Swarm and Evolutionary Computation 49 (2019) 234–244

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

journal homepage: www.elsevier.com/locate/swevo

Differential evolution based on reinforcement learning with fitness ranking for solving multimodal multiobjective problems

Zhihui Li [a], Li Shi [b], Caitong Yue [a], Zhigang Shang [a,*], Boyang Qu [c]

[a] School of Electrical Engineering, Zhengzhou University, Zhengzhou 450001, China
[b] Department of Automation, Tsinghua University, Beijing 100000, China
[c] School of Electronic and Information Engineering, Zhongyuan University of Technology, Zhengzhou 450007, China

ARTICLE INFO

Keywords:
Multimodal multiobjective optimization problem
Differential evolution
Reinforcement learning
Fitness ranking
Q-learning

ABSTRACT

In multimodal multiobjective optimization problems (MMOOPs), there is more than one Pareto-optimal Set (PS) in the decision space corresponding to the same Pareto Front(PF). How to dynamically adjust the evolution direction of the population adaptively is a key problem, to ensure approaching the PF in the global sense with good convergence while finding out more PSs. In this paper, a novel Differential Evolution algorithm based on Reinforcement Learning with Fitness Ranking (DE-RLFR) is proposed. The DE-RLFR is based on the Q-learning framework, and each individual in the population is considered an agent. The fitness ranking values of each agent are used to encode hierarchical state variables. Three typical DE mutation operations are employed as optional actions for the agent. Based on the analysis of the distribution characteristics of the population in objective space, decision space and fitness-ranking space, we design a reward function of the ⟩*state*, *action*⟨ pairs to guide the population to move to the PF asymptotically. According to its reinforcement learning experience represented by the corresponding *Q* table value, each agent could adaptively select a mutation strategy to generate offspring individuals. The evaluation results on eleven MMOOP test functions show that DE-RLFR could quickly and effectively find multiple PSs in the decision space, and approach PF in the global sense.

1. Introduction

The multiobjective optimization problem (MOOP) has an important theoretical and practical application value and has been the focus of many fields such as evolutionary computation, reinforcement learning and optimal scheduling [1–6]. In practical applications, sometimes there is more than one global or local Preto-optimal sets (PSs) for many MOOPs, which are at different positions in the decision space. Such an optimization problem is called a multimodal multiobjective optimization problem (MMOOP). These PSs may meet the different needs of decision makers. For a general multiobjective optimization algorithms, finding one complete PS in the decision space indicates that the problem has been solved. However, in the MMOOP, as many PSs should be retained as possible to provide decision makers with more choices.

Some scholars have studied MMOOP. Deb proposed a MMOOP when designing the test problem in Ref. [7]. Afterwards, he designed the Omni-optimizer [8] algorithm to solve different types of optimization problems, including single/multi-objective, single/multimodal optimization. Rudolph et al. [9] analyzed the ability of multiple evolutionary multiobjective optimization algorithms to retain multiple PSs, and presented several multimodal multiobjective test functions. Ishibuchi et al. [10] proposed multimodal multiobjective test problems to visualize the diversity in the decision space. P. Kerschke analyzed multimodality of multiobjective landscapes in Ref. [11]. Liang et al. proposed some multimodal multiobjective test problems in Refs. [12–14] and a MMOPs benchmark suite with various properties in Ref. [15] to promote development in this field. Some new algorithms focused on solving multimodal multiobjective optimization problems were proposed. Recently, Liang et al. [12] designed a DN-NSGAII algorithm, which combined a new crowding distance with NSGAII to improve the diversity in the decision space for such problems. Subsequently, Yue et al. [13] proposed a multiobjective particle swarm optimizer using ring topology (MO_Ring_PSO_SCD) which used a niching method in the decision space to maintain the diversity for the decision space without degrading their distribution in the objective space to solve MMOOPs. A decomposition-based MOEA is proposed by associating two or more solutions to each

* Corresponding author.
E-mail addresses: lizhrain@zzu.edu.cn (Z. Li), shili@zzu.edu.cn (L. Shi), zhigang_shang@zzu.edu.cn (Z. Shang).

https://doi.org/10.1016/j.swevo.2019.06.010
Received 15 December 2018; Received in revised form 26 June 2019; Accepted 28 June 2019
Available online 6 July 2019

decomposed sub-problem to maintain diversity in the decision space in Ref. [16]. In Ref. [14], Liang proposed a multimodal multiobjective differential evolution algorithm (MMODE) with a decision-variable preselection scheme that promotes diversity of solutions in both the decision and objective space. An improved PSO algorithm with a self-organizing map network (SMPSO-MM) [17] is proposed to build a more reasonable neighborhood in the decision space and improve the diversity in the decision space with the help of a self-organizing map. Recently, TriMOEA-TA&R, which employed a clustering strategy and a niche-based clearing strategy, was proposed to guarantee the diversity in both the objective and decision spaces [5]. Liu et al. [18] proposed a double-niched evolutionary algorithm for MMOOP, which emploied a niche sharing method to increase the diversity of the solution set in both the objective and decision spaces. More progress and open challenges concerning in multiobjective optimization and multimodal optimization are reviewed systematically in Ref. [3]. Although these improved algorithms perform well, the obtained optimal solution sets for some problems are not complete, the distribution of PSs is uneven, and thus better algorithms are still needed to solve such problems.

The difficulty of solving the MMOOP lies in how to maintain the diversity of population in both decision and objective space. It is necessary not only to ensure that the PSs are found, but also to prevent premature of population with convergence to a specific PS. In order to solve this problem, this paper proposes a hierarchical evolutionary strategy in combination with a reinforcement learning mechanism. Each individual in the population is regarded as an agent. Based on a Q-learning framework, the agent selects a differential evolutionary mutation strategy using the Q table, according to its hierarchical state encoded by fitness-ranking in the population, to generate offspring individuals. The specifically designed reward function will guide the individual to go to the adjacent higher hierarchical state zone, or to stay in the current hierarchical state zone which possibly has PS. Through heuristic trial-and-error and evolution, the entire population will explore the decision space gradually, maintain good diversity, and converge to the true PSs.

The structure of the rest of this paper is as follows. The second section introduces the related definitions and work. The third section analyzes the distribution characteristics of the population of MMOOP in objective space, decision space and fitness-ranking space. Section 4 introduces the proposed DE-RLFR algorithm for MMOOP in detail. The fifth section gives the performance evaluation results of the algorithm on 11 multimodal multiobjective optimization test functions. Finally, the conclusions and future work are given in section 6.

2. Related definitions and works

2.1. Multimodal multiobjective optimization

Multimodal multiobjective optimization belongs to MOOP. MOOP refers to those which have two or more conflicting objectives to be optimized, which is usually described as:

$$\begin{aligned} &\text{Mimimize} \quad F(X) = [f_1(X), f_2(X), \ldots, f_m(X)] \\ &\text{Subject to} \quad X \in S \end{aligned} \tag{1}$$

where $f_i : \mathbb{R}^n \to \mathbb{R}$ are $m(\geq 2)$ fitness functions to describe m conflicting goals that need to be minimized at the same time, $X = [x_1, x_2, \ldots x_n]$ is a solution vector in a nonempty feasible domain $S \subset \mathbb{R}^n$, which is defined by constraints. For the non-constrained MOOP, the feasible domain is the definition domain of X. In multiobjective optimization, the improvement of one fitness function may cause the performance degradation of another fitness function or several others. When comparing the value of different solutions in multiobjective optimization, the dominant relationship is generally adopted.

Definition 1. Let X and Y be two solutions to the MOOP. X dominates Y (*record as*$X \prec Y$) when both of the following two conditions are met.

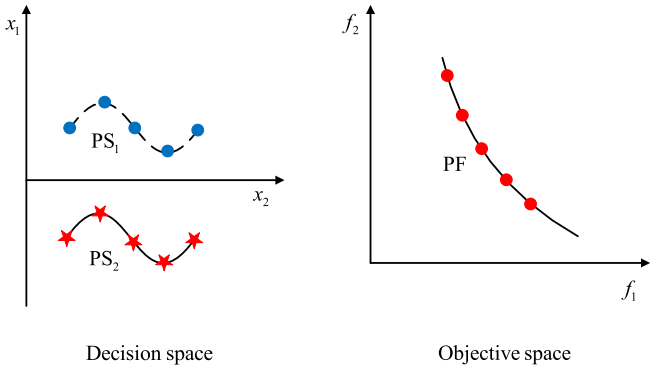


Fig. 1. Many-to-one mapping feature between PSs and PF in MMOOP.

Condition 1: X is not worse than Y according to all the objectives;

$$f_i(X) \leq f_i(Y) \quad i = 1, 2, \ldots, n \tag{2}$$

Condition 2: X is absolutely better than Y according to at least one objective.

$$f_j(X) < f_j(Y) \quad j \in \{1, \ldots, n\} \tag{3}$$

The set composed of all non-dominant solutions in the decision space is called a Pareto-optimal set (PS), and the set of vectors corresponding to PS in the objective space is called a Pareto Front (PF). MMOOP has multiple PSs in the decision space corresponding to the same PF in the objective space. A typical MMOOP is shown in Fig. 1, where PS_1 and PS_2 correspond to the same PF. Because of the many-to-one mapping feature from the decision space to the objective space, it is more difficult to balance the global and local search in MMOOP than the general MOOP.

At present, for general MOOP problems, there are mainly three types of evolutionary algorithms: 1) The multiobjective evolutionary algorithm based on non-dominant sorting. NSGA-II is the most representative one of these [19]. This kind of algorithm sorts the particles according to the non-dominant relation and encourages the population to converge to the PF. Its advantage is that it is easy to understand and can be applied to a wide application range, while its disadvantage is high computational complexity and poor performance in many-objective optimization. Thus, Deb proposed NSGAIII in 2014 to solve the many-objective optimization problem [20]. 2) The decomposition-based multiobjective optimization algorithm, such as the representative algorithm MOEA/D [21]. This, decomposes the MOOP into multiple single objective optimization problems. It is low in computational complexity, but is sensitive to the shape of PF. 3) The indicator-based multiobjective evolutionary algorithms [22]. The representative algorithms are SPEA2 [23] and HV based MOO [24]. Such algorithms assign a reasonable indicator value to each individual, and the population is sorted according to indicator value. The advantage of this algorithm is that it converges fast, but the disadvantage is that it is easy to fall into the local optima. For MMOOP, more attention should be paid to the distribution characteristics of the population in the decision space, and these algorithms do not give too much consideration to this aspect.

2.2. Differential evolution algorithm

The differential evolution (DE) algorithm is a parallel and global search method based on differential steps of the individuals. It solves the optimization problem through cooperation and competition among individuals within the population. Optimization search in DE is carried out by mutation, crossover and selection in each iteration.

Table 1
Three typical DE mutation operators.

	Differential Expression
DE/rand/1/bin	$v_i = x_{r1} + F(x_{r2} - x_{r3})$
DE/best/1/bin	$v_i = x_{best} + F(x_{r1} - x_{r2})$
DE/rand to best/bin	$v_i = x_i + F(x_{best} - x_i) + F(x_{r1} - x_{r2})$

∗where x_{best} is the current best individual.

2.2.1. Mutation

For each target individual x_i, different mutation strategies can be selected to generate a trial individual v_i. The basic form of the mutation operation is:

$$v_i = x_{r1} + F(x_{r2} - x_{r3}) \tag{4}$$

where x_{r1}, x_{r2}, x_{r3} are the non-equal individuals randomly selected, F is called scaling factor or mutation factor, and the value range is usually [0.41.5]. Three typical mutation strategies adopted as action options of the agent in this paper are shown in Table 1.

2.2.2. Crossover

In order to maintain the population diversity, the DE algorithm intersects the target individual and its corresponding trial individual to generate candidate individual of the target individual. Binomial crossover operation is usually adopted:

$$u_{ij} = \begin{cases} v_{ij} & \text{if } (rand(0,1) \le Cr) \text{ or } (j = j_{rand}) \\ x_{ij} & \text{otherwise} \end{cases} \tag{5}$$

where Cr is the crossover factor and the value is range [0.51].

2.2.3. Selection

The selection operation is used to compare the value of the target individuals and candidate individuals for population evaluation. Individuals with better fitness value will be selected to enter the next iteration in MOOP:

$$x_i(G+1) = \begin{cases} u_i & \text{if } u_i \prec x_i \\ x_i & \text{else} \end{cases} \tag{6}$$

The DE algorithm is widely used because of its low computational complexity and good global search ability. The selection or adaptive adjustment of a mutation strategy, scaling factor and crossover factor are the key factors affecting algorithm performance. In the existing multiobjective DE algorithms, niche, non-dominant sorting and crowded distance selection are adopted to maintain population diversity [3,14,25]. There are also applications to solve single objective optimization problems combined with reinforcement learning [26], but no applications have been seen in solving multimodal multiobjective problems. Since DE performs well on improving the diversity of the population, and reinforcement learning has good adaptability, the combination of these two methods may achieve excellent performance on multimodal multiobjective optimization.

2.3. Reinforcement learning

Reinforcement learning (RL) is a learning process in which the agent interacts with the environment in order to obtain the optimal action strategy to get the maximum long term rewards. The basic framework of RL is shown in Fig. 2. At each time step t, the agent perceives the environment's state S_t, and on that basis selects an action A_t. One time step later, the agent receives a reward R_{t+1} (positive or negative) and moves to the next state S_{t+1}. The agent acquires the optimal action strategy by this trial-and-error approach.

As a classical reinforcement learning algorithm, Q-learning has many advantages, such as 1) its structure is very simple, 2) it does not need prior knowledge, and 3) it does not need to wait until the completion of tasks. This paper takes it as the implementation framework of RL. Q-learning uses Q table to represent the value of different $\langle state, action \rangle$ pairs. As shown in Table 2, agents have m possible states and n available actions in each state. The agent determines the probability of choosing different behaviors in different states according to the Q table.

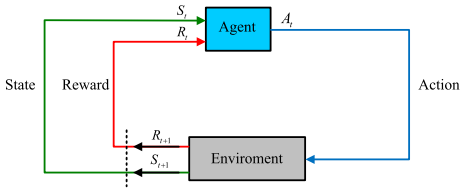


Fig. 2. The agent-environment interaction of reinforcement learning.

Table 2
The form of Q table.

State	Action			
	a_1	a_2	…	a_n
s_1	$Q(s_1, a_1)$	$Q(s_1, a_1)$	…	$Q(s_1, a_n)$
s_2	$Q(s_2, a_1)$	$Q(s_2, a_2)$	…	$Q(s_2, a_n)$
⋮	⋮	⋮	⋱	⋮
s_m	$Q(s_m, a_1)$	$Q(s_m, a_2)$	…	$Q(s_m, a_n)$

For each agent, the probability of selecting a_j action in state s_i is usually determined by Softmax strategy:

$$\pi(s_i, a_j) = \frac{e^{Q_t(s_i,a_j)/T}}{\sum_{j=1}^{n} e^{Q_t(s_i,a_j)/T}} \tag{7}$$

where $Q_t(s_i, a_j)$ is the corresponding value in the Q table at time t, and T is a control parameter.

Q-learning is an iterative learning process. The formula for updating Q value is as follows:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\left(R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)\right) \tag{8}$$

Where, R_{t+1} is the reward value obtained after A_t action of the agent in the S_t state; α is the learning rate,$\alpha \in [0,1]$; γ is called the discount rate, $\gamma \in [0,1]$.

The combination of reinforcement learning and evolutionary computation has received more and more attention in recent years [3,27–29]. Reinforcement learning can improve the performance of evolutionary algorithms because it is able to get an optimal long-term reward by trying different behavioral strategies, and is helpful in keeping a balance between exploration and exploitation adaptively.

3. Hierarchical state encoding and distribution of population in decision space, objective space and fitness-ranking space

In the MMOOP, because there are multiple PSs in the decision space, the conflict between global exploration and fine exploitation is more severe. For different optimization problems, the decision space and the objective space have different mapping patterns, and the iterative optimization process can easily lead to the random and discontinuous transition of individual in the decision space. If the evolutionary algorithm cannot learn adaptively or there is a diversity maintenance mechanism lacking, premature convergence may occur, resulting in the loss of part of the PSs. To solve this problem, a gradual transition strategy based on hierarchical state is proposed to guide the individual to move to the adjacent higher probability region or stay in the current high probability region where lie non-dominated solutions. In this way, the continu-

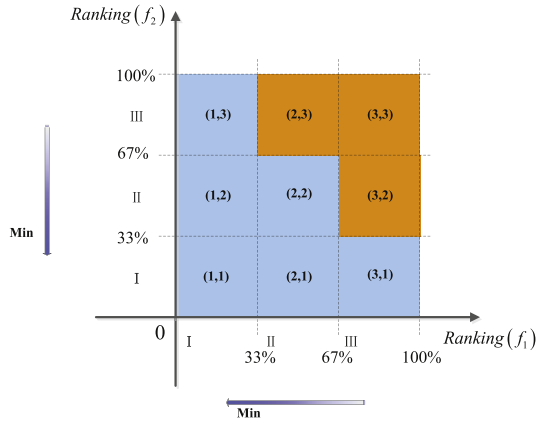


Fig. 3. The illustration of hierarchical states in the fitness-ranking space.

ity of individual transitions in the iteration could be better guaranteed, so as to overcome premature convergence and improve the distribution diversity of the optimal solution set.

In this paper, the hierarchical state of each individual in a population is encoded according to its fitness function values. For MOOP, the fitness value of each objective is sorted from small to large, the population is divided into three levels equally: excellent (rank 0 ~ 33.3%, denoted as I), medium (rank 33.3% ~ 66.7%, denoted as II) and poor (rank 66.7% ~ 100%, denoted as III). Taking the two objective functions as examples, the population may have nine hierarchical states, each of which is denoted as $(g_1, g_2)(1 \le g_1 \le 3, 1 \le g_2 \le 3)$, where g_1 and g_2 represent the corresponding grades of fitness values of the two objective functions respectively. The nine hierarchical states distribution zone and their corresponding number are shown in Fig. 3. Considering the conflict of each objective in MOOP, it is not possible to have non-dominated solutions in the hierarchical state zone $(2,3),(3,2),(3,3)$ shown in orange in Fig. 3. Other blue zones have a high probability of containing non-dominated solutions. When the population comprises all non-dominated solutions, the hierarchical states of the population will only be distributed in zones $(1,3),(2,2),(3,1)$ equally.

In order to observe how the different topological mapping relationships between the decision space and the objective space as to how the hierarchical state distribution of population is affected, many MMOOP test functions (see Ref. [10] for details) are used in the experiments. According to the above hierarchical state encoding method, two test functions MMF5 and MMF10 are taken as examples to show the hierarchical state distribution of the population in the decision space, objective space and fitness-ranking space. The results are shown in Fig. 4. Different colors in the color axis correspond to the hierarchical states. The black square is the true PF and PSs. Fig. 4 $(a),(b)$ and (c) depict the hierarchical states' distribution of the test functions MMF5 (left) and MMF10 (right) in three spaces respectively. It can be seen in the decision space, that there are many PSs of whose hierarchical states are distributed in $(1,1),(1,2),(1,3),(2,1)$ or $(3,1)$, and the hierarchical states' population distribution depends on the property of objective functions and the ratio of non-dominated solutions in the population. Similar results can be seen in objective space and fitness-ranking space. The distribution in fitness-ranking space could be taken as the normalization of ones in the objective space, and this transformation is regardless of the subtle difference of fitness value so that it helps to speed up convergence.

More interestingly, the hierarchical state of individual change gradually rather than drastically. Individuals located near the PSs in the decision space tend to have better fitness function values near the PF in the objective space. It can be seen that individuals with adjacent hierarchical states are neighbors in both the decision and objective spaces, namely the individuals with state $(3,3)$ are always near $(3,2),(2,3)$ and $(2,2)$, and the individuals with state $(1,1)$ are always near states $(1,2),(2,1)$ or $(2,2)$, and so on. This property could be used to prevent premature population development while keeping directional evolution. For example, an individual whose hierarchical state is $(2,3)$ should move to its adjacent higher hierarchical states, such as $(1,3),(1,2)$ and $(2,2)$, but can not jump directly into the highest hierarchical state $(1,1)$.

In order to evaluate the retaining chance for each individual in population, on the basis of analysis of the hierarchical state population distribution of in the fitness-ranking space, an indicator named retaining factor (RF) is proposed as follows:

$$RF = 1 - \frac{d_{toN}}{d_{toD}} \tag{9}$$

where d_{toN} is the distance between the individual and it's nearest non-dominated solution in the current population in the fitness-ranking space, and d_{toD} is the distance between the individual and the diagonal line in the fitness-ranking space. As shown in Fig. 5, the purple stars are dominated solutions and the red points are non-dominated solutions. In the fitness-ranking space, if there are no dominated solutions in the population, all points should lie in the diagonal line (blue one) and their RF values are all 1. For a dominated solution, where it lies above the diagonal line, it will mean neither of its objective value is not good enough to retain. Thus, the d_{toD} of these solutions will be set at infinity. For a dominated solution, where it lies under the diagonal line, the farther away from the diagonal line it is, the more chance for retaining there will be.

Therefore, this paper proposes a hierarchical evolution strategy to guide individuals to the adjacent hierarchical states or stay in the current hierarchical state zone which possibly has non-dominated solutions. In an iterative evolution process, if the selection operation is completed only according to the fitness value or the dominant relationship between the parent generation and the child generation, an abrupt transition of the individual is likely to occur, which may lead to the potential PS region being missed. The introduction of a hierarchical evolution strategy can lead to better adaptation to the continuous change trend of multiple objective function values at the same time, which is helpful in effectively maintaining the population diversity in the process of evolution.

4. Differential evolution algorithm based on reinforcement learning with fitness-ranking

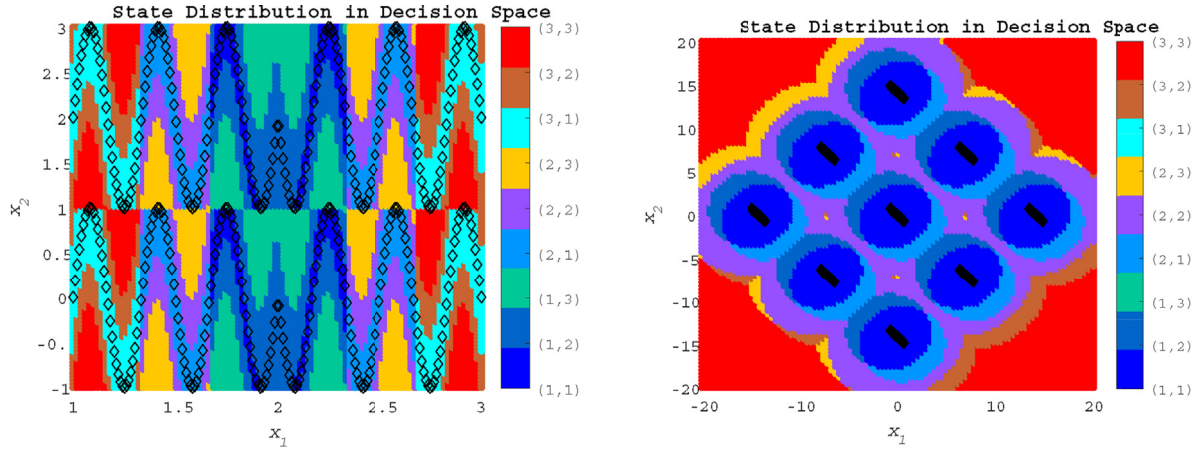
The distribution characteristics of the current population hierarchical states are related to the distribution of the population in the decision space and the mapping feature of objective functions. Based on reinforcement learning idea, each individual is regarded as an agent. Each agent adaptively learns the optimal evolutionary behavior strategy to be adopted in different states, according to the rewards brought by the action in the current state. In reinforcement learning, a good state, appropriate action and a proper reward mechanism are the keys to a successful learning process.

4.1. State encoding

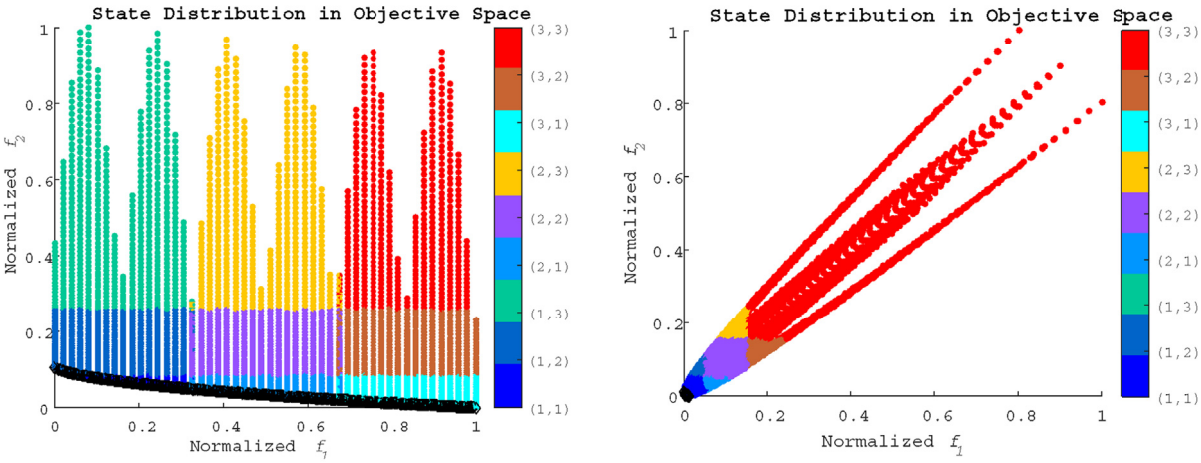
The state of each agent is encoded according to the above hierarchical state division method. Taking the two-objective optimization problem as an example, each optimization objective function is divided into three levels equally. Then, the total number of hierarchical states is 9, namely $s \in \{g_1, g_2\}$ $g_1 = 1,2,3$; $g_2 = 1,2,3$.

4.2. Action options and Q table

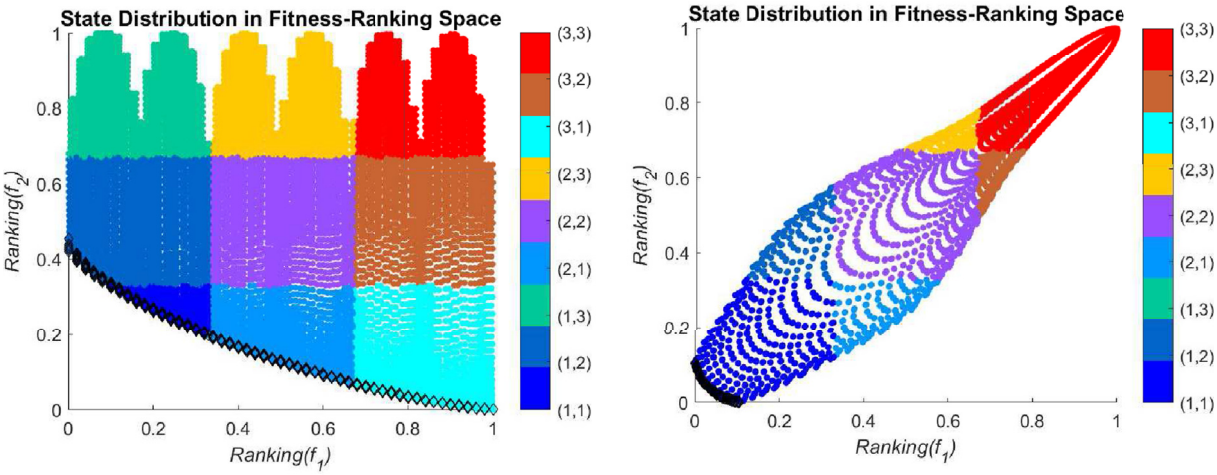
In this paper, DE is adopted as the basic evolutionary algorithm. Each agent has three mutation operations to choose from, namely



(a) Hierarchical state population distribution in the decision space.



(b) Hierarchical state population distribution in the objective space.



(c) Hierarchical state population distribution in the fitness-ranking space.

Fig. 4. Hierarchical state population distribution in the decision space, objective space and fitness-ranking space. (left: MMF5, right: MMF10).

DE/rand/1/bin, DE/best/1/bin, and DE/rand to best/bin. Table 3 is the Q value table of $\langle state, action \rangle$ pairs, which is a 9 × 3 matrix. In its current state, each agent samples a mutation operation according to the probabilistic distribution calculated by Eq. (4), and generates a candidate solution by combining a crossover operation.

4.3. Reward function table

In order to complete the idea of hierarchical evolution, the design of the reward function table should guide agents in different hierarchical states to adjacent higher hierarchical states or to stay in the current hierarchical state zone which possibly have a PS, in order to evolve

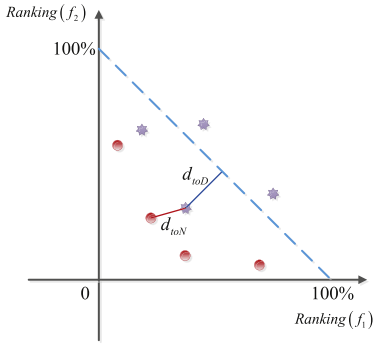


Fig. 5. The illustration of how to measure the distances for acquiring retaining factor.

Table 3
Q value table of $\langle state, action \rangle$ pairs.

State	Strategy		
	DE/rand/1/bin	DE/best/1/bin	DE/rand to best/bin
(1, 1)	Q_{11}	Q_{12}	Q_{13}
(1, 2)	Q_{21}	Q_{22}	Q_{23}
(1, 3)	Q_{31}	Q_{32}	Q_{33}
(2, 1)	Q_{41}	Q_{42}	Q_{43}
(2, 2)	Q_{51}	Q_{52}	Q_{53}
(2, 3)	Q_{61}	Q_{62}	Q_{63}
(3, 1)	Q_{71}	Q_{72}	Q_{73}
(3, 2)	Q_{81}	Q_{82}	Q_{83}
(3, 3)	Q_{91}	Q_{92}	Q_{93}

to non-dominant solutions. The expected ranking-grade transition path of individuals in different hierarchical state zones is shown in Fig. 6, where the black arrow indicates the evolutionary direction in which the agent will be rewarded. For example, agents in the state $(3, 3)$ are rewarded if their offsprings states are $(2, 3), (3, 2)$ or $(2, 2)$. For agents in state $(1, 1)$, if the states of a newly generated solution are retained as $(1, 1)$, they will be rewarded.

The reward function table designed is shown in Table 4. When the agent transfers from the current state to the next state, the reward value obtained is shown in the element of the corresponding position (current state, next state) in Table 4.

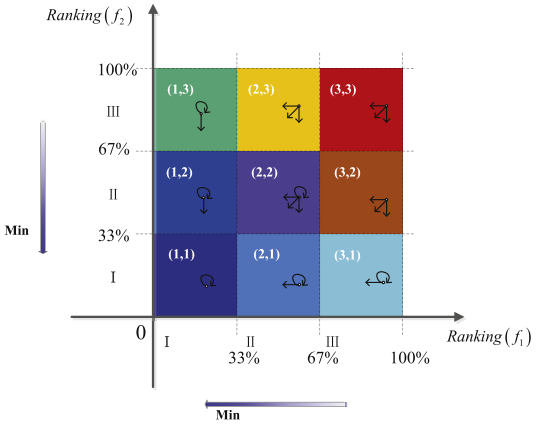


Fig. 6. Designed grade transition path of individuals in different hierarchical state zone.

Table 4
Reward function table.

State	Next state								
	(1, 1)	(1, 2)	(1, 3)	(2, 1)	(2, 2)	(2, 3)	(3, 1)	(3, 2)	(3, 3)
(1, 1)	10	0	0	0	0	0	0	0	0
(1, 2)	10	10	0	0	0	0	0	0	0
(1, 3)	0	10	10	0	0	0	0	0	0
(2, 1)	10	0	0	10	0	0	0	0	0
(2, 2)	10	10	0	10	10	0	0	0	0
(2, 3)	0	10	10	0	10	10	0	0	0
(3, 1)	0	0	0	10	0	0	10	0	0
(3, 2)	0	0	0	10	10	0	10	0	0
(3, 3)	0	0	0	0	10	10	0	10	0

4.4. Proposed algorithm and pseudocode

A novel differential evolution algorithm based on reinforcement learning with fitness-ranking (DE-RLFR) is proposed, which combines a DE algorithm with an RL mechanism. The DE-RLFR is based on a Q-learning framework, and each individual in the population is considered as an agent. Each agent has its hierarchical state variable assigned by its ranking order of multi fitness function values, and has three typical DE mutation operations to use. In generation, those individuals who successfully get to or stay at a higher hierarchical state zone are rewarded, and the experience of the agent is retained in the Q table and updated with iterations, which is used to guide all agents to choose a suitable mutation strategy to generate offspring individuals in the next iteration.

The general procedure for implementing the DE-RLFR algorithm is as follows in Algorithm 1. First, the population and Q table are randomly initialized, and all control parameters are set. Then, for each iteration, the hierarchical state value is assigned to each individual according to their ranking order of multi fitness function values. For each target individual(agent) $x_{k,g}$, its mutation operator is drawn using a Q table by Eq. (7), and generates a candidate solution $u_{k,g}$ after mutation and crossover operation. The hierarchical state value of $u_{k,g}$ in the current population can be acquired as the next state, and then the Q table is updated by Eq. (8). All individuals of the current population X_g and their corresponding candidate solutions U_g are combined as X_{buffer} sorted by their retaining factor value in descending order. Then the first Np (population size) individuals in X_{buffer} become the next iteration population. Meanwhile, the corresponding fitness value and non-dominated solutions enter the next iteration. When the maximum iteration number is reached, the loop is ended and the founded PS (non-dominated solutions) and corresponding PF (non-dominated front) are outputted.

5. Performance evaluation of DE-RLFR

In order to test the performance of DE-RLFR, 11 test problems designed by Deb [8], Rudolph [9], Liang [12] and Yue [13] are used to evaluate the proposed algorithm. The number of PSs of these 11 test problems varies from 2 to 27. PFs have convex and concave shapes, PSs have linear and nonlinear shapes, and the dimensions of the decision space and objective space are 2 and 3 dimensions. The details of these test problems are shown in Table 5.

Four indicators HV, IGDF, IGDX and PSP are used to compare the performances of different algorithms. Among them, HV and IGDF reflect the performance in the objective space, while IGDX and PSP reflect the performance in the decision space. For HV and PSP, the larger values represent better performance. As for IGDF and IGDX, the smaller values mean better performance. The definitions and details of these indicators are seen in Ref. [13].

For the purpose of unbiased comparison, the maximal number of evaluations is set to 80,000 for all algorithms. Each algorithm used for

Algorithm 1 Differential evolution based on Reinforcement Learning with Fitness Ranking (DE-RLFR).

Initialization: R- reward matrix, Np- population size,G_{max}- maximum iteration number, F- mutation factor, Cr - crossover factor, α- learning rate, γ-discount factor, $Q(S_i, A_j)$-Q table, X-Population, f-Fitness Vector of Population,h_1, h_2-hierarchical state encoding threshold;
for *all* $g \in \{1, \ldots, G_{max}\}$ **do**
  Encoding the hierarchical state S_x of all individuals using f_{xg} of X_g, h_1 and h_2 ;
  $\{PF, PS\}$ = Get Non-dominated solutions;
  for *all* $k \in \{1, \ldots, Np\}$ **do**
    Draw a mutation operator by $Q(S_k i, A_k j)$ with equation(7);
    mutation operation: $v_{k,g} = Mutation(x_{k,g})$;
    crossover operation: $u_{k,g} = Crossover(x_{k,g}, v_{k,g})$;
    compute fitness $f(u_{k,g})$;
    Encoding the hierarchical state S_u of $u_{k,g}$ using $f(u_{k,g})$, h_1 and h_2 ;
    update Q table using equation (8);
  end
  set $X_{buffer} = \{X_g, U_g\}, f_{buffer} = \{f_x g, f_u g\}$;
  sort X_{buffer} and f_{buffer} descending by Retaining Factor ;
  $X_{g+1} = X_{buffer}(1 : NP), f_{g+1} = f_{buffer}(1 : NP)$;
end
Output: PF,PS

Table 5
Characteristics of the 11 test problems.

Function name	Number of PSs	Number of decision variables	Number of objectives	Pareto set geometry	Pareto front geometry
MMF1	2	2	2	Convex	Nonlinear
MMF2	2	2	2	Convex	Nonlinear
MMF3	2	2	2	Convex	Nonlinear
MMF4	4	2	2	Non-Convex	Nonlinear
MMF5	4	2	2	Convex	Nonlinear
MMF6	4	2	2	Convex	Nonlinear
MMF7	2	2	2	Convex	Nonlinear
MMF8	4	2	2	Non-Convex	Nonlinear
SYM-PART simple	9	2	2	Convex	Linear
SYM-PART rotated	9	2	2	Convex	Linea
Omni-test (n = 3)	27	3	2	Convex	Linear

comparison was tested on the eleven test problems 20 times. The mean and standard deviation of 20 times runnings of DE-RLFR are given in Table 6. These results indicate that the proposed DE-RLFR algorithm is robust, because the variances of each indicator on all 11 test functions are small. Furthermore, the mean values of HV and PSP are relatively large, while IGDF and IGDX are relatively small that means the DE-RLFR algorithm performs well in both the target space and the decision space for optimization demand.

Figs. 7–10 are used to compare the performances of DE-RLFR with eight other good or popular MOOP/MMOOP algorithms using the above

Table 6
The mean and standard deviation of different indicators for DE-RLFR.

Function Name	Performance Indicator			
	HV	IGDF	IGDX	PSP
MMF1	3.664 ± 0.0009	0.0009 ± 0.00005	0.0167 ± 0.0004	60.03 ± 3.286
MMF2	3.639 ± 0.0231	0.0080 ± 0.0037	0.0167 ± 0.0102	73.32 ± 27.966
MMF3	3.644 ± 0.0149	0.0060 ± 0.0022	0.0128 ± 0.0061	88.68 ± 30.373
MMF4	3.329 ± 0.0028	0.0009 ± 0.00001	0.0090 ± 0.0007	111.57 ± 8.357
MMF5	3.664 ± 0.0007	0.0008 ± 0.00004	0.0311 ± 0.0017	32.22 ± 1.782
MMF6	3.664 ± 0.0007	0.0008 ± 0.00003	0.0285 ± 0.0010	35.03 ± 1.299
MMF7	3.664 ± 0.0007	0.0012 ± 0.00009	0.0131 ± 0.0025	78.38 ± 14.896
MMF8	3.211 ± 0.0010	0.0009 ± 0.00004	0.0226 ± 0.0035	45.00 ± 6.806
SYM-PART simple	1.321 ± 0.008	0.006 ± 0.001	0.016 ± 0.0013	59.99 ± 4.568
SYM-PART rotated	1.317 ± 0.008	0.007 ± 0.001	0.034 ± 0.0027	29.47 ± 2.295
Omni-test	62.023 ± 0.011	0.0084 ± 0.0008	0.062 ± 0.0027	16.11 ± 0.713

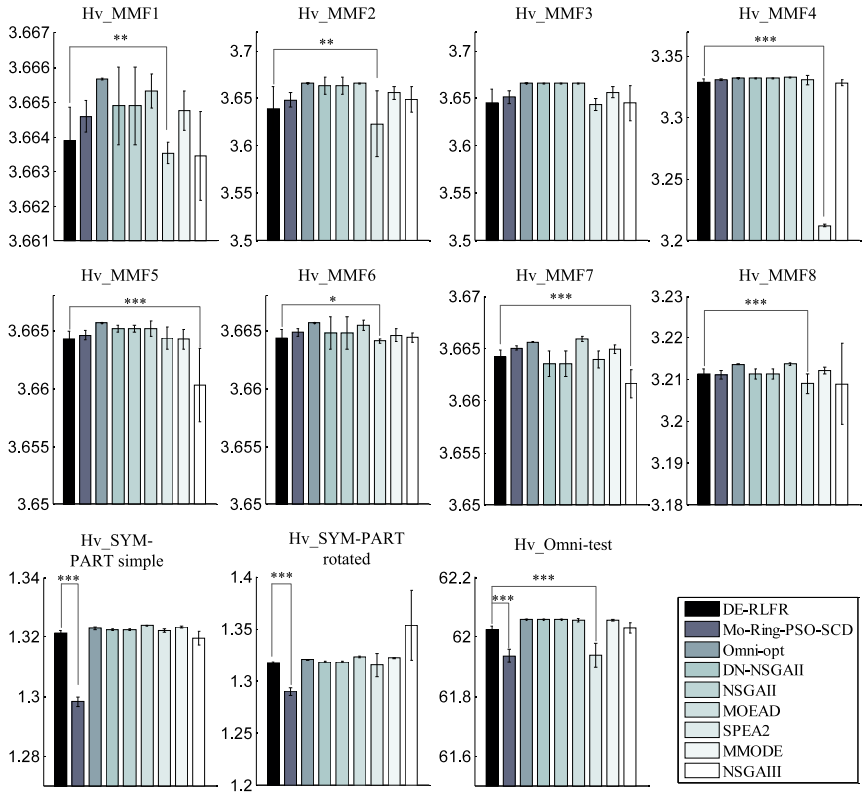


Fig. 7. The bar-plots of HV values of different algorithms on 11 test functions with significance test.

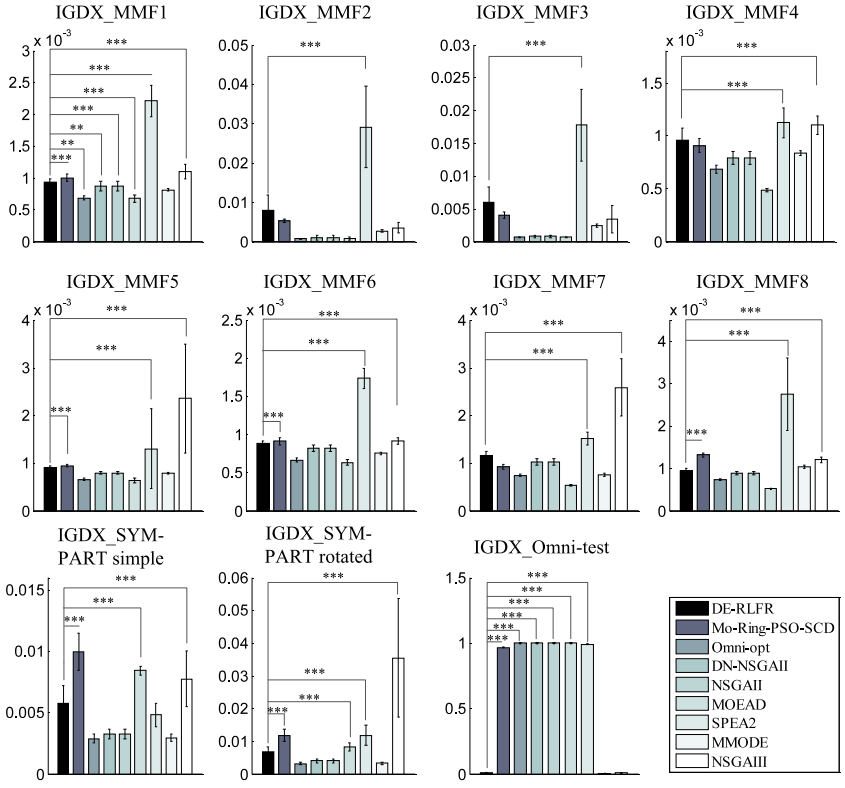


Fig. 8. The bar-plots of IGDX values of different algorithms on 11 test functions with significance test.

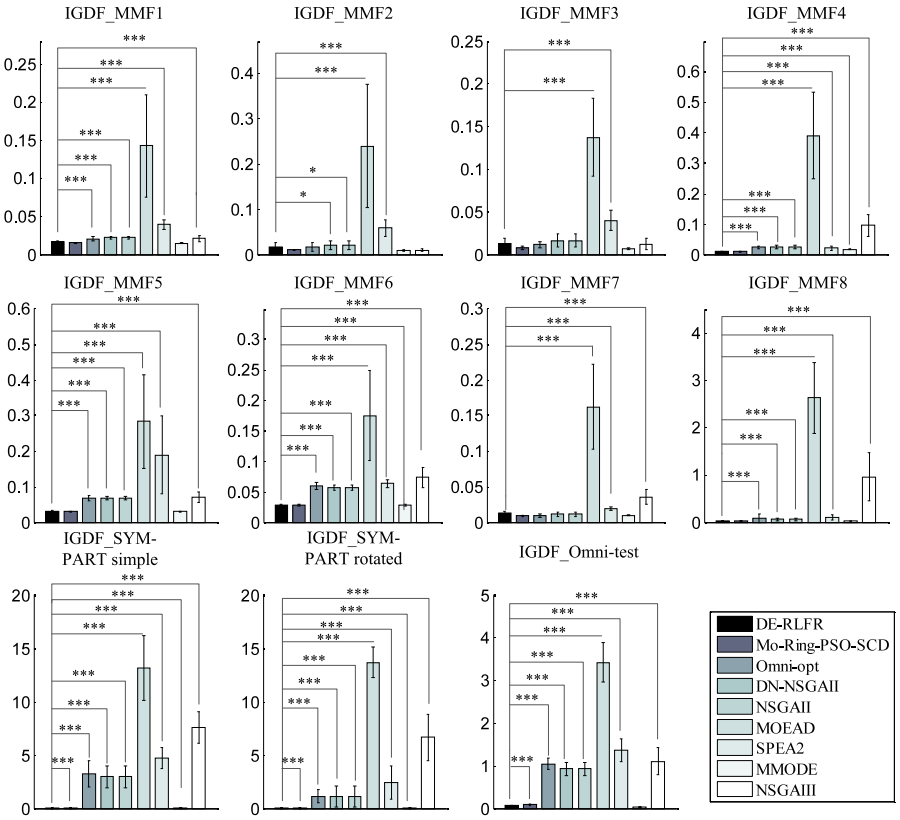


Fig. 9. The bar-plots of IGDF values of different algorithms on 11 test functions with significance test.

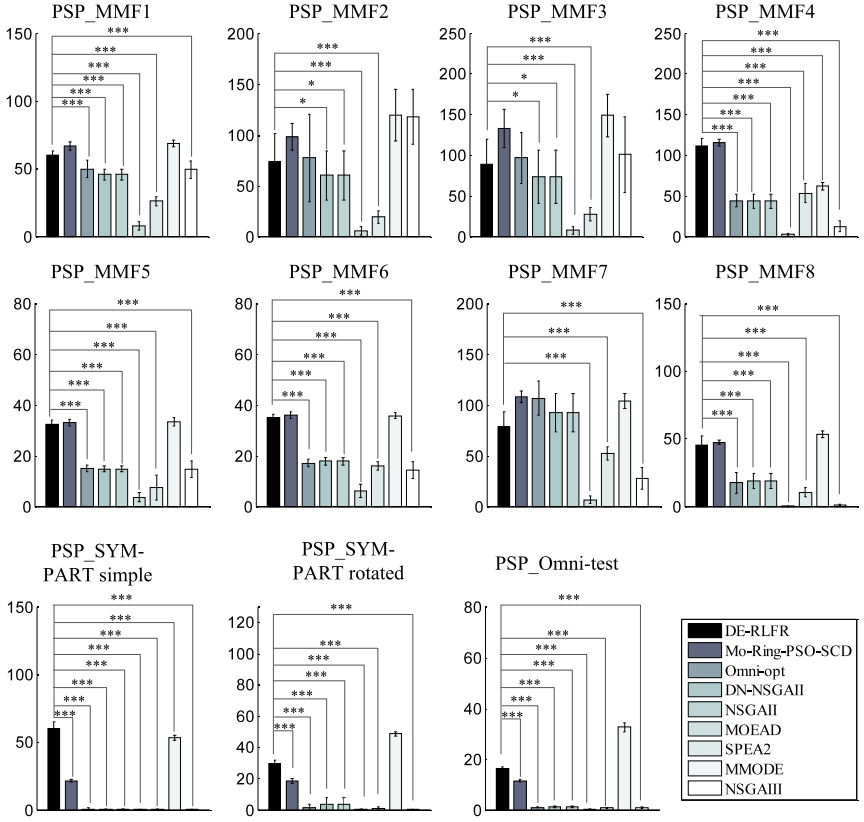


Fig. 10. The bar-plots of PSP values of different algorithms on 11 test functions with significance test.

four key indicators respectively. The rank sum test is used for a statistical test, and $'^{***}'$ indicates $p < 0.001$, $'^{**}'$ indicates $p < 0.01$, and $'^{*}'$ indicates $p < 0.05$. In general, there was a significant improvement of the DE-RLFR compared with other MOOP/MMOOP algorithms. In Fig. 7, the HV indicator values of DE-RLFR, MO_Ring_PSO_SCD [13], Omni-optimizer [8], DN-NSGAII [12], NSGAII [19], MOEA/D [16], SPEA2 [23], MMODE [14] and NSGAIII [20] algorithms are compared. It can be seen that DE-RLFR outperforms some algorithms, especially on test function MMF5, MMF6 and MMF7. It is worth noting that the variances of DE-RLFR are significantly smaller than that of other algorithms, showing its good robustness.

Figs. 8 and 9 compare the distribution characteristics of PSs in the decision space and PF in the objective space respectively, obtained different algorithms to solve different MMOOPs. As can be verified from the results, the IGDX values and IGDF values obtained by DE-RLFR are smaller than most other algorithms on most test functions. It indicates that the PSs and PF found by DE-RLFR have good global or uniform distribution properties.

Indicator PSP reflects the similarity between the obtained PSs and the true PSs. In Fig. 10, the mean PSP values of DE-RLFR are significantly higher than all other contrasting algorithms on test function SYM-PART simple. On the test functions SYM-PART rotated and Omni-test, the PSP value of DE-RLFR are only smaller than that of MMODE. On the test function MMF4, PSP value of DE-RLFR are significantly higher than all other contrast algorithms except MO_Ring_PSO_SCD. On the test function MMF1,MMF5, MMF6 and MMF8, PSP value of DE-RLFR are significantly higher than all other contrast algorithms except MO_Ring_PSO_SCD and MMODE. All these good results show that the combination of the reinforcement learning idea and an evolutionary algorithm can further improve the algorithm's adaptability and global search performance.

6. Conclusion and discussion

DE-RLFR, a combination of reinforcement learning and differential evolution, is proposed in this paper. The hierarchical state of each individual in a population is encoded according to its joint ranking order value of multi fitness functions. The hierarchical state is used to determine evolution direction and mutation strategy in DE. Since reinforcement learning with fitness-ranking helps guide the population to converge gradually and DE helps improve the diversity, DE-RLFR shows good performance in solving MMOOPs. It can not only find a Pareto optimal solution set stably and effectively, but also ensures good distribution characteristics. DE-RLFR was tested on eleven multimodal multi-objective test problems and compared with eight commonly used algorithms. It can be concluded from the results that the evolutionary algorithm combined with a reinforcement learning framework, can significantly improve the efficiency of the population search in the solution space. According to different hierarchical states in the solution space take different evolution behavior which long term returns could be used to accumulate experience, and they adapt to choose effective behavior to evolution in a probability sense. This approach will increase the adaptivity of evolutionary algorithms in choices of evolutionary strategies or control parameters.

The values of the Q Table in the early stage (first iteration, after 800 times updating) and later stage (100 iterations, after 80,000 times updating) of the population evolutionary process are listed in Tables 7 and 8 respectively.

It can be seen that there are some $\langle state, action \rangle$ pair values which are not changed, such as the solutions with state $(1,1)$ and that means there are no solutions in the population with state $(1,1)$, which indicates that the distribution of non-dominated solutions are more close to the diagonal line and also converge. For solutions with hierarchical state $(2,3)$ or $(3,2)$ which are all dominated, it can be seen that they tend to choose the second mutation strategy after 800 updating, and this tendency becomes more obvious after 80,000 times updating. For the solutions with hierarchical state $(1,3)$,$(3,1)$ and $(2,2)$ which are highly possible non-dominated, it can be seen that there is no obvious preference for any strategy. The influence of different R table settings is also compared in our test. The R table designed (Table 4) according to the hierarchical state distribution property of the population in the ranking-fitness space, has good performance in convergence and better adaptability than with other R settings.

In future, the influence of hierarchical encoding criteria, variation strategy selection and DE control parameters on the algorithm effect will be further studied. Combining reinforcement learning with evolutionary algorithms is a promising direction. In the next step, this hierarchical evolutionary reinforcement learning mechanism will be introduced into other evolutionary algorithms to evaluate the effectiveness of solving different types of optimization problems.

Table 7
Q value table after the first iteration (updated for 800 times).

State	Strategy		
	DE/rand/1/bin	DE/best/1/bin	DE/rand to best/bin
(1, 1)	4.1536	4.0656	7.5994
(1, 2)	11.2840	11.8259	12.5818
(1, 3)	14.4423	11.5746	11.4255
(2, 1)	11.1045	10.7417	10.1574
(2, 2)	12.6658	11.7412	11.2222
(2, 3)	9.0114	14.0620	9.9320
(3, 1)	11.2811	13.8213	12.6173
(3, 2)	8.3144	11.1555	8.9073
(3, 3)	9.1490	9.9949	10.7592

Table 8
Q value table after 100 iterations (updated for 80,000 times).

State	Strategy		
	DE/rand/1/bin	DE/best/1/bin	DE/rand to best/bin
(1, 1)	4.1536	4.0656	7.5994
(1, 2)	14.4037	14.8349	15.9722
(1, 3)	15.5399	17.9288	18.3047
(2, 1)	14.0347	15.1333	13.9585
(2, 2)	15.5841	15.5317	17.0400
(2, 3)	10.6190	17.5729	9.9320
(3, 1)	14.3472	15.4824	15.0042
(3, 2)	11.3243	16.6859	13.8672
(3, 3)	9.1490	9.9949	12.0648

Acknowledgment

This work is supported by the National Natural Science Foundation of China (61876169,U1304602,61673404,61305080).

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.swevo.2019.06.010.

References

[1] S. Bandaru, A.H.C. Ng, K. Deb, Data mining methods for knowledge discovery in multi-objective optimization: Part A - Survey, Expert Syst. Appl. 70 (2017) 139–159, https://doi.org/10.1016/j.eswa.2016.10.015.
[2] Y. Rizk, M. Awad, E.W. Tunstel, Decision making in multi-agent systems: a survey, IEEE Trans. Cogn. Dev. Syst. 10 (3) (2018) 514–529, https://doi.org/10.1109/TCDS.2018.2840971.
[3] J.D. Ser, E. Osaba, D. Molina, X.-S. Yang, S. Salcedo-Sanz, D. Camacho, Das Swagatam, P.N. Suganthan, C.A.C. Coello, F. Herrera, Bio-inspired computation: where we stand and whats next, Swarm Evol. Comput. 48 (2019) 220–250, https://doi.org/10.1016/j.swevo.2019.04.008.

[4] D.J. Lizotte, E.B. Laber, Multi-objective Markov decision processes for data-driven decision support, J. Mach. Learn. Res. 17 (2016) 1–28.
[5] Y. Liu, G. G.Yen, D. Gong, A multi-modal multi-objective EvolutionaryAlgorithm using two-archive and recombination strategies, IEEE Trans. Evol. Comput. https://doi.org/10.1109/TEVC.2018.2879406.
[6] G. Yi-Nan, C. Jian, L. Sha, G. Dun-Wei, X. Yu, Robust dynamic multi-objective vehicle routing optimization method, IEEE ACM Trans. Comput. Biol. Bioinform 15 (6) (2018) 1891–1903, https://doi.org/10.1109/TCBB.2017.2685320.
[7] K. Deb, Multi-objective genetic algorithms: problem difficulties and construction of test problems, Evol. Comput. 7 (3) (1999) 205–230, https://doi.org/10.1162/evco.1999.7.3.205.
[8] D. Kalyanmoy, T. Santosh, Omni-optimizer: A Procedure for Single and Multi-Objective Optimization, 2005, pp. 47–61, https://doi.org/10.1007/978-3-540-31880-4_4.
[9] G. Rudolph, B. Naujoks, M. Preuss, Capabilities of EMOA to detect and preserve equivalent Pareto subsets, in: International Conference on Evolutionary Multi-Criterion Optimization, 2007, pp. 36–50, https://doi.org/10.1007/978-3-540-70928-2_7.
[10] H. Ishibuchi, N. Akedo, Y. Nojima, A many-objective test problem for visually examining diversity maintenance behavior in a decision space, in: Conference on Genetic and Evolutionary Computation, 2011, pp. 649–656, https://doi.org/10.1145/2001576.2001666.
[11] P. Kerschke, W. Hao, M. Preuss, C. Grimme, A. Deutz, H. Trautmann, M. Emmerich, Towards analyzing multimodality of continuous multiobjective landscapes, in: International Conference on Parallel Problem Solving from Nature, 2016, pp. 962–972, https://doi.org/10.1007/978-3-319-45823-6_90.
[12] J. Liang, C.T. Yue, B.Y. Qu, Multimodal multi-objective optimization: a preliminary study, in: Evolutionary Computation, 2016, pp. 2454–2461, https://doi.org/10.1109/CEC.2016.7744093.
[13] C. Yue, B. Qu, J. Liang, A multiobjective particle swarm optimizer using ring topology for solving multimodal multiobjective problems, IEEE Trans. Evol. Comput. (2018) 805–817, https://doi.org/10.1109/TEVC.2017.2754271.
[14] J. Liang, W. Xu, C. Yue, K. Yu, H. Song, O.D. Crisalle, B. Qu, Multimodal Multiobjective Optimization with Differential Evolution, Swarm and Evolutionary Computation, 2019, pp. 1028–1059, https://doi.org/10.1016/j.swevo.2018.10.016.
[15] C. Yue, B. Qu, K. Yu, J. Liang, X. Li, A novel scalable test problem suite for multimodal multiobjective optimization, Swarm Evol. Comput. (2019), https://doi.org/10.1016/j.swevo.2019.03.011.
[16] R. Tanabe, H. Ishibuchi, A decomposition-based evolutionary algorithm for multi-modal multi-objective optimization, in: International Conference on Parallel Problem Solving from Nature, 2018, pp. 249–261.
[17] J. Liang, Q. Guo, C. Yue, B. Qu, K. Yu, A self-organizing multi-objective particle swarm optimization algorithm for multimodal multi- objective problems, in: International Conference on Swarm Intelligence, 2018, pp. 550–560.
[18] Y. Liu, H. Ishibuchi, Y. Nojima, N. Masuyama, K. Shang, A double-niched evolutionary algorithm and its behaviors on polygon-based problems, in: Proceedings of Conference on Parallel Problem Solving from Nature, 2018, pp. 262–273, https://doi.org/10.1007/978-3-319-99253-_21.
[19] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197, https://doi.org/10.1109/4235.996017.
[20] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: solving problems with box constraints, IEEE Trans. Evol. Comput. 18 (4) (2014) 577–601, https://doi.org/10.1109/TEVC.2013.2281535.
[21] Q. Zhang, L. Hui, MOEA/D: a multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evol. Comput. 11 (6) (2007) 712–731, https://doi.org/10.1109/TEVC.2007.892759.
[22] E. Zitzler, S. Knzli, Indicator-based selection in multiobjective search, Lect. Notes Comput. Sci. 3242 (2004) 832–842, https://doi.org/10.1007/978-3-540-30217-9_84.
[23] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: improving the strength Pareto evolutionary algorithm, Evol. Methods Des., Optimiz. Contr. 2002 (2002) 95–100, https://doi.org/10.1007/978-3-540-30217-9_84.
[24] D. Brockhoff, E. Zitzler, Improving hypervolume-based multiobjective evolutionary algorithms by using objective reduction methods, in: IEEE Congress on Evolutionary Computation, 2007, pp. 2086–2093, https://doi.org/10.1109/CEC.2007.4424730.
[25] S. Das, S.S. Mullick, P. Suganthan, Recent advances in differential evolution an updated survey, Swarm Evol. Comput. 27 (2016) 1–30, https://doi.org/10.1016/j.swevo.2016.01.004.
[26] P. Rakshit, A. Konar, P. Bhowmik, I. Goswami, S. Das, L.C. Jain, A.K. Nagar, Realization of an adaptive memetic algorithm using differential evolution and Q-learning: a case study in multirobot path planning, IEEE Trans. Syst. Man Cybern. Syst. 43 (4) (2013) 814–831, https://doi.org/10.1109/TSMCA.2012.2226024.
[27] H. Samma, C.P. Lim, J.M. Saleh, A new reinforcement learning-based memetic particle swarm optimizer, Appl. Soft Comput. 43 (C) (2016) 276–297, https://doi.org/10.1016/j.asoc.2016.01.006.
[28] M.M. Drugan, Reinforcement learning versus evolutionary computation: a survey on hybrid algorithms, Swarm Evol. Comput. (2019), https://doi.org/10.1016/j.swevo.2018.03.011.
[29] Y. Zeng Hsieh, M.C. Su, A Q-learning-based swarm optimization algorithm for economic dispatch problem, Neural Comput. Appl. 27 (8) (2015) 2333–2350, https://doi.org/10.1007/s00521-015-2070-1.