HYBRID MULTI-OBJECTIVE REINFORCEMENT LEARNING ALGORITHM FOR HYBRID FLOW SHOP WITH LIMITED WORKERS AND TIME-OF-USE ELECTRICITY TARIFFS

Anonymous authors
Paper under double-blind review

ABSTRACT

With the increasing requirements of lower carbon emission, energy-related objectives have gained increasing attention in both industrial and management decision makers. This study proposed a Q-learning-driven hybrid reinforcement learning algorithm to solve the hybrid flow shop scheduling problem, wherein limited resources, worker energy consumption, and time-of-use (TOU) electricity tariffs are considered simultaneously. Three objectives are to be minimized, including makespan, total energy consumption, and TOU electricity pricing. First, a mathematical model based on the mixed integer linear programming (MILP) is formulated. A dynamic decoding approach is then developed to choose suitable machines and workers. Next, six variations of Nawaz–Enscore–Ham heuristics are proposed to enhance the initialization approach with diversity capabilities. Considering the multi-objective handling method, an adaptive-space-based crowding distance heuristic is embedded to select solutions resulted in higher global searching abilities. Additionally, two lemmas considering machine turn off and worker rest are designed to improve certain objectives. Furthermore, a Q-learning driven hyper-heuristic combining with VNS and IG is developed to balance the exploration and exploitation abilities. Finally, detailed experimental comparisons with state-of-the-art algorithms are tested to verify the efficiency and effectiveness of the proposed algorithm.

1 INTRODUCTION

As one of the most widely used scheduling problems (1), the hybrid flow shop scheduling (HFS) problem has gained increasing attention. Meanwhile, numerous realistic constraints and objectives have been considered in HFSs to make the problem close to the reality, such as limited human resources (2), low-carbon emission or total energy consumption (TEC) (3), and time-of-use (TOU) electricity tariff (4).

During recent years, plenty of methods have been developed to solve different versions of HFSs, including constraint programming (CP), multi-objective evolutionary algorithm (MOEA), ant colony optimization (ACO), differential evolution (DE), variable neighborhood search (VNS), iterated greedy (IG), reinforcement learning (RL) and deep RL (DRL). Table 1 briefly reports the related works of different types of HFS problems and solving methods. Observing from the related studies about the HFS with limited resource, TOU electricity pricing, multi-objective optimization methods, and the reinforcement learning based methods, it is obviously that there is an urging requirement to consider all these constraints, objectives, and optimization methods together. Therefore, this is the motivation of this study.

2 PROBLEM DESCRIPTION

In the classical HFS, there are generally n jobs to be processed through ϑ stages, and there are M_k parallel machines at stage $k = 1, \ldots, \vartheta$. The routes of all jobs are the same, i.e., from the first stage

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: Literature review

Constraints	Objectives	Methods	Ref.
HFS+resource/assemble	total cost	CP	(5)
HFS+resource	makespan/TEC	Q-learning MOEA	(6)
HFS+worker fatigue	makespan	Agent-based	(7)
HFS+TOU	makespan/TEC	multi-objective ACO	(8)
HFS+distributed+setup	makespan	Imitation learning	(9)
HFS+lot streaming	makespan/TEC/tardiness	Q-learning/hyper-heuristic/multi-objective	(10)
HFS	makespan/TEC	Q-learning PSO	(11)
HFS+lot streaming	makespan	Q-learning heuristic	(12)
HFS+fuzzy+block	makespan/TEC	Q-learning MOEA	(13)
HFS+distributed	TEC/tardiness	dueling DRL	(14)
HFS+resource	TEC/cost	Q-learning+IG+VNS	(15)
HFS+deteriorating	makespan/tardiness	bi-population method	(16)
HFS+distributed	makespan/TEC	knowledge MOEA	(17)
HFS+worker+TOU	makespan/TEC/electricity cost	knowledge based Q-learning+IG+VNS	our

to the last one. At each stage k, a unique machine should be selected from the M_k parallel machines to process each operation. Therefore, compared with the classical flow shop scheduling problem, there is one more task for machine assignment in HFS.

The considered problem is a special case of the classical HFS with limited workers (hereinafter referred to as HFS-LW). To make the problem close to the reality, the following assumptions are considered: (a) Each job should select exactly one machine from the set of parallel machines at each stage, while one worker should be chosen to process the operation. (b) Each machine can be occupied with one job, and one worker can do one operation at a time. (c) An overlap between two consecutive operations of the same job, an overlap between two operations processing on the same machine, and an overlap between two consecutive operations on the same worker are not permitted. (d) Each operation can be started its processing after the available of the selected machine and worker. (e) Each operation should not be interrupted by other operations until its completion. (f) Different levels of speed on each machine are considered. (g) TOU electricity price is considered. (h) Different type of energy consumption of each worker is considered as part of the objectives.

2.1 OBJECTIVES

The detailed mathematical model can be found in Appendix. Given the indices as follows: n: Total number of jobs; ϑ: Total number of stages; j, j_2: Indices of job; $C_{j,k}$: Completion time of the kth task of job j; ξ_{wp}: Energy consumed during the processing operation of workers; ξ_{pt}: Energy consumed during the processing operations; ξ_{idle}: Energy consumed during the idle time of machine and workers; and ρ_e: Total electricity prices, this study considers the minimization of the following three objectives, i.e., makespan, total energy consumption (TEC), and total electricity price (TEP), using Eq. (3), respectively.

$$f_1 = \max\{C_{j,\vartheta}\}, \quad \forall j \tag{1}$$

$$f_2 = \xi_{wp} + \xi_{pt} + \xi_{idle} \tag{2}$$

$$f_3 = \rho_e \tag{3}$$

3 METHODOLOGY

The framework of the Q-learning-driven multi-population collaborative algorithm is given in Algorithm 1. It can be seen from Lines 16-22 in Algorithm 1 that, the Q-learning mechanism is invoked during the VNS and IG procedure, wherein the selection of the operators for VNS and the destruction length for IGD are learned and provided by the Q-learning approach. The framework and flowchart of the proposed algorithm can be found in Fig. 1.

3.1 ENCODING AND DECODING METHOD

This study introduces a simple and naïve encoding method. Each solution is represented by a one-dimensional vector which discloses the job sequence in the first stage.

Considering the decoding method, the following tasks should be completed: (1) assigning an available machine for each operation; (2) deciding the job sequence in the following stages; and (3) selecting a suitable worker for each operation. Unlike the commonly used first-available-machine

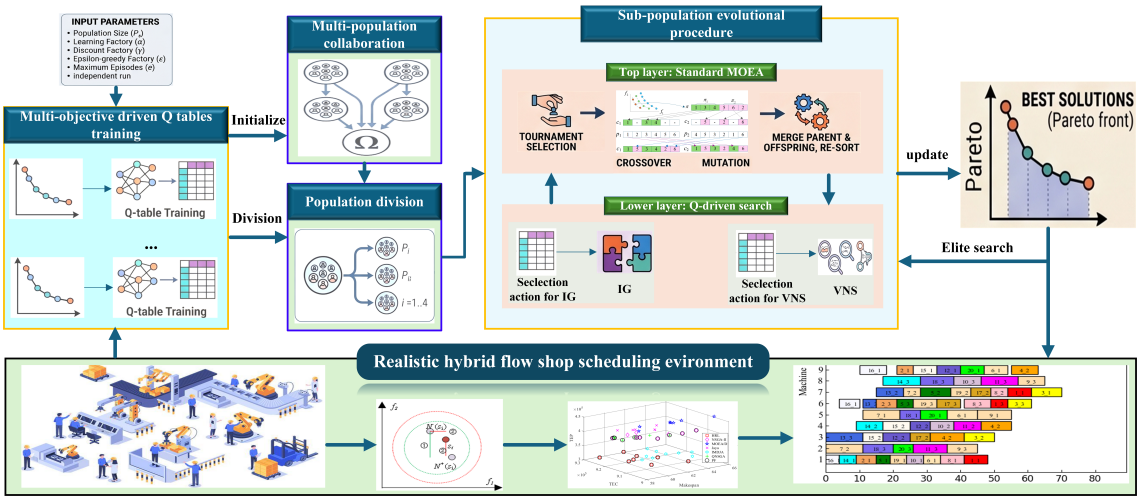

Figure 1: Flowchart of the proposed algorithm

(FAM) rule, this study used the ECT rule with the following main reasons: (1) ECT selects the machine with the earliest completion time for the candidate operation; and (2) considering different processing capabilities of the parallel machines, the suitable machine chosen by the FAM rule may have longer processing time and later completion time. If more than one machine satisfies, then one machine is randomly selected.

For the second task, all the jobs are sorted according to a non-decreasing order of their completion time in the previous stage. Algorithm 2 is used to choose the most suitable worker for each operation.

3.2 Adaptive-space-based crowding distance heuristic

The crowding-distance method in the classical NSGA-II is too simple to consider space features of the neighboring solution. To address this issue, a multi-population-space-based crowding distance (MSCD) method is designed with the following steps:

$$d_1 = \left(\frac{\left| \overrightarrow{N^-(s_i)N^+(s_i)} \right|}{2} \right)^m \tag{4}$$

$$d_2 = \left(\frac{\left| \overrightarrow{s_i N^+(s_i)} \right| + \left| \left| \overrightarrow{s_i N^-(s_i)} \right| \right|}{2} \right)^m - d_1 \tag{5}$$

$$d_k = \lambda d_1 + (1-\lambda) d_2 \tag{6}$$

where $|\overrightarrow{p_1 p_2}|$ is the distance between the two points p_1 and p_2. Different types of distances are used to construct an adaptive heuristic, which includes the following types of distances: (a) Euclidean distance: $d = \sqrt{\sum_{k=1}^{m} (x_{1k} - x_{2k})^2}$; (b) Standardized Euclidean distance: $d = \sqrt{\sum_{k=1}^{m} \left(\frac{x_{1k} - x_{2k}}{s_k} \right)^2}$; (c) Chebyshev distance: $d = \max_i (|x_{1i} - x_{2i}|)$; (d) Cosine distance: $d = \frac{\sum_{k=1}^{m} x_{1k} x_{2k}}{\sqrt{\sum_{k=1}^{m} x_{1k}^2} \sqrt{\sum_{k=1}^{m} x_{2k}^2}}$.

are the two fitness values in the kth objective of the two neighboring solutions before and after s_i. It should be noted that all the solutions must be sorted according to the kth objective, and the neighboring solutions $f_k^{N^-(s_i)}$ and $f_k^{N^+(s_i)}$ of s_i are different for each objective. If the solution s_i is at the first or last position considering the kth objective, then its corresponding distance for it is set to ∞ . The distance between $f_k^{N^-(s_i)}$ and $f_k^{N^+(s_i)}$ is calculated by using the Euclidean method. However, the spatial information will not be considered by using this method. Additionally, the Euclidean method is not sufficient to obtain the features of all the solutions.

3.3 Machine turn off and worker rest heuristic

A machine on standby that is not shut down or turned off will consume a certain number of standby energies. In addition, a worker that stands in a longer period without any operation will get tired con-

Algorithm 1: Framework of the proposed HRL algorithm

Input: parameters including: population size P_s, learning factory α, discount factory γ, Epsilon-greedy factory ε, and maximum episodes e
Output: the Pareto set of solutions

```
Perform the Q tables training heuristic (c.f. Algorithm 3);
% Perform evolutionary process using Q tables;
Divide the whole population into eight sub-populations in a random way;
foreach independent run do
    Perform the initialization heuristic to generate the whole population P;
    Divide P into four sub-populations P_i (i = 1, 2, 3, 4) in a random way;
    Let Ω = ∅;
    for i = 1 to 4 do
        Calculate the fitness values for each individual in P_i;
        Perform the non-dominated sorting function on P_i and obtain the Pareto level front f_i
          and rank r_i;
        for j = 1 to g do
            Perform tournament selection function based on f_i and r_i;
            Randomly select a crossover operator and apply it to two randomly-selected
              solutions;
            Randomly select a mutation operator for each individual in the current population to
              generate the offspring solutions;
            Combine the parent and offspring populations and perform the non-dominated
              sorting function to obtain the next parent population;
            % Perform the Q-table driven searching function;
            Choose two actions a_1 and a_2 for the current population based on Q_1^i, Q_2^i,
              i = 1, 2, 3, 4;
            Apply the destruction_construction heuristic based on a_1;
            Apply the VNS heuristic based on a_2 ;
            Calculate the weighted objective value based on the ith weighted;
            Update the current individuals with the neighboring solutions;
            Update the best solution and insert it to the Pareto front;
        end
        % Multi-population collaboration heuristic;
        Merge the current sub-population P_i into Ω: Ω = Ω ∪ P_i;
    end
    Divide the whole population Ω into four sub-populations P_i (i = 1, 2, 3, 4) in a random
      way;
end
```

suming more waiting energy will be consumed. Therefore, two heuristics are proposed to decrease the energy consumption.

(1) Machine turn off heuristic

Lemma 1: For any idle duration t_d on any machine i, if the following condition is satisfied, the machine should be turned off:

$$(SM_{i,k,r+1} - EM_{i,k,r}) \times Us_i > Et_i^{on} + Et_i^{off}, \forall k, i, r \in 2, ..n \tag{7}$$

Proof: Given the two consecutive processing positions r and r+1, the starting time of the following position is $SM_{i,k,r+1}$, while the completion time of the previous position is $EM_{i,k,r}$. Therefore, the idle time of this duration is $SM_{i,k,r+1} - EM_{i,k,r}$, and the standby energy consumption is $(SM_{i,k,r+1} - EM_{i,k,r}) \times Us_i$. Without affecting the makespan, if the current machine is turned off, the addition energy consumption is $Et_i^{on} + Et_i^{off}$, and the standby energy consumption $(SM_{i,k,r+1} - EM_{i,k,r}) \times Us_i$ can be avoided. Therefore, if the condition $(SM_{i,k,r+1} - EM_{i,k,r}) \times Us_i > Et_i^{on} + Et_i^{off}, \forall k, i, r \in 2, ..n$, the machine can be turned off to decrease the total energy without affecting the other objectives.

(2) Worker rest heuristic

Algorithm 2: Decoding heuristic

Input: an encoding representation
Output: a feasible solution

for $k = 1$ **to** ϑ **do**
  for $j = 1$ **to** n **do**
    Let EM_i and EW_w be the completion time of the last operation on machine i and worker w, respectively;
    Select the machine i^* according to ECT rule:
    $i^* = \arg\min \left\{ \mu_i = EM_i + \left(\frac{pt_{j,k,w}}{v_{j,k}} \right),\ i \in \{1, \ldots, m\} \right\}$
    Get the possible starting time:
    $S^*_{j,k} = \max\{C_{j,k-1}, EM_{i^*}\}$
    Record the workers with the available time earlier than $S^*_{j,k}$ in a set S_w:
    $S_w = \left\{ w \mid EW_w - S^*_{j,k} < 0,\ w \in \{1, \ldots, W\} \right\}$
    if $|S_w| > 0$ **then**
      $w^* = \arg\max \left\{ EW_w - S^*_{j,k} \mid w \in S_w \right\}$
    else
      $w^* = \arg\min \left\{ EW_w - S^*_{j,k} \mid w \in \{EW_w - S^*_{j,k} > 0\} \right\}$
    end
    Update the starting and completion time of $O_{j,k}$:
    $S_{j,k} = \max\{S^*_{j,k}, EW_{w^*}\}, \qquad C_{j,k} = S_{j,k} + pt_{j,k,w^*}$
    Update the available time of the selected machine and worker:
    $EM_{i^*} = EW_{w^*} = C_{j,k}$
  end
end
Calculate the three objectives according to Equations (1)–(3)

Lemma 2: For any idle duration on any worker w, if the following condition satisfied, the worker should be allowed to reset:

$$(S_{j2,k2} - C_{j,k}) \times UEs_w > EWr_w, if \ \ U_{j2,k2,w,p+1} = U_{j,k,w,p} = 1, \quad \forall w, j, k, j2, k2, p \in 1, nm - 1 \tag{8}$$

Proof: Given the two consecutive processing positions p and p+1, the starting time of the following position is $S_{j2,k2}$, while the completion time of the previous position is $C_{j,k}$ and $U_{j2,k2,w,p+1} = U_{j,k,w,p} = 1$ means that $O_{j,k}$ and $O_{j2,k2}$ are processed by the worker consecutively. Therefore, the if the worker is standing from $C_{j,k}$ to $S_{j2,k2}$, then the standby energy consumption is $(S_{j2,k2} - C_{j,k}) \times UEs_w$. Without affecting the makespan, the worker is allowed to rest, the addition energy consumption is EWr_w , and the standby energy consumption $(S_{j2,k2} - C_{j,k}) \times UEs_w$ can be avoided. Therefore, if the condition $(S_{j2,k2} - C_{j,k}) \times UEs_w > EWr_w$, the worker can be allowed to rest to decrease the total energy without affecting the other objectives.

3.4 Q-LEARNING-DRIVEN MULTI-POPULATION COLLABORATIVE ALGORITHM

To adapt to solving the three objectives efficiently, four populations with different Q-learning components are designed, wherein the first three components are designed for each objective, respectively, while the last Q-learning component is used for the multi-objective one. All the three Q-learning components possess the same structures, and the only difference is the use of a different objective to calculate the reward value. The collaborative approach among the four population is performed using the crossover operators.

(1) State and actions

Two states are embedded wherein state S_1 represents the shaking procedure in VNS, and state S_2 contains all neighboring structures, i.e., NI-I to NI-VII (c.f. Appendix).

(2) Reward function

Let x_t be the current solution at step t, and x_{t+1} be the neighboring solution generated after applying the chosen action. Let $f^i_{\min}, i = 1, 2, 3$ be the best value of the ith objective. Considering the four Q-learning components, different reward calculations are used as follows: (a) Considering the ith objective, the reward value is calculated: $r^i_t = \frac{f^i_{\min}}{f^i_{x_t}} \left(\frac{f^i_{x_t} - f^i_{x_{t+1}}}{f^i_{\max} - f^i_{\min}} \right), \forall i = 1, 2, 3$; (b) Considering the fourth Q-learning components, the reward value is calculated as: $r^4_t = \sum_{i=1}^{3} \frac{f^i_{\min}}{f^i_{x_t}} \left(\frac{f^i_{x_t} - f^i_{x_{t+1}}}{f^i_{\max} - f^i_{\min}} \right)$;

(3) Update of the Q-table

The updating value of $Q(s, a)$ is given as follows:

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha \begin{bmatrix} r^i_t + \\ \gamma \max_{a'} Q(s_{t+1}, a_{t+1}) \\ -Q(s_t, a_t) \end{bmatrix}, i \in 1, 2, 3, 4 \tag{9}$$

where s_t and s_{t+1} are the two states at step t and t+1, while a_t and a_{t+1} are the two actions at step t and t+1; r^i_t is the reward value obtained for the ith Q-learning component; γ is the discount factor that is used for the future reward $\max_{a'} Q(s_{t+1}, a_{t+1})$, and α is the learning rate to accept the new state information.

4 Experimental Comparisons and Analysis

This section illustrates the detailed experimental comparisons, which include the experimental settings to set the testing instances and choose the comparison algorithms, the tuning of parameters to decide the most suitable parameters for the proposed algorithm, the ablation experiments to test the efficiency of each component, and the experimental analysis compared with the state-of-the-art efficient algorithms.

4.1 Experiment settings

The testing instances are modified based on the benchmark instances from the Ref (15). Each instance is marked with three numbers, including the number of jobs, number of stages, and number of machines at each stage. The detailed experiment settings are described as follows.

(1) The TOU electricity function is shown as follows:

$$TOU(t) = \begin{cases} 250, & 0 \le t\%24 < 8 \\ 1000, & 8 \le t\%24 < 11 \\ 600, & 11 \le t\%24 < 17 \\ 1000, & 17 \le t\%24 < 22 \\ 600, & 22 \le t\%24 < 24 \end{cases} \quad (\mathrm{CNY/MWh}) \tag{10}$$

(2) The details of other experiment settings are presented in Table 2.

Table 2: Experimental settings

Factors	Levels	Factors	Levels
n	10, 20, 50	$h_{j,k,1}$	0.2
ϑ	3, 5	$h_{j,k,2}$	1.6
m	3, 4, 6	W_w	U[50, 100](kg)
n	10, 20, 50	$h_{j,k,1}$	0.2
$pt_{j,k,w}$	U[5, 10](hour)	V_w	2
Up_i	U[4, 15](10^5W)	UEs_w	0.2
Us_i	2	EWr_w	4
Et^{on}_i	4	Lw_j	U[50, 100](kg)
Et^{off}_i	4	$v_{j,k}$	0.8, 1.0, 1.2

4.2 Ablation experiments

To test the performance of different components embedded in the proposed framework, we performed detailed ablation experiments as follows: (a) for the proposed adaptive-space-based crowding distance heuristic two algorithms are coded, i.e., HRL_NCR algorithm which contains all the

components of the proposed HRL algorithm with the crowding distance heuristic in NSGA-II; (b) for dynamic worker selection mechanism comparisons, two comparison algorithms, including HRL_NW with a randomly selection of workers and the HRL algorithm with the dynamic worker selection heuristic; (c) for multi-population Q-learning components, two algorithms are coded, including the HRL_NQ algorithm without the proposed multi-population Q-learning method, and HRL with all the components; and (d) for the initialization heuristic, we implemented two comparison algorithms, i.e., the HRL_NI algorithm with a random initialization method.

The detailed comparison results are listed in Appendix. To verify the statistical significance, an analysis of variance (ANOVA) is conducted between all the two pair of algorithms for the 12 relatively large-scale instances. The ANOVA comparison results are shown from Fig. 2 (a)-(d), wherein the obtained p-values are less than 0.05. Therefore, the efficiency of the proposed heuristic can be verified.

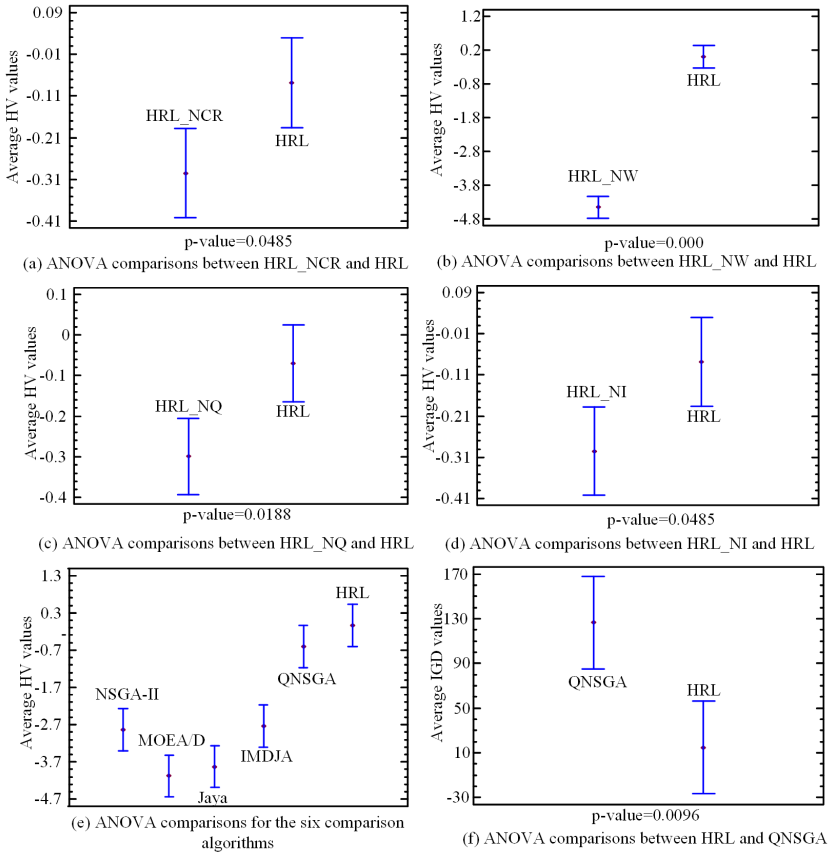

Figure 2: ANOVA comparison results

4.3 Comparisons with the state-of-the-art algorithms

Five state-of-the-art algorithms, i.e., NSGA-II (19), modified MOEA/D (20), Jaya (21), IMDJA (22), and QNSGA (15), are selected to make detailed comparisons with the proposed HRL algorithm. These algorithms were selected due to the following reasons: (1) NSGA-II is a widely used multi-objective optimization algorithms which has been verified to be efficient for solving different types of multi-objective optimization problems; (2) the modified MOEA/D and the Jaya algorithms are designed for a distributed flow shop considering minimized different types of objectives, which can be easily adjusted to solve the considered problem in this study; (3) the IMDJA algorithm is designed for solving job shop scheduling problems with considering limited workers and ergonomic factors with a similar worker energy consumption as this study. Therefore, the IMDJA algorithm can also be modified to solve the considered problem in this study easily; and (4) the QNSGA algorithm is designed to solve multi-objective HFS problems without considering the worker ergonomic factors

and TOU electronic tariffs. Therefore, detailed comparisons can be made with QNSGA easily and fairly.

To make fair comparisons with the five state-of-the-art algorithms and the proposed HRL algorithm, the following settings are given: (1) to adapt to solving the considered problem, all the comparison algorithms are set the same stopping criterion, i.e., the CPU time is set as seconds for each instance, where n and m represent the number of jobs and machines, respectively; (2) to adapt to the considered problem, all the comparison algorithms are embedded the required components, such as the encoding/decoding methods. All the other components of the comparison algorithms listed in their references have been used directly if they can adapt to solving the considered problem; (3) after performing 30 independent runs for each instance, all the non-dominated solutions obtained by each comparison algorithm are collected to construct the final Pareto front to calculate the HV and IGD indicators.

Table 3 presents the comparison results of the HV indicators for the six comparison algorithms. It can be observed from Table 3 that: (1) the proposed HRL algorithm obtained 14 better solutions, while the second-best algorithm QNSGA obtained 4 better values. The other four comparison algorithms obtained no better values. Meanwhile, considering the four types of instances, the proposed HRL algorithm obtained slightly worse results compared with those of the QNSGA algorithm; (2) considering the average RPI values shown in the last line, the proposed HRL algorithm obtained a value of -0.0375, which is about 20 times higher than the result of QNSGA. It is worth noting that, for the HV indicator, a higher RPI value means a better result; and (3) in a nutshell, the proposed HRL algorithm exhibits an efficient performance in convergence, diversity and robustness compared with the other five comparison algorithms.

Fig. 2(e) shows the ANOVA comparisons for the six algorithms considering the HV indicator. It can be observed from Fig. 2(e) that the first two algorithms, i.e., QNSGA and HRL algorithms are significantly better than the other four algorithms. To further verify the significant difference between the QNSGA and HRL algorithms, an ANOVA comparison is also performed considering the IGD indicator in Fig. 2(f), showing that the HRL algorithm is significantly competitive than the QNSGA algorithm.

Table 3: Comparison results with NSGA-II, MOEA/D, Jaya, IMDJA, QNSGA, and HRL

Inst	Best	HV values					
		NSGA-II	MOEA/D	Jaya	IMDJA	QNSGA	HRL
10-3-3	0.5589	0.5483	0.5422	0.5434	0.5513	0.5567	0.5589+
10-3-4	0.5608	0.5463	0.5321	0.5382	0.5497	0.5484	0.5608+
10-3-6	0.5785	0.5502	0.5406	0.5489	0.5527	0.5687	0.5785+
10-5-3	0.5570	0.5308	0.5257	0.5277	0.5314	0.5531	0.5570+
10-5-4	0.5724	0.5323	0.5276	0.5308	0.5324	0.5707	0.5724+
10-5-6	0.5974	0.5519	0.5488	0.5488	0.5559	0.5937	0.5974+
20-3-3	0.5379	0.5338	0.5306	0.5265	0.5365	0.5343	0.5379+
20-3-4	0.5499	0.5407	0.5314	0.5353	0.5419	0.5499+	0.5471
20-3-6	0.5627	0.5472	0.5378	0.5372	0.5493	0.5574	0.5627+
20-5-3	0.5512	0.5378	0.5308	0.5274	0.5367	0.5512+	0.5511
20-5-4	0.5599	0.5439	0.5345	0.5349	0.5435	0.5599+	0.5598
20-5-6	0.5682	0.5419	0.5321	0.5342	0.5406	0.563	0.5682+
50-3-3	0.5316	0.5297	0.5253	0.5256	0.5293	0.5276	0.5316+
50-3-4	0.5367	0.5335	0.5254	0.527	0.5308	0.5336	0.5367+
50-3-6	0.5382	0.5325	0.5274	0.5307	0.5335	0.5356	0.5382+
50-5-3	0.5386	0.53	0.5256	0.5247	0.5283	0.5368	0.5386+
50-5-4	0.5440	0.5336	0.5279	0.5298	0.5326	0.5424	0.5440+
50-5-6	0.5419	0.5331	0.5272	0.5274	0.5308	0.5419+	0.5412
Avg	0.5548	0.5388	0.5318	0.5333	0.5393	0.5514	0.5546+

- "+" means the better values

4.4 Comparisons of realistic instances with generalization

A realistic instance from a glass-ceramics production system has been tested in Ref. (23), where there are five parallel machines at Stage 1 and a furnace at Stage 2. The TOU energy price machine

processing and standby energy consumption, and processing time information are given in Ref. (23). Based on the detailed information of the realistic case, we extended it to generate 10 different-scale instances and also with the detailed worker information. All these instances can be downloaded from the website: https://github.com/DATAdriving/ . The two recently-published DRL-driven multi-objective optimization algorithms, including MMEAPSL (24) and DRLOSEMCMO (25), are used to make detailed comparisons. In addition, to test the generalization abilities of the Q-learning methods, we directly used the training Q-tables for the instance "50-5-6" in Section V-J. Fig. 3. ANOVA comparison results

The comparison results of the three compared algorithms are reported in Appendix. The ANOVA comparisons shown in Fig. 3 further verify the significant competitive performance of the proposed algorithm.

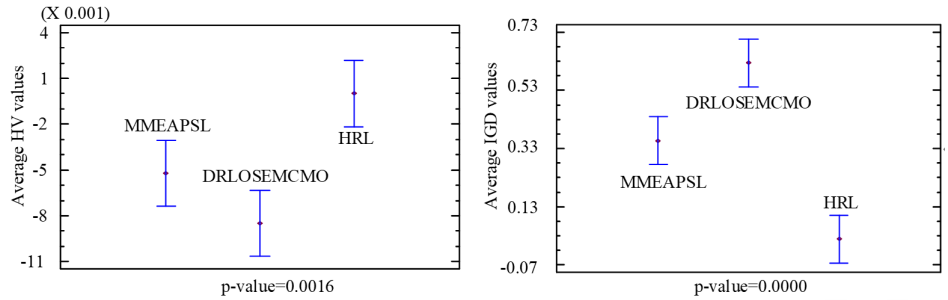

Figure 3: ANOVA comparisons for the 10 different-scale realistic instances

4.5 Experimental analysis

From the experimental comparisons, we can conclude that the proposed algorithm is efficient for solving different types of HFS problems with limited workers and time-of-use electricity tariffs. The main advantages of the proposed algorithm are as follows: (1) six extended versions of NEH heuristics are embedded to generate initialization population with higher performance; (2) the adaptive crowding distance method is embedded to enhance the space perception abilities; (3) a multi-population collaboration Q-learning training and evolving mechanism is used to enhance the diversity and convergence abilities simultaneously; (4) a dynamic decoding heuristic is utilized to further improve the resulted solution; and (5) two problem-specific lemmas are embedded to enhance the capabilities for optimizing the considered problem.

It can be observed from the structure of the proposed algorithm that, by using the extended initialization heuristics, multi-population collaboration method, and the dynamic decoding heuristic, the searching abilities of the proposed algorithm can be enhanced. After applying the adaptive crowding distance method and the problem-specific lemmas, the population diversity can be improved. The main limitations of the proposed algorithm are as follows: (1) dynamic events such as machine breakdown and random job arrivals are not considered; and (2) how to balance the multi-objective during the training of the Q-table should be enhanced.

5 Conclusions

This study considered a special case of HFS, wherein the worker ergonomic factors and TOU electronic tariffs simultaneously. To efficiently solve the considered problem, the following components are embedded, i.e., six versions of NEH initialization heuristics, a dynamic decoding method, two problem-specific lemmas, a multi-population Q-learning collaboration mechanism, and an adaptive-space crowding distance approach. All these components have been tested in the corresponding ablation experiments. Finally, the proposed HRL algorithm has also been verified with five state-of-the-art algorithms.

AI use statement

No AI tools were involved.

ETHICS STATEMENT

No human and animal subjects were involved.

REPRODUCIBILITY STATEMENT

The code and instance can be found in https://github.com/DATAdriving/ICLR_code_data.git.

REFERENCES

[1] Zhang, C., Cao, Z., Song, W., Wu, Y., Zhang, J. (2024, May). Deep reinforcement learning guided improvement heuristic for job shop scheduling. In International Conference on Learning Representations (Vol. 2024, pp. 45696-45716).

[2] C. Lu, R. Gao, L. Yin, and B. Zhang, "Human–Robot Collaborative Scheduling in Energy-Efficient Welding Shop," IEEE Transactions on Industrial Informatics, vol. 20, no. 1, pp. 963-971, 2024.

[3] D. Lei, L. Gao, and Y. Zheng, "A novel teaching-learning-based optimization algorithm for energy-efficient scheduling in hybrid flow shop. IEEE Transactions on Engineering Management, vol. 65, no. 2, 330-340, 2018.

[4] W. Shao, Z. Shao, and D. Pi, "An Ant Colony Optimization Behavior-Based MOEA/D for Distributed Heterogeneous Hybrid Flow Shop Scheduling Problem Under Nonidentical Time-of-Use Electricity Tariffs," IEEE Transactions on Automation Science and Engineering, vol. 19, no. 4, pp. 3379-3394, 2022.

[5] C. Zeng, J. Liu, and Q. Li, "A constraint programming approach for resource-constrained flexible assembly flow shop scheduling problem with batch direct delivery," Computers & Operations Research, vol. 173, 2025.

[6] H. Song, J. Li, Z. Du, X. Yu, Y. Xu, Z. Zheng, and J. Li, "A Q-learning driven multi-objective evolutionary algorithm for worker fatigue dual-resource-constrained distributed hybrid flow shop," Computers & Operations Research, vol. 175, 2025.

[7] Y. Liu, W. Shen, C. Zhang, and X. Sun, "Agent-based simulation and optimization of hybrid flow shop considering multi-skilled workers and fatigue factors," Robotics and Computer-Integrated Manufacturing, vol. 80, 2023.

[8] H. Luo, B. Du, G. Huang, H. Chen, and X. Li, "Hybrid flow shop scheduling considering machine electricity consumption cost," International journal of production economics, vol. 146, no. 2, pp. 423-439, 2013.

[9] J. Li, J. Li, K. Gao, and P. Duan, "A Hybrid Graph-Based Imitation Learning Method for a Realistic Distributed Hybrid Flow Shop With Family Setup Time," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 54, no. 12, pp. 7291-7304, 2024.

[10] Y. Chen, J. Du, J. Mumtaz, J. Zhong, and M. Rauf, "An efficient Q-learning integrated multi-objective hyper-heuristic approach for hybrid flow shop scheduling problems with lot streaming," Expert Systems with Applications, vol. 262, 2025.

[11] W. Zhang, C. Li, M. Gen, W. Yang, and G. Zhang, "A multiobjective memetic algorithm with particle swarm optimization and Q-learning-based local search for energy-efficient distributed heterogeneous hybrid flow-shop scheduling problem," Expert Systems with Applications, vol. 237, 2024.

[12] B. Lu, K. Gao, Y. Ren, D. Li, and A. Slowik, "Combining meta-heuristics and Q-learning for scheduling lot-streaming hybrid flow shops with consistent sublots," Swarm and Evolutionary Computation, vol. 91, 2024.

[13] Z. Shao, W. Shao, J. Chen, and D. Pi, "MQL-MM: A Meta-Q-Learning-Based Multi-Objective Metaheuristic for Energy-Efficient Distributed Fuzzy Hybrid Blocking Flow-Shop Scheduling Problem," IEEE Transactions on Evolutionary Computation, pp. 1-1, 2024.

[14] F. Zhao, F. Yin, L. Wang, Y. Yu, "A co-evolution algorithm with dueling reinforcement learning mechanism for the energy-aware distributed heterogeneous flexible flow-shop scheduling problem," IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2024.

[15] P. Li, Q. Xue, Z. Zhang, J. Chen, and D. Zhou, "Multi-objective energy-efficient hybrid flow shop scheduling using Q-learning and GVNS driven NSGA-II," Computers & Operations Research, vol. 159, 106360, 2023.

[16] Y. Fu, M. Zhou, X. Guo, and L. Qi, "Scheduling Dual-Objective Stochastic Hybrid Flow Shop With Deteriorating Jobs via Bi-Population Evolutionary Algorithm," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 50, no. 12, pp. 5037-5048, 2020.

[17] J. Li, X. Chen, P.-Y. Duan, and J.-H. Mou, "KMOEA: A Knowledge-Based Multiobjective Algorithm for Distributed Hybrid Flow Shop in a Prefabricated System," IEEE Transactions on Industrial Informatics, vol. 18, no. 8, pp. 5318-5329, 2022.

[18] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Trans. Evol. Comput. vol.6, no. 2, 182-197, 2002.

[19] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," IEEE Trans. Evol. Comput. vol.6, no. 2, 182-197, 2002.

[20] G. Wang, X. Li, L. Gao, P. Li, "Energy-efficient distributed heterogeneous welding flow shop scheduling problem using a modified MOEA/D", Swarm and Evolutionary Computation, vol. 62, 100858, 2021.

[21] Y. Pan, K. Gao, Z. Li, N. Wu, "Solving biobjective distributed flow-shop scheduling problems with lot-streaming using an improved Jaya algorithm," IEEE Transactions on Cybernetics, vol. 53, no. 6, 3818-3828, 2023.

[22] S. Usman, and C. Lu, "Job-shop scheduling with limited flexible workers considering ergonomic factors using an improved multi-objective discrete Jaya algorithm," Computers Operations Research, vol. 162, 106456, 2024.

[23] S. Wang, X. Wang, F. Chu, J. Yu, "An energy-efficient two-stage hybrid flow shop scheduling problem in a glass production," International Journal of Production Research, vol. 58, no. 8, pp. 2283-2314, 2020.

[24] F. Ming, W. Gong, Y. Jin, "Growing neural gas network-based surrogate-assisted pareto set learning for multimodal multi-objective optimization," Swarm and Evolutionary Computation, vol. 87, 101541, 2024.

[25] F. Ming, W. Gong, L. Wang, Y. Jin, Y. "Constrained multi-objective optimization with deep reinforcement learning assisted operator selection," IEEE/CAA Journal of Automatica Sinica, vol. 11, no. 4, pp. 919-931, 2024.

A Appendix

A.1 Problem models

A.1.1 Indices

n: Total number of jobs;

ϑ: Total number of stages;

m: Total number of machines;

N: Total number of hours of each period;

ζ: Total number of speeds;

τ: Total number of TOU zones;

W: Total number of workers;

j, j_2: Indices of job;
i: Index of machine;
k: Index of stage;
u: Index of TOU zone;
r: Index of processing positions on each machine, $r = 1, \ldots, n$;
p: Index of processing positions on each worker, $p = 1, \ldots, nm$;
s: Index of speed;
w: Index of worker;
t: Index of hour in each period;

A.1.2 PARAMETERS AND VARIABLES

$O_{j,k}$: The kth task of job j being processed at stage k;
$pt_{j,k,w}$: Processing time of the kth task of job j by worker w;
$\overline{pt}_{j,k,w}$: Real processing time of the kth task of job j by worker w;
LW_j: Load weight of job j;
$Dh_{j,k}$: Horizontal movement distance of operation $O_{j,k}$;
$Fp_{j,k}$: Force of push/pull movement of operation $O_{j,k}$;
$h_{j,k,1}$: Minimum height of $O_{j,k}$, from which the job should be raised;
$h_{j,k,2}$: Maximum height of $O_{j,k}$, from which the job should be lowered;
ξ_{wp}: Energy consumed during the processing operation of workers;
ξ_{pt}: Energy consumed during the processing operations;
ξ_{idle}: Energy consumed during the idle time of machine and workers;
ξ^m_{idle}: Energy consumed during the machine idle time;
ξ^w_{idle}: Energy consumed during the worker idle time;
UP_i: Unit processing energy consumption of machine i;
US_i: Unit standby or idle energy consumption of machine i;
Et^{on}_i: Energy consumption for turning on machine i;
Et^{off}_i: Energy consumption for turning off machine i;
UES_w: Unit of energy consumption for standby of worker w without any operation;
EWr_w: Energy consumption for letting worker w rest, which is used for the worker recovery from sleep;
GoS: Grade of the walking surface;
G_w: Gender of worker w;
W_w: Weight of worker w;
V_w: Walking speed of worker w;
$v_{j,k}$: Processing speed of job j at stage k;
$S_{j,k}$: Starting time of the kth task of job j;
$C_{j,k}$: Completion time of the kth task of job j;
pz_t: Price of the tth TOU zone;
M_k: The set of machines at stage k;
$SM_{i,k,r}$: Starting time of the rth position on ith machine at stage k;
$EM_{i,k,r}$: Completion time of the rth position on ith machine at stage k;
$SW_{w,k,p}$: Starting time of the pth position on wth worker at stage k;
$EW_{w,k,p}$: Completion time of the pth position on wth worker at stage k;
ρ_e: Total electricity prices;
L: A very large number.

A.1.3 DECISION VARIABLES

$X_{j,k,t}$: A binary value that is set to 1 if $O_{j,k}$ is processed in stage k at time t; otherwise, it is set to 0.

$Y_{i,j,k,r}$: A binary value that is set to 1 if $O_{j,k}$ is processed at rth position on machine i; otherwise, it is set to 0.

$U_{j,k,w,p}$: A binary value that is set to 1 if $O_{j,k}$ is processed by the worker w at the pth position; otherwise, it is set to 0.

$Z_{j,k,i,w}$: A binary value that is set to 1 if $O_{j,k}$ is processed by the worker w on machine i; otherwise, it is set to 0.

A.1.4 CALCULATION OF THE REAL PROCESSING TIME

The calculation of the real processing time is calculated using Eq. (11).

$$\overline{pt}_{j,k,w} = pt_{j,k,w}/v_{j,k}, \quad \forall j,k,w \tag{11}$$

where $v_{i,k}$ is the processing speed of machine i at state k.

A.1.5 CONSTRAINTS ON STARTING AND COMPLETION TIMES

Constraints (12) and (13) guarantee that the starting and completion times of each operation should be positive values, constraints (14) and (15) make sure that the starting and completion times of each processing position on each machine should not be negative values. Constraint (16) ensures that the overlap between any two consecutive positions on each machine should not be violated. Constraints (17) and (18) make sure that the completion time should be equal to the sum of the starting time and the real processing times of the operation assigned on it. Constraint (19) ensures that there is no overlap between two consecutive operations of the same job.

$$S_{j,k} \geq 0, \quad \forall j,k \tag{12}$$

$$C_{j,k} \geq 0, \quad \forall j,k \tag{13}$$

$$SM_{i,k,r} \geq 0, \quad \forall i,k,r \tag{14}$$

$$EM_{i,k,r} \geq 0, \quad \forall i,k,r \tag{15}$$

$$EM_{i,k,r} \leq SM_{i,k,r+1}, \quad \forall i,k,\ r = 1,\ldots,n-1 \tag{16}$$

$$SM_{i,k,r} + \overline{pt}_{j,k,w} \geq EM_{i,k,r} - L\left(2 - Y_{i,j,k,r} - Z_{j,k,i,w}\right), \quad \forall i \in M_k, k, r, w \tag{17}$$

$$SM_{i,k,r} + \overline{pt}_{j,k,w} \leq EM_{i,k,r} + L\left(2 - Y_{i,j,k,r} - Z_{j,k,i,w}\right), \quad \forall i \in M_k, k, r, w \tag{18}$$

$$C_{j,k} \leq S_{j,k+1}, \quad \forall j,\ k = 1,\ldots,\vartheta \tag{19}$$

A.1.6 CONSTRAINTS ON MACHINE ASSIGNMENT

Constraint (20) guarantees that each operation should not be assigned to more than one machine. Constraint (21) ensures that each processing position of each machine should not be assigned with more than one operation. Therefore, both constraints (20) and (21) ensure unique assignment. Constraint (22) ensures that the previous processing positions on each machine should be assigned with a high priority, that is, no idle position exists between two assigned positions. Constraints (23)–(26) ensure that the starting and completion times of the processing position on each machine should be equal to those of the assigned operation. Constraints (27)–(30) ensure that the starting and completion times of the processing position on each worker should be equal to those of the assigned operation.

$$\sum_{i \in M_k} \sum_{r=1}^{n} Y_{i,j,k,r} \leq 1, \quad \forall j,k \tag{20}$$

$$\sum_{j=1}^{n}\sum_{k=1}^{\vartheta} Y_{i,j,k,r} \leq 1, \quad \forall i, r \tag{21}$$

$$\sum_{j=1}^{n} Y_{i,j,k,r} \geq \sum_{j=1}^{n} Y_{i,j,k,r+1}, \quad \forall i, k,\ r = 1, \ldots, n \tag{22}$$

$$SM_{i,k,r} \leq S_{j,k} + L\left(1 - Y_{i,j,k,r}\right), \quad \forall i, j, k, r \tag{23}$$

$$SM_{i,k,r} \geq S_{j,k} - L\left(1 - Y_{i,j,k,r}\right), \quad \forall i, j, k, r \tag{24}$$

$$EM_{i,k,r} \leq C_{j,k} + L\left(1 - Y_{i,j,k,r}\right), \quad \forall i, j, k, r \tag{25}$$

$$EM_{i,k,r} \geq C_{j,k} - L\left(1 - Y_{i,j,k,r}\right), \quad \forall i, j, k, r \tag{26}$$

$$SW_{w,k,p} \leq S_{j,k} + L\left(1 - U_{j,k,w,p}\right), \quad \forall w, j, k, p \tag{27}$$

$$SW_{w,k,p} \geq S_{j,k} - L\left(1 - U_{j,k,w,p}\right), \quad \forall w, j, k, p \tag{28}$$

$$EW_{w,k,p} \leq C_{j,k} + L\left(1 - U_{j,k,w,p}\right), \quad \forall w, j, k, p \tag{29}$$

$$EW_{w,k,p} \geq C_{j,k} - L\left(1 - U_{j,k,w,p}\right), \quad \forall w, j, k, p \tag{30}$$

A.1.7 CONSTRAINTS ON TOU

Constraint (31) guarantees that if the current time t locates between the starting and completion times of $O_{j,k}$, then the decision variable $X_{j,k,t}$ should be set to 1. Constraint (32) ensures that the sum of $X_{j,k,t}$ should be equal to the real processing time of $O_{j,k}$.

$$t \geq S_{j,k} \ \wedge\ t \leq C_{j,k} \ \Rightarrow\ X_{j,k,t} = 1, \quad \forall j, i \tag{31}$$

$$\sum_{t} X_{j,k,t} = C_{j,k} - S_{j,k}, \quad \forall j, k \tag{32}$$

A.1.8 CONSTRAINTS ON THE RANGE OF THE DECISION VARIABLES

The range of the decision variables is set in constraints (33) (36).

$$X_{j,k,t} \in \{0, 1\}, \quad \forall j, k, t \tag{33}$$

$$Y_{i,j,k,r} \in \{0, 1\}, \quad \forall i, j, k, r \tag{34}$$

$$Z_{j,k,i,w} \in \{0, 1\}, \quad \forall j, k, i, w \tag{35}$$

$$U_{j,k,w,p} \in \{0, 1\}, \quad \forall j, k, w, p \tag{36}$$

A.1.9 CONSTRAINTS RELATED TO THE WORKERS

The following constraints are set for the workers, wherein Constraint (37) ensures that each operation $O_{j,k}$ can be assigned to exactly one position on one worker. Constraint (38) guarantees that the number of operations assigned on one position of each worker should be no more than one. Constraint (39) makes sure that the processing positions on each worker should be assigned from left to right, and no blank position should occur. Constraint (40) ensures that an overlap is not permitted between two operations assigned on the two consecutive positions of the same worker. Constraints (41) and (42) guarantee that the completion time of $O_{j,k}$ should be the sum of the starting time and the realistic processing times on the assigned worker. Constraint (43) ensures that each operation should be assigned to exactly one machine and one worker.

$$\sum_{w=1}^{W}\sum_{p=1}^{nm} U_{j,k,w,p} = 1, \quad \forall j, k \tag{37}$$

$$\sum_{j=1}^{n}\sum_{k=1}^{\vartheta} U_{j,k,w,p} \leq 1, \quad \forall w, p \tag{38}$$

$$\sum_{j=1}^{n}\sum_{k=1}^{\vartheta} U_{j,k,w,p} \geq \sum_{j_2=1}^{n}\sum_{k_2=1}^{\vartheta} U_{j_2,k_2,w,p+1}, \quad \forall w,\ p=1,\ldots,nm-1 \tag{39}$$

$$S_{j_2,k_2} \geq C_{j,k} - L\left(2 - U_{j_2,k_2,w,p+1} - U_{j,k,w,p}\right), \quad \forall w,\ p=1,\ldots,nm-1,\ j,j_2 \in [1,n],\ k,k_2 \in [1,\vartheta] \tag{40}$$

$$S_{j,k} + \overline{pt}_{j,k,w} \geq C_{j,k} - L\left(2 - \sum_{i=1}^{m} Z_{j,k,i,w}\right), \quad \forall j,k,w \tag{41}$$

$$S_{j,k} + \overline{pt}_{j,k,w} \leq C_{j,k} + L\left(2 - \sum_{i=1}^{m} Z_{j,k,i,w}\right), \quad \forall j,k,w \tag{42}$$

$$\sum_{i=1}^{m}\sum_{w=1}^{W} Z_{j,k,i,w} = 1, \quad \forall j,k \tag{43}$$

A.1.10 Calculation of the energy-related objectives

Equations (44)–(54) are used to calculate the energy consumed by workers for standing, walking, carrying load, holding load, squat lifting, arm lifting, bench pushing, arm lowering, squat lowering, lateral movement, and forward movement, respectively.

Equation (55) is used to calculate the processing energy consumed by the workers, while Eq. (56) describes the processing energy consumed by the machines. Equation (59) calculates the standby or idle energy consumption, which includes two parts, i.e., standby energy consumed by machines (Eq. 57) and that of workers (Eq. 58).

$$\xi_1 = \left(Z_{j,k,i,w} \times 0.024 \times W_w \times \overline{pt}_{j,k,w}\right) \tag{44}$$

$$\xi_{21} = \Big(Z_{j,k,i,w} \times 0.01 \times \overline{pt}_{j,k,w} \times \left[51 + 2.54 \times W_w \times (V_w)^2 + 0.379 \times W_w \times GoS \times V_w\right]\Big) \tag{45}$$

$$\xi_{22} = \Big(Z_{j,k,i,w} \times 0.01 \times \overline{pt}_{j,k,w} \times \left[68 + 2.54 \times W_w \times V_w + 4.08 \times LW_j \times (V_w)^2 + 11.4 \times LW_j + 0.379 \times (LW_j + W_w) \times GoS \times V_w\right]\Big) \tag{46}$$

$$\xi_{23} = \left(Z_{j,k,i,w} \times 0.037 \times \overline{pt}_{j,k,w} \times LW_j\right) \tag{47}$$

$$\xi_{24} = \Big(Z_{j,k,i,w} \times 0.01 \times (h_{j,k,2} - h_{j,k,1}) \times \left[0.514 \times W_w \times (0.81 - h_{j,k,1}) + (2.19 \times LW_j + 0.62 \times G_w)\right]\Big) \tag{48}$$

$$\xi_{25} = \Big(Z_{j,k,i,w} \times 0.01 \times (h_{j,k,2} - h_{j,k,1}) \times \left[0.062 \times W_w \times (h_{j,k,2} - 0.81) + (3.19 \times LW_j - 0.52 \times G_w \times LW_j)\right]\Big) \tag{49}$$

$$\xi_{26} = \Big(Z_{j,k,i,w} \times 0.01 \times Dh_{j,k} \times \left[0.112 \times W_w + 1.15 \times Fp_{j,k} + 0.505 \times G_w \times Fp_{j,k}\right]\Big) \tag{50}$$

$$\xi_{27} = \Big(Z_{j,k,i,w} \times 0.01 \times (h_{j,k,2} - h_{j,k,1}) \times \left[0.093 \times W_w \times (h_{j,k,2} - 0.81) + (1.02 \times LW_j + 0.37 \times G_w \times LW_j)\right]\Big) \tag{51}$$

$$\xi_{28} = \Big(Z_{j,k,i,w} \times 0.01 \times (h_{j,k,2} - h_{j,k,1}) \times \left[0.511 \times W_w \times (0.81 - h_{j,k,1}) + (0.701 \times LW_j)\right]\Big) \tag{52}$$

$$\xi_{29} = \left(Z_{j,k,i,w} \times 0.01 \times [2.54 + 1.1 \times LW_j + 0.248 \times G_w \times Fp_{j,k}] \right) \tag{53}$$

$$\xi_{2\,10} = \left(Z_{j,k,i,w} \times 0.01 \times Dh_{j,k} \times [6.3 + 2.71 \times LW_j] \right) \tag{54}$$

$$\xi_{wp} = \xi_1 + \sum_{i=1}^{10} \xi_{2i} \tag{55}$$

$$\xi_{pt} = \sum_{j=1}^{n} \sum_{k=1}^{\vartheta} \sum_{i \in M_k} \sum_{t=1}^{N} \sum_{w=1}^{W} \sum_{r=1}^{n} Y_{i,j,k,r} \times UP_i \times \overline{pt}_{j,k,w} \tag{56}$$

$$\xi_{idle}^{m} = \sum_{k=1}^{\vartheta} \sum_{i \in M_k} \sum_{r=2}^{n} (SM_{i,k,r+1} - EM_{i,k,r}) \times US_i, \quad \forall i \tag{57}$$

$$\xi_{idle}^{w} = \sum_{k=1}^{\vartheta} \sum_{i \in M_k} \sum_{p=2}^{nm} (SW_{w,k,p+1} - EW_{w,k,p}) \times UES_w, \quad \forall w \tag{58}$$

$$\xi_{idle} = \xi_{idle}^{m} + \xi_{idle}^{w} \tag{59}$$

A.1.11 Calculation of the electricity price objectives

Equation (60) is used to calculate the TEP ρ_e.

$$\rho_e = \sum_{t=1}^{\tau} \left(pz_t \times \sum_{j=1}^{n} \sum_{k=1}^{\vartheta} X_{j,k,t} \right) \tag{60}$$

A.2 Methodology

A.2.1 Initialization method

The initialization method involves the generation of the first encoding type, which is based on the well-known NEH method. Unlike the classical NEH heuristic, the following six variations of NEH are embedded considering different types of objectives.

(1) NEH-I is used to generate a solution considering the minimization of the makespan. Similar to the classical NEH method, first, all the jobs are sorted according to the non-decreasing order of their average processing time at the first stage. The first two jobs are then selected as the initial of part schedule. Next, a loop is performed to insert the following jobs one by one, and in each loop the task is to find the best suitable position thus minimizing the makespan of the partial schedule. Finally, the final encoding is obtained when all the jobs have been inserted.

(2) NEH-II is used for the minimization of TEC. The main difference between NEH-II and NEH-I is that when inserting one job into the partial schedule, the suitable position is the one with the minimum TEC objective.

(3) NEH-III is used for the minimization of TEP objective. Unlike NEH-I, NEH-III inserts the unscheduled job at the position with the minimum TEP value.

(4) NEH-IV is an improved version of the classical NEH method, which first sorts all the jobs according to the non-increasing order of their average processing times. Each unscheduled job is then inserted considering the minimization of makespan.

(5) NEH-V first calculates the mean processing time $\overline{p}$ of all the jobs at the first stage. The absolute distance value $|\widetilde{p_j} - \overline{p}|$ between the average processing time $\widetilde{p_j}$ of job j and $\overline{p}$ is then calculated. Next, all the jobs are sorted according to the non-decreasing order of their $|\widetilde{p_j} - \overline{p}|$ values. The suitable position for each unscheduled job is then chosen according to the minimization of the TEC objective.

(6) NEH-VI first sorts all the jobs according to the non-increasing order of their $|\widetilde{p_j} - \overline{p}|$ values. The TEP objective is then used to insert the unscheduled jobs. Based on the six variations of NEH

heuristics, the initial population is generated as follows: first, six solutions are generated using NEH-I to NEH-VI, respectively; and then, other solutions are generated using a random method.

The main advantages of this type of initial method are as follows: first, using six different version of NEH method to generate high performance of initial solutions; then, using random method to generate the other solutions can enhance the population diversity and therefore can improve the global searching abilities. Algorithm 3 shows the detailed steps of the initialization heuristic.

Algorithm 3: Initialization heuristic

Input: instances, system parameters, e.g., population size P_s
Output: the initial population P_{init}
Generate six initial solutions using NEH-I to NEH-VI, and store them in P_{init} ;
for $i = 1$ **to** P_s *-6* **do**
  Generate a solution using a random method and record it in P_{init} ;
end
For each solution in P_{init} , decode the solution using ECT rule for machine assignment.

A.2.2 NEIGHBORHOOD STRUCTURES

The following neighborhood structures are embedded:

• NI-I: Operation swap mutation (OSM) operator, which randomly selects two jobs and swaps them to generate a new neighboring solution;

• NI-II: Operation insertion mutation (OIM) operator, which randomly selects two jobs and inserts one of them to the position of the other one;

• NI-III: Operation reverse mutation (ORM) operator, which randomly selects two positions and reverses the jobs between;

• NI-IV: critical swap mutation (CSM) operator, which swaps a randomly-selected critical operation and a randomly-selected non-critical operation;

• NI-V: critical insertion mutation (CIM) operator, which inserts a randomly-selected critical operation before a randomly-selected non-critical operation;

• NI-VI: critical reverse mutation (CRM) operator, which reverses all the elements between a critical operation and a non-critical operation;

• NI-VII: three-point cutting mutation (TCM) operator, which first generates three points and cuts the solution into three parts; next, which are chosen in a random sequence to construct a completed solution;

A.2.3 CROSSOVER OPERATORS

Crossover operators are embedded for the scheduling vectors. The two-point order crossover (2p-OXO) is directly introduced in the proposed algorithm.

A.3 BI-Q-TABLE TRAINING HEURISTIC

A Bi-Q-table heuristic is embedded to enhance the learning abilities of the proposed algorithm, as shown in Algorithm 4.

A.3.1 Q-TABLE DRIVEN DESTRUCTION AND CONSTRUCTION HEURISTIC

The iterated greedy (IG) algorithm that involves the following steps: (1) destruction phase: in which a certain number of jobs are randomly deleted from the current solution, with the set of deleted jobs named J_d and the remaining partial solution denoted as S_r ; and (2) construction phase, in which each job in J_d is inserted into the most suitable position into S_r considering certain objectives, until J_d is empty.

Algorithm 4: A bi-Q-table training heuristic

Input: parameters including: population size P_s, learning factory α, discount factory γ, Epsilon-greedy factory ε, and maximum episodes e
Output: the two trained Q tables

```
% Q-table training process
Initialize the following Q tables: Q_1^i, Q_2^i, i = 1, 2, 3, 4;
Initialize the population and record the global best solution x*;
% training for Q_1^i;
Randomly select the initial solution x_1, state s_1, and action a_1;
for t = 1 to e do
    for i = 1 to 4 do
        % four Q-learning components;
        Generate a neighboring solution x by applying a_1 to x_1;
        if x < x_1 then
            Calculate r;
            Update the Q_1^i tabl;
            s_1, a_1 = ε(Q_1^i, s_1, a_1);
            x_1 = x;
            Update x* with x;
        end
    end
end
% training for Q_2^i;
Let A_2 = {NI-I, NI-II, NI-III, NI-IV, NI-V, NI-VI, NI-VII} (c.f. Section IV-D);
Perform lines 5–17 with Q_2^i, state s_2, and action a_2;
end;
```

The number of deleted jobs, i.e., $|J_{\mathrm{d}}|$, is a critical parameter that can affect the performance of IG. An objective-oriented Q-table driven destruction and construction heuristic is developed in this study, wherein four Q tables are designed with the weight values set as [1.0, 0.0, 0.0], [0.0, 1.0, 0.0], [0.0, 0.0, 1.0], and [0.4, 0.3, 0.3], respectively. The first three Q tables are designed for makespan, TEC, and TEP, respectively, while the last one focuses on a weighed sum of the three objectives.

A.3.2 Q-TABLE DRIVEN VARIABLE NEIGHBORHOOD SEARCH HEURISTIC

VNS is typical and efficient algorithm that selects more promising neighborhood structure operators to perform local search tasks. The main task of VNS is to select the precise neighborhood structure.

In most works, a fixed weight value is set for the Q table training process, which leads to lower performance considering different objectives. To tackle this issue, a four-Q-table driven VNS heuristic is developed in this study: (1) First, four Q tables are developed with the weight values set as [1.0, 0.0, 0.0], [0.0, 1.0, 0.0], [0.0, 0.0, 1.0], and [1/3, 1/3, 1/3], respectively. (2) For each Q table, the training process is the same except the weight values for the three objectives are different. Using different weight values, the four Q tables can adapt to solving different objectives.

A.3.3 MULTI-POPULATION COLLABORATION HEURISTIC

To search in a promising space considering different objectives, a multi-population collaboration heuristic is proposed. The main ideas are as follows: (1) considering each objective, we introduce three independent sub-populations, which perform the search tasks aiming at minimization of the three objectives, respectively; and (2) considering the combination of the three objectives, a sub-population is embedded to minimize the weighted value of the three objectives.

After performing the search task independently, the four sub-populations are combined to construct a whole population, which is then divided into four sub-populations to perform a loop until the stop condition is satisfied.

A.4 Experimental Analysis

This section illustrates the detailed experimental comparisons, which include the experimental settings to set the testing instances and choose the comparison algorithms, the tuning of parameters to decide the most suitable parameters for the proposed algorithm, the ablation experiments to test the efficiency of each component, and the experimental analysis compared with the state-of-the-art efficient algorithms.

A.4.1 Comparison metrics

The comparison metrics include the spacing matrix (SM) , the inverted generational distance (IGD), and the hypervolume metric (HV).

(1) The SM, presented in Eq. 61, is used to indicate the population diversity, and the details are as follows.

$$SM\left(P_C\right)=\sqrt{\frac{\sum_{x\in P_C}\left(d_x-\overline{d}\right)^2}{|P_C|}} \tag{61}$$

(2) The IGD is used to indicate the convergence and diversity abilities of the compared algorithm, as shown in Eq. (62), where $d\left(x,y\right)$ is the Euclidean distance between the two solutions x and y.

$$\mathrm{IGD}(P_C,P^*)=\frac{\sum_{x\in P^*}\left(\min\limits_{y\in P_C} d\left(x,y\right)\right)}{|P^*|} \tag{62}$$

(3) The HV is also used to indicate the convergence and diversity abilities of the compared algorithm, as shown in Eq. (63), where v_q is the hypervolume constituted by the solution x and the reference point RP , e.g. $RP=(1.0,1.0,1.0)$.

$$\mathrm{HV}(P_C)=\lambda\left(\cup\limits_{x}^{|P_C|} v_x\right) \tag{63}$$

(4) The relative percentage increase (RPI) is used to evaluate the performance of the comparison algorithms, which is given as follows:

$$\mathrm{RPI}=\frac{f_c-f_b}{f_b}\times 100\% \tag{64}$$

where, f_b is the best value among the comparison algorithms, which can be represented by SM, HV, or IGD values; and f_c is the obtained value by the comparison algorithm.

A.5 Parameter tuning

The following five key parameters are important for the proposed algorithm, including the population size P_s , the crossover probability p_c , the mutation probability p_m, the CPU time ($CT\times n\times m$ seconds), and the Q-learning heuristic stop condition Q_{iter} . In order to make a fair parameter tuning independent from the other experiments, we randomly generate eight different-scale instances to conduct the parameter tuning procedure, wherein the number of jobs ranges in 10, 20, number of stages ranges in 3, 5, and the number of parallel machines ranges in 9, 12, 15, 20. tabularx multirow multicol

The levels for these five key parameters are presented in Table 5. The orthogonal array $L_{27}\left(3^5\right)$ is tested to determine the most suitable parameters. From the results of the main effect plot of parameter tuning shown in Fig. 4, the parameters should be set as follows: P_s = 80, p_c = 0.8, p_m = 0.3, CT = 0.5, and Q_{iter} = 3.

A.6 Comparison of the mathematical model

A set of small-scale instances is used to make detailed comparisons between CPLEX and HRL to verify the effectiveness of the proposed mathematical model. The stop criterion for CPLEX is 10

Table 4: Levels of parameters

Factors	IGD values		
	1	2	3
P_s	40	80	120
p_c	0.6	0.8	0.9
p_m	0.1	0.3	0.5
CT	0.2	0.5	0.8
Q_{iter}	2	3	5

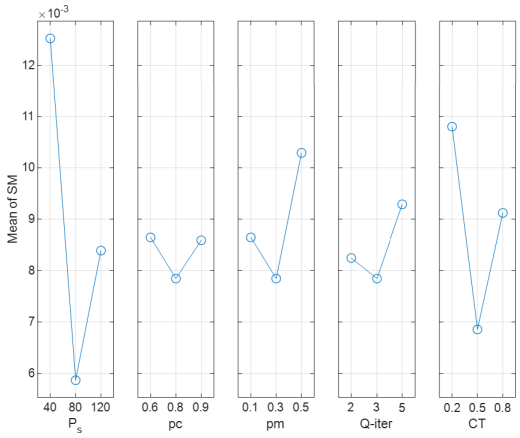

Figure 4: Main effect plot of parameter tuning

hours for each instance, while 20 seconds for HRL. The range of the number of jobs, stages, and machines are set to 3, 4, 5, 6, 7, 2, 3, and 4, 6, respectively. Totally, 8 groups of different instances are tested and the results are collected in Table IV.

It can be observed from Table III that: (1) the proposed algorithm shows competitive performance for solving instances with jobs smaller than six. For example, for the two types of instances, i.e., "3-2-4" and "5-2-4", HRL obtained non-dominated solutions; (2) for relative large-scale instances, CPLEX cannot obtain any solutions within 10 hours, while the proposed HRL can obtain non-dominated solutions in short period.

A.7 Effectiveness of the crowding distance mechanism

Two types of algorithms were coded to test the efficiency of the proposed adaptive-space-based crowding distance heuristic. Those include HRL_NCR algorithm which contains all the components of the proposed HRL algorithm with the crowding distance heuristic in NSGA-II, rather than the adaptive-space crowding distance mechanism in HRL; and the HRL algorithm with all the components discussed in Section IV. The two compared algorithms were performed with the same computational environments. After 30 independent runs for each instance, the Pareto solutions were collected to make detailed comparisons.

Table 6 presents the computational results collected by HRL_NCR and HRL algorithms, wherein the SM indicator is used to analyze the performance. The first column of Table 6 lists the problem scale that contains three numbers representing the number of jobs, stages, and parallel machines in each stage. The second column lists the best SM values collected from HRL_NCR and HRL algorithms. The following two columns list the SM values for HRL_NCR and HRL algorithms, respectively, while the last two columns show the RPI values for both algorithms.

It can be concluded from Table 6 that: (1) considering the SM values, the proposed HRL algorithm obtained 12 better results out of the given 18 instances, while the HRL_NCR can obtain only 6 better values; (2) the average SM values displayed in the last line show that the HRL algorithm obtained

Table 5: Comparison results with CPLEX and HRL

Inst	Pareto solutions	
	CPLEX	HRL
3-2-4	(4, 68748.6, 330),(8, 66172.5, 280)	(4, 68459.6, 340)
4-2-4	(8, 84017.9, 240),(4, 89498.7, 220)	(4, 100537.3, 280)
5-2-4	(4, 102124.8, 330),(10, 91363.2, 330)	(4, 99708.0, 330),(5, 101769.5, 300)
5-3-6	(4, 100454.7, 400),(10, 86802.8, 410)	(5, 120208.5, 500)
6-3-6	-	(6, 147825.3, 430),(7, 147520.5, 410)
6-3-6	-	(10, 250562.8, 790)
7-3-6	-	(10, 252278.0, 690),(11, 251762.3, 720)
7-3-6	-	(13, 336897.8, 830),(14, 333712.6, 830)
Num of PF	7	10

Table 6: Comparison Results of The Crowding Distance Mechanism

Inst	Best	SM values		RPI values	
		HRL_NCR	HRL	HRL_NCR	HRL
10-3-3	0.193	0.193+	0.239	0.000	23.834
10-3-4	0.159	0.159+	0.230	0.000	44.654
10-3-6	0.140	0.14+0	0.199	0.000	42.143
10-5-3	0.145	0.198	0.145+	36.552	0.000
10-5-4	0.196	0.196+	0.252	0.000	28.571
10-5-6	0.117	0.117+	0.137	0.000	17.094
20-3-3	0.147	0.190	0.147+	29.252	0.000
20-3-4	0.157	0.186	0.157+	18.471	0.000
20-3-6	0.154	0.154+	0.206	0.000	33.766
20-5-3	0.104	0.211	0.104+	102.885	0.000
20-5-4	0.101	0.132	0.101+	30.693	0.000
20-5-6	0.136	0.151	0.136+	11.029	0.000
50-3-3	0.109	0.140	0.109+	28.440	0.000
50-3-4	0.122	0.123	0.122+	0.820	0.000
50-3-6	0.182	0.204	0.182+	12.088	0.000
50-5-3	0.098	0.108	0.098+	10.204	0.000
50-5-4	0.199	0.220	0.199+	10.553	0.000
50-5-6	0.132	0.176	0.132+	33.333	0.000
Avg	0.144	0.167	0.161+	18.018	10.559

- "+" means the better values

a better average value; (3) the RPI values displayed in the last two columns further verify the effectiveness and efficiency of the adaptive-space-based crowding distance heuristic in the proposed HRL algorithm; and (4) considering the relatively large-scale instances with 20 and 50 jobs, the HRL algorithm obtained 11 better values out of the 12 instances, which verify that the proposed crowding distance method is efficient for solving large-scale instances.

A.7.1 Effectiveness of the two lemmas

The two lemmas are used to decrease certain objectives while maintaining the other objectives, thus improving the given solution. To test the efficiency of the proposed lemmas, we implemented the two comparison algorithms, including the HRL_NL algorithm without the two lemmas and the HRL algorithm with all the components list in Section IV. Both the two comparison algorithms performed 30 independent runs for each instance. The Pareto front is constructed by collecting the non-dominated solutions obtained by the two comparison algorithms for each instance. Both HV and IGD indicators are used to make detailed comparisons between HRL_NL and HRL algorithms.

Table 7: Comparison results of the two lemmas

Inst	HV values			IGD values		
	Best	HRL_NL	HRL	Best	HRL_NL	HRL
10-3-3	0.5589	0.5513	0.5589+	0.907	1.000	0.907+
10-3-4	0.5608	0.5497	0.5608+	0.608	1.000	0.608+
10-3-6	0.5785	0.5527	0.5785+	0.474	1.000	0.474+
10-5-3	0.5348	0.5037	0.5348+	0.000	1.000	0.000+
10-5-4	0.5715	0.5316	0.5715+	0.412	1.000	0.412+
10-5-6	0.5855	0.5451	0.5855+	0.711	1.000	0.711+
20-3-3	0.5376	0.5362	0.5376+	0.747	0.747+	1.000
20-3-4	0.5471	0.5419	0.5471+	0.446	1.000	0.446+
20-3-6	0.5627	0.5493	0.5627+	0.938	1.000	0.938+
20-5-3	0.5475	0.5333	0.5475+	0.481	1.000	0.481+
20-5-4	0.5602	0.5448	0.5602+	0.942	0.942+	1.000
20-5-6	0.5659	0.5376	0.5659+	0.617	1.000	0.617+
50-3-3	0.5320	0.5297	0.5320+	0.446	1.000	0.446+
50-3-4	0.5367	0.5308	0.5367+	0.913	1.000	0.913+
50-3-6	0.5324	0.5277	0.5324+	0.498	1.000	0.498+
50-5-3	0.5367	0.5265	0.5367+	0.402	1.000	0.402+
50-5-4	0.5428	0.5313	0.5428+	0.328	1.000	0.328+
50-5-6	0.5402	0.5298	0.5402+	0.746	1.000	0.746+
Avg	0.5518	0.5363	0.5518+	0.590	0.983	0.607+

- "+" means the better values

Table 7 presents the comparison results between HRL_NL and HRL algorithms, which has the following observations: (1) considering the HV values, the proposed HRL algorithm obtained all the better values for all types of instances, which verified the diversity and convergence abilities of the proposed algorithm after applying the two lemmas; (2) considering the IGD values, the proposed algorithm with the two lemmas obtained 16 better results, while the HRL_NL algorithm only obtained two better values; and (3) the average values obtained by the HRL algorithm are significantly better than those obtained by the HRL$_N Algorithm both in terms of HV and IGD indicators.$

A.7.2 Effectiveness of the dynamic worker selection mechanism

A dynamic decoding method that contains a dynamic worker selection heuristic is proposed in Algorithm 2. To test the efficiency of this heuristic, we implemented two comparison algorithms, including HRL_NW with a randomly selection of workers and the HRL algorithm with the dynamic worker selection heuristic. All the other components of the two algorithms are set the same.

Table 8 reports the comparison results of the HV values between HRL_NW and HRL algorithms, which verifies that after applying the dynamic worker selection mechanism, the proposed HRL shows significantly performance.

A.7.3 Effectiveness of the multi-population Q-learning method

To test the efficiency of the multi-population Q-learning method, we also made a detailed comparison between two algorithms, i.e., the HRL_NQ algorithm without the proposed multi-population Q-learning method, and HRL with all the components.

Table 9 shows the detailed comparison results in HV and IGD indicators between HRL_NQ and HRL algorithms. It can be seen from Table 9 that: (1) the comparisons result of the HV values show that the proposed HRL algorithm obtained 14 better results while the HRL_NQ algorithm can only obtain four better values; (2) the proposed HRL obtained 16 better values considering the IGD indicator which is significantly better than the HRL_NQL algorithm; and (3) the average values for both HV and IGD indicators also verify the robustness of the proposed multi-population Q-learning method.

Table 8: Comparison results of the dynamic worker selection mechanism

Inst	HV values			RPI		
	Best	HRL_NW	HRL	Best	HRL_NW	HRL
10-3-3	0.5978	0.5861	0.5978	0.0000	-1.9572	0.0000+
10-3-4	0.5910	0.5752	0.5910	0.0000	-2.6734	0.0000+
10-3-6	0.5929	0.5748	0.5929	0.0000	-3.0528	0.0000+
10-5-3	0.5834	0.5686	0.5834	0.0000	-2.5369	0.0000+
10-5-4	0.5784	0.5473	0.5784	0.0000	-5.3769	0.0000+
10-5-6	0.5841	0.5689	0.5841	0.0000	-2.6023	0.0000+
20-3-3	0.5915	0.5706	0.5915	0.0000	-3.5334	0.0000+
20-3-4	0.5757	0.5461	0.5757	0.0000	-5.1416	0.0000+
20-3-6	0.5906	0.5586	0.5906	0.0000	-5.4182	0.0000+
20-5-3	0.5705	0.5372	0.5705	0.0000	-5.8370	0.0000+
20-5-4	0.5737	0.5458	0.5737	0.0000	-4.8632	0.0000+
20-5-6	0.5854	0.5575	0.5854	0.0000	-4.7660	0.0000+
50-3-3	0.5873	0.5583	0.5873	0.0000	-4.9379	0.0000+
50-3-4	0.5885	0.5618	0.5885	0.0000	-4.5370	0.0000+
50-3-6	0.5949	0.5667	0.5949	0.0000	-4.7403	0.0000+
50-5-3	0.5797	0.5434	0.5797	0.0000	-6.2619	0.0000+
50-5-4	0.5837	0.5503	0.5837	0.0000	-5.7221	0.0000+
50-5-6	0.5806	0.5451	0.5806	0.0000	-6.1144	0.0000+
Avg	0.5850	0.5590	0.5850	0.0000	-4.4485	0.0000+

- "+" means the better values

Table 9: Comparison results of the multi-population Q-learning method

Inst	HV values			IGD values		
	Best	HRL_NQ	HRL	Best	HRL_NQ	HRL
10-3-3	0.5478	0.5478+	0.5477	0.430	1.000	0.430+
10-3-4	0.5563	0.5535	0.5563+	0.695	1.000	0.695+
10-3-6	0.5490	0.5490+	0.5467	0.401	1.000	0.401+
10-5-3	0.5353	0.5341	0.5353+	0.426	1.000	0.426+
10-5-4	0.5323	0.5308	0.5323+	0.350	1.000	0.350+
10-5-6	0.5439	0.5412	0.5439+	0.401	0.401+	1.000
20-3-3	0.5332	0.5320	0.5332+	0.577	1.000	0.577+
20-3-4	0.5369	0.5369+	0.5343	0.534	1.000	0.534+
20-3-6	0.5537	0.5456	0.5537+	0.315	1.000	0.315+
20-5-3	0.5330	0.5305	0.5330+	0.330	1.000	0.330+
20-5-4	0.5384	0.5373	0.5384+	0.244	1.000	0.244+
20-5-6	0.5400	0.5360	0.5400+	0.388	1.000	0.388+
50-3-3	0.5269	0.5266	0.5269+	0.096	1.000	0.096+
50-3-4	0.5291	0.5278	0.5291+	0.453	1.000	0.453+
50-3-6	0.5272	0.5265	0.5272+	0.000	1.000	0.000+
50-5-3	0.5279	0.5277	0.5279+	0.817	1.000	0.817+
50-5-4	0.5329	0.5313	0.5329+	0.723	0.723+	1.000
50-5-6	0.5281	0.5281+	0.5263	0.554	1.000	0.554+
Avg	0.5373	0.5357	0.5370+	0.430	0.951	0.478+

- "+" means the better values

A.7.4 EFFECTIVENESS OF THE INITIALIZATION HEURISTIC

In the proposed algorithm, six variations of the NEH heuristic have been embedded to enhance the performance of the initial population. Table 10 presents the comparison results between the two comparison algorithms, i.e., the HRL_NI algorithm with a random initialization method, and the HRL algorithm with the proposed initialization method.

It can be observed from Table 10 that: (1) the proposed HRL algorithm obtained 11 better results for HV indicator, and 13 better values for IGD indicator, verifying the efficiency of the proposed initialization method; and (2) the average HV and IGD values displayed in the last line further show the robustness of the proposed initialization heuristic.

Table 10: Comparison results of the initialization heuristic

Inst	HV values			IGD values		
	Best	HRL_NI	HRL	Best	HRL_NI	HRL
10-3-3	0.5470	0.5470+	0.5451	0.201	1.000	0.201+
10-3-4	0.5527	0.5465	0.5527+	0.445	1.000	0.445+
10-3-6	0.5460	0.5412	0.5460+	0.298	1.000	0.298+
10-5-3	0.5353	0.5348	0.5353+	0.290	1.000	0.290+
10-5-4	0.5344	0.5344+	0.5341	0.549	0.549+	1.000
10-5-6	0.5507	0.5507+	0.5503	0.313	0.313+	1.000
20-3-3	0.5364	0.5364+	0.5349	0.382	0.382+	1.000
20-3-4	0.5371	0.5356	0.5371+	0.761	1.000	0.761+
20-3-6	0.5555	0.5537	0.5555+	0.935	1.000	0.935+
20-5-3	0.5337	0.5337+	0.5330	0.183	1.000	0.183+
20-5-4	0.5370	0.5355	0.5370+	0.828	1.000	0.828+
20-5-6	0.5361	0.5288	0.5361+	0.000	1.000	0.000+
50-3-3	0.5298	0.5284	0.5298+	0.312	1.000	0.312+
50-3-4	0.5290	0.5258	0.5290+	0.845	1.000	0.845+
50-3-6	0.5270	0.5267	0.5270+	0.169	1.000	0.169+
50-5-3	0.5277	0.5277+	0.5276	0.577	1.000	0.577+
50-5-4	0.5304	0.5302	0.5304+	0.528	0.528+	1.000
50-5-6	0.5304	0.5304+	0.5277	0.600	0.600+	1.000
Avg	0.5376	0.5360	0.5371+	0.456	0.854	0.602+

- ”+” means the better values