change is detected.

7) In all the algorithms, we randomly select 10 individuals as the detectors when the environmental changes are detected.

4.4. Experimental results and analysis

To thoroughly investigate the performance of the proposed RL-DMOEA, we conduct the experiments on all the test problems. The obtained average VMIGD, VMHV, and SP statistical results under different environment changes are summarized in Tables 3-5, respectively. In particular, we show the best average values in bold face. In addition, we further investigate and analyze the efficiency of the proposed RL-DMOEA.

Since VMIGD metrics depicts the differences between the approximate and the true POFs, we can use VMIGD to assess the performance of proposed algorithm. As can be observed in Table 3, obviously, in most cases, RL-DMOEA performs best on most of FDA, HE, and DMOP benchmark functions when evaluating VMIGD metric. It shows that RL-DMOEA has better convergence than the other five competing algorithms. However, Tr-NSGA-II performs better than RL-DMOEA on FDA4. For DIMP2 and DMOP2$_{dec}$, RL-DMOEA is slightly inferior to MOEA/D-KF and Tr-NSGA-II. In addition, RL-DMOEA also offers some better results on HE2, HE7 and HE9, which means that its Q-learning approach may be effective in dealing with type III DMOPs.

Table 4 shows the VMHV statistical results of six competing algorithms on all the test problems. The VMHV results are somewhat different from VMIGD values shown in Table 3. Different from Table 3, it shows that RL-DMOEA performs the best on six test problems but barely loses to Tr-NSGA-II on FDA4, DMOP2$_{dec}$ and variants of FDA5 in Table 4. RL-DMOEA shows a little worse performance compared with Tr-NSGA-II on DMOP2$_{dec}$. Clearly, the obtained VMHV values of DMS and KPEA are similar, which are reported in Table 4. Additionally, RL-DMOEA shows promising performance maybe because RL-DMOEA could predict the population according to the severity degree of environment changes.

For the SP metric, in Table 5, we can find that RL-DMOEA wins the first place on testing most FDA and DMOP problems, which indicates that it preserves a good distribution in most cases. It obtains worse results than Tr-NSGA-II on FDA4, FDA5$_{dec}$ and DIMP2. The DMS, PPS and MOEA/D-KF

perform better than RL-DMOEA for DMOP3, HE2 and HE9, respectively. Tr-NSGA-II wins second place on most test problems. For all the tested instances, the performances of DMS and MOEA/D-KF are almost similar.

According to the experimental study of VMIGD, VMHV, and SP on these competing algorithms, the following observations are obtained:

1) From Tables 3 and 4, the results measured by the VMIGD and VMHV metrics indicate that RL-DMOEA obtains significantly better results on most tested FDA, DMOP, and HE instances. The Table 5 shows that a good SP value is not necessarily related to VMIGD metric, which is given in the cases of Tr-NSGA-II on FDA4, FDA5 and its variants.

2) For most performance metrics and test instances, RL-DMOEA and Tr-NSGA-II are in the first and second places. The results show that RL-DMOEA could search for a well-converged and well-distributed POF. RL-DMOEA shows better performance mainly because reinforcement learning evolutionary framework selects the suitable response mechanism.

Compared with our previous proposed KPEA, the experimental results show that RL-DMOEA has clear advantages in handling DMOPs especially pursuing a well-converged and well-distributed Pareto front. The main reasons are illustrated as follows. RL-DMOEA focuses on searching for a well-distributed Pareto front and utilizes the reinforcement learning-based method, which learns more information from environment and accelerates the evolution process of searching for Pareto front. The reinforcement learning helps to enhance the searching ability of solving dynamic multi-objective optimization problems, which promotes a better performance in the experiments. KPEA aims to maintain non-dominated solutions around knee and boundary regions, ignoring the diversity during the evolutionary process. Although KPEA reduces the computational complexity, it seems not to obtain a well-distributed Pareto front which is reflected on worse VMIGD, VMHV and SP values.

Table 3: VMIGD values of five algorithms compared with the proposed RL-DMOEA.

VMIGD	RL-DMOEA	Tr-NSGA-II	DMS	PPS	MOEA/D-KF	KPEA
FDA4	1.058E-1(2.251E-2)	**7.218E-2(1.420E-2)**	1.128E-1(2.832E-2)	1.397E-1(2.209E-2)	1.902E-1(3.122E-2)	1.918E-1(2.724E-2)
FDA5	**1.363E-1(2.044E-2)**	1.724E-1(2.414E-2)	1.889E-1(2.440E-2)	1.747E-1(2.938E-2)	1.915E-1(3.016E-2)	1.897E-1(3.313E-2)
$FDA5_{iso}$	**1.124E-1(2.058E-2)**	1.365E-1(2.735E-2)	2.419E-1(3.019E-2)	2.080E-1(2.114E-2)	2.351E-1(3.106E-2)	2.844E-1(1.969E-2)
$FDA5_{dec}$	**3.020E-1(2.640E-2)**	3.048E-1(3.046E-2)	3.682E-1(2.550E-2)	3.406E-1(4.514E-2)	3.477E-1(3.613E-2)	3.862E-1(2.725E-2)
DIMP2	3.602E-1(2.317E-2)	4.760E-1(4.970E-2)	3.551E-1(3.541E-2)	3.776E-1(2.970E-2)	**3.237E-1(2.080E-2)**	5.351E-1(1.753E-2)
DMOP2	**1.681E-1(1.43E-2)**	2.089E-1(2.320E-2)	4.160E-1(2.011E-2)	3.710E-1(2.390E-2)	3.664E-1(2.850E-2)	5.615E-1(6.602E-2)
$DMOP2_{iso}$	**3.292E-1(1.330E-2)**	4.520E-1(1.962E-2)	5.065E-1(1.260E-2)	4.351E-1(3.571E-2)	4.803E-1(2.639E-2)	4.896E-1(3.417E-2)
$DMOP2_{dec}$	3.187E-1(2.702E-2)	**2.170E-1(2.151E-2)**	5.848E-1(3.052E-2)	5.070E-1(2.236E-2)	4.360E-1(2.177E-2)	5.545E-1(3.246E-2)
DMOP3	**1.086E-1(1.960E-2)**	3.062E-1(2.715E-2)	5.089E-1(3.193E-2)	4.324E-1(2.678E-2)	4.289E-1(2.405E-2)	5.252E-1(2.729E-2)
HE2	**6.221E-2(9.260E-3)**	1.583E-1(3.232E-2)	2.965E-1(3.964E-2)	3.881E-1(2.933E-2)	3.786E-1(2.807E-2)	3.781E-1(2.206E-2)
HE7	**5.152E-2(1.811E-2)**	1.653E-1(2.364E-2)	1.806E-1(3.242E-2)	2.020E-1(2.902E-2)	2.234E-1(2.368E-2)	2.996E-1(2.780E-2)
HE9	1.376E-1(2.903E-2)	1.684E-1(1.948E-2)	**1.144E-1(9.959E-3)**	2.714E-1(1.862E-2)	2.464E-1(3.420E-2)	3.609E-1(1.751E-2)

Table 4: VMHV values of five algorithms compared with the proposed RL-DMOEA.

VMHV	RL-DMOEA	Tr-NSGA-II	DMS	PPS	MOEA/D-KF	KPEA
FDA4	6.365E-1(1.068E-2)	**6.981E-1(1.071E-2)**	6.541E-1(1.247E-2)	6.381E-1(1.478E-2)	5.799E-1(1.014E-2)	5.852E-1(1.232E-2)
FDA5	5.944E-1(2.023E-2)	6.175E-1(1.064E-2)	5.918E-1(1.331E-2)	6.169E-1(1.147E-2)	**6.397E-1(2.009E-2)**	5.907E-1(1.780E-2)
$FDA5_{iso}$	5.660E-1(1.654E-2)	**5.753E-1(1.214E-2)**	5.625E-1(1.297E-2)	5.752E-1(1.388E-2)	5.744E-1(2.089E-2)	5.631E-1(1.574E-2)
$FDA5_{dec}$	5.643E-1(1.798E-2)	**5.872E-1(1.282E-2)**	5.472E-1(2.003E-2)	5.781E-1(1.713E-2)	5.612E-1(1.931E-2)	5.598E-1(1.106E-2)
DIMP2	**5.041E-1(1.489E-2)**	4.513E-1(2.047E-2)	4.699E-1(1.513E-2)	4.574E-1(1.812E-2)	4.642E-1(1.099E-2)	4.590E-1(2.036E-2)
DMOP2	**6.165E-1(1.669E-2)**	6.001E-1(1.355E-2)	5.951E-1(2.207E-2)	5.737E-1(2.472E-2)	5.424E-1(1.195E-2)	5.412E-1(1.806E-2)
$DMOP2_{iso}$	**4.399E-1(1.338E-2)**	4.379E-1(1.094E-2)	4.276E-1(1.350E-2)	4.383E-1(1.055E-2)	4.241E-1(1.403E-2)	4.294E-1(1.084E-2)
$DMOP2_{dec}$	5.312E-1(1.368E-2)	**5.531E-1(1.283E-2)**	4.658E-1(2.007E-2)	4.528E-1(1.528E-2)	4.235E-1(1.069E-2)	4.566E-1(1.816E-2)
DMOP3	**6.344E-1(1.203E-2)**	6.316E-1(1.438E-2)	5.955E-1(1.411E-2)	5.912E-1(2.056E-2)	5.767E-1(1.912E-2)	5.810E-1(1.244E-2)
HE2	**6.457E-1(1.446E-2)**	6.434E-1(1.905E-2)	6.319E-1(1.573E-2)	6.425E-1(1.529E-2)	6.260E-1(2.034E-2)	6.446E-1(1.195E-2)
HE7	**8.235E-1(1.501E-2)**	8.021E-1(1.659E-2)	7.357E-1(1.379E-2)	7.411E-1(1.282E-2)	7.486E-1(1.309E-2)	7.382E-1(1.026E-2)
HE9	7.184E-1(1.436E-2)	7.160E-1(1.410E-2)	7.024E-1(2.051E-2)	7.057E-1(1.654E-2)	**7.295E-1(1.943E-2)**	6.943E-1(1.714E-2)

Table 5: SP values of five algorithms compared with the proposed RL-DMOEA.

SP	RL-DMOEA	Tr-NSGA-II	DMS	PPS	MOEA/D-KF	KPEA
FDA4	5.521E-2(2.440E-3)	**5.430E-2(2.302E-3)**	6.247E-2(1.112E-3)	5.644E-2(2.843E-3)	9.796E-2(1.595E-3)	9.718E-2(4.304E-3)
FDA5	**4.266E-2(3.125E-3)**	6.761E-2(1.248E-2)	7.668E-2(4.875E-3)	6.604E-2(2.980E-3)	6.872E-2(2.061E-3)	7.069E-2(3.551E-3)
$FDA5_{iso}$	**8.671E-2(2.526E-3)**	9.516E-2(1.089E-2)	1.317E-1(3.341E-2)	1.439E-1(3.073E-3)	8.774E-2(3.817E-3)	1.335E-1(1.304E-2)
$FDA5_{dec}$	7.862E-2(5.334E-3)	**7.661E-2(3.676E-3)**	8.501E-2(5.368E-3)	9.392E-2(5.259E-3)	9.382E-2(4.950E-3)	1.007E-1(2.096E-2)
DIMP2	3.852E-2(2.492E-3)	**3.768E-2(1.144E-3)**	5.329E-2(3.689E-3)	6.479E-2(4.237E-3)	8.012E-2(3.185E-3)	8.099E-2(4.457E-3)
DMOP2	**4.658E-2(3.792E-3)**	6.372E-2(2.302E-3)	5.954E-2(2.782E-3)	8.696E-2(4.134E-3)	6.544E-2(3.280E-3)	6.909E-2(2.177E-3)
$DMOP2_{iso}$	**3.153E-2(4.290E-3)**	5.962E-2(5.270E-3)	5.236E-2(7.247E-2)	6.414E-2(5.363E-3)	5.763E-2(4.787E-3)	7.559E-2(5.215E-3)
$DMOP2_{dec}$	**3.291E-2(4.833E-3)**	2.941E-2(3.317E-3)	4.211E-2(1.204E-3)	8.356E-2(4.778E-3)	6.553E-2(3.419E-3)	7.249E-2(3.170E-3)
DMOP3	9.635E-2(2.394E-3)	9.353E-2(1.843E-3)	**4.874E-2(1.609E-3)**	9.872E-2(4.283E-3)	7.475E-2(2.958E-3)	8.435E-2(3.634E-3)
HE2	5.264E-2(3.286E-3)	4.925E-2(9.982E-3)	5.329E-2(3.783E-2)	**4.228E-2(2.996E-3)**	8.061E-2(3.264E-3)	7.898E-2(2.946E-3)
HE7	**6.451E-2(2.603E-3)**	6.724E-2(3.792E-3)	6.926E-2(3.844E-3)	6.982E-2(4.624E-3)	7.713E-2(4.084E-3)	7.880E-2(3.553E-3)
HE9	7.722E-2(8.9162E-3)	9.986E-2(1.469E-3)	8.137E-2(1.989E-3)	5.664E-2(2.028E-3)	**5.078E-2(8.440E-4)**	8.982E-2(2.907E-3)

4.5. Effectiveness of different components of RL-DMOEA

In addition, the effectiveness of different components of RL-DMOEA has been investigated. RL-DMOEA has following key components, namely the Q-learning evolutionary framework, knee-based prediction, center-based prediction and the indicator-based local search. In order to further investigate each component for dynamic optimization, we build three variants based on the original RL-DMOEA. The first variant (RL-DMOEA-s1) does not use Q-learning evolutionary framework, but randomly selects prediction mechanisms when detecting environment changes. The second variant (RL-DMOEA-s2) uses the Q-learning evolutionary framework. Specifically, RL-DMOEA-s2 detects and responds to environment changes through adopting center-based prediction and indicator-based local search mechanisms (without knee-based prediction). RL-DMOEA-s3 is another version which retains Q-learning evolutionary framework but using knee-based prediction and indicator-based local search mechanisms (without center-based prediction). The mentioned variants are compared with proposed RL-DMOEA on five test instances to verify the effectiveness.

Table 6 presents the results on RL-DMOEA and its variants on these three performance metrics. In terms of three metrics, RL-DMOEA shows the best

performance on most test problems. It means that all the key components are critical to the tracking ability of RL-DMOEA. For HE2, there is a significant difference between RL-DMOEA and competing variants on all the performance metrics. RL-DMOEA-s3 shows worse SP and VMIGD results than RL-DMOEA, while it obtains better VMHV values on HE2. In addition, the differences between RL-DMOEA-s1 and RL-DMOEA for all the test problems clearly indicate that the use of Q-learning framework can significantly enhance the tracking ability of RL-DMOEA. The obtained results indicate that RL-DMOEA shows better performance than other competing variants. This observation confirms the advantage of the Q-learning framework and change response mechanisms in RL-DMOEA.

We can conclude that RL-DMOEA generally outperforms other compared variants based on the experimental study. The observation obviously shows that each component is indispensible in addressing environmental changes. In addition, we specifically explain the role for each component. The MMD knee-based prediction and center-based prediction explores high-severity and medium-severity environmental changes and respond rapidly to enhance the convergence speed. The indicator-based local search exploits the information of slight-severity environmental changes and relocates individuals near the POF, thus tracking the varying POF quickly. The Q-learning framework promotes rapid convergence speed and responds to varying degrees of environment changes. Besides, it uses these change response mechanisms to feed back on different environment changes and obtains a delicate balance in exploration-exploitation trade-off. In summary, all these key components of RL-DMOEA play important roles in responding to changes and adapting rapidly to various dynamic environments.

4.6. Computational time

In order to examine the computational efficiency of the proposed RL-DMOEA, the statistical results on the total computational time are obtained by the compared algorithms when (τ_t, n_t) are set as (5,10), (10,10), and (20,10). All the algorithms run independently on benchmark problems with parameter configu-

Table 6: Performance results on RL-DMOEA variants

Problem	Performance metrics	RL-DMOEA-s1	RL-DMOEA-s2	RL-DMOEA-s3	RL-DMOEA
FDA5	VMIGD	3.408E-1(3.128E-2)	3.023E-1(3.468E-2)	2.477E-1(2.362E-2)	**1.363E-1(2.044E-2)**
	VMHV	5.322E-1(1.672E-2)	5.663E-1(1.828E-2)	5.897E-1(1.918E-2)	**5.944E-1(2.023E-2)**
	SP	8.788E-2(3.405E-3)	6.344E-2(4.492E-3)	5.891E-2(4.083E-3)	**4.266E-2(3.125E-3)**
DMOP2	VMIGD	3.584E-1(4.136E-2)	2.839E-1(3.143E-2)	3.026E-1(2.647E-2)	**1.681E-1(1.43E-2)**
	VMHV	5.486E-1(1.631E-2)	5.859E-1(1.072E-2)	6.078E-1(2.343E-2)	**6.165E-1(1.669E-2)**
	SP	5.915E-2(3.604E-3)	5.098E-2(5.438E-3)	5.023E-2(4.381E-3)	**4.658E-2(3.792E-3)**
HE7	VMIGD	1.278E-1(2.341E-2)	1.091E-1(3.932E-2)	9.265E-2(2.011E-2)	**5.152E-2(1.811E-2)**
	VMHV	7.649E-1(1.925E-2)	7.975E-1(1.153E-2)	7.988E-1(2.028E-2)	**8.235E-1(1.501E-2)**
	SP	7.983E-2(2.874E-3)	7.308E-2(3.780E-3)	7.112E-2(3.092E-3)	**6.451E-2(2.603E-3)**
HE2	VMIGD	1.032E-1(2.413E-2)	1.001E-1(1.409E-2)	8.011E-2(3.776E-2)	**6.221E-2(9.260E-3)**
	VMHV	6.287E-1(1.882E-2)	6.398E-1(1.124E-2)	**6.747E-1(2.171E-2)**	6.457E-1(1.446E-2)
	SP	8.309E-2(3.671E-3)	6.617E-2(2.584E-3)	7.049E-2(3.751E-3)	**5.264E-2(3.286E-3)**
DMOP3	VMIGD	1.429E-1(2.509E-2)	1.119E-1(2.733E-2)	1.099E-1(2.093E-2)	**1.086E-1(1.960E-2)**
	VMHV	6.253E-1(1.149E-2)	6.280E-1(2.474E-2)	6.323E-1(1.924E-2)	**6.344E-1(1.203E-2)**
	SP	1.251E-1(2.391E-3)	1.080E-1(4.843E-3)	1.057E-1(3.460E-3)	**9.635E-2(2.394E-3)**

rations (τ_t, n_t). As reported in Table 7, it can be concluded that KPEA shows the best computational results, which is considered the fastest algorithm. RL-DMOEA wins the second place, which is a little worse than KPEA, validated by the average running time on all test instances. The performances of DMS and PPS are almost similar. MOEA/D-KF is the most time-consuming design compared with other competing algorithms. The reason why MOEA/D-KF and Tr-NSGA-II consume much time may lie in employing complex prediction models. The Kalman Filter-based model takes long time to relocate the individuals in the new environment in MOEA/D-KF. Tr-NSGA-II utilizes the transfer learning model to predict the new locations of optimal solutions. On the other hand, KPEA is integrated with MCDM strategies, which mainly focus on searching solely around the knee-guided region. Instead of maintaining a number of non-dominated solutions, it directly searches for the knee and boundary regions to speed up the convergence and reduce the computational complexity. Therefore, KPEA outperforms other compared algorithms on computational time. RL-DMOEA applies a relatively time-saving RL-based prediction framework, which consumes more time than KPEA but much less time than other prediction models. This is the main reason why RL-DMOEA is the second fastest algorithm compared with other competing algorithms.

4.7. Parameter sensitivity analysis

In this subsection, the influence of four parameters which include the population size, the learning rate, the discount rate, and the severity is discussed. The sensitivity of four parameters in RL-DMOEA is analyzed and all obtained results are averaged over 20 independent runs.

4.7.1. Influence of the population size

In order to analyze the sensitivity of the population size, we set it to 50, 100 and 150, respectively. Experimental results on the VMIGD performance metric and computational time are shown in Table 8. It can be observed that RL-DMOEA performs convincingly well when the population size is set to 100.

Table 7: Computational time of five algorithms compared with the proposed RL-DMOEA.

Prob.	RL-DMOEA	Tr-NSGA-II	DMS	PPS	MOEA/D-KF	KPEA
FDA4	83.177	308.124	252.797	250.084	1735.25	**60.291**
FDA5	83.692	311.556	256.226	255.192	1904.66	**60.194**
FDA5iso	84.713	313.243	256.813	255.478	2008.39	**65.237**
FDA5dec	83.208	312.998	257.002	256.385	2010.644	**66.325**
DIMP2	77.966	307.161	253.489	250.657	1666.945	**58.825**
DMOP2	79.09	304.133	252.166	251.121	1678.324	**60.997**
DMOP2iso	79.067	305.752	252.362	251.444	1687.35	**61.214**
DMOP2dec	80.195	305.91	253.998	252.229	1690.098	**61.214**
DMOP3	73.48	306.898	252.104	251.208	1645.898	**60.607**
HE2	70.688	302.199	250.321	249.19	1632.79	**63.1**
HE7	70.858	302.672	250.483	249.478	1634.144	**64.198**
HE9	69.908	303.254	250,05	249.004	1644.22	**63.144**

Obviously, reducing the population size will result into less computational time but cause a poor convergence in RL-DMOEA. The VMIGD values on the chosen test problems are similar when the population size is set to 100 and 150, while the computational time is appreciably different. The computational time increases from 83.692 seconds to 111.072 seconds when the population size increases from 100 to 150 on FDA5. As a compromise, RL-DMOEA is considered more effective in solving these DMOP benchmark problems when the population size is set to 100.

4.7.2. Influence of the learning rate

The learning rate α is an important parameter influencing the efficiency of the proposed RL-DMOEA. The larger the learning rate α , the more information is learned from the reward. If α is small, the proposed RL-DMOEA is not able to learn the environmental changes very well and may not select the appropriate prediction-based strategy to generate non-dominated solutions. The

Table 8: Experimental results of different population size in RL-DMOEA.

Prob.	Population size	VMIGD	CPU time
FDA5	50	2.036E-1(1.438E-2)	64.72
	100	1.363E-1(2.044E-2)	83.692
	150	1.362E-1(1.038E-2)	111.072
DMOP2	50	2.206E-1(1.518E-2)	54.724
	100	1.681E-1(1.43E-2)	79.09
	150	1.679E-1(1.071E-2)	98.387
HE2	50	1.178E-1(1.119E-2)	57.65
	100	6.221E-2(9.260E-3)	70.688
	150	6.220E-2(8.912E-3)	91.713
DMOP3	50	1.787E-1(1.632E-2)	53.287
	100	1.086E-1(1.960E-2)	73.48
	150	1.083E-1(1.495E-2)	92.326

performance of RL-DMOEA on four selected test problems for different learning rate values is examined. It can be seen that the RL-DMOEA obtains the best results when the learning rate is set to 0.9 in Table 9. As the learning rate increases, the obtained statistic results are better at the cost of convergence rate, which indicates that the learning rate has an influence on the proposed algorithm. From the experimental results, we recommend that the learning rate should be set to 0.9.

4.7.3. Influence of the discount rate

The discount rate γ affects the future rewards and the convergence of the algorithm [35]. The larger the γ , the more focus the algorithm will concentrate on the future rewards. If γ is small, the algorithm only focuses on the immediate reward. This parameter needs to be tuned for the proposed RL-DMOEA. The statistic results of sensitivity analysis for γ are shown in Table 10. Four different test problems including FDA5, DMOP2, HE2 and DMOP3 are selected for conducting the comparison experiments. It can be observed from Table 10 that the VMIGD and VMHV values are the best when the discount rate is set to 0.6. A median discount rate indicates that it not only facilitates the convergence

Table 9: Experimental results of different learning rate in RL-DMOEA.

Prob.	Learning rate	VMIGD	VMHV
FDA5	0.3	2.061E-1(1.902E-2)	5.305E-1(1.704E-2)
	0.6	1.797E-1(1.467E-2)	5.560E-1(1.359E-2)
	0.9	**1.363E-1(2.044E-2)**	**5.944E-1(2.023E-2)**
DMOP2	0.3	2.310E-2(1.474E-2)	5.918E-1(1.056E-2)
	0.6	2.048E-1(1.831E-2)	5.934E-1(1.559E-2)
	0.9	**1.681E-1(1.43E-2)**	**6.165E-1(1.669E-2)**
HE2	0.3	1.537E-1(2.001E-2)	6.183E-1(1.723E-2)
	0.6	1.191E-1(1.418E-2)	6.206E-1(1.055E-2)
	0.9	**6.221E-2(9.260E-3)**	**6.457E-1(1.446E-2)**
DMOP3	0.3	1.808E-1(1.317E-2)	6.119E-1(1.817E-2)
	0.6	1.614E-1(2.011E-2)	6.126E-1(1.236E-2)
	0.9	**1.086E-1(1.960E-2)**	**6.344E-1(1.203E-2)**

of the RL-DMOEA, but also learns enough information on the future rewards. Therefore, the discount rate γ is recommended to be set to 0.6.

4.7.4. Influence of the change severity

To investigate the influence of different change severity levels, we carry out the experiments on FDA5 and HE2 test problems. The change frequency τ_t is set to 10, and the change severity n_t is set to 5, 10, and 20, respectively. The VMIGD values are plotted in Fig. 4, which compare the proposed RL-DMOEA with the best algorithm Tr-NSGA-II among the competing algorithm on test problems considered. As shown in Fig. 4, our proposed RL-DMOEA shows better performance than Tr-NSGA-II. The change severity reduces as n_t increases, which is reflected on the obtained VMIGD values. In Fig. 4, the box describes the distribution of the VMIGD values and the lower quartile, the median, and the upper quartile values. The whiskers are lines extending from each end of the box to show the extent of the remaining data. The box-plots indicate that the proposed RL-DMOEA is robust under different severity of environmental changes.

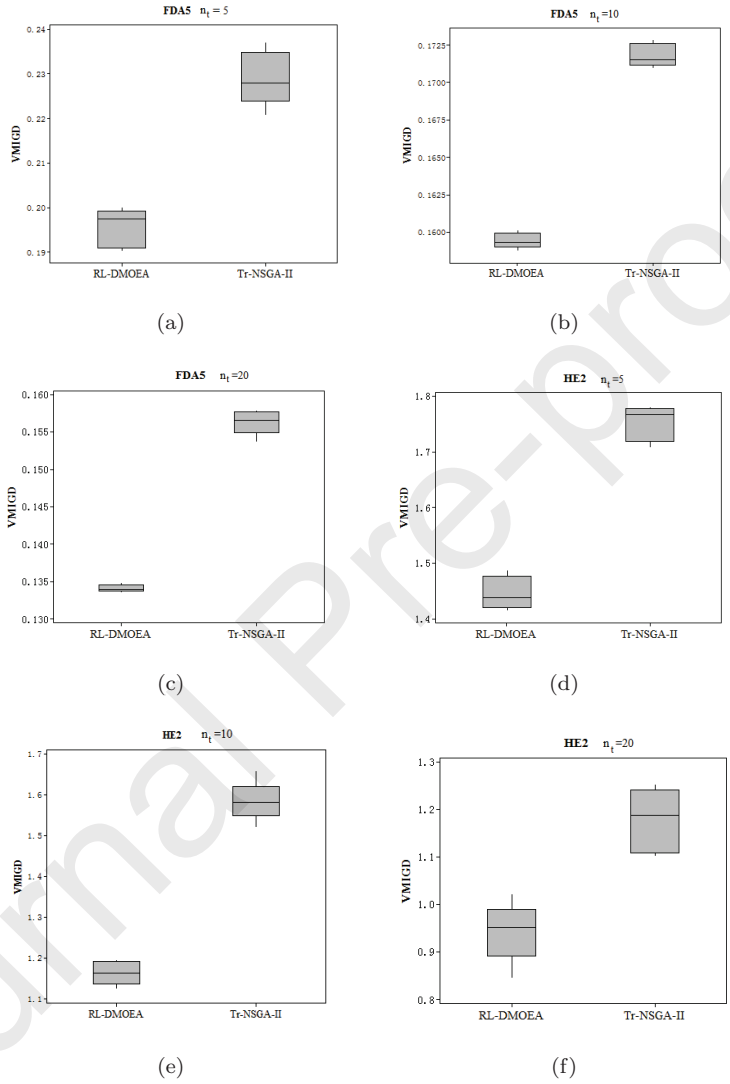

Fig. 4. Influence of the change severity on FDA5 and HE2 problems. τ_t is set to 10. The figures show the box plot of VMIGD values of the proposed RL-DMOEA and its competing algorithm Tr-NSGA-II when n_t is set to 5, 10, and 20, respectively.

Table 10: Experimental results of different discount rate in RL-DMOEA.

Prob.	discount rate	VMIGD	VMHV
FDA5	0.3	1.974E-1(2.095E-2)	5.779E-1(1.933E-2)
	0.6	**1.363E-1(2.044E-2)**	**5.944E-1(2.023E-2)**
	0.9	2.131E-1(1.288E-2)	5.737E-1(1.295E-2)
DMOP2	0.3	2.047E-1(1.614E-2)	5.964E-1(1.474E-2)
	0.6	**1.681E-1(1.43E-2)**	**6.165E-1(1.669E-2)**
	0.9	2.312E-1(1.244E-2)	5.930E-1(1.620E-2)
HE2	0.3	1.297E-1(1.159E-2)	6.332E-1(1.001E-2)
	0.6	**6.221E-2(9.260E-3)**	**6.457E-1(1.446E-2)**
	0.9	1.517E-1(1.196E-2)	6.238E-1(1.771E-2)
DMOP3	0.3	1.978E-1(2.039E-2)	6.192E-1(1.598E-2)
	0.6	**1.086E-1(1.960E-2)**	**6.344E-1(1.203E-2)**
	0.9	1.979E-1(1.504E-2)	6.258E-1(1.399E-2)

5. Conclusion

This article devised an RL-DMOEA algorithm to address dynamic multi-objective optimization problems which have time-dependent features. Different from other approaches, a reinforcement learning technique is incorporated to enhance the tracking ability. The change response mechanisms including knee-based prediction, center-based prediction and indicator-based local search are incorporated to improve both local and global tracking ability, facilitate the convergence speed and preserve population diversity. This algorithm framework can perceive the state of the current environment, and select the appropriate prediction mechanism in the evolutionary process based on obtained information and experience by the Q-learning agent. The experiments with chosen representative algorithms verify the efficacy of RL-DMOEA on CEC 2015 benchmark problems. The experimental studies have shown that RL-DMOEA can track their changing POFs effectively on most of the test problems considered.

We will concentrate on combining machine learning methods with DMOPs in the future. In addition, we believe that the proposed RL-DMOEA is useful to tackle real-world problems with various uncertainties.

Acknowledgments

This work was supported by the Fund for Innovative Research Groups of the National Natural Science Foundation of China (71621061), the Major International Joint Research Project of the National Natural Science Foundation of China (71520107004), the Major Program of National Natural Science Foundation of China (71790614) and the 111 Project (B16009).

References

[1] V. S. Aragón, S. C. Esquivel, and C. Coello Coello. Evolutionary multiobjetive optimization in non-stationary environments. *Journal of Computer Science & Technology*, 5, 2005.

[2] 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*, 21(4):885–906, 2017.

[3] J. Branke, K. Deb, H. Dierolf, and M. Osswald. Finding knees in multiobjective optimization. In *International conference on parallel problem solving from nature*, pages 722–731. Springer, 2004.

[4] Y. Bravo, G. Luque, and E. Alba. Global memory schemes for dynamic optimization. *Natural Computing*, 15(2):319–333, 2016.

[5] M. Cámara, J. Ortega, and F. de Toro. A single front genetic algorithm for parallel multi-objective optimization in dynamic environments. *Neurocomputing*, 72(16-18):3570–3579, 2009.

[6] R. Chen, K. Li, and X. Yao. Dynamic multiobjectives optimization with a changing number of objectives. *IEEE Transactions on Evolutionary Computation*, 22(1):157–171, 2017.

[7] W.-Y. Chiu, G. G. Yen, and T.-K. Juan. Minimum manhattan distance approach to multiple criteria decision making in multiobjective optimization

problems. *IEEE Transactions on Evolutionary Computation*, 20(6):972–985, 2016.

[8] C. A. C. Coello, G. T. Pulido, and M. S. Lechuga. Handling multiple objectives with particle swarm optimization. *IEEE Transactions on evolutionary computation*, 8(3):256–279, 2004.

[9] C. Cruz, J. R. González, and D. A. Pelta. Optimization in dynamic environments: a survey on problems, methods and measures. *Soft Computing*, 15(7):1427–1448, 2011.

[10] K. Deb, S. Karthik, et al. Dynamic multi-objective optimization and decision-making using modified nsga-ii: a case study on hydro-thermal power scheduling. In *International conference on evolutionary multi-criterion optimization*, pages 803–817. Springer, 2007.

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

[12] C.-K. Goh and K. C. Tan. A coevolutionary paradigm for dynamic multiobjective optimization. In *Evolutionary Multi-objective Optimization in Uncertain Environments*, pages 153–185. Springer, 2009.

[13] M. Helbig and A. Engelbrecht. Benchmark functions for cec 2015 special session and competition on dynamic multi-objective optimization. *Tech. Rep.*, 2015.

[14] M. Jiang, Z. Huang, L. Qiu, W. Huang, and G. G. Yen. Transfer learning-based dynamic multiobjective optimization algorithms. *IEEE Transactions on Evolutionary Computation*, 22(4):501–514, 2018.

[15] S. Jiang and S. Yang. A steady-state and generational evolutionary algorithm for dynamic multiobjective optimization. *IEEE Transactions on evolutionary Computation*, 21(1):65–82, 2016.

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

[17] H. Li and Q. Zhang. Multiobjective optimization problems with complicated pareto sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, 13(2):284 – 302, 2009.

[18] K. Li, S. Kwong, J. Cao, M. Li, J. Zheng, and R. Shen. Achieving balance between proximity and diversity in multi-objective evolutionary algorithm. *Information Sciences*, 182(1):220–242, 2012.

[19] H. Liao, Q. Wu, and L. Jiang. Multi-objective optimization by reinforcement learning for power system dispatch and voltage stability. In *2010 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT Europe)*, pages 1–8. IEEE, 2010.

[20] M. Liu, J. Zheng, J. Wang, Y. Liu, and L. Jiang. An adaptive diversity introduction method for dynamic evolutionary multiobjective optimization. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 3160–3167. IEEE, 2014.

[21] R. Liu, J. Li, C. Mu, L. Jiao, et al. A coevolutionary technique based on multi-swarm particle swarm optimization for dynamic multi-objective optimization. *European Journal of Operational Research*, 261(3):1028–1051, 2017.

[22] R. Liu, W. Zhang, L. Jiao, F. Liu, and J. Ma. A sphere-dominance based preference immune-inspired algorithm for dynamic multi-objective optimization. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 423–430. ACM, 2010.

[23] A. Muruganantham, K. C. Tan, and P. Vadakkepat. Evolutionary dynamic multiobjective optimization via kalman filter prediction. *IEEE transactions on cybernetics*, 46(12):2862–2873, 2015.

[24] Z. Peng, J. Zheng, and J. Zou. A population diversity maintaining strategy based on dynamic environment evolutionary model for dynamic multiobjective optimization. In *2014 IEEE Congress on Evolutionary Computation (CEC)*, pages 274–281. IEEE, 2014.

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

[26] M. Ruiz-Montiel, L. Mandow, and J.-L. Perez-de-la Cruz. A temporal difference method for multi-objective reinforcement learning. *Neurocomputing*, 263:15–25, 2017.

[27] H. Samma, C. P. Lim, and J. M. Saleh. A new reinforcement learning-based memetic particle swarm optimizer. *Applied Soft Computing*, 43:276–297, 2016.

[28] J. R. Schott. Fault tolerant design using single and multicriteria genetic algorithm optimization. Technical report, AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH, 1995.

[29] A. Schwartz. A reinforcement learning method for maximizing undiscounted rewards. In *Proceedings of the tenth international conference on machine learning*, volume 298, pages 298–305, 1993.

[30] R. Shang, L. Jiao, Y. Ren, L. Li, and L. Wang. Quantum immune clonal coevolutionary algorithm for dynamic multiobjective optimization. *Soft Computing*, 18(4):743–756, 2014.

[31] X.-N. Shen, L. L. Minku, N. Marturi, Y.-N. Guo, and Y. Han. A Q-learning-based memetic algorithm for multi-objective dynamic software project scheduling. *Information Sciences*, 428:1 – 29, 2018.

[32] K. Sindhya. Hybrid evolutionary multi-objective optimization with enhanced convergence and diversity. *Jyväskylä studies in computing*, (131), 2011.

[33] S. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvári. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine learning*, 38(3):287–308, 2000.

[34] R. S. Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

[35] R. S. Sutton and A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[36] G. Tesauro. Practical issues in temporal difference learning. In *Advances in neural information processing systems*, pages 259–266, 1992.

[37] Y. Wang and B. Li. Investigation of memory-based multi-objective optimization evolutionary algorithm in dynamic environment. In *2009 IEEE Congress on Evolutionary Computation*, pages 630–637. IEEE, 2009.

[38] C. J. Watkins and P. Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[39] Y. G. Woldesenbet and G. G. Yen. Dynamic evolutionary algorithm with variable relocation. *IEEE Transactions on Evolutionary Computation*, 13(3):500–513, 2009.

[40] Y. Wu, Y. Jin, and X. Liu. A directed search strategy for evolutionary dynamic multiobjective optimization. *Soft Computing*, 19(11):3221–3235, 2015.

[41] J. Zhang, Z.-h. Zhan, Y. Lin, N. Chen, Y.-j. Gong, J.-h. Zhong, H. S. Chung, Y. Li, and Y.-h. Shi. Evolutionary computation meets machine learning: A survey. *IEEE Computational Intelligence Magazine*, 6(4):68–75, 2011.

[42] X. Zhang, Y. Tian, and Y. Jin. A knee point-driven evolutionary algorithm for many-objective optimization. *IEEE Transactions on Evolutionary Computation*, 19(6):761–776, 2014.

[43] Z. Zhang. Multiobjective optimization immune algorithm in dynamic environments and its application to greenhouse control. *Applied Soft Computing*, 8(2):959–971, 2008.

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

[45] A. Zhou, Y. Jin, Q. Zhang, B. Sendhoff, and E. Tsang. Prediction-based population re-initialization for evolutionary dynamic multi-objective optimization. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 832–846. Springer, 2007.

[46] E. Zitzler and S. Künzli. Indicator-based selection in multiobjective search. In *International Conference on Parallel Problem Solving from Nature*, pages 832–842. Springer, 2004.

[47] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on evolutionary computation*, 7(2):117–132, 2003.

[48] F. Zou, G. G. Yen, and L. Tang. A knee-guided prediction approach for dynamic multi-objective optimization. *Information Sciences*, 509:193–209, 2020.