Information Sciences 678 (2024) 120863

Contents lists available at ScienceDirect

Information Sciences

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

A reinforcement learning assisted evolutionary algorithm for constrained multi-task optimization

Yufei Yang[a], Changsheng Zhang[a,c,*], Bin Zhang[a], Jiaxu Ning[b]

[a] Software College, Northeastern University, Shenyang, 110169, China
[b] School of Information Science and Engineering, Shenyang Ligong University, Shenyang, 110159, China
[c] College of Computer Science and Engineering, Ningxia Institute of Science and Technology, Shizuishan, 753000, China

ARTICLE INFO

Keywords:
Constrained multi-task optimization
Reinforcement learning
Operator selection
Knowledge transfer

ABSTRACT

Multi-task optimization problems in the real world often contain constraints. When dealing with these problems, it is necessary to consider multiple tasks and their respective constraints simultaneously. However, most of existing research on multi-task optimization neglects the influence of constraints, which leads to slow convergence speed and susceptibility to local optima. To address the aforementioned issues, this paper proposes a reinforcement learning assisted constrained multi-task evolutionary algorithm. First, to meet the different requirements of different tasks and constraints, an adaptive operator selection strategy based on reinforcement learning is proposed. Second, to enhance population diversity, a multi-population method with different constraint handling techniques is introduced. This method assigns two independent populations to each task. The main population aims to find feasible solutions, while the auxiliary population focuses on exploring the entire search space. Finally, considering the individual differences between tasks, a dimension-based knowledge transfer is employed to facilitate positive information exchange. Compared with other state-of-the-art constrained evolutionary algorithms, the experimental results on constrained multi-task benchmark suite demonstrate the superiority of the proposed algorithm. The source code can be obtained from https://github.com/yufeiyng/RL-CMTEA.

1. Introduction

Multi-task optimization (MTO) [1] is an emerging research topic in evolutionary computing, aiming to solve multiple optimization tasks simultaneously. These tasks are often interrelated, and the common knowledge obtained from solving one optimization task can be helpful in solving other related tasks. In real-world applications, interrelated optimization tasks [2,3] are ubiquitous. MTO uses this common knowledge to solve multiple tasks simultaneously in a unified search space, thereby achieving better performance in solving multi-task optimization problems (MTOPs).

Notably, real-world optimization problems often incorporate constraints. For example, in urban transportation planning problems, factors such as road capacity, restrictions on the number of public transportation vehicles, and signal light cycles need to be taken into account to improve urban transportation efficiency and optimize the service scope of public transportation. Similarly, when solving other practical applications, constraints also need to be considered, such as supply chain design [4], and the multi-task

* Corresponding author.
E-mail address: zhangchangsheng@mail.neu.edu.cn (C. Zhang).

https://doi.org/10.1016/j.ins.2024.120863
Received 6 December 2023; Received in revised form 19 March 2024; Accepted 30 May 2024

Available online 3 June 2024
0020-0255/© 2024 Published by Elsevier Inc.

shape optimization problem [5]. Therefore, effectively solving constrained multi-task optimization problems (CMTOPs) is crucial for meeting real-world needs. Without loss of generality, a minimized CMTOP can be defined as Eq. (1).

$$\begin{aligned} & x_t^* = \arg\min f_t(x), t = 1, 2, \ldots, T \\ & subject\ to\ g_{t,i}(x) \le 0, i = 1, \ldots, p \\ & \qquad\qquad h_{t,j}(x) = 0, j = 1, \ldots, q \end{aligned} \tag{1}$$

where $f_t(x)$ denotes the objective function of task $\mathcal{T}_t$, and $g_{t,i}$ and $h_{t,j}$ represent p inequality and q equality constraints, respectively. Due to the difficulty in satisfying equality constraints, these constraints are usually transformed into inequality constraints, as presented in Eq. (2).

$$|h\_(t,j)(x) - \varepsilon| \le 0 \tag{2}$$

where ε denotes a very small value to convert an equality constraint to an inequality constraint. The goal of constrained multi-task optimization (CMTO) is to find a set of feasible solutions $\{x_1^*, x_2^*, \ldots, x_T^*\}$ for a CMTOP that satisfies the constraints.

Although CMTO is of great significance in solving practical problems, research on CMTO in recent years is scarce. In comparison with MTO, CMTO introduces different constraints in each task, leading to the following differences: MTO focuses on the optimization of objective functions, while CMTO considers both objective functions and their corresponding constraints. Furthermore, the presence of infeasible regions results in great differences between individuals in the different tasks of CMTO. How to effectively traverse these infeasible regions and find as many feasible solutions with good convergence as possible increases the difficulty of designing CMTO algorithms.

To solve CMTOPs, Li et al. [6] designed a benchmark set and integrated the feasibility priority (FP) rule into multi-task evolutionary algorithms (MTEAs). The FP rule can be easily embedded into evolutionary algorithms without modifying the algorithm logic. This rule can rapidly locate feasible regions by prioritizing feasible solutions and discarding infeasible ones. Xing et al. [7] proposed an adaptive archive-based multifactorial evolutionary algorithm, namely, A-CMFEA, which uses infeasible solutions to accelerate the convergence rate and adaptively adjust the random mating probability (rmp) to promote positive knowledge transfer. In addition, they proposed a new mutation strategy that replaces the individual with the largest constraint violation with the individual generated by random individual mutations.

However, the current works have not considered the difficulty of FP-driven populations in exploring infeasible regions, resulting in the following problems that have not been effectively solved:

Difficulty of operator selection. Different tasks have different requirements for operators, which are influenced by changes in their objective functions and constraints. For example, when the current feasible region is far from the feasible optima, the operator should exhibit global exploration capability. In scenarios with irregular shapes in feasible regions, the operator needs to possess strong local exploitation capability.

Insufficient exploration of infeasible regions. The current utilization of infeasible solutions only archives infeasible solutions discarded by the FP-driven population, lacking effective exploration of the search space around these infeasible solutions. This insufficiency may result in the inability of the population to escape local optima, especially when the local optima are far from the global optima.

Ineffectiveness of individual-based knowledge transfer. Because the objective functions and constraints of different tasks are quite different, the solutions that perform well in the source task may perform poorly in the target task, or even fail to meet the constraints.

To address the aforementioned problems, it is imperative to select suitable operators for tasks with different constraints, which can balance the exploitation of feasible regions and the exploration of infeasible regions. Reinforcement learning (RL) [8,9] provides a method for operator selection by training an agent to learn the association between the characteristics of different tasks, constraints and the performance of operators, thereby selecting suitable operators for populations with different optimization goals. In addition, knowledge transfer achieved through crossover operators relies on the use of appropriate operators to enhance adaptability from the source task to the target task.

Based on these considerations, this paper proposes an RL assisted constrained multi-task evolutionary algorithm (RL-CMTEA) for solving CMTOPs. The main contributions of this paper can be summarized as follows:

(1) An adaptive operator selection strategy based on RL is proposed. The selection of operators depends on the individual improvements brought by the offspring population generated by each operator. This strategy aims to enhance the adaptability of the algorithm in different tasks and constraints by adopting four different types of operators as candidates.
(2) A multi-population method with different constraint handling techniques is proposed. Two independent populations are used to handle each task. The main population considers all constraints and focuses on finding feasible solutions, whereas the auxiliary population ignores constraints, allowing for free exploration of the entire search space and providing diversity and feasibility for the main population.
(3) A knowledge transfer strategy based on similar dimensions is introduced. Information exchange is achieved through offspring of individual blocks with similar dimensions, which are generated through RL selection operators. This strategy helps improve the accuracy and adaptability of knowledge transfer, and better cope with the tasks and constraint challenges in CMTO.
(4) To demonstrate the effectiveness of RL-CMTEA, this paper employs the benchmark test suite introduced in [6] and conducts comparisons with other state-of-the-art constrained evolutionary algorithms. The experimental results confirm the superior performance of the proposed algorithm in solving CMTOPs.

The rest of this paper is organized as follows. Section 2 introduces the related work. Section 3 describes the proposed RL-CMTEA algorithm. Section 4 presents the experimental results and discussions. Finally, Section 5 provides the conclusions and future work.

2. Related work

2.1. Existing algorithmic strategies for MTO

In recent years, MTO has received considerable attention. Due to the lack of prior knowledge, it is difficult to determine whether there is any correlation between tasks. When the inter-task similarity is low, the performance of MTO may be affected, even leading to negative transfer [10]. Therefore, knowledge transfer has always been the focus of MTO research, and its classification mainly includes when-to-transfer and how-to-transfer.

When-to-transfer: Bali et al. [11] developed an *rmp* matrix to determine the pairwise *rmp* value of all tasks, thereby minimizing negative interactions. Xu et al. [12] introduced cultural transmission into MTO and adaptively adjusted the *rmp* value based on the success rate of knowledge transfer within and between tasks. In addition, Tang et al. [13] used a population to represent the best performing individuals and grouped tasks through genotype and phenotype analysis. Tasks within the same population use the same *rmp* to achieve knowledge transfer. Furthermore, Lin et al. [14] employed information entropy to quantify the uncertainty of evolutionary search and used this information to adjust parameters.

How-to-transfer: Li et al. [15] regarded the knowledge of how to generate high-quality solutions as meta-knowledge and used this knowledge to achieve knowledge transfer. Wang et al. [16] introduced neighborhoods as a conduit for knowledge transfer, facilitating information exchange between distinct neighborhoods of sub-problems. Cui et al. [17] considered the convergence interval of the decision variables to make transferred knowledge more suitable for the target task. Wu et al. [18] proposed the shift invariance to measure the inter-task similarity and applied it to a two-stage transferable adaptive differential evolution algorithm. Li et al. [19] proposed a framework to incorporate multiple types of knowledge and provided a transfer adaptation strategy to control the type and quantity of knowledge.

Some researchers also use different crossover operators to achieve knowledge transfer. For instance, Feng et al. [20] used mating methods in particle swarm optimization and differential evolution as crossover operators to achieve knowledge transfer. Similar approaches have been applied in other algorithms such as RPB-MO-MFEA [21], MT-CPSO [22], and MM-DE [23].

These strategies facilitate effective communication and learning between different tasks, thus improving the overall performance of MTO, leading to the application of MTO to various real-world problems such as production group decision-making problems [24], and logistics distribution systems [25].

2.2. RL techniques in evolutionary algorithms

RL [26] is a machine learning method that involves agent learning to make decisions by interacting with the environment so as to maximize a specific reward. Because RL does not rely on labeled training data, some researchers have applied RL techniques in multi-objective evolutionary algorithms and hyper-heuristic evolutionary algorithms.

Kruekaew et al. [27] combined an RL technique with the artificial bee colony algorithm for load balancing MOPs in cloud computing environments. Palm et al. [28] designed a hybrid multi-objective evolutionary algorithm, which uses an RL technique to train an agent for dynamically selecting and combining the multi-objective optimization search strategies. Tian et al. [29] applied deep Q-Network to select the operator for solving MOPs. Dang et al. [30] used the RL technique to allocate computing resources for multi-modal MOPs. In addition, Zuo et al. [31] integrated RL and three change response mechanisms to solve dynamic MOPs. Ming et al. [32] used RL and deep RL to select the most appropriate auxiliary task for the main task for solving constrained MOPs. Furthermore, Leite et al. [33] proposed a neuroevolutionary multi-objective RL method for solving an energy resource management problem.

Recently, RL has also been applied in hyper-heuristic as a high-level strategy. Ji et al. [34] applied RL to select a suitable low-level heuristic from a comprehensive strength-based neighborhood search for dynamic task allocation of crowdsensing. Zhang et al. [35] employed RL to guide the selection of low-level heuristics constructed from various heuristic algorithms to solve distributed flexible job shop scheduling problems.

Notably, this paper only reviews some relevant research on RL in multi-objective evolutionary algorithms and hyper-heuristic evolutionary algorithms. In addition, to the best of our knowledge, there is no research exploring the application of RL techniques in CMTO.

2.3. Motivation

The current research mainly focuses on unconstrained MTO. However, constraints are very common [3] in practical applications. Traditional MTO methods usually adopt simple strategies, such as removing individuals that violate constraints or adding penalty terms to the objective function. Unfortunately, these methods often result in a sharp reduction in population diversity and even difficulty in converging to feasible regions. Some researchers have introduced the FP rule and infeasible solution assisted archiving to solve CMTOPs, but these methods clearly do not consider the difficulty of FP driven MTEA in addressing the constraint challenges in CMTO.

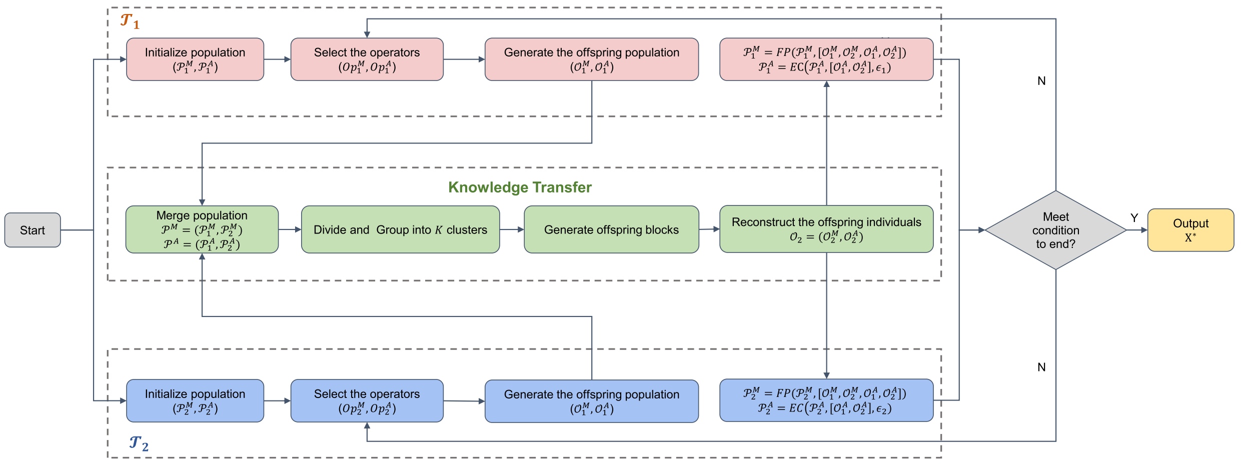

$\mathcal{T}_1$
Initialize population ($\mathcal{P}_1^M,\mathcal{P}_1^A$)
Select the operators (Op_1^M, Op_1^A)
Generate the offspring population ($\mathcal{O}_1^M,\mathcal{O}_1^A$)
$\mathcal{P}_1^M = FP(\mathcal{P}_1^M,[\mathcal{O}_1^M,\mathcal{O}_2^M,\mathcal{O}_1^A,\mathcal{O}_2^A])$
$\mathcal{P}_1^A = EC(\mathcal{P}_1^A,[\mathcal{O}_1^A,\mathcal{O}_2^A],\epsilon_1)$
Knowledge Transfer
Start
Merge population $\mathcal{P}^M = (\mathcal{P}_1^M,\mathcal{P}_2^M)$ $\mathcal{P}^A = (\mathcal{P}_1^A,\mathcal{P}_2^A)$
Divide and Group into K clusters
Generate offspring blocks
Reconstruct the offspring individuals $\mathcal{O}_2 = (\mathcal{O}_2^M,\mathcal{O}_2^A)$
N
Meet condition to end?
Y
Output X*
N
Initialize population ($\mathcal{P}_2^M,\mathcal{P}_2^A$)
Select the operators (Op_2^M, Op_2^A)
Generate the offspring population ($\mathcal{O}_1^M,\mathcal{O}_1^A$)
$\mathcal{P}_2^M = FP(\mathcal{P}_2^M,[\mathcal{O}_1^M,\mathcal{O}_2^M,\mathcal{O}_1^A,\mathcal{O}_2^A])$
$\mathcal{P}_2^A = EC(\mathcal{P}_2^A,[\mathcal{O}_1^A,\mathcal{O}_2^A],\epsilon_2)$
$\mathcal{T}_2$

Fig. 1. The framework of the proposed RL-CMTEA. The evolutionary processes of $\mathcal{T}_1$ and $\mathcal{T}_2$ are represented in red and blue, respectively. Each task has two populations $\mathcal{P}^M$ and $\mathcal{P}^A$ using different constraint handling techniques, and RL selects operators for these populations. The knowledge transfer, represented in green, involves clustering similar dimensions among individuals in both tasks to generate offspring.

First, the FP-driven population is difficult to fully explore infeasible regions, which may lead to the algorithm falling into local optima. To achieve trade-off between diversity, convergence, and feasibility in CMTO, this paper adopts a multi-population method, where each population has different constraint handling techniques to ensure comprehensive exploration of the search space. In addition, recognizing the potential ineffectiveness of knowledge transfer strategies from MTO to CMTO, this paper employs a dimension-based knowledge transfer strategy. This strategy involves clustering similar dimensions between individuals in different tasks and using crossover operators to achieve knowledge transfer. Furthermore, to meet the performance requirements of operators for different tasks and constraints in CMTO, this paper proposes an RL based adaptive operator selection strategy, which selects suitable operators for different populations based on the operators' degree of improvement for each population.

3. The proposed method

3.1. *The framework of the proposed method*

This section provides detailed information about the proposed RL-CMTEA. Specifically, the framework and pseudocode of RL-CMTEA are presented in Fig. 1 and Algorithm 1 respectively. The detailed process is outlined as follows:

Population initialization (Lines 3-7): The populations consisting of solutions are generated using random initialization, which is a conventional setting adopted in many existing works, including the main population $\mathcal{P}_t^M$ and the auxiliary population $\mathcal{P}_t^A$ of task $\mathcal{T}_t$. Next, the fitness values of these solutions are evaluated.

Initialization tables for RL (Lines 10-11): Two tables, $\mathcal{Q}$ and $\mathcal{UCB}$, are initialized to record the rewards generated by each population using different operators and the corresponding $\mathcal{UCB}$ values.

The main loop of RL-CMTEA is from Lines 13 to 37 as follows:

- Select suitable operators for each population (Lines 15-18): At the beginning of each iteration, the agent uses $\mathcal{UCB}$ to select the operators Op_t^M and Op_t^A for $\mathcal{P}_t^M$ and $\mathcal{P}_t^A$ respectively. The selection count $\mathcal{N}_a$ of the selected operators is increased by 1.
- Offspring generation (Lines 20-21): Generate the offspring population, $\mathcal{O}_1^M$ and $\mathcal{O}_1^A$, for each population of task $\mathcal{T}_t$ using the selected operators.
- Knowledge transfer (Lines 23-24): Merge the main population of all tasks and use the current selected operator Op_t^M to generate offspring $\mathcal{O}_2^M$ for the main population $\mathcal{P}_t^M$ of task $\mathcal{T}_t$, thereby achieving knowledge transfer. The auxiliary population $\mathcal{P}_t^A$ obtains its offspring $\mathcal{O}_2^A$ in the same way. The detailed process of knowledge transfer is presented in Section 3.4.
- Environmental selection (Lines 26-29): $\mathcal{O}_1^M$ and $\mathcal{O}_2^M$ are considered to be the offspring of $\mathcal{P}_t^M$. N individuals are selected using the FP rule to form the next generation of main population. Similarly, $\mathcal{O}_1^A$ and $\mathcal{O}_2^A$ are considered to be the offspring of $\mathcal{P}_t^A$, and N individuals are selected using the ϵ-Constraint (EC) [36] to form the next generation of auxiliary population.
- Update the two tables (Lines 31-34): Update $\mathcal{Q}$ and $\mathcal{UCB}$ based on the proportion of generated offspring selected into the next generation as the reward and the selected times of each operator in the population.

The above loop body is repeatedly executed until the termination condition is met. Finally, the best feasible solution is output for each task. The key components of RL-CMTEA are described in detail in the following sections.

Algorithm 1: The pseudocode of RL-CMTEA

```
   Input:
       $T$: number of tasks;
       $N$: population size;
   Output:
       $\{\mathbf{x}_1^*, \mathbf{x}_2^*, ..., \mathbf{x}_T^*\}$: the optimal feasible solution of task $\{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_T\}$
1  for $t = 1 \rightarrow T$ do
2      // Population initialization process
3      $\mathcal{P}_t^M \leftarrow$ Generate the main population for task $\mathcal{T}_t$;
4      $\mathcal{P}_t^A \leftarrow$ Generate the auxiliary population for task $\mathcal{T}_t$;
5      Evaluate fitness of individuals in $\mathcal{P}_t^M$;
6      Evaluate fitness of individuals in $\mathcal{P}_t^A$;
7      $\mathcal{E}_t \leftarrow$ The maximum constraint violation of $\mathcal{P}_t^A$;
8  end
9  // RL initialization process
10 Initialize the tables $\mathcal{Q}$ and $\mathcal{UCB}$;
11 Initialize the selected count $\mathcal{N}_a$;
12 while termination conditions are not met do
13     for $t = 1 \rightarrow T$ do
14         // Select suitable operators for different populations
15         $Op_t^M \leftarrow$ Select operator for $\mathcal{P}_t^M$ according to $\mathcal{UCB}$;
16         $Op_t^A \leftarrow$ Select operator for $\mathcal{P}_t^A$ according to $\mathcal{UCB}$;
17         $\mathcal{N}_a(Op_t^M) \leftarrow \mathcal{N}_a(Op_t^M) + 1$;
18         $\mathcal{N}_a(Op_t^A) \leftarrow \mathcal{N}_a(Op_t^A) + 1$;
19         // Generate offspring population
20         $\mathcal{O}_1^M \leftarrow$ Generate $N$ offspring of $\mathcal{P}_t^M$ using $Op_t^M$;
21         $\mathcal{O}_1^A \leftarrow$ Generate $N$ offspring of $\mathcal{P}_t^A$ using $Op_t^A$;
22         // Knowledge Transfer
23         $\mathcal{O}_2^M \leftarrow$ Generate $N$ offspring of $\mathcal{P}_t^M$ using $Op_t^M$;
24         $\mathcal{O}_2^A \leftarrow$ Generate $N$ offspring of $\mathcal{P}_t^A$ using $Op_t^A$;
25         // Environmental Selection
26         $\mathcal{O}^M \leftarrow \mathcal{O}_1^M \cup \mathcal{O}_2^M$;
27         $\mathcal{O}^A \leftarrow \mathcal{O}_1^A \cup \mathcal{O}_2^A$;
28         $\mathcal{P}_t^M \leftarrow FP(\mathcal{P}_t^M, [\mathcal{O}^M, \mathcal{O}^A])$;
29         $\mathcal{P}_t^A \leftarrow EC(\mathcal{P}_t^A, \mathcal{O}^A, \mathcal{E}_t)$;
30         // Update Q-table and UCB-table
31         $\mathcal{R}^M \leftarrow$ Calculate the reward of $Op_t^M$;
32         $\mathcal{R}^A \leftarrow$ Calculate the reward of $Op_t^A$;
33         $\mathcal{Q} \leftarrow$ Update $\mathcal{Q}$ based on $\mathcal{R}^M$ and $\mathcal{R}^A$ refer to Eq. (6) ;
34         $\mathcal{UCB} \leftarrow$ Update $\mathcal{UCB}$ based on $\mathcal{Q}$ and $\mathcal{N}_a$ refer to Eq. (7) ;
35         $\mathbf{x}_t^* \leftarrow$ Update the optimal feasible solution for $\mathcal{T}_t$;
36     end
37     $\mathcal{E} \leftarrow$ Update the epsilon value refer to Eq. (9);
38 end
39 Return the optimal feasible solution set: $\{\mathbf{x}_1^*, \mathbf{x}_2^*, ..., \mathbf{x}_T^*\}$;
```

3.2. The adaptive operator selection strategy

Selection of a suitable operator is quite difficult for CMTOPs. A CMTOP consists of different tasks and constraints, making it impractical to rely on a universal operator suitable for all problems. To effectively address this challenge, this paper proposed an adaptive operator selection strategy based on Q-learning with the upper confidence bound.

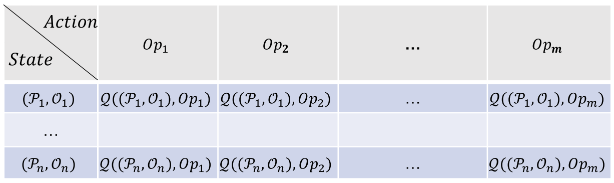
Q-learning exhibits a weak inductive bias and operates without reliance on specific model structures. However, this characteristic also implies the need to learn a substantial number of samples to achieve satisfactory results. In this paper, individuals in the population are considered as states, and operator selection for different populations is considered as action. Because the states and actions are finite and discrete, it is desirable to use Q-learning in the proposed algorithm.

The core idea is employing the improvement of offspring individual concerning the parent population as the reward. This allows the agent to take actions based on the received reward, promoting adaptive selection. The definitions and formulas associated with Q-learning are as follows:

The action is to select an operator for each population, as presented in Eq. (3).

$$A = \{a | a \in \{(\mathcal{P}_1, Op_1), (\mathcal{P}_2, Op_2), \ldots, (\mathcal{P}_n, Op_n)\}\} \tag{3}$$

where the set of action A includes different actions a. For example, $(\mathcal{P}_1, Op_1)$ means assigning the operator Op_1 to the population $\mathcal{P}_1$. The operator pool consists of the following candidate operators:



Action / State	Op_1	Op_2	…	Op_m
$(\mathcal{P}_1, \mathcal{O}_1)$	$\mathcal{Q}((\mathcal{P}_1, \mathcal{O}_1), Op_1)$	$\mathcal{Q}((\mathcal{P}_1, \mathcal{O}_1), Op_2)$	…	$\mathcal{Q}((\mathcal{P}_1, \mathcal{O}_1), Op_m)$
…				
$(\mathcal{P}_n, \mathcal{O}_n)$	$\mathcal{Q}((\mathcal{P}_n, \mathcal{O}_n), Op_1)$	$\mathcal{Q}((\mathcal{P}_n, \mathcal{O}_n), Op_2)$	…	$\mathcal{Q}((\mathcal{P}_n, \mathcal{O}_n), Op_m)$

Fig. 2. The $\mathcal{Q}$-table for Q-Learning.

(1) Simulated binary crossover
The simulated binary crossover (SBX) operator simulates the crossover process designed for binary coding, but it is adapted for real number coding. SBX is advantageous in handling multi-modal problems, particularly those where maintaining diversity is crucial.

(2) DE/rand/1
The DE/rand/1 operator generates offspring by randomly selecting three individuals for mutation and using linear combinations. Its simplicity and intuitive nature enable superior performance with limited computational resources, making it suitable for some straightforward global search problems. However, its search capability may be limited when applied to more complex problems.

(3) DE/rand/2
Compared with DE/rand/1, the DE/rand/2 operator introduces more individual information by using a linear combination to generate offspring, which involves the random selection of five different individuals for mutation. This modification makes this operator more suitable for problems that demand a broader search of the solution space. However, the increased computational cost compared with DE/rand/1 may result in slower convergence rate.

(4) DE/best/1
The DE/best/1 operator uses the optimal individual in the current population as a parent and then randomly selects two more individuals for mutation. This strategy enhances the accuracy of convergence to local optimal and is effective for problems that are more sensitive to local search. However, it may encounter challenges in dealing with global search problems, potentially falling into local optima.

In general, these different operators have their own advantages and are suitable for specific scenarios. Thus, the selection of these operators can be tailored based on the different requirements of the specific problem.

The state is the current population and its offspring populations, as presented in Eq. (4).

$$\mathcal{S} = \{s | s \in \{(\mathcal{P}_1, \mathcal{O}_1), (\mathcal{P}_2, \mathcal{O}_2), \dots, (\mathcal{P}_m, \mathcal{O}_m)\}\} \tag{4}$$

where m denotes the number of operators in the operator candidate pool, the set $\mathcal{S}$ of states includes different states s, and $(\mathcal{P}_1, \mathcal{O}_1)$ represents the offspring population $\mathcal{O}_1$ generated by population $\mathcal{P}_1$.

The reward is the individual changes brought by the offspring generated using the selected operator, as presented in Eq. (5).

$$R_t = \frac{|\mathcal{O}_t|}{|\mathcal{P}_t|} \tag{5}$$

Then, the $\mathcal{Q}$-table is updated using the Bellman function, as presented in Eq. (6).

$$\mathcal{Q}(s,a) = \mathcal{Q}(s,a) + \alpha[r + \gamma(\max \mathcal{Q}(s', a') - \mathcal{Q}(s,a))] \tag{6}$$

where α denotes the learning rate; γ, the discount factor; and $\mathcal{Q}(s,a)$, the expected reward for taking action a in state s. The actual reward received by the action is guided by r, and s' refers to the next state. The highest expected reward of all possible actions a' in state s' is represented by $\max \mathcal{Q}(s', a')$.

The Bellman function considers both the real-time reward for performing an action in a specific state and the expected future reward for taking the next action, thus updating the values in the $\mathcal{Q}$-table, as shown in Fig. 2. The algorithm uses the upper confidence interval definition strategy and $\mathcal{Q}$ to update the $\mathcal{UCB}$, as presented in Eq. (7).

$$\mathcal{UCB}(s,a) = \mathcal{Q}(s,a) + \mathcal{U}(a) \tag{7}$$

where $\mathcal{Q}(s,a)$ denotes the effect and reflects the benefit of action a, and $\mathcal{U}(a)$ represents the upper boundary of the confidence interval, as presented in Eq. (8).

$$\mathcal{U}(a) = \sqrt{(2log(MaxT))/\mathcal{N}_a(a)} \tag{8}$$

where $\mathcal{U}(a)$ denotes the uncertainty of the effect of action a; $MaxT$, the total number of selections; and $\mathcal{N}_a(a)$, the number of times action a is selected.

When $\mathcal{N}_a(a)$ is low, the corresponding value of $\mathcal{U}(a)$ tends to be higher. This indicates that when there are only a limited number of choices for action a, it becomes challenging to assess the potential advantages it can offer to the algorithm. Consequently, action a is more likely to undergo further exploration.

As the selected times increase, the uncertainty of action a diminishes over time. The influence of the uncertainty term gradually diminishes, and the probability of selecting the action becomes increasingly dominated by the Q-value. When actions with higher rewards are consistently selected more frequently, the Q-value tends to stabilize at a relatively high level. Even as the term $\mathcal{U}(a)$ diminishes, the likelihood of selecting the action remains high. Conversely, for actions with lower rewards that are selected more often, the Q-value stabilizes at a lower level. As the term $\mathcal{U}(a)$ diminishes further, there is a diminishing chance of selecting these actions.

The upper confidence bound can not only quickly give up those actions with low rewards, but also take care of those actions with less times of being selected, providing them with a chance to stand out.

3.3. The multi-population method with different constrained handling techniques

This section provides a detailed description of how the proposed algorithm problem uses multiple population to solve CMTOPs. For each task, two different populations are established, each utilizing different constraint handling techniques:

The main population uses the FP rule to handle constraints. The goal of the main population is to find feasible solutions and ensure that the basic requirements of each task are met.

The auxiliary population uses the EC rule to handle constraints. This rule introduces an epsilon value ϵ, as presented in Eq. (9).

$$\epsilon(g) = \begin{cases} \epsilon(0)(1 - \frac{g}{T_c})^{cp}, & if \quad 0 \le g \le T_c \\ 0, & if \ \ g > T_c \end{cases} \tag{9}$$

where g denotes the current generation, and cp denotes the control factor. During the iteration process, the algorithm gradually reduces the epsilon value, allowing the auxiliary population to gradually approach the feasible regions. Once g exceeds Tc, the epsilon value is set to zero to only obtain feasible solutions. If all constraint violations of a solution are less than this value, the solution is considered epsilon-feasible.

The auxiliary population provides information on solutions with good convergence and diversity, facilitating the main population's efficient exploration of feasible regions. Due to the different techniques of the main and auxiliary populations in handling constraints, their requirements for operators may vary. In the evolutionary process, RL allocates the same or different operators based on their respective states to achieve their own evolutionary goals.

Compared with the single-population method, the multi-population method has the following obvious advantages when dealing with CMTOPs:

(1) Task-specific constraint handling: The multi-population method allows the allocation of an independent population to each task. Consequently, different tasks can use distinct techniques to handle constraints, which are tailored to meet the unique requirements. For instance, certain tasks may need stringent constraints, whereas others may benefit from more lenient constraints. This task-specific constraint handling technique enhances the performance of each task.
(2) Avoid cross interference: The multi-population method ensures that populations for each task run independently. This independence is vital to the prevention of cross-interference between different tasks, as the constraint handling for one task may negatively affect others. Contrarily, the single-population method may encounter challenges in effectively controlling cross interference between different tasks.
(3) Task separation and modularity: The inherent task-separation characteristic of the multi-population method fosters modular designs, enabling the development and adaptation of each population independently without causing interference to other tasks. This modularity simplifies system management and maintenance, allowing parallel processing of different tasks.

3.4. Knowledge transfer based on similar dimensions

Knowledge transfer plays a crucial role in CMTO. Different from MTO, the constraints make the inter-task similarities contingent on the intersection degree of feasible regions between tasks. Substantial differences exist in the distribution of individuals in different tasks, which may result in ineffective individual-based knowledge transfer.

Compared with individual-based knowledge transfer, the dimension-based knowledge transfer reflects the common characteristics of two tasks through the similarity dimensions between different individuals. This approach considers constraints of different tasks to achieve a more precise information exchange. The pseudocode of this method is presented in Algorithm 2, which specifically includes the following steps:

- First, merge the populations $\{\mathcal{P}_1, ..., \mathcal{P}_T\}$ from tasks $\{\mathcal{T}_1, ..., \mathcal{T}_T\}$ into one population, denoted as $\mathcal{P}$.
- The individuals are divided into individual blocks, each with a length of B. If the last segment of an individual is shorter than B, it is padded with zeros to meet the specified length. There are w blocks are formed, as presented in Eq. (10).

$$w = \sum_{t=1}^{T} \left\lceil \frac{D_t}{B} \right\rceil \cdot N \tag{10}$$

where T denotes the number of tasks; N, the population size; D_t, the dimension of task $\mathcal{T}_t$.

Algorithm 2: Knowledge Transfer Based on Similar Dimensions

```
Input:
    $N$: population size ;
    $Op$: the operator of the current population;
    $\{\mathcal{P}_1, \mathcal{P}_2, ..., \mathcal{P}_T\}$: the population of task $\{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_T\}$;
    $B, K$: the current value of block size $B$ and cluster number $K$;
    $B_{min}, B_{max}$: the minimum and maximum values of $B$;
    $K_{min}, K_{max}$: the minimum and maximum values of $K$;
Output: $\mathcal{O}$: the offspring population generated by knowledge transfer
1  $\mathcal{P} \leftarrow \mathcal{P}_1 \cup \mathcal{P}_2 \cup ... \cup \mathcal{P}_T$ ;
2  $\mathcal{B} \leftarrow$ Divide individuals in $\mathcal{P}$ into blocks of size $B$;
3  $C \leftarrow$ Cluster $\mathcal{B}$ using K-means into $K$ clusters;
4  for $k = 1 \rightarrow K$ do
5      $\mathcal{O}_k^{\mathcal{B}} \leftarrow$ Generate offspring blocks in $C_k$;
6  end
7  $\mathcal{O} \leftarrow$ Reconstruct $N$ offspring by placing $\mathcal{O}^{\mathcal{B}}$ into the corresponding dimensions;
8  if the optimal solution is updated then
9      $B \leftarrow randi(B-1, B+1)$;
10     $K \leftarrow randi(K-1, K+1)$;
11 else
12     $B \leftarrow randi(B_{min}, B_{max})$;
13     $K \leftarrow randi(K_{min}, K_{max})$;
14 end
15 Return the offspring population: $\mathcal{O}$;
```

- The K-means algorithm is employed to cluster these w individual blocks, forming K clusters based on similar dimensions.
- Perform crossover and mutation in the same cluster to achieve knowledge transfer. To ensure the effectiveness of the selected operator in handling constraints, RL is used to select suitable operators for the knowledge transfer of each task.
- Combine the offspring blocks $\mathcal{O}^{\mathcal{B}}$ into their corresponding dimensions and reconstruct the offspring individuals. These offspring individuals are incorporated into environmental selection, contributing to the formation of the next generation population.
- Update the values of B and K: If the offspring individuals generated by the knowledge transfer exhibit positive results, B and K randomly sample in the values of the previous generation. Contrarily, B and K are reinitialized.

This strategy facilitates the grouping of similar dimensions from different tasks into the same cluster, thereby allowing positive knowledge transfer. Simultaneously, it allows the grouping of similar individuals within the same task into the same cluster, promoting knowledge transfer within the task. Furthermore, dissimilar individual blocks are segregated into different clusters, mitigating the risk of negative transfer between irrelevant dimensions.

3.5. *Computational complexity of RL-CMTEA*

According to Algorithm 1, the time complexity of the proposed RL-CMTEA is mainly determined by the five steps in the generation of offspring for each task, i.e., operator selection, offspring generation, knowledge transfer, environment selection, reward calculation, and updating of tables. Specifically, in the operator selection process, updating the Q-values of three different populations using four different operators has a constant time complexity of $O(1)$ for each state-action pair. The time complexity for offspring generation is $O(ND)$, and the time complexity for environment selection is $O(M)$. For the knowledge transfer, clustering two populations into K clusters has a time complexity of $O(IKND)$, where I denotes the number of iterations. The time complexity of recombination is $O(KND)$. Thus, the total time complexity of each task in one generation is $O(IKND)$.

4. Experiments and numerical analysis

To evaluate the effectiveness of RL-CMTEA, a comprehensive set of experiments are conducted.

Initially, RL-CMTEA is compared with FP-driven variants, including GA [37], DE [38], MFEA [39], MFDE [20], and BLKT-DE [40], and A-CMFEA [7]. Then, comparisons are made with several advanced constrained single-task evolutionary algorithms (CSTEAs), namely C2oDE [41], CAL-SHADE [42], CORCO [43], DeCODE [44], ECHT-DE [36], MTV-DE [45], VMCH_LSHADE44 [46], and LSHADE44 [47]. Subsequently, the robustness of the proposed algorithm is assessed by comparing its performance with other algorithms in different dimensions. Then, a sensitivity analysis is underway on the parameters of the algorithm, covering both the operator parameters of the candidate operator pool and the Q-learning parameters. Finally, various variants of RL-CMTEA are introduced to evaluate the effectiveness of each component.

The experimental benchmark is the constrained multi-task benchmark set CMT [6]. All experiments are conducted on the MToP platform [48], ensuring the reliability and consistency of the experimental results.

4.1. *Experimental setup*

(1) Population size: $N = 50 \times T$, Dimension: $D = 100$.

(2) The maximum number of evaluations of the function: $1000 \times 100 \times T$.
(3) DE parameters:
  a) scale factor: $F = 0.5$,
  b) crossover rate: $CR = 0.5$.
  SBX crossover:
  a) crossover probability: $p_c = 1$,
  b) distribution index: $\eta_c = 2$.
  Polynomial mutation:
  a) mutation probability: $p_m = 1$,
  b) distribution index: $\eta_m = 5$.
  A-CMFEA parameters:
  a) probability of variable swap: $probwap = 0$,
  b) initial random mating probability: $rmp_0 = 0.3$,
  c) constant c: $c = 0.1$.
(4) RL-CMTEA parameters:
  a) learning rate: $\alpha = 0.01$,
  b) discount factor: $\gamma = 0.9$,
  c) the minimum and maximum number of clusters: $K_{min} = 2$, $K_{max} = N/2$,
  d) the minimum and maximum individual block lengths: $B_{min} = 1$, $B_{max} = D$.

4.2. Results and discussion

4.2.1. Comparison with other constrained evolutionary algorithms

For each task in CMT, this section uses RL-CMTEA and other 14 algorithms to find the optimal feasible solutions. Table 1 and Table 2 present the average fitness values and standard deviations of CMTEAs and CSTEAs, respectively. To evaluate the performance of RL-CMTEA, the Friedman aligned test[1] is employed to rank the fitness value of each test case. A smaller ranking indicates superior performance. The 'NaN' value indicates that no feasible solution has been found or there is a situation where a feasible solution cannot be found over 30 independent runs.

As shown in Table 1 and Table 2, RL-CMTEA exhibits significant performance advantages when compared with other CMTEAs.

Compared with FP-driven MTEAs, RL-CMTEA ranks first in 18 tasks. Out of these 18 tasks, it achieved optimal results on 11 tasks. Compared with A-CMFEA, RL-CMTEA performed superiorly on 11 tasks, inferior to it on 5 tasks, and equivalent to it on 2 tasks. Compared with other PF-driven MTEAs, RL-CMTEA shows a significant performance advantage, highlighting its ability to outperform current CMTEAs.

Compared with CSTEAs, RL-CMTEA similarly ranks first in 18 tasks. Out of these 18 tasks, the proposed algorithm achieved optimal results on 14 tasks. Obviously, the proposed algorithm is substantially better than other CSTEAs in solving CMT.

Based on the experimental results, the proposed algorithm is the top-ranking algorithm, highlighting its superior performance in solving CMTOPs. The success of RL-CMTEA can be attributed to the following factors:

(1) The effectiveness of knowledge transfer. Knowledge transfer changes accordingly with the degree of overlap between feasible regions between tasks. When the feasible regions' overlap of tasks is large, individuals of different tasks have more similar dimensions, which is conducive to their grouping into the same cluster, so that knowledge transfer occurs more on similar dimensions between different tasks. For example, in CMT1 to CMT3, RL-CMTEA improves the degree of knowledge transfer between tasks by performing mutation and crossover on similar dimensions of different individuals. When the feasible regions of tasks intersect only partially, the number of individuals with similar dimensions between different tasks decreases, and therefore, the knowledge transfer between different tasks decreases. Finally, when the feasible regions of tasks do not overlap at all, there are almost no individuals with similar dimensions between different tasks; thus, the knowledge transfer mainly occurs within the same task, effectively preventing negative transfer.
(2) The auxiliary population can provide the main population with information on feasible solutions with good convergence, particularly when the distance between the unconstrained optimal solution and the feasible region is very close. The role of the auxiliary population is to accelerate the main population through the infeasible regions so that it can reach the vicinity of the feasible regions faster, thereby more effectively finding feasible solutions with better convergence.
(3) The adaptive operator selection based on action rewards. The agent selects the operators for different populations that are more suitable in the current task. Although trying out each operator is required in the early stages of evolution, the effort is worth it in the long-term as it helps find the operator that is best suited to solve the problem. Judging from the standard deviation results, the proposed algorithm exhibits good stability, which also reflects the advantages of RL in long-term returns.

To highlight the convergence performance of the proposed algorithm, Fig. 3 and Fig. 4 present the convergence curves of RL-CMTEA compared with FP-driven CMTEAs and CSTEAs, respectively. Taking the convergence curve of CMT1-T_1 as an example, in

[1] The Friedman test is implemented by the KEEL Software Tool available at https://sci2s.ugr.es/keel/.

Table 1

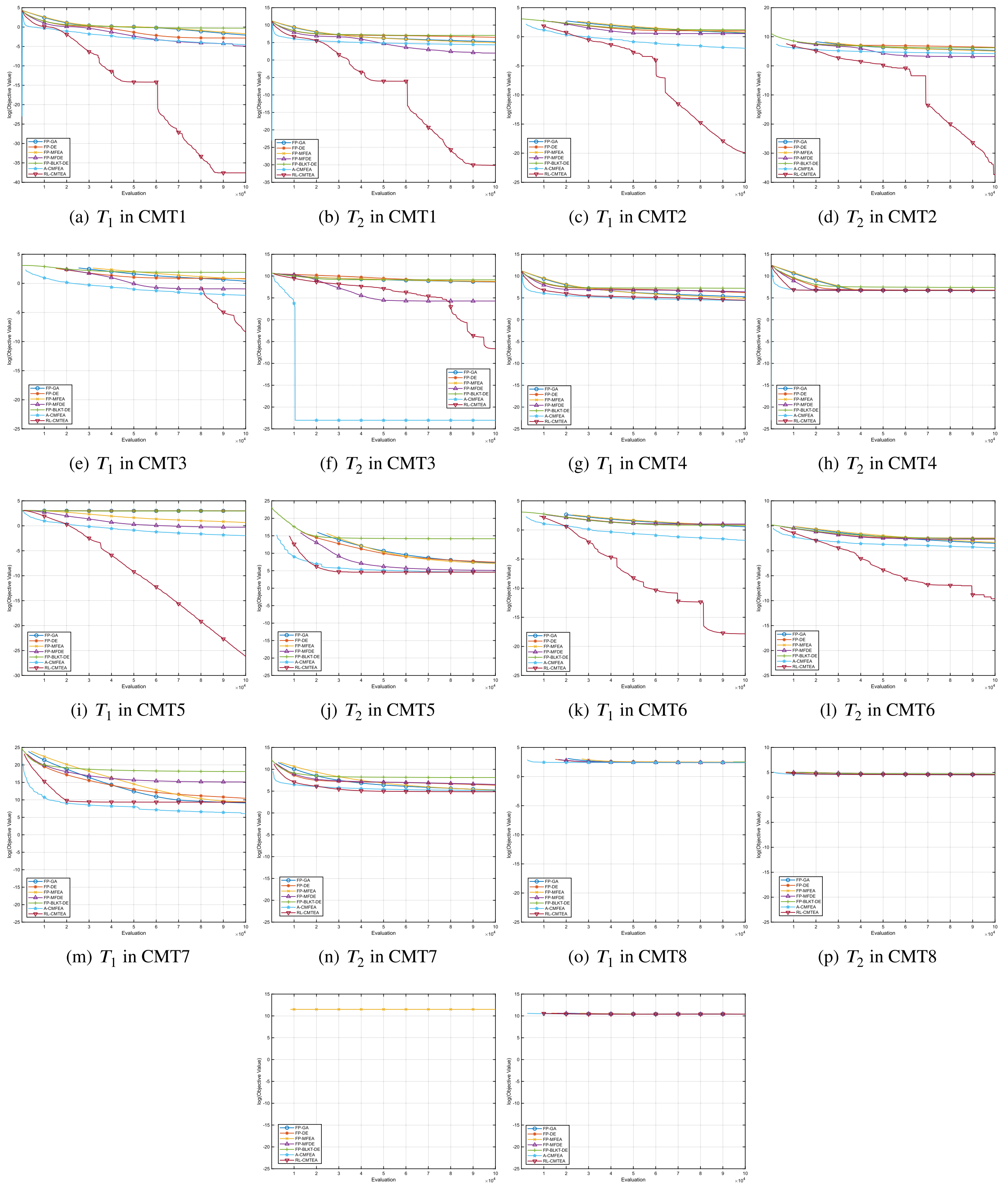
Average fitness values and standard deviations of FP-GA, FP-DE, FP-MFEA, FP-MFDE, FP-BLKT-DE, A-CMFEA and RL-CMTEA on CMT among 30 runs. The best results in each row are highlighted.

Problem	FP-GA	FP-DE	FP-MFEA	FP-MFDE	FP-BLKT-DE	A-CMFEA	RL-CMTEA
CMT1-T1	1.25e-01 (2.61e-02)	5.83e-02 (1.06e-01)	1.79e-01 (3.63e-02)	7.37e-03 (1.95e-02)	7.41e-01 (8.21e-01)	1.09e-02 (1.25e-02)	**4.81e-17 (1.19e-16)**
CMT1-T2	1.93e+02 (2.95e+01)	6.64e+02 (2.09e+02)	1.41e+02 (1.65e+01)	7.16e+00 (1.08e+01)	1.15e+03 (1.34e+03)	7.69e+01 (7.27e+01)	**7.98e-14 (1.63e-13)**
CMT2-T1	1.86e+00 (2.20e-01)	2.77e+00 (8.51e-01)	2.14e+00 (1.94e-01)	1.72e+00 (3.37e-01)	3.30e+00 (4.03e+00)	1.38e-01 (1.00e-01)	**2.19e-09 (2.85e-09)**
CMT2-T2	2.00e+02 (2.49e+01)	5.74e+02 (2.01e+02)	1.57e+02 (1.81e+01)	2.42e+01 (9.75e+00)	4.98e+02 (6.81e+02)	6.71e+01 (6.32e+01)	**5.92e-17 (3.24e-16)**
CMT3-T1	1.44e+00 (3.30e-01)	2.27e+00 (7.26e-01)	2.07e+00 (3.84e-01)	3.85e-01 (1.24e+00)	6.61e+00 (6.95e+00)	1.29e-01 (1.71e-01)	**2.28e-04 (5.86e-04)**
CMT3-T2	5.66e+03 (7.05e+02)	5.95e+03 (1.51e+03)	6.64e+03 (2.38e+03)	7.24e+01 (2.63e+02)	9.34e+03 (6.62e+03)	**-6.21e+02 (1.27e+03)**	1.30e-03 (9.05e-05)
CMT4-T1	1.93e+02 (2.23e+01)	4.97e+02 (1.78e+02)	1.34e+02 (1.42e+01)	5.95e+02 (1.88e+02)	1.35e+03 (1.38e+03)	**8.06e+01 (5.88e+01)**	8.79e+01 (8.74e+01)
CMT4-T2	8.43e+02 (1.18e+01)	8.50e+02 (8.76e+01)	8.53e+02 (1.17e+01)	8.23e+02 (1.86e+01)	1.62e+03 (2.06e+03)	**7.83e+02 (3.43e+01)**	8.15e+02 (1.88e+01)
CMT5-T1	1.99e+01 (8.04e-02)	2.00e+01 (1.43e-04)	1.94e+00 (1.86e-01)	7.52e-01 (7.04e-01)	1.95e+01 (2.92e+00)	1.39e-01 (9.30e-02)	**4.29e-12 (6.70e-12)**
CMT5-T2	1.37e+03 (2.79e+02)	1.71e+03 (6.54e+03)	1.19e+03 (3.37e+02)	1.62e+02 (7.01e+01)	1.41e+06 (4.35e+06)	9.91e+01 (3.34e+01)	**9.74e+01 (6.48e-01)**
CMT6-T1	1.74e+00 (3.31e-01)	2.64e+00 (6.41e-01)	2.11e+00 (1.83e-01)	2.77e+00 (6.80e-01)	2.23e+00 (3.56e+00)	1.68e-01 (9.56e-02)	**1.79e-08 (5.53e-08)**
CMT6-T2	4.41e+00 (3.58e-01)	1.26e+01 (3.66e+00)	5.36e+00 (3.79e-01)	1.01e+01 (3.07e+00)	1.15e+01 (2.16e+01)	1.76e+00 (8.43e-01)	**6.60e-05 (3.52e-04)**
CMT7-T1	9.19e+03 (2.51e+03)	3.43e+04 (9.16e+04)	1.28e+04 (3.82e+03)	3.69e+06 (8.83e+06)	7.24e+07 (2.00e+08)	**4.41e+02 (1.06e+03)**	1.13e+04 (7.75e+03)
CMT7-T2	1.95e+02 (1.71e+01)	5.79e+02 (2.09e+02)	1.82e+02 (1.46e+01)	6.83e+02 (2.31e+02)	3.36e+03 (1.03e+04)	1.65e+02 (4.83e+01)	**1.29e+02 (5.46e+00)**
CMT8-T1	1.14e+01 (6.13e-02)	**1.10e+01 (2.57e-02)**	1.20e+01 (1.26e-01)	1.11e+01 (2.15e-01)	1.25e+01 (2.08e+00)	1.10e+01 (3.74e-02)	1.61e+01 (4.95e+00)
CMT8-T2	9.94e+01 (1.26e+00)	9.78e+01 (1.71e+00)	1.00e+02 (1.57e+00)	9.83e+01 (1.84e+00)	1.09e+02 (9.74e+00)	**9.18e+01 (1.58e+00)**	9.19e+01 (3.00e+00)
CMT9-T1	9.94e+04 (1.09e+04)	1.12e+05 (0.00e+00)	9.92e+04 (1.07e+04)	NaN (NaN)	NaN (NaN)	5.40e+04 (5.22e+04)	**1.94e+01 (6.62e+01)**
CMT9-T2	3.35e+04 (9.72e+01)	3.33e+04 (1.50e+02)	3.37e+04 (1.15e+02)	3.35e+04 (3.59e+02)	3.42e+04 (1.40e+03)	3.36e+04 (3.86e+02)	**3.32e+04 (4.33e+01)**
Ranking	68.1389	74.3056	67.5556	74.9444	94.0278	46.8611	18.6667

Table 2

Average fitness values and standard deviations of C2oDE, CAL-SHADE, CORCO, DeCODE, ECHT-DE, LSHADE44, MTV-DE, VMCH-LSHADE44 and RL-CMTEA on CMT among 30 runs. The best results in each row are highlighted.

Problem	C2oDE	CAL-SHADE	CORCO	DeCODE	ECHT-DE	LSHADE44	MTV-DE	VMCH-LSHADE44	RL-CMTEA
CMT1-T1	1.47e-03 (4.63e-03)	3.46e-02 (3.81e-02)	2.63e-03 (6.83e-03)	1.97e-03 (5.54e-03)	1.33e+00 (9.58e-02)	1.32e-02 (3.16e-02)	5.68e-02 (1.47e-01)	1.43e-02 (2.22e-02)	**4.81e-17 (1.19e-16)**
CMT1-T2	7.11e+02 (3.78e+01)	2.09e+01 (4.83e+00)	5.70e+02 (4.02e+01)	5.38e+02 (6.37e+01)	1.40e+03 (1.19e+02)	1.16e+02 (1.44e+02)	7.40e+02 (1.20e+02)	1.40e+02 (1.62e+02)	**7.98e-14 (1.63e-13)**
CMT2-T1	1.30e+01 (3.55e-01)	1.43e+00 (2.68e+00)	6.01e-01 (5.41e-01)	9.19e-01 (4.94e-01)	9.44e+00 (9.54e-01)	4.05e+00 (8.24e-01)	1.94e+00 (4.17e-01)	4.09e+00 (8.86e-01)	**2.19e-09 (2.85e-09)**
CMT2-T2	2.63e+03 (2.25e+02)	1.30e+01 (5.56e+00)	5.61e+02 (3.77e+01)	5.36e+02 (5.06e+01)	1.64e+03 (2.40e+02)	1.77e+02 (1.81e+02)	7.85e+02 (1.58e+02)	1.59e+02 (1.55e+02)	**5.92e-17 (3.24e-16)**
CMT3-T1	7.17e+00 (4.64e+00)	1.27e+01 (6.78e-01)	5.54e-01 (1.22e+00)	8.70e-01 (5.64e-01)	6.72e+00 (5.81e-01)	3.97e+00 (6.70e-01)	2.01e+00 (6.82e-01)	4.38e+00 (1.00e+00)	**2.28e-04 (5.86e-04)**
CMT3-T2	3.22e+04 (2.00e+03)	3.00e+04 (6.17e+03)	1.77e+04 (8.74e+03)	1.36e+04 (1.06e+03)	2.92e+04 (1.84e+03)	1.57e+03 (2.04e+03)	6.93e+03 (1.56e+03)	2.36e+03 (2.16e+03)	**1.30e-03 (9.05e-05)**
CMT4-T1	6.96e+02 (3.93e+01)	**2.30e+01 (1.02e+01)**	5.51e+02 (4.99e+01)	5.43e+02 (4.09e+01)	1.39e+03 (8.93e+01)	1.97e+02 (1.99e+02)	7.69e+02 (1.36e+02)	1.16e+02 (1.23e+02)	8.79e+01 (8.74e+01)
CMT4-T2	8.53e+02 (8.88e+00)	8.24e+02 (1.75e+01)	8.24e+02 (1.12e+01)	8.11e+02 (8.47e+00)	1.47e+03 (4.18e+02)	7.94e+02 (1.03e+01)	8.33e+02 (1.95e+01)	**7.94e+02 (8.26e+00)**	8.15e+02 (1.88e+01)
CMT5-T1	2.00e+01 (3.68e-02)	3.07e+00 (1.30e+00)	1.17e+01 (9.57e+00)	2.00e+01 (2.21e-02)	2.00e+01 (3.29e-04)	1.99e+01 (7.96e-02)	2.00e+01 (1.72e-04)	1.99e+01 (6.30e-02)	**4.29e-12 (6.70e-12)**
CMT5-T2	3.68e+06 (5.30e+05)	3.02e+02 (5.50e+01)	2.99e+02 (6.79e+01)	3.03e+02 (7.93e+01)	8.59e+05 (3.21e+05)	1.85e+02 (5.53e+01)	1.66e+04 (5.92e+04)	1.74e+02 (4.13e+01)	**9.74e+01 (6.48e-01)**
CMT6-T1	1.18e+01 (3.52e-01)	8.90e-01 (8.89e-01)	6.68e-01 (5.71e-01)	6.60e-01 (5.99e-01)	8.41e+00 (7.46e-01)	3.82e+00 (9.59e-01)	1.93e+00 (4.67e-01)	3.81e+00 (8.26e-01)	**1.79e-08 (5.53e-08)**
CMT6-T2	1.05e+02 (3.57e+00)	1.38e+01 (2.59e+01)	8.14e-01 (8.61e-01)	4.79e+00 (1.88e+00)	5.66e+01 (5.89e+00)	2.28e+01 (5.12e+00)	9.44e+00 (2.99e+00)	2.34e+01 (4.74e+00)	**6.60e-05 (3.52e-04)**
CMT7-T1	4.36e+03 (2.70e+03)	3.64e+03 (1.92e+03)	1.68e+03 (1.00e+03)	**1.41e+03 (7.61e+02)**	1.86e+06 (7.59e+05)	2.79e+03 (2.23e+03)	1.96e+05 (4.09e+05)	2.94e+03 (2.50e+03)	1.13e+04 (7.75e+03)
CMT7-T2	6.97e+02 (3.61e+01)	1.37e+02 (3.77e+00)	5.72e+02 (3.15e+01)	5.77e+02 (3.09e+01)	1.28e+03 (9.18e+01)	3.12e+02 (1.54e+02)	7.69e+02 (1.52e+02)	2.63e+02 (1.69e+02)	**1.29e+02 (5.46e+00)**
CMT8-T1	1.71e+01 (6.78e-01)	2.01e+01 (6.88e-01)	NaN (NaN)	**1.10e+01 (5.19e-03)**	1.62e+01 (8.72e-01)	1.10e+01 (8.18e-03)	1.11e+01 (4.03e-02)	1.10e+01 (9.52e-03)	1.61e+01 (4.95e+00)
CMT8-T2	1.50e+02 (2.98e+00)	1.67e+02 (2.21e+00)	NaN (NaN)	1.08e+02 (4.00e+00)	1.46e+02 (2.87e+00)	9.67e+01 (4.82e+00)	9.85e+01 (2.06e+00)	9.66e+01 (4.66e+00)	**9.19e+01 (3.00e+00)**
CMT9-T1	8.41e+04 (1.66e+04)	5.13e+04 (1.41e+04)	1.84e+03 (6.93e+02)	3.48e+03 (2.33e+03)	1.23e+05 (0.00e+00)	9.74e+04 (1.32e+04)	8.01e+04 (2.51e+04)	9.50e+04 (1.30e+04)	**1.94e+01 (6.62e+01)**
CMT9-T2	4.06e+04 (3.24e+02)	4.22e+04 (3.76e+02)	3.39e+04 (6.84e+02)	3.34e+04 (2.80e+02)	4.07e+04 (4.42e+02)	3.36e+04 (3.75e+02)	3.33e+04 (1.76e+02)	3.35e+04 (3.62e+02)	**3.32e+04 (4.33e+01)**
Ranking	102.9722	76.25	88.25	74.8056	113.3056	80	95.9722	81.0556	20.8889



(a) T_1 in CMT1 (b) T_2 in CMT1 (c) T_1 in CMT2 (d) T_2 in CMT2

(e) T_1 in CMT3 (f) T_2 in CMT3 (g) T_1 in CMT4 (h) T_2 in CMT4

(i) T_1 in CMT5 (j) T_2 in CMT5 (k) T_1 in CMT6 (l) T_2 in CMT6

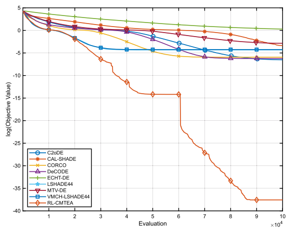
(m) T_1 in CMT7 (n) T_2 in CMT7 (o) T_1 in CMT8 (p) T_2 in CMT8

(q) T_1 in CMT9 (r) T_2 in CMT9

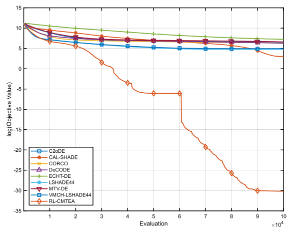
Fig. 3. Convergence curves obtained by RL-CMTEA and other compared CMTEAs on CMT among 30 runs.

the early stage, the convergence rate of RL-CMTEA did not exceed that of A-CMFEA. However, the proposed algorithm has the fastest convergence rate and the best convergence trend after about 20,000 evaluation times. Similar results have also been obtained on other tasks.

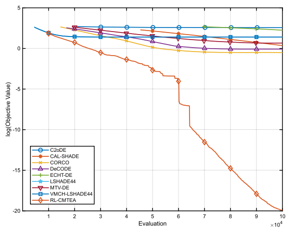
The initially observed slower convergence rate of RL-CMTEA can be attributed to the learning phase of reinforcement learning, which requires a considerable number of samples to evaluate the performance of different operators for specific tasks. This observation aligns with the underlying hypothesis posited in this paper. The exploration of the search space by the auxiliary population also contributes to the consumption of evaluation times, further affecting the initial convergence rate. However, as RL adapts and



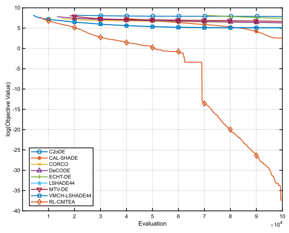
(a) T_1 in CMT1



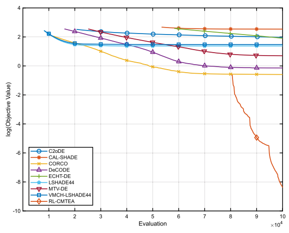
(b) T_2 in CMT1



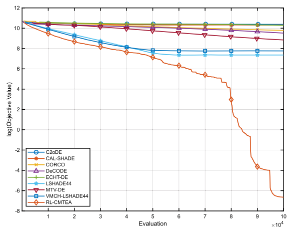
(c) T_1 in CMT2



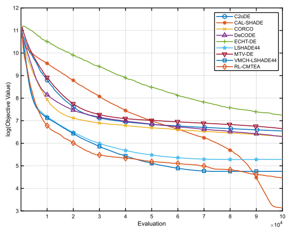
(d) T_2 in CMT2



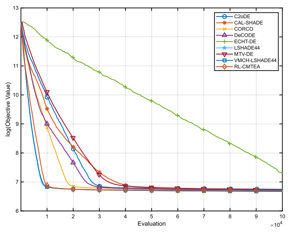
(e) T_1 in CMT3



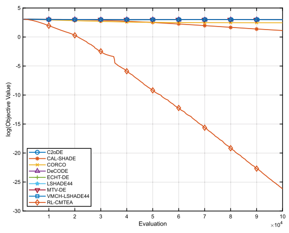
(f) T_2 in CMT3



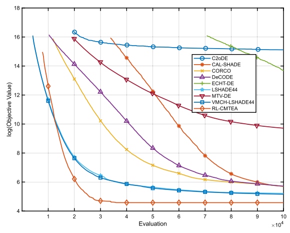
(g) T_1 in CMT4



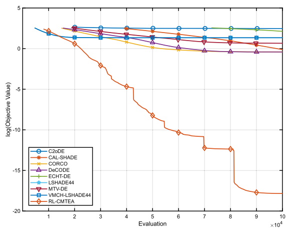
(h) T_2 in CMT4



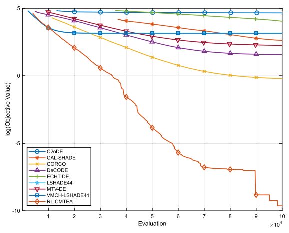
(i) T_1 in CMT5



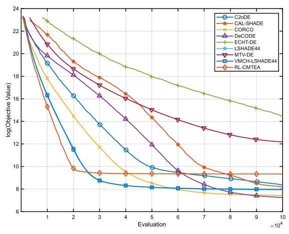
(j) T_2 in CMT5



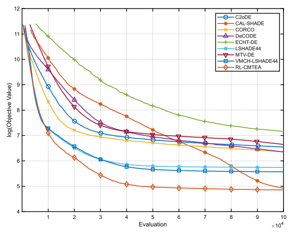
(k) T_1 in CMT6



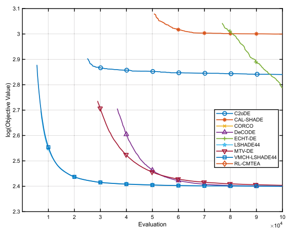
(l) T_2 in CMT6



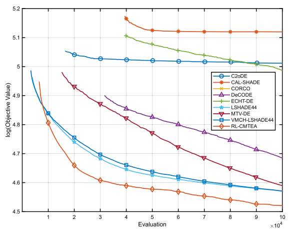
(m) T_1 in CMT7



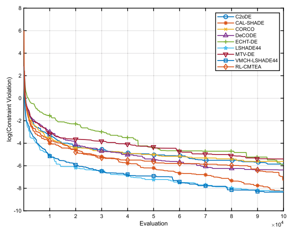
(n) T_2 in CMT7



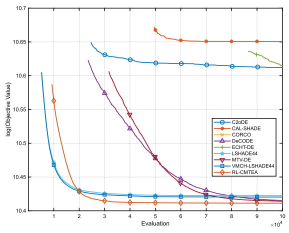
(o) T_1 in CMT8



(p) T_2 in CMT8



(q) T_1 in CMT9



(r) T_2 in CMT9

Fig. 4. Convergence curves obtained by RL-CMTEA and other compared CSTEAs on CMT among 30 runs.

selects the suitable operator for the task, coupled with the diverse solutions provided by the auxiliary population, the performance of the algorithm substantially improves in subsequent iterations. This observation concurs with the experimental results presented in Table 1 and Table 2, providing additional confirmation of the effectiveness of the proposed algorithm.

Table S-1 and Table S-2 in the supplementary file present the adjusted p-values for the fitness values obtained through the Friedman aligned test using RL-CMTEA as the control algorithm. To assess the statistical differences between RL-CMTEA and the other 14 algorithms, a series of post-hoc tests are conducted. To control the family-wise error rate, several post-hoc procedures [49] are employed, including the Holm, Hochberg, Hommel, Holland, Rom, Finner, and Li procedures. The results of these tests

Table 3
Average fitness values and standard deviations of RL-CMTEA-a1, RL-CMTEA-a2, RL-CMTEA-a3, RL-CMTEA-a4, and RL-CMTEA on CMT among 30 runs. The best results in each row are highlighted.

Problem	RL-CMTEA-a1	RL-CMTEA-a2	RL-CMTEA-a3	RL-CMTEA-a4	RL-CMTEA
CMT1-T1	1.40e-11 (7.67e-11)	8.64e-10 (4.73e-09)	6.27e-06 (3.31e-05)	3.29e-06 (1.77e-05)	**4.81e-17 (1.19e-16)**
CMT1-T2	1.22e-07 (6.66e-07)	4.26e-06 (2.33e-05)	3.63e-02 (1.94e-01)	2.66e-02 (1.44e-01)	**7.98e-14 (1.63e-13)**
CMT2-T1	3.17e-04 (1.71e-03)	1.78e-04 (6.60e-04)	1.15e-04 (4.26e-04)	8.34e-04 (2.62e-03)	**2.19e-09 (2.85e-09)**
CMT2-T2	3.43e-03 (1.88e-02)	5.50e-04 (2.71e-03)	2.32e-04 (9.74e-04)	8.45e-03 (3.31e-02)	**5.92e-17 (3.24e-16)**
CMT3-T1	9.78e-01 (3.09e+00)	6.78e-01 (2.36e+00)	1.47e+00 (4.01e+00)	1.21e+00 (3.84e+00)	**2.28e-04 (5.86e-04)**
CMT3-T2	2.54e+02 (7.13e+02)	1.77e+02 (8.59e+02)	1.91e+02 (5.37e+02)	1.79e+02 (5.37e+02)	**1.30e-03 (9.05e-05)**
CMT4-T1	1.36e+02 (7.09e+01)	1.50e+02 (8.21e+01)	1.44e+02 (6.91e+01)	1.55e+02 (7.86e+01)	**8.79e+01 (8.74e+01)**
CMT4-T2	8.30e+02 (1.57e+01)	8.23e+02 (2.05e+01)	8.30e+02 (1.79e+01)	8.22e+02 (1.40e+01)	**8.15e+02 (1.88e+01)**
CMT5-T1	8.88e-03 (4.00e-02)	2.18e-05 (9.85e-05)	1.08e-07 (3.44e-07)	7.42e-02 (3.55e-01)	**4.29e-12 (6.70e-12)**
CMT5-T2	9.81e+01 (2.49e+00)	9.76e+01 (1.88e-01)	9.76e+01 (1.40e-01)	1.33e+02 (1.90e+02)	**9.74e+01 (6.48e-01)**
CMT6-T1	2.60e-07 (4.81e-07)	2.58e-07 (5.13e-07)	1.92e-07 (2.74e-07)	5.35e-07 (2.83e-06)	**1.79e-08 (5.53e-08)**
CMT6-T2	2.00e-03 (4.26e-03)	1.35e-03 (4.27e-03)	9.84e-04 (1.03e-03)	4.16e-04 (1.55e-03)	**6.60e-05 (3.52e-04)**
CMT7-T1	1.58e+04 (8.64e+03)	1.54e+04 (8.24e+03)	1.29e+04 (8.95e+03)	1.94e+04 (8.19e+03)	**1.13e+04 (7.75e+03)**
CMT7-T2	1.28e+02 (4.78e+00)	1.29e+02 (7.80e+00)	**1.28e+02 (4.74e+00)**	1.28e+02 (2.97e+00)	1.29e+02 (5.46e+00)
CMT8-T1	1.48e+01 (2.43e+00)	**1.39e+01 (2.87e+00)**	1.43e+01 (2.24e+00)	1.45e+01 (2.38e+00)	1.61e+01 (4.95e+00)
CMT8-T2	9.27e+01 (3.06e+00)	9.43e+01 (3.76e+00)	9.37e+01 (2.99e+00)	9.55e+01 (5.05e+00)	**9.19e+01 (3.00e+00)**
CMT9-T1	2.00e+04 (3.35e+04)	1.38e+04 (2.42e+04)	1.35e+04 (2.68e+04)	2.24e+04 (3.01e+04)	**1.94e+01 (6.62e+01)**
CMT9-T2	3.33e+04 (7.32e+01)	3.33e+04 (1.13e+02)	3.33e+04 (1.12e+02)	3.32e+04 (8.48e+01)	**3.32e+04 (4.33e+01)**
Ranking	49.8611	53.1111	50.6667	58.25	15.6111

consistently led to the conclusion that RL-CMTEA considerably outperforms the other 14 algorithms. The results of these tests conclusively indicate that RL-CMTEA substantially outperforms the other peer algorithms.

4.2.2. *Comparison in different dimensions*

To evaluate the performance of RL-CMTEA in different dimensions, experiments are conducted in both 50-dimension (50-D) and 150-dimension (150-D) settings. All other experimental conditions are consistently maintained to ensure fair comparisons. Table S-3 and Table S-4 in the supplementary file present the average fitness values and standard deviation of RL-CMTEA compared with other CMTEAs and CSTEAs in 50-D, respectively. Furthermore, Tables S-5 and Table S-6 in the supplementary file provide the average fitness values and standard deviation of RL-CMTEA compared with other comparative algorithms in 150-D, respectively.

As can be seen from the tables, the proposed algorithms rank first in both 50-D and 150-D. Specifically, in 50-D, compared with other peer CMTEAs, the proposed algorithm obtains 11 optimal results. Compared with A-CMFEA, RL-CMTEA shows remarkable advantages in 13 tasks but slightly inferior in 5 tasks. Compared with CSTEAs, nine optimal results are achieved. Compared with other CMTEAs and CSTEAs, the proposed algorithm obtains 11 and 13 optimal results in 150-D, respectively, showing similar performance.

Fig. S-1 to Fig. S-2, and Fig. S-3 to Fig. S-4 in the supplementary file show the convergence curves of RL-CMTEA and other compared CMTEAs algorithms in 50-D and 150-D, respectively. Observing the convergence curve, the proposed algorithm achieved the fastest convergence rate in most problems after approximately 10,000 and 30,000 evaluation times in 50-D and 150-D, respectively. This can be attributed to problems with lower dimensions exhibit relatively simple features, allowing RL to find the suitable operator for the task after learning from a small number of samples. However, as the dimensions of the problem increase, the task becomes more complex and requires more samples for learning.

The experimental results indicate that the proposed algorithm has obvious advantages in handling both low-dimension and high-dimension problems. This is particularly important for practical applications with complex constraints and different dimensional features.

4.3. *Parameter sensitivity analysis*

To evaluate the effectiveness of parameter settings in the different components of RL-CMTEA, 30 independent experiments on the CMT benchmark suite were conducted in this section under consistent conditions to evaluate the performance of each parameter in different settings.

4.3.1. *Sensitivity analysis of RL*

The learning rate α determines the step size taken by the algorithm to update the estimated Q-value during each update. In this section, while keeping other parameters constant, the value of α is changed to produce the variants RL-CMTEA-a1, RL-CMTEA-a2, RL-CMTEA-a3, and RL-CMTEA-a4, with α values of 0.1, 0.3, 0.5 and 0.7, respectively.

Table 3 presents the average fitness values and standard deviations obtained by RL-CMTEA and its variants in different α settings. For detailed insights into the convergence curves, please refer to Fig. S-5 and Fig. S-6 in the supplementary file. Notably, smaller learning rates often lead to improved results. A smaller α enables the algorithm to gradually converge to the optimal strategy. This might demand more operator attempts. However, in the long-term, this approach proves to be beneficial. The analysis of the convergence curves reveals that the variants with lower learning rates exhibit slower convergence in the early stages, but after a

Table 4
Average fitness values and standard deviations of RL-CMTEA-g1, RL-CMTEA-g2, RL-CMTEA-g3, RL-CMTEA-g4, and RL-CMTEA on CMT among 30 runs. The best results in each row are highlighted.

Problem	RL-CMTEA-g1	RL-CMTEA-g2	RL-CMTEA-g3	RL-CMTEA-g4	RL-CMTEA
CMT1-T1	3.28e-08 (1.45e-07)	1.56e-08 (8.55e-08)	6.91e-07 (3.77e-06)	1.35e-11 (4.40e-11)	**4.81e-17 (1.19e-16)**
CMT1-T2	1.70e-04 (7.88e-04)	7.03e-05 (3.85e-04)	3.16e-03 (1.72e-02)	6.58e-08 (2.13e-07)	**7.98e-14 (1.63e-13)**
CMT2-T1	8.57e-06 (2.69e-05)	3.93e-05 (2.13e-04)	6.45e-06 (2.75e-05)	3.77e-08 (1.04e-07)	**2.19e-09 (2.85e-09)**
CMT2-T2	9.57e-07 (4.26e-06)	5.61e-05 (3.07e-04)	9.59e-07 (5.03e-06)	1.45e-11 (5.18e-11)	**5.92e-17 (3.24e-16)**
CMT3-T1	2.42e+00 (4.72e+00)	7.99e-01 (2.85e+00)	7.88e-01 (2.49e+00)	7.79e-01 (2.81e+00)	**2.28e-04 (5.86e-04)**
CMT3-T2	6.86e+02 (1.52e+03)	3.52e+02 (1.38e+03)	2.99e+02 (1.15e+03)	2.57e+02 (1.24e+03)	**1.30e-03 (9.05e-05)**
CMT4-T1	9.38e+01 (6.58e+01)	8.99e+01 (5.82e+01)	9.42e+01 (5.99e+01)	1.15e+02 (7.25e+01)	**8.79e+01 (8.74e+01)**
CMT4-T2	8.20e+02 (1.72e+01)	8.18e+02 (1.81e+01)	8.15e+02 (1.78e+01)	8.21e+02 (2.08e+01)	**8.15e+02 (1.88e+01)**
CMT5-T1	1.86e-07 (3.82e-07)	2.02e-07 (4.23e-07)	1.02e-07 (1.53e-07)	1.06e-04 (5.80e-04)	**4.29e-12 (6.70e-12)**
CMT5-T2	**9.74e+01 (6.68e-01)**	9.74e+01 (2.70e-01)	9.75e+01 (1.97e-01)	9.75e+01 (1.69e-01)	9.74e+01 (6.48e-01)
CMT6-T1	2.25e-07 (2.38e-07)	1.57e-07 (1.06e-07)	2.68e-07 (7.42e-07)	3.73e-07 (1.49e-06)	**1.79e-08 (5.53e-08)**
CMT6-T2	3.76e-04 (1.94e-04)	3.17e-04 (2.19e-04)	3.51e-04 (3.13e-04)	1.19e-04 (2.21e-04)	**6.60e-05 (3.52e-04)**
CMT7-T1	1.56e+04 (8.78e+03)	1.25e+04 (8.82e+03)	1.49e+04 (9.07e+03)	1.49e+04 (7.96e+03)	**1.13e+04 (7.75e+03)**
CMT7-T2	**1.27e+02 (4.65e+00)**	1.29e+02 (4.86e+00)	1.28e+02 (4.63e+00)	1.28e+02 (4.90e+00)	1.29e+02 (5.46e+00)
CMT8-T1	**1.30e+01 (1.67e+00)**	1.35e+01 (1.93e+00)	1.40e+01 (1.82e+00)	1.44e+01 (1.94e+00)	1.61e+01 (4.95e+00)
CMT8-T2	9.28e+01 (2.90e+00)	9.28e+01 (2.23e+00)	9.22e+01 (2.70e+00)	9.28e+01 (3.36e+00)	**9.19e+01 (3.00e+00)**
CMT9-T1	1.06e+04 (1.90e+04)	1.26e+04 (2.63e+04)	4.01e+03 (1.12e+04)	4.05e+02 (1.07e+03)	**1.94e+01 (6.62e+01)**
CMT9-T2	3.32e+04 (3.99e+01)	3.32e+04 (4.33e+01)	3.32e+04 (4.44e+01)	**3.32e+04 (3.68e+01)**	3.32e+04 (4.33e+01)
Ranking	57.1667	55.6944	55.5556	40.4722	18.6111

certain number of attempts, they achieve better convergence performance. This also conforms to the assumptions made in establishing the Q-learning model.

The discount factor γ dictates the weight given to future rewards, and the appropriate selection depends on the specific problem. This section explores different variants while keeping other parameters constant ($\alpha = 0.01$), result in four variants: RL-CMTEA-g1 ($\gamma = 0.1$), RL-CMTEA-g2 ($\gamma = 0.3$), RL-CMTEA-g3 ($\gamma = 0.5$), and RL-CMTEA-g4 ($\gamma = 0.7$).

Table 4 presents the average fitness values and standard deviations obtained by RL-CMTEA and its variants in different γ settings. For a more detailed understanding of the convergence curves, please refer to Fig. S-7 and Fig. S-8 in the supplementary file. The experimental results indicate that when γ is close to 1, the agent prioritizes future rewards, highlighting the maximization of long-term cumulative rewards. Conversely, when γ approaches 0, the agent leans toward immediate rewards and short-term gains. Notably, focusing on the γ value for future rewards often leads to improved results when evaluations are sufficient.

4.3.2. Sensitivity analysis of candidate operators

In the context of differential mutation strategy, the scaling factor (F) plays a pivotal role in determining the scaling degree of the mutation vector. Maintaining constant experimental conditions and other parameters, this section sets the variants RL-CMTEA-F1, RL-CMTEA-F2, RL-CMTEA-F3, and RL-CMTEA-F4, with F of 0.1, 0.3, 0.7, and 0.9, respectively.

Table S-7 and Fig. S-9 to Fig. S-10 in the supplementary file present the experimental numerical results, and convergence curves of RL-CMTEA and its variants in different scaling factor settings, respectively. The experimental results indicate that RL-CMTEA achieved 5 best results on 18 questions whereas RL-CMTEA-F1 achieved the best results on 7 tasks. Compared with RL-CMTEA-F1, RL-CMTEA performed slightly worse on five problems but performed better on seven tasks, achieving similar results on six tasks. Overall, this paper has determined the scale factor to be 0.5.

The crossover rate (CR) is another pivotal parameter influencing the proportion of offspring generated through the mutation vector during crossover operations. Under the same experimental conditions and parameter settings, this section sets up the variants RL-CMTEA-CR1, RL-CMTEA-CR2, RL-CMTEA-CR3, and RL-CMTEA-CR4, with CR of 0.1, 0.3, 0.7, and 0.9, respectively.

Table S-8 and Fig. S-11 to Fig. S-12 in the supplementary file present the experimental numerical results, and the convergence curves in different crossover rate settings, respectively. The numerical results indicate that RL-CMTEA achieved the best performance on 10 tasks, whereas RL-CMTEA-CR1 and RL-CMTEA-CR4 performed poorly, with the other two values slightly inferior to RL-CMTEA. Compared with F, the value of CR has a smaller impact on algorithm performance. When the CR value is too high, although the diversity is good, it may affect the convergence rate, but when the CR is small, it will quickly reduce population diversity, which is not conducive to global optimization. Therefore, this paper determines the value of CR as 0.5.

The distribution index (η_c) in the SBX crossover operation influences the shape of the offspring distribution. Keeping experimental conditions and other parameter settings constant, this section sets variants RL-CMTEA-c1, RL-CMTEA-c2, RL-CMTEA-c3, and RL-CMTEA-c4, with η_c of 1, 1.3, 1.5, and 1.7, respectively.

Table S-9 and Fig. S-13 to Fig. S-14 in the supplementary file present the experimental numerical results and the convergence curves in different distribution index settings, respectively. RL-CMTEA achieved eight best results. More uniform changes often lead to better results and more stable performance in sufficient evaluations settings. Therefore, this paper identifies η_c as 2.

4.4. Ablation studies

To evaluate the effectiveness of each component of RL-CMTEA, this section designs three variants for detailed ablation studies:

Table 5
Average fitness values and standard deviations of RL-CMTEA and its different component variants on CMT among 30 runs. The best results in each row are highlighted.

Problem	RL-CMTEA-w/o-QL	RL-CMTEA-w/o-SD	RL-CMTEA-w/o-AP	RL-CMTEA
CMT1-T1	1.97e-06 (9.76e-06)	1.23e+00 (5.45e-02)	1.11e-01 (2.29e-01)	**4.81e-17 (1.19e-16)**
CMT1-T2	5.37e-03 (2.25e-02)	6.47e+02 (1.38e+02)	3.62e+01 (6.68e+01)	**7.98e-14 (1.63e-13)**
CMT2-T1	1.05e-06 (4.25e-06)	8.34e+00 (2.85e+00)	9.90e-01 (6.48e-01)	**2.19e-09 (2.85e-09)**
CMT2-T2	2.36e-08 (1.24e-07)	7.56e+02 (6.25e+02)	1.27e+01 (1.00e+01)	**5.92e-17 (3.24e-16)**
CMT3-T1	4.39e-01 (2.32e+00)	NaN (NaN)	1.95e-01 (8.25e-01)	**2.28e-04 (5.86e-04)**
CMT3-T2	1.59e+01 (8.68e+01)	1.56e+04 (1.54e+03)	2.49e+01 (1.30e+02)	**1.30e-03 (9.05e-05)**
CMT4-T1	**5.26e+01 (3.81e+01)**	6.54e+02 (1.18e+02)	1.62e+02 (3.90e+01)	8.79e+01 (8.74e+01)
CMT4-T2	8.11e+02 (1.49e+01)	1.08e+03 (1.82e+02)	**7.87e+02 (1.51e+01)**	8.15e+02 (1.88e+01)
CMT5-T1	1.26e-06 (6.87e-06)	6.80e+00 (8.69e-01)	1.15e+00 (1.07e+00)	**4.29e-12 (6.70e-12)**
CMT5-T2	9.76e+01 (1.86e-01)	3.97e+05 (3.87e+05)	6.44e+02 (1.00e+03)	**9.74e+01 (6.48e-01)**
CMT6-T1	**8.36e-09 (4.16e-08)**	1.20e+01 (1.72e+00)	1.06e+00 (9.49e-01)	1.79e-08 (5.53e-08)
CMT6-T2	3.36e-03 (1.20e-02)	4.01e+01 (4.71e+00)	4.91e+00 (4.70e+00)	**6.60e-05 (3.52e-04)**
CMT7-T1	1.19e+04 (6.69e+03)	5.22e+08 (3.03e+08)	3.03e+05 (1.11e+06)	**1.13e+04 (7.75e+03)**
CMT7-T2	**1.27e+02 (3.74e+00)**	6.28e+03 (2.18e+03)	1.78e+02 (4.81e+01)	1.29e+02 (5.46e+00)
CMT8-T1	1.34e+01 (1.71e+00)	NaN (NaN)	**1.11e+01 (1.83e-01)**	1.61e+01 (4.95e+00)
CMT8-T2	**9.18e+01 (3.26e+00)**	1.30e+02 (1.21e+01)	9.57e+01 (4.34e+00)	9.19e+01 (3.00e+00)
CMT9-T1	4.92e+03 (1.36e+04)	3.19e+04 (2.99e+04)	9.35e+04 (3.10e+04)	**1.94e+01 (6.62e+01)**
CMT9-T2	**3.32e+04 (4.34e+01)**	4.05e+04 (1.81e+03)	3.36e+04 (3.38e+02)	3.32e+04 (4.33e+01)
Ranking	27.0833	58.5556	44.6111	15.75

(1) RL-CMTEA-w/o-QL: In this variant, the operators are randomly selected to confirm the influence of the adaptive operator selection in RL-CMTEA.
(2) RL-CMTEA-w/o-SD: This variant adopts an elite-based knowledge transfer strategy, aiming to explore the superiority of the dimension-based knowledge transfer strategy in RL-CMTEA relative to the individual-based knowledge transfer strategy.
(3) RL-CMTEA-w/o-AP: This variant abstains from using auxiliary populations, allowing each task to use only one population driven by FP for independent evolution so as to assess the guiding impact of the auxiliary population on the main population.

Table 5 presents the statistical results of the average fitness values and standard deviation over 30 independent runs. Fig. S-15 and Fig. S-16 in the supplementary file present the convergence curves of RL-CMTEA and its three variants on these nine problems. The results highlight the notable superiority of RL-CMTEA over the other variants. The following analysis and conclusions can be drawn from the results:

(1) Comparison with RL-CMTEA-w/o-QL: Random selection of operators leads to a lack of guidance during the evolution process, whereas inappropriate operators may waste evaluation. Contrarily, the Q-learning-based operator selection strategy adapts the operators according to the performance of each of them, enhancing the robustness and overall performance of the algorithm.
(2) Comparison with RL-CMTEA-w/o-KT: Knowledge transfer based on similar dimensions demonstrates superior performance, particularly when the intersection of feasible regions is substantial. This observation highlights the suitability of dimension-based strategies over elite-based strategies, particularly in addressing intricate constraint problems. In scenarios where feasible areas considerably overlap, the strategy enables effective knowledge transfer between tasks with high similarity in dimensions. Conversely, when the inter-task similarity is low, the strategy mainly focuses on knowledge transfer within the same task, thereby avoiding the potential for negative knowledge transfer.
(3) Comparison with RL-CMTEA-w/o-AP: It can be seen from the convergence curve that RL-CMTEA with auxiliary population guidance exhibits a better convergence trend and traverses the infeasible region faster. This helps the main population quickly find feasible regions with good convergence and avoid consuming too many evaluation times in feasible regions containing local optimal solutions. This indicates that the auxiliary population plays a key role in improving the overall performance of the algorithm.

5. Conclusion

In this paper, a reinforcement learning assisted constrained multi-task evolutionary algorithm (RL-CMTEA) is proposed for constrained multi-task optimization. First, the adaptive operator selection strategy dynamically selects operators based on their performance in each task, thereby improving the adaptability and performance of the algorithm in different tasks. Second, the multi-population method uses independent evolution of auxiliary populations using different constraint handling techniques to maintain the diversity and feasibility of the main population. Finally, the dimension-based knowledge transfer strategy employs crossover operators to transfer knowledge between similar dimensions, promoting effective information exchange between tasks.

The experiments conducted on the CMT test suite clearly demonstrate that RL-CMTEA is significantly superior to other peer algorithms in solving CMTOPs. The analysis of fitness values and convergence curves under different parameter settings highlights the excellent performance of RL-CMTEA. Additionally, ablation studies further confirmed the effectiveness of its key components.

However, it is important to acknowledge that the proposed algorithm has certain limitations, which provide valuable insights for future research directions. Notably, the algorithm exhibits a tendency for knowledge transfer to predominantly occur within the same task under conditions of low similarity between tasks. While this approach mitigates negative transfer, it may inadvertently lead to the generation of an excessive number of offspring. In addition, challenges arise when the distance between feasible and unconstrained optima is considerable, impacting the effectiveness of the guidance of the auxiliary population and increasing the consumption of evaluation times. Tailoring knowledge transfer strategies and constraint handling techniques specifically adapted to CMTO are crucial for enhancing algorithm capabilities and effectively addressing the inherent challenges.

CRediT authorship contribution statement

Yufei Yang: Writing – original draft, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Changsheng Zhang:** Writing – review & editing, Supervision, Resources, Funding acquisition, Conceptualization. **Bin Zhang:** Resources, Data curation, Validation. **Jiaxu Ning:** Writing – review & editing, Investigation, Visualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This study received support from the Ningxia Natural Science Foundation Project (2023AAC03361).

Appendix A. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ins.2024.120863.

References

[1] Yanchi Li, Wenyin Gong, Shuijia Li, Multitasking optimization via an adaptive solver multitasking evolutionary framework, Inf. Sci. 630 (2023) 688–712.
[2] Jun Yao, Yandong Nie, Zihao Zhao, Xiaoming Xue, Yongfei Yang, Self-adaptive multifactorial evolutionary algorithm for multitasking production optimization, J. Pet. Sci. Eng. 205 (4) (2021) 108900.
[3] Abhishek Gupta, Lei Zhou, Yew-Soon Ong, Zefeng Chen, Yaqing Hou, Half a dozen real-world applications of evolutionary multitasking, and more, IEEE Comput. Intell. Mag. 17 (2) (2022) 49–66.
[4] Hossein Gitinavard, Mohsen Akbarpour Shirazi, Mohammad Zarandi, A possibilistic programming approach for biomass supply chain network design under hesitant fuzzy membership function estimation, Sci. Iran. 12 (2021).
[5] Heng Xiao, Gen Yokoya, Toshiharu Hatanaka, Multifactorial pso-fa hybrid algorithm for multiple car design benchmark, in: 2019 IEEE International Conference on Systems, Man and Cybernetics (SMC), IEEE, 2019, pp. 1926–1931.
[6] Yanchi Li, Wenyin Gong, Shuijia Li, Evolutionary constrained multi-task optimization: benchmark problems and preliminary results, in: Proceedings of the Genetic and Evolutionary Computation Conference Companion, 2022, pp. 443–446.
[7] Caixiao Xing, Wenyin Gong, Shuijia Li, Adaptive archive-based multifactorial evolutionary algorithm for constrained multitasking optimization, Appl. Soft Comput. 143 (2023) 110385.
[8] Loukia Avramelou, Paraskevi Nousi, Nikolaos Passalis, Anastasios Tefas, Deep reinforcement learning for financial trading using multi-modal features, Expert Syst. Appl. 238 (2024) 121849.
[9] Benjamin Smith, Anahita Khojandi, Rama K. Vasudevan, Bias in reinforcement learning: a review in healthcare applications, ACM Comput. Surv. 56 (2023) 1–17.
[10] Bingshui Da, Abhishek Gupta, Yew-Soon Ong, Curbing negative influences online for seamless transfer evolutionary optimization, IEEE Trans. Cybern. 49 (12) (2019) 4365–4378.
[11] Kavitesh Kumar Bali, Yew-Soon Ong, Abhishek Gupta, Puay Siew Tan, Multifactorial evolutionary algorithm with online transfer parameter estimation: MFEA-II, IEEE Trans. Evol. Comput. 24 (1) (2020) 69–83.
[12] Xu Zhiwei, Liu Xiaoming, Zhang Kai, He Juanjuan, Cultural transmission based multi-objective evolution strategy for evolutionary multitasking, Inf. Sci. 582 (2022) 215–242.
[13] Jing Tang, Yingke Chen, Zixuan Deng, Yanping Xiang, Colin Paul Joy, A group-based approach to improve multifactorial evolutionary algorithm, in: Twenty-Seventh International Joint Conference on Artificial Intelligence IJCAI-18, 2018.
[14] Ting Yee Lim, Choo Jun Tan, Wai Peng Wong, Chee Peng Lim, An information entropy-based evolutionary computation for multi-factorial optimization, Appl. Soft Comput. 114 (2022) 108071.
[15] Jian-Yu Li, Zhi-Hui Zhan, Kay Chen Tan, Jun Zhang, A meta-knowledge transfer-based differential evolution for multitask optimization, IEEE Trans. Evol. Comput. 26 (4) (2021) 719–734.
[16] Xianpeng Wang, Zhiming Dong, Lixin Tang, Qingfu Zhang, Multiobjective multitask optimization-neighborhood as a bridge for knowledge transfer, IEEE Trans. Evol. Comput. 27 (1) (2022) 155–169.
[17] Zhihua Cui, Ben Zhao, Tianhao Zhao, Xingjuan Cai, Jinjun Chen, Adaptive multi-task evolutionary algorithm based on knowledge reuse, Inf. Sci. 648 (2023) 119568.
[18] Sheng-Hao Wu, Zhi-Hui Zhan, Kay Chen Tan, Jun Zhang, Transferable adaptive differential evolution for many-task optimization, IEEE Trans. Cybern. (2023).
[19] Yanchi Li, Wenyin Gong, Multiobjective multitask optimization with multiple knowledge types and transfer adaptation, IEEE Trans. Evol. Comput. (2024).
[20] L. Feng, W. Zhou, L. Zhou, S.W. Jiang, J.H. Zhong, B.S. Da, Z.X. Zhu, Y. Wang, An empirical study of multifactorial PSO and multifactorial DE, in: 2017 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2017, pp. 921–928.

[21] Huynh Thi Thanh Binh, Nguyen Quoc Tuan, Doan Cao, Thanh Long, A multi-objective multi-factorial evolutionary algorithm with reference-point-based approach, in: 2019 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2019, pp. 2824–2831.
[22] Mei-Ying Cheng, Abhishek Gupta, Yew-Soon Ong, Zhi-Wei Ni, Coevolutionary multitasking for concurrent global optimization: with case studies in complex engineering design, Eng. Appl. Artif. Intell. 64 (2017) 13–24.
[23] Yongliang Chen, Jinghui Zhong, Mingkui Tan, A fast memetic multi-objective differential evolution for multi-tasking optimization, in: 2018 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2018, pp. 1–8.
[24] Hamid Ghaderi, Hossein Gitinavard, Mansour Mehralizadeh, An intuitionistic fuzzy DEA cross-efficiency methodology with an application to production group decision-making problems, J. Qual. Eng. Prod. Optim. 5 (2) (2020) 69–86.
[25] Hossein Gitinavard, Vahid Mohagheghi, Seyed Mousavi, Ahmad Makui, A new bi-stage interactive possibilistic programming model for perishable logistics distribution systems under uncertainty, Expert Syst. Appl. 238 (122121) (2023) 10.
[26] Jorge Ramírez, Wen Yu, Adolfo Perrusquía, Model-free reinforcement learning from expert demonstrations: a survey, Artif. Intell. Rev. (2022) 1–29.
[27] Boonhatai Kruekaew, Warangkhana Kimpan, Multi-objective task scheduling optimization for load balancing in cloud computing environment using hybrid artificial bee colony algorithm with reinforcement learning, IEEE Access 10 (2022) 17803–17818.
[28] Herbert Palm, Lorin Arndt, Reinforcement learning-based hybrid multi-objective optimization algorithm design, Information 14 (5) (2023) 299.
[29] Ye Tian, Xiaopeng Li, Haiping Ma, Xingyi Zhang, Kay Chen Tan, Yaochu Jin, Deep reinforcement learning based adaptive operator selection for evolutionary multi-objective optimization, IEEE Trans. Emerg. Top. Comput. Intel. (2022).
[30] Qian-Long Dang, Wei Xu, Yang-Fei Yuan, A dynamic resource allocation strategy with reinforcement learning for multimodal multi-objective optimization, Mach. Intel. Res. 19 (2) (2022) 138–152.
[31] Fei Zou, Gary G. Yen, Lixin Tang, Chunfeng Wang, A reinforcement learning approach for dynamic multi-objective optimization, Inf. Sci. 546 (2021) 815–834.
[32] Fei Ming, Wenyin Gong, Liang Gao, Adaptive auxiliary task selection for multitasking-assisted constrained multi-objective optimization [feature], IEEE Comput. Intell. Mag. 18 (2) (2023) 18–30.
[33] G.M.C. Leite, S. Jiménez-Fernández, S. Salcedo-Sanz, C.G. Marcelino, C.E. Pedreira, Solving an energy resource management problem with a novel multi-objective evolutionary reinforcement learning method, Knowl.-Based Syst. 280 (2023) 111027.
[34] Jian-Jiao Ji, Yi-Nan Guo, Xiao-Zhi Gao, Dun-Wei Gong, Ya-Peng Wang, Q-learning-based hyperheuristic evolutionary algorithm for dynamic task allocation of crowdsensing, IEEE Trans. Cybern. 53 (4) (2021) 2211–2224.
[35] Zi-Qi Zhang, Fang-Chun Wu, Bin Qian, Rong Hu, Ling Wang, Huai-Ping Jin, A Q-learning-based hyper-heuristic evolutionary algorithm for the distributed flexible job-shop scheduling problem with crane transportation, Expert Syst. Appl. 234 (2023) 121050.
[36] Rammohan Mallipeddi, Ponnuthurai N. Suganthan, Ensemble of constraint handling techniques, IEEE Trans. Evol. Comput. 14 (4) (2010) 561–579.
[37] John H. Holland, Genetic algorithms, Sci. Am. 267 (1) (1992) 66–73.
[38] Kenneth V. Price, Differential evolution, in: Handbook of Optimization: From Classical to Modern Approach, Springer, 2013, pp. 187–214.
[39] Abhishek Gupta, Yew-Soon Ong, Liang Feng, Multifactorial evolution: toward evolutionary multitasking, IEEE Trans. Evol. Comput. 20 (3) (2015) 343–357.
[40] Yi Jiang, Zhi-Hui Zhan, Kay Chen Tan, Jun Zhang, Block-level knowledge transfer for evolutionary multitask optimization, IEEE Trans. Cybern. (2023).
[41] Bing-Chuan Wang, Han-Xiong Li, Jia-Peng Li, Yong Wang, Composite differential evolution for constrained evolutionary optimization, IEEE Trans. Syst. Man Cybern. Syst. 49 (7) (2018) 1482–1495.
[42] Aleš Zamuda, Adaptive constraint handling and success history differential evolution for cec 2017 constrained real-parameter optimization, in: 2017 IEEE Congress on Evolutionary Computation (CEC), IEEE, 2017, pp. 2443–2450.
[43] Yong Wang, Jia-Peng Li, Xihui Xue, Bing-Chuan Wang, Utilizing the correlation between constraints and objective function for constrained evolutionary optimization, IEEE Trans. Evol. Comput. 24 (1) (2019) 29–43.
[44] Bing-Chuan Wang, Han-Xiong Li, Qingfu Zhang, Yong Wang, Decomposition-based multiobjective optimization for constrained evolutionary optimization, IEEE Trans. Syst. Man Cybern. Syst. 51 (1) (2018) 574–587.
[45] Efrén Mezura-Montes, C.A. Coello Coello, Jesús Velázquez-Reyes, Lucıa Munoz-Dávila, Multiple trial vectors in differential evolution for engineering design, Eng. Optim. 39 (5) (2007) 567–589.
[46] Guohua Wu, Xupeng Wen, Ling Wang, Witold Pedrycz, Ponnuthurai Nagaratnam Suganthan, A voting-mechanism-based ensemble framework for constraint handling techniques, IEEE Trans. Evol. Comput. 26 (4) (2022) 646–660.
[47] Radka Poláková, L-SHADE with competing strategies applied to constrained optimization, in: 2017 IEEE Congress on Evolutionary Computation (CEC), 2017, pp. 1683–1689.
[48] Yanchi Li, Wenyin Gong, Fei Ming, Tingyu Zhang, Shuijia Li, Qiong Gu, MToP: a MATLAB optimization platform for evolutionary multitasking, arXiv preprint, arXiv:2312.08134, 2023.
[49] Joaquín Derrac, Salvador García, Daniel Molina, Francisco Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (1) (2011) 3–18.