Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2024.049756

ARTICLE

Energy-Saving Distributed Flexible Job Shop Scheduling Optimization with Dual Resource Constraints Based on Integrated Q-Learning Multi-Objective Grey Wolf Optimizer

Hongliang Zhang[1,2], Yi Chen[1], Yuteng Zhang[1] and Gongjie Xu[3,*]

[1]School of Management Science and Engineering, Anhui University of Technology, Ma'anshan, 243032, China

[2]Laboratory of Multidisciplinary Management and Control of Complex Systems of Anhui Higher Education Institutes, Anhui University of Technology, Ma'anshan, 243032, China

[3]Department of Industrial Engineering, School of Mechanical Engineering, Northwestern Polytechnical University, Xi'an, 710072, China

*Corresponding Author: Gongjie Xu. Email: gjxu19@gmail.com

Received: 17 January 2024 Accepted: 27 March 2024 Published: 20 May 2024

ABSTRACT

The distributed flexible job shop scheduling problem (DFJSP) has attracted great attention with the growth of the global manufacturing industry. General DFJSP research only considers machine constraints and ignores worker constraints. As one critical factor of production, effective utilization of worker resources can increase productivity. Meanwhile, energy consumption is a growing concern due to the increasingly serious environmental issues. Therefore, the distributed flexible job shop scheduling problem with dual resource constraints (DFJSP-DRC) for minimizing makespan and total energy consumption is studied in this paper. To solve the problem, we present a multi-objective mathematical model for DFJSP-DRC and propose a Q-learning-based multi-objective grey wolf optimizer (Q-MOGWO). In Q-MOGWO, high-quality initial solutions are generated by a hybrid initialization strategy, and an improved active decoding strategy is designed to obtain the scheduling schemes. To further enhance the local search capability and expand the solution space, two wolf predation strategies and three critical factory neighborhood structures based on Q-learning are proposed. These strategies and structures enable Q-MOGWO to explore the solution space more efficiently and thus find better Pareto solutions. The effectiveness of Q-MOGWO in addressing DFJSP-DRC is verified through comparison with four algorithms using 45 instances. The results reveal that Q-MOGWO outperforms comparison algorithms in terms of solution quality.

KEYWORDS

Distributed flexible job shop scheduling problem; dual resource constraints; energy-saving scheduling; multi-objective grey wolf optimizer; Q-learning

1 Introduction

With the continuous development of the manufacturing industry, the flexible job shop scheduling problem (FJSP) has become one of the core problems in production scheduling. The increasing diversification of market demands and the shortening of product life cycles have prompted manufacturing enterprises to gradually shift from the traditional single-factory production pattern to the multi-factory collaborative production pattern. Under this pattern, optimizing the processing sequence of multiple jobs on multiple machines has become an important challenge for the manufacturing industry. To meet this challenge, the manufacturing industry has begun seeking more efficient and flexible scheduling schemes. The distributed flexible job shop scheduling problem (DFJSP) has thus become the focus of manufacturing industry and academia. Although many scholars have studied DFJSP, most of the previous research focused on machine constraints within the factories, with relatively little attention paid to worker constraints.

In actual production, the collaborative work of workers and machines is crucial to improving production efficiency. By considering the skills of workers and the characteristics of machines, resources can be allocated more rationally to avoid waste of resources [1]. Meanwhile, with the growing environmental issues, manufacturing enterprises have begun to pay attention to energy-saving manufacturing. As a specific measure of energy-saving manufacturing, energy-saving scheduling can minimize resource waste [2,3].

Therefore, the DFJSP with dual resource constraints (DFJSP-DRC) is investigated in this paper, with the objectives of minimizing makespan and total energy consumption. By considering the skills of workers and the characteristics of machines comprehensively, we construct a mathematical model of DFJSP-DRC, which enriches the existing DFJSP model and provides references for subsequent DFJSP research. Through considering the synergistic effect of worker and machine resources in actual production, this model not only enhances the practicality and applicability of scheduling theory but also provides new perspectives and tools to solve complex production scheduling problems.

It is difficult to get exact solutions using traditional mathematical methods. To obtain high-quality solutions, a novel and effective Q-learning-based multi-objective grey wolf optimizer (Q-MOGWO) is designed, which adds a local search strategy to multi-objective grey wolf optimizer (MOGWO) and uses a Q-learning strategy to dynamically adjust the local search strategy according to the population state. The main contributions of this paper are as follows:

(1) DFJSP-DRC is studied, and a multi-objective mathematical model aiming at minimizing the makespan and total energy consumption is established.

(2) A hybrid population initialization strategy is introduced to enhance the quality and diversity of the initial population, and an improved active decoding strategy that fully utilizes the public idle time of machines and workers is designed to transform solutions into efficient scheduling schemes.

(3) Two improved wolf predation strategies and a local search strategy based on Q-learning are proposed to extend the search space of solutions.

The rest of this paper is organized as follows: Section 2 describes the related works. Section 3 illustrates the multi-objective mathematical model of DFJSP-DRC. Section 4 details the proposed Q-MOGWO. Experiments in Section 5 evaluate the performance of Q-MOGWO. Section 6 provides the conclusions and future works.

2 Related Works

2.1 Distributed Flexible Job Shop Scheduling Problem

Some scholars have studied DFJSP with the objective of minimizing makespan [4–7]. Meanwhile, increasingly scholars pay attention to environmental issues, and DFJSP considering energy consumption is studied. Luo et al. [8] developed a mathematical framework for optimizing makespan, maximum workload, and total energy consumption of DFJSP. Du et al. [9] used a hybrid heuristic algorithm to optimize the makespan and total energy consumption for DFJSP considering crane transportation. Xu et al. [10] considered DFJSP, which requires operation outsourcing for some jobs, and established a mathematical model with four optimization objectives. Li et al. [11] proposed a two-stage knowledge-driven evolutionary algorithm to solve a multi-objective DFJSP with type-2 fuzzy processing time. The above studies can guide enterprises to realize energy-saving scheduling, but they only focus on machine resources and ignore worker resources. The role of workers is indispensable in the multi-variety and small-lot production model. Rational arrangement of workers can increase work efficiency and reduce costs [12].

To better simulate the real production scenario, both machines and workers should be considered in FJSP, which is referred to as the dual resource constrains flexible job shop scheduling problem (DRCFJSP). To solve the problem, Gong et al. [12] proposed a hybrid artificial bee colony algorithm with a specific local search strategy to expand the search space. Tan et al. [13] considered worker fatigue in DRCFJSP and proposed a multi-objective optimization model. Zhao et al. [14] proposed a hybrid discrete multi-objective imperial competition algorithm to solve DRCFJSP considering job transportation time and worker learning effects. Shi et al. [15] studied DRCFJSP, where employees are boredom-aware in allocating resources and scheduling tasks, and built a two-layer dictionary model to solve the problem. Luo et al. [16] used an improved mayfly algorithm based on the non-dominated sorting genetic algorithm-II (NSGA-II) structure to solve the DFJSP considering worker arrangements. Although the above literature considers both machine and worker resources in FJSP [17,18], few studies consider these resources in DFJSP. At the same time, the main optimization objective of the above literature research is makespan, and the objective of energy consumption is rarely considered.

2.2 Optimization Algorithms for DFJSP

The intelligent optimization algorithm is an effective method to solve different types of FJSP. Lin et al. [19] used a genetic algorithm with incomplete chromosome representation and shaded chromosomes to solve DFJSP. Li et al. [20] proposed an improved grey wolf optimizer for DFJSP. Xie et al. [21] proposed a hybrid genetic tabu search algorithm to address DFJSP. Li et al. [22] used an adaptive memetic algorithm to solve energy-saving DFJSP. Zhu et al. [23] applied a reformative memetic algorithm to address DFJSP considering order cancellations. Although various intelligent optimization algorithms have been employed to tackle scheduling problems, there are prevalent limitations, such as a lack of local search capability, the inability to ensure global optimal solutions, and difficulties in parameter adjustment.

Integrating reinforcement learning (RL) with intelligent optimization algorithms can effectively guide the intelligent optimization algorithms' search process, improve solution quality and accelerate convergence [24]. Several studies have combined RL and intelligent algorithms to solve scheduling problems. Cao et al. [25] suggested integrating a cuckoo search algorithm with RL modeling to address the scheduling problem in semiconductor terminal testing. Chen et al. [26] introduced a self-learning genetic algorithm for FJSP, incorporating state-action-reward-state-action (SARSA) and Q-learning as adaptive learning methods for intelligent adjustment of critical parameters. Cao et al. [27] introduced a cuckoo search algorithm based on the SARSA to solve the FJSP. Li et al. [28] used an RL-based multi-objective evolution algorithm based on decomposition (MOEA/D) to solve multi-objective FJSP with fuzzy processing time. Li et al. [29] proposed a Q-learning artificial bee colony algorithm with heuristic initialization rules, which uses Q-learning to prefer high-quality neighborhood structures.

The above literature proves that the combination of RL and intelligent optimization algorithms can help intelligent algorithms to find better solutions and accelerate convergence speed when solving various job shop scheduling problems. As a new intelligent optimization algorithm, MOGWO has been used to solve a variety of scheduling problems [30,31]. In addition, MOGWO has the advantages of fewer parameters, global search and strong adaptability, but it also lacks the ability of adaptive adjustment and local search. By combining the adaptive learning ability of Q-learning, MOGWO can dynamically adjust the search strategy, which can help MOGWO find a better solution in the complex search space. Therefore, the Q-MOGWO is designed to address the DFJSP-DRC in this study.

3 Description and Mathematical Modeling of DFJSP-DRC

3.1 Problem Definition

There are n jobs processed in p factories, and each factory is considered a flexible manufacturing unit. Job J_i contains n_i operations. Every factory has m machines and w workers and jobs have processing sequence constraints. Workers have different skill levels and can operate different machines. The same machine is operated by different workers to process the same job will produce different processing times. Therefore, the DFJSP-DRC comprises four coupled subproblems: operation sequence, factory selection, machine selection and worker selection. To clarify the proposed problem, the following assumptions are considered: (1) All factories, machines, workers and jobs are available at 0 moment. (2) There are sequential constraints between different operations for one job. (3) Each job can be processed in multiple factories but only assigned to one factory for processing. (4) Each worker can operate multiple machines, and each machine can process multiple jobs. (5) There is no preemptive operation. (6) The machine breakdown and transportation time of jobs are not considered.

To explain DFJSP-DRC, Fig. 1 depicts a Gantt chart for an instance involving 4 jobs, 4 machines, 1 factory and 3 workers. Information for jobs processing is given in Table 1, in which '1/3' indicates that the processing machine is M_1, processing time is 3, and '-' indicates that this worker cannot process this operation.

Table 1: Information for jobs processing

Jobs	Operations	W_1	W_2	W_3
J_1	O_{11}	1/3, 3/4	–	1/2, 3/4
	O_{12}	3/7, 4/8	2/5, 4/2	3/2
J_2	O_{21}	1/2, 4/3	4/5	1/6
	O_{22}	3/2, 4/3	4/6	3/4
	O_{23}	–	2/5	2/4
J_3	O_{31}	1/2, 3/9, 4/5	4/5	1/2, 3/7
	O_{32}	1/4, 3/4	–	1/5, 3/7
	O_{33}	4/2	4/7	–
J_4	O_{41}	1/7, 3/8, 4/9	2/6, 4/5	1/5, 2/7, 3/9
	O_{42}	1/4	2/4	1/6, 2/5
	O_{43}	4/5	4/9	–

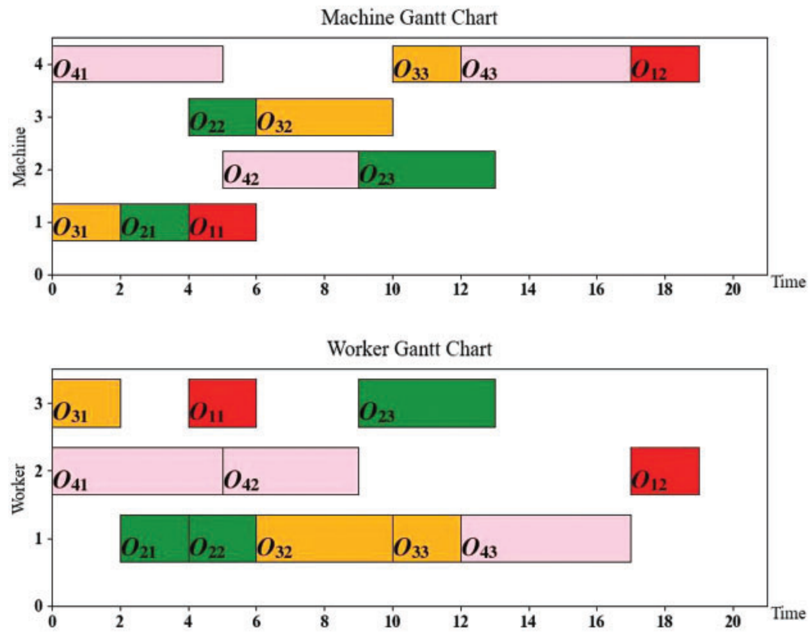

Figure 1: A Gantt chart of an instance

3.2 Mathematical Model of DFJSP-DRC

Based on the problem definition and assumptions in Section 3.1, the sets and indices, decision variables and parameters used in the mathematical model of DFJSP-DRC are as follows:

Sets and indices

i: Index of job

j: Index of operation

k: Index of machine

f: Index of factory

s: Index of worker

M_{ij}: The set of available machines to operate O_{ij}

W_{ijk}: The set of available workers who can operate machine k for operation O_{ij}

F_j: The set of available factories to operate J_i

Parameters

J_i: The i-th job

M_k: The k-th machine

W_s: The s-th worker

n_i: Total number of operations for J_i

O_{ij}: The j-th operation of J_i

T_{ijks}: The processing time of O_{ij} processed by worker s on machine k

ST_{ij}: Start processing time of O_{ij}

ET_{ij}: End processing time of O_{ij}

T_{ijksf}: The processing time of O_{ij} processed by worker s on machine k in factory f

ST_{ijf}: Starting processing time of O_{ij} in factory f

ET_{ijf}: End processing time of O_{ij} in factory f

c_{ijks}: Completion time of O_{ij} on machine k by worker s

C_i: Completion time of J_i

C_{max}: Makespan

Me_k: Process energy consumption per unit time for machine k

Re_k: Idle energy consumption per unit time for machine k

PE: Total processing energy consumption

IE: Total idle energy consumption

TEC: Total energy consumption

ε: A sufficiently sizeable positive number

Decision variables

W_{if}: 0-1 decision variables, take value 1 when J_i is processed in factory f; otherwise, the value is 0.

X_{ijksf}: 0-1 decision variables, take value 1 when O_{ij} is processed in factory f by worker s operating machine k; otherwise, the value is 0.

$Y_{i'j'ijkf}$: 0-1 decision variables, take value 1 when O_{ij} and $O_{i'j'}$ are processed by machine k in factory f, $O_{i'j'}$ is processed immediately before O_{ij}; otherwise, the value is 0.

$Z_{i'j'ijsf}$: 0-1 decision variables, take a value of 1 to indicate that O_{ij} and $O_{i'j'}$ are processed by worker s in factory f, $O_{i'j'}$ is processed immediately before O_{ij}; otherwise, the value is 0.

Combining the symbol description and problem definition, the mathematical model of DFJSP-DRC is developed:

$$f_1 = \min\left(\max\left(C_i\right)\right) \tag{1}$$

$$f_2 = \min\left(TE\right) \tag{2}$$

$$\sum_{f=1}^{p} W_{if} = 1 \tag{3}$$

$$\sum_{f=1}^{p}\sum_{k=1}^{m}\sum_{s=1}^{w} X_{ijksf} = 1 \tag{4}$$

$$C_i \geq c_{ijks} \tag{5}$$

$$ET_{ijkf} - ST_{ijkf} = \sum_{k=1}^{m}\sum_{s=1}^{w} X_{ijksf} \cdot T_{ijksf} \tag{6}$$

$$ST_{i(j+1)kf} \geq ET_{ijk'f} \tag{7}$$

$$ST_{ijf} + \varepsilon\left(1 - Y_{i'j'ijkf}\right) \geq ET_{i'j'f} \tag{8}$$

$$ST_{ijf} + \varepsilon\left(1 - Z_{i'j'ijsf}\right) \geq ET_{i'j'f} \tag{9}$$

$$ST_{ijf} \geq 0 \tag{10}$$

$$ET_{ijf} \geq 0 \tag{11}$$

$$PE = \sum_{f=1}^{p} \sum_{i=1}^{n} \sum_{j=1}^{v_i} \sum_{k=1}^{m} \sum_{s=1}^{w} X_{ijksf} \cdot T_{ijksf} \cdot Me_k \tag{12}$$

$$IE = \sum_{f=1}^{p} \sum_{i=1}^{n} \sum_{j=1}^{v_i} \sum_{k=1}^{m} \sum_{s=1}^{w} X_{ijksf} \cdot \left(ST_{ijkf} - ET_{i'j'kf}\right) \cdot Re_k \tag{13}$$

$$TEC = PE + IE \tag{14}$$

where Eqs. (1) and (2) are the objectives of minimizing makespan and total energy consumption, respectively. Eq. (3) indicates that the job cannot be processed across factories. Eq. (4) indicates that only one worker operates one machine for each operation. Eq. (5) is the completion time of J_i. Eq. (6) indicates that the processing of each job cannot be interrupted. Eq. (7) denotes the presence of sequence constraints on the operations of the same job. Eq. (8) denotes that the same machine can only process a job at any moment. Eq. (9) indicates that the same worker can only process a job at any moment. Eq. (10) shows that machines and workers can start processing at 0 moment. Eq. (11) denotes that the start time and completion time of any operation are greater than or equal to 0. Eqs. (12)–(14) indicate the total processing energy consumption, total idle energy consumption and total energy consumption, respectively.

4 Q-MOGWO for DFJSP-DRC

4.1 The Canonical MOGWO

In the grey wolf optimizer (GWO), the grey wolf population is composed of four species based on social leadership mechanisms: α wolf, β wolf, δ wolf and ω wolves. In which α, β and δ are the first, second and third head wolves, and the rest of the wolves are ω. The ω wolves obey these head wolves. Hunting in the optimization process is directed by α, β and δ, ω follows head wolves in the pursuit of an optimal solution. The positions of α, β and δ are recorded after each iteration, and ω is guided to update its position accordingly.

MOGWO is built on GWO by adding two components. The first component is an archive that stores non-dominated Pareto optimal solutions acquired thus far. The second component is a leader selection strategy employed to aid in selecting α, β and δ as the leaders within the hunting process from the archive. The conventional MOGWO can be referred to in the literature [32].

4.2 Framework of the Proposed Q-MOGWO

The pseudo-code of Q-MOGWO is described in Algorithm 1. The main steps of Q-MOGWO include four-layer encoding, active decoding based on public idle time, hybrid initialization strategy, wolf pack search strategy and neighborhood structure based on Q-learning. The iteration of Q-MOGWO is as follows. Firstly, the initial population is generated by a hybrid initialization strategy and the head wolves are selected in the population. Secondly, the head wolves lead the evolution of the population, and the evolved population and the initial population are merged by the elite strategy to obtain the external archive and the new generation of population. Finally, the local search strategy based on Q-learning is applied to the external archive, and the external archive is updated by the elite strategy.

Algorithm 1: Q-MOGWO

Input: population size N, external archive length E, number of iterations M, learning rate a, discount factor γ, greedy factor ε, number of training rounds *tr-ep*, number of testing round *te-ep*, maximum number of steps *max-ep*.
Output: optimal Pareto Front (PF)
Initialize **MOGWO:** population size N, external archive length E, number of iterations M
Initialize **RL:** learning rate a, discount factor γ, greedy factor ε, number of training rounds *tr-ep*, number of testing round *te-ep*, maximum number of steps *max-ep*
Set current iteration number t=0. Calculate the fitness of all individuals
While $t \leq M$ **do:**
    Select the head wolves and head wolves lead the pack in hunting
    Update populations using elite strategies with archiving mechanisms
    Select a local search strategy for solutions in the archive using Q-learning
    Perform a local search strategy to generate new solutions
    Update stock states and the Q-table
    Use elite strategies for new solutions and solutions in the archive
    t=t+1
Else
    Output optimal Pareto Front

4.3 Four-Layer Encoding

The feasible solutions for DFJSP-DRC are represented using a four-layer coding scheme, which includes vectors for the operations sequence (OS), factories sequence (FS), machines sequence (MS) and workers sequence (WS). A four-layer encoding scheme of 4 jobs and 3 factories is shown in Fig. 2. J_1 and J_4 have two operations, J_2 and J_3 have three operations. Each factory has three machines and two workers. OS consists of integers from 1 to n, and each integer i corresponds to J_i. The 2 in the fifth position indicates operation O_{22}. The second layer is FS, in which each number represents the processing factory for each operation. The sequence length of FS is the same as the length of OS, and it is clear that J_2 and J_4 are processed in F_1, J_1 is processed in F_2, J_3 is processed in F_3. The MS and WS structure is similar to the FS, with the third number in the MS and WS indicating that O_{41} is processed by machine 1 and worker 1.

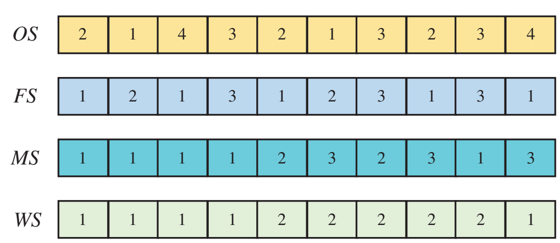

Figure 2: Schematic diagram of four-layer coding

4.4 Active Decoding Based on Public Idle Time

A good decoding strategy not only can rationalize the arrangement of jobs, machines and workers, but also obtain a high-quality scheduling scheme. Based on the literature of Kacem et al. [33], an improved active decoding strategy is devised, with the efficient use of the public idle time of machines and workers as its core. This reduces makespan by arranging processing jobs to the earliest public idle time slot. In the improved active decoding, it is necessary to determine whether the idle time of the current processing machine and the worker overlap and then determine the size of the overlap time and the processing time. When the two conditions are satisfied, the following two cases will appear. Algorithm 2 describes the pseudo-code for two cases.

Algorithm 2: Improved active decoding strategy

Input: total number of processes TP, machines and workers processing state.
Output: start processing time ST_{ij}, end processing time ET_{ij}, machine processing time $[MT_s, MT_e]$, worker processing time $[WT_s, WT_e]$.

For tp=1 to TP:
  $O_{ij}=OS(tp)$; $flag=0$; // *flag* is used to determine whether to perform insertion decoding
  Obtain O_{ij} optional processing machine k and worker s, and the corresponding time
  Processing time for machine k and worker s $[MT_{ks}, MT_{ke}]$, $[WT_{ss}, WT_{se}]$ and the corresponding idle time slot $[AMT_{ks}, AMT_{ke}]$, $[AWT_{ss}, AWT_{se}]$
  Calculate all public idle time slots for machine k and worker s $[T_s, T_e]$
  If $T_e-T_s\geq T_{ijks}$; // Perform improved active decoding
    If $T_s\geq ET_{i(j-1)}$
      $ST_{ij}=T_s$, $ET_{ij}=T_s+T_{ijks}$, flag=1
    Else if $ET_{i(j-1)}\geq T_s$
      $ST_{ij}=ET_{i(j-1)}$, $ET_{ij}=ET_{i(j-1)}+T_{ijks}$, flag=1
  Else
    O_{ij} does not perform improved active decoding
  End if
  Update the earliest ET_{ij} of process O_{ij}, the processing time, public time of machine k and worker s
End for

Note: If there is more than one free time slot, select the min(T_{s0}, T_{s1}, $T_{s2} \ldots T_{sn}$) and compare with $ET_{i(j-1)}$.

Case 1: As shown in Fig. 3, the public idle processing time slot of a machine and a worker is $[T_s, T_e]$, $T_e-T_s\geq T_{ijks}$ and $ET_{i(j-1)}\geq T_s$. In this case, the processing time slot of the operation is denoted as $[ET_{i(j-1)}, ET_{i(j-1)}+T_{ijks}]$.

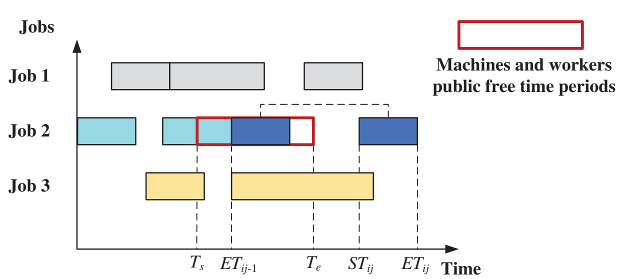

Figure 3: Active decoding case 1

Case 2: As shown in Fig. 4, the public idle processing time slot of a machine and a worker is $[T_s, T_e]$, $T_e-T_s\geq T_{ijks}$ and $T_s\geq ET_{i(j-1)}$. In this case, the processing time slot of the operation is denoted as $[T_s, T_s+T_{ijks}]$.

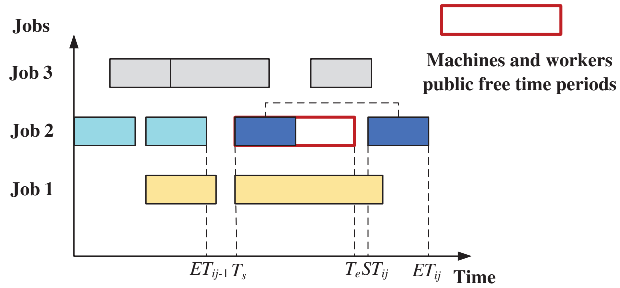

Figure 4: Active decoding case 2

4.5 Hybrid Initialization Strategy

(1) Initialization strategy for the OS. To ensure the diversity and randomness of the population, the initialization coding for OS adopts the positional ascending rule. Firstly, generate a list of basic OS. Secondly, generate a random number ranging from 0 to 1 for each element in the OS. Finally, the OS is rearranged in ascending order of these random numbers to obtain the initial OS.

(2) Initialization strategy for the FS. The following two strategies each account for 50% of the population size. The first strategy prioritizes the assignment to the factories with few jobs. One factory is randomly chosen if there are multiple optional factories for selection. The second strategy involves the random assignment of jobs to a factory.

(3) Initialization strategy for MS and WS. The processing time for the DFJSP-DRC depends on both the machine and the worker. Taking into account the time differences that arise from different operators using the same machine, a principle of machine-worker integration is formulated. First, determine the set of processing machines to operate, and then determine the available processing workers for each machine. For example, the available processing machines for O_{21} of J_2 consist of M_1 and M_2. M_1 and M_2 can be operated by worker W_1, and M_2 can be operated by worker W_2. The set of available machines and workers for O_{21} of J_2 are [(M_1, W_1), (M_2, W_1) (M_2, W_2)]. Each operation has three strategies: randomly selecting a machine and worker, selecting the machine and worker combinations with the shortest processing time, and selecting the machine and worker combinations with the least energy consumption.

4.6 Wolf Pack Search Strategy

MOGWO mimics the grey wolf population predation strategy, utilizing the three head wolves to guide the position update of the population. However, this strategy cannot be applied directly to DFJSP-DRC. Therefore, in Q-MOGWO, the two modified search operators are adopted for global search to ensure the feasibility of the DFJSP-DRC solution. The social leadership mechanism proposed by Lu et al. [34] is used to obtain α, β and δ.

The first search operator comprises improved precedence operation crossover (IPOX) [35] and multiple point crossover (MPX). IPOX is performed for OS, where P_1 and P_2 represent the two paternal chromosomes that undergo crossover to generate offspring C_1 and C_2, as shown in Fig. 5. After the IPOX crossover is performed for the OS, MS and WS adopt MPX. MPX is a process in which multiple crossover points are randomly set up in two somatic chromosomes and then genes are exchanged, as shown in Fig. 6. If the case of infeasible solutions appears, machines and workers are randomly selected.

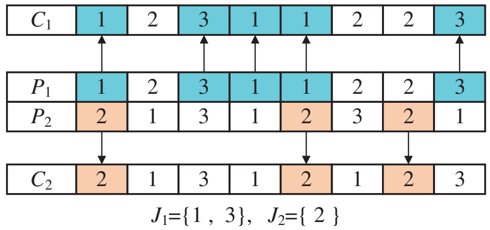

Figure 5: IPOX crossover

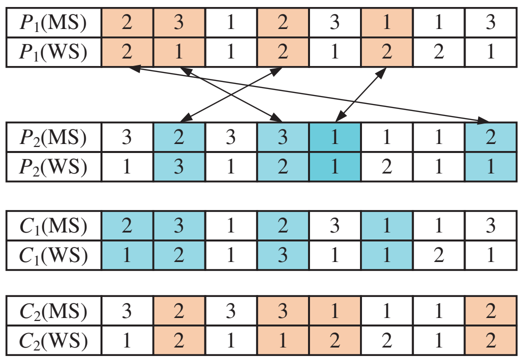

Figure 6: Multi-point intersection of machines and workers

The second search operator is the improved IPOX, which executes the crossover for OS, MS and WS, as shown in Fig. 7. Binding the three together while performing the crossover can avoid infeasible solutions.

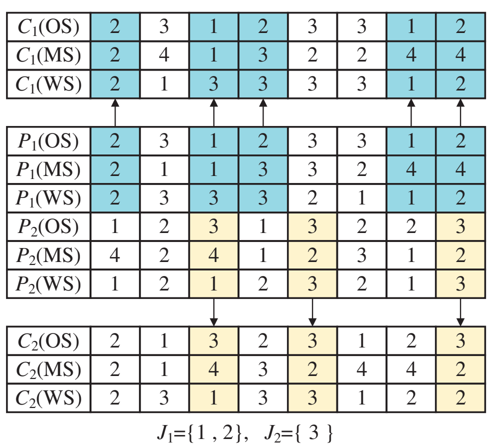

Figure 7: Improved IPOX operator

4.7 Neighborhood Structure Based on Q-Learning

4.7.1 Brief Introduction of Q-Learning

The main components of RL include agent, environment, actions, rewards and states. The agent in the RL algorithm gets as much reward as possible through trial and error of the environment. The agent takes action by its state S_t at time t within the environment, subsequently receiving a reward R_{t+1} and transitioning to state S_{t+1}.

Q-learning is an effective algorithm that improves the solution diversity of the algorithm by choosing appropriate local search operators during iteration. Q-learning is a greedy algorithm where the agent selects the action with the highest Q value to maximize rewards. The agent can fine-tune the disparity between the actual and estimated Q values by computing the difference between them. Learning rate (a) and the discount factor (γ) are both in the range of 0 to 1, as γ approaches 0, the current state influences the Q value, and γ is more concerned with the future state as it approaches 1. The current state s_t can influence the later state $s_{t+1,}$ and r_t is the reward after performing an action a_t. The Q value is updated according to the Eq. (15).

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha\left[r_{t+1} + \gamma maxQ(s_{t+1}, a) - Q(s_t, a_t)\right] \tag{15}$$

4.7.2 Agent and Action Definition

In Q-MOGWO, the PF is the set of optimal solutions, which can reflect the comprehensive ability of the algorithm. Q-learning guides the algorithm to choose the optimal local search strategy. Therefore, the solution set in the external archive acts as an agent to reflect the success of the local search strategy.

4.7.3 State Definition

The state change can give feedback to the agent and determine whether the action performed can improve the overall quantity of the PF. In Q-MOGWO, MOGWO is viewed as the environment. To better construct the state of the environment, the comprehensive performance of the PF, and the other is the degree of excellence degree of the α are taken.

Whether or not the α is good depends on whether the α of the previous generation dominates the α of the current generation. The integrative performance of the PF is calculated by Inverse Generational Distance (IGD) [36]. IGD measures the diversity and convergence of PF. Eqs. (16) and (17) calculate IGD and ΔIGD, respectively:

$$IGD = \frac{\sum_{x \in PF*} dist(x, PF)}{|PF*|} \tag{16}$$

$$\Delta IGD = IGD_i - IGD_{i-1} \tag{17}$$

where $PF*$ is the true frontier of the Pareto solution set, $|PF*|$ represents the total count of elements in $PF*$, meanwhile $dist(x, PF)$ signifies the minimum Euclidean distance between point x and the closest element in PF. IGD_i indicates the IGD value of the i-generation PF.

There are three outcomes of ΔIGD and two outcomes of α dominance in the iterative process, at thus. Six states can be obtained by combining these results: (1) State 1: $\Delta IGD > 0$, α_i does not dominate α_{i-1}; (2) State 2: $\Delta IGD > 0$, α_i dominates α_{i-1}; (3) State 3: $\Delta IGD = 0$, α_i dominates α_{i-1}; (4) State 4: $\Delta IGD = 0$, α_i does not dominate α_{i-1}; (5) State 5: $\Delta IGD < 0$, α_i dominates α_{i-1}; (6) State 6: $\Delta IGD < 0$, α_i does not dominate α_{i-1}.

4.7.4 Reward Definition

Upon performing an action, the agent receives a reward, which may be positive or negative. The definition of reward is as Eq. (18). The chosen action (local search strategy) is rewarded, and the Q-table is updated if PF exhibits superior overall performance; otherwise, the reward is set to 0.

$$Reward = \begin{cases} 2, \Delta IGD < 0; \alpha_i \, dominates \, \alpha_{i-1} \\ 0, \Delta IGD \geq 0; \alpha_{i0} \, does \, not \, dominate \\ 1, other \, states \end{cases} \tag{18}$$

4.7.5 Q-Learning for Neighborhood Structure

Local search strategy is a crucial technique to improve resource utilization, but it consumes a lot of computing resources. Executing the local search strategy randomly leads to a low success rate. However, RL offers selection strategies to guide agents in choosing the local search strategy with the highest likelihood of success.

Based on the literature of Zhang et al. [37], this paper identifies two types of critical factories: one is related to the makespan and the other is related to the maximum energy consumption. For the local search strategy, two local search operators are proposed: (1) Remove a job from the critical factory and insert the job into the factory with the minimum makespan or energy consumption; (2) Reschedule the jobs in the critical factory.

Combining two different local search operators, three local search strategies are proposed. Local search strategy 1: Select the factory with the makspan. Local search strategy 2: Select the factory with the maximum energy consumption. Local search strategy 3: Randomly selected factory. According to the above description, an adaptive local search strategy based on Q-learning(Q-ALS) is designed, and Algorithm 3 provides the corresponding pseudo-code.

Algorithm 3: Q-ALS

Input: external archive P, greed factor ε, learning rate a, discount factor γ, maximum step *max-step*
Output: excellent solution in the updated external archive

```
Initialize parameters and Set current step i=0.
Choose a random action a_i, set a←a_i, calculate the state s_i of MOGWO, set s←s_i.
Q-table(6×3)←0
While i ≤ max-step do:
    Confirm the agent's state s_i
    If rand number < ε
        Select the action a_i with max Q(S_i, A_i)
    Else
        Randomly select an action a_i
    Execute action a_i for MOGWO to update P and get PF
    Calculate IGD_i and ΔIGD
    Get action a_i's reward R(s_i,a_i)
    Calculate the new solution x_new's state s_{i+1}
    Update Q-table
    s_i←s_{i+1}
    i=i+1
```

5 Experimental Results

A series of experimental instances are designed to assess Q-MOGWO's performance. The Q-MOGWO and comparison algorithms are coded in Python on an Intel Core i7 8550 CPU @1.80 GHz and 8G RAM. To be fair, each algorithm collects the results after 20 independent runs and then calculates the average for performance comparison.

MOEA/D [38], MOGWO [32], NSGA-II [39] and memetic algorithm (MA) [40] are chosen to verify the Q-MOGWO effectiveness. Three multi-objective algorithmic measures: IGD, Spread [39] and Hyper Volume (HV) [41] are used to evaluate the obtained Pareto solutions. The IGD formulation is given in Section 4.7.3. The formulas for the other two metrics are as follows:

(1) Spread measures the degree of propagation between the found solutions, and its formula is:

$$Spread = \frac{\sum_{j=1}^{n_o} d_j^e + \sum_{i=1}^{|PF|} \left| d_i - \overline{d} \right|}{\sum_{j=1}^{n_o} d_j^e + |PF| \cdot \overline{d}} \tag{19}$$

in Eq. (19), d_i represents the Euclidean distance between each point in the real PF and its nearest neighbor within the front. $\overline{d}$ is the average of all d_i, the d_j^e denotes the Euclidean distance between the extreme solution of the j-th objective and the boundary solution of the obtained PF. |PF| represents the number of points within the PF, while no stands for the number of objectives.

(2) HV serves as a metric for assessing the overall performance of an algorithm. It quantifies the volume or area within the objective space enclosed by the resulting non-dominant solution set and reference points. The formula of HV is:

$$HV(P, r) = \bigcup_{X \in P}^{P} v(\boldsymbol{X}, r) \tag{20}$$

in Eq. (20), P represents the PF computed by the algorithm, $r = (1, 1)$ is the reference point, and $\boldsymbol{X}$ denotes a normalized non-dominated solution in the PF. The variable ($\boldsymbol{X}$, r) signifies the volume of the hypercube formed by $\boldsymbol{X}$ and r. A higher HV indicates improved convergence and diversification of the algorithm.

5.1 Experimental Instances and Parameters Setting

For there is no specific instance of the DFJSP-DRC, the flexible job shop scheduling problem benchmark [42] is extended to consider production environments with 2, 3, and 4 factories, with the same number of machines and workers in each factory. 45 test instances are generated, and the worker machine information in each factory is shown in the link https://pan.baidu.com/s/1vIwX5MszpleEIm6pQ7XOFw?pwd=zxoi. Worker processing time T_{ijks} is randomly generated within [T_{ij}, T_{ij} +δ_{ij}], where the operation processing time T_{ij} is given by the benchmarking algorithm and $\delta_{ij} \in$ [2,8] [43]. The unit processing energy consumption of each machine ranges from 5 to 10, and the unit standby energy consumption with of each machine ranges from 1 to 5.

The parameter configuration affects the algorithm's performance in solving the problem. Q-MOGWO contains three primary parameters: the length of the external archive (denoted by E), the maximum step (denoted by *max-step*), and the size of the population (denoted by N). Taguchi's experimental approach can systematically assess parameter impact on algorithm performance, facilitating the identification of optimal parameter combinations. Therefore, the Taguchi experiment is used to obtain the optimal combination of the three parameters of Q-MOGWO. Each parameter exhibits three levels, and Table 2 displays the specific parameter values. The $L_9(3^4)$ orthogonal table is employed to conduct experiments based on the designated levels and the number of parameters.

Table 2: Parameters level

Level	*N*	*E*	*Max-step*
1	100	30	100
2	200	50	200
3	300	80	300

Q-MOGWO runs 20 times under each parameter combination to ensure fairness, and the average IGD values from these 10 runs are collected. Experiments are conducted on the Mk-3-01 instance, employing IGD to evaluate parameter combinations, as presented in Table 3. Fig. 8 illustrates the trend chart delineating each parameter level concerning the results outlined in Table 3. It can be observed that the optimal configuration for the parameter setting values is $N = 300$, $E = 80$, and $max\text{-}step = 300$.

Table 3: Orthogonal table

Number	Parameters			IGD
	Max-step	*E*	*N*	
1	1	1	1	72.918
2	1	2	2	53.365
3	1	3	3	42.634
4	2	1	2	33.547
5	2	2	3	31.562
6	2	3	1	45.074
7	3	1	3	24.532
8	3	2	1	50.612
9	3	3	2	25.169
Level 1	56.306	43.666	56.201	
Level 2	36.728	45.180	37.360	
Level 3	33.438	37.626	32.909	
Range	22.868	7.554	23.292	
Rank	2	3	1	

5.2 Effectiveness of the Proposed Strategy

The proposed strategy's effectiveness is validated through experiments on 15 instances. Algorithms Q-MOGWO1, Q-MOGWO2 and Q-MOGWO3 denote the local search strategy for the makespan factory, the maximum energy consumption factory and the randomized factory, respectively. The IGD and Spread values of Q-MOGWO, Q-MOGWO1, Q-MOGWO2 and Q-MOGWO3 are shown in Table 4. The better results are highlighted in bold for each instance. From Table 4, it can be seen that Q-MOGWO has lower IGD values compared with all the other three algorithms. Table 4 shows that Q-MOGWO has lower Spread values than all the other three algorithms. The Spread values correspond precisely to the IGD values. It can be seen clearly that the solutions identified through the RL selective local search strategy demonstrate superior breadth and comprehensive performance compared with those obtained via the deterministic local search strategy, thereby confirming the effectiveness of using RL to select critical factories.

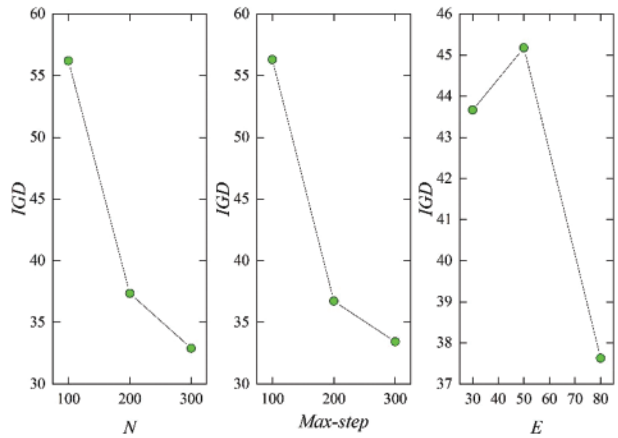

Figure 8: The trend chart of each parameter level

Table 4: The average values of IGD metric and spread metric for 12 instances

	IGD				Spread			
	Q-MOGWO	Q-MOGWO1	Q-MOGWO2	Q-MOGWO3	Q-MOGWO	Q-MOGWO1	Q-MOGWO2	Q-MOGWO3
MK-2-01	**17.792**	55.234	44.184	47.315	**0.852**	1.409	0.900	1.226
MK-2-04	**53.754**	97.492	87.787	101.213	**0.850**	1.157	0.984	1.135
MK-2-09	**120.780**	594.551	331.678	395.348	**0.852**	1.800	0.964	1.166
MK-2-12	**422.012**	480.252	568.373	628.657	**0.931**	1.236	1.482	1.015
MK-2-15	**131.864**	565.720	614.396	641.083	**1.025**	1.046	1.077	1.095
MK-3-01	**13.913**	66.432	70.455	25.947	**0.843**	1.155	1.023	0.935
MK-3-04	**64.624**	111.541	72.101	58.187	**0.983**	1.105	1.008	0.987
MK-3-09	**224.815**	482.553	315.329	189.537	**0.864**	1.415	1.103	1.115
MK-3-12	**241.600**	459.829	439.840	401.252	**0.857**	1.211	0.906	1.035
MK-3-15	**215.525**	372.663	758.286	809.903	**0.820**	1.013	1.007	0.975
MK-4-01	**19.492**	55.706	55.242	44.981	**0.880**	0.905	0.919	0.961
MK-4-04	**33.174**	67.482	106.403	69.193	**1.018**	1.153	1.043	1.093
MK-4-09	**170.921**	219.599	205.870	289.636	**0.819**	0.898	0.998	0.976
MK-4-12	**329.979**	409.506	398.500	443.178	**0.843**	1.141	1.089	1.064
MK-4-15	**336.915**	291.637	489.670	709.550	**0.860**	0.995	0.916	0.955

5.3 Evaluation of the Proposed Q-MOGWO

To further evaluate the effectiveness of Q-MOGWO, four multi-objective optimization algorithms, MOEA/D, MA, NSGA-II and MOGWO, are selected as compared algorithms. Regarding the parameter setting of the compared algorithms, refer to the literature [44], and detailed data is shown in Table 5.

Table 5: The parameter setting of compared algorithms

Algorithm	Parameter setting
MOEA/D	Population_num = 300, generation_num = 300, pc_max = 0.8, pm_max = 0.1, pc_min = 0.4, pm_min = 0.02, T = 10, H = 300
MA	Population_num = 300, generation_num = 300, pc_max = 0.8, pm_max = 0.1, pc_min = 0.4, pm_min = 0.02
NSGA-II	Population_num = 300, generation_num = 300, pc_max = 0.8, pm_max = 0.1, pc_min = 0.4, pm_min = 0.02, external archive length = 80
MOGWO	Population_num = 300, generation_num = 300, external archive length = 80

Tables 6 and 7 present the IGD and HV results. The better results are highlighted in bold for each instance. Table 6 reveals that Q-MOGWO consistently exhibits smaller average IGD values across all instances compared with other algorithms, indicating superior convergence and diversity in the solutions obtained by Q-MOGWO. Table 7 shows the average HV values for 45 instances, with Q-MOGWO consistently showing larger values compared with other algorithms. This proves that Q-MOGWO has better comprehensive performance and can obtain Pareto solutions with better coverage and distribution in the solution space. The IGD and HV results for 45 instances show that Q-MOGWO outperforms the compared algorithms. The boxplots of HV and IGD indicators are given in Figs. 9 and 10 to visualize the excellent performance of Q-MOGWO. Boxplots show that the Pareto solutions from Q-MOGWO consistently outperform those of compared algorithms, exhibiting superior maximum, minimum, median, and quartile values.

Table 6: The average values of IGD metric for 45 instances

	Q-MOGWO	MA	NSGA-II	MOEA/D	MOGWO
MK-2-01	**37.981**	79.778	85.983	66.391	138.772
MK-2-02	**11.457**	35.413	72.422	112.391	82.105
MK-2-03	**72.862**	361.266	328.452	1352.319	656.802
MK-2-04	**25.846**	272.001	140.738	170.961	118.303
MK-2-05	**14.731**	245.971	167.056	424.825	157.336
MK-2-06	**24.462**	101.209	284.131	786.891	342.987
MK-2-07	**32.662**	265.343	125.819	572.337	221.306
MK-2-08	**10.221**	338.599	1392.342	3738.575	1667.969
MK-2-09	**73.879**	263.779	1925.326	4696.575	1998.552
MK-2-10	**4.687**	250.418	1167.949	2786.030	1114.156
MK-2-11	**3.026**	480.000	878.320	1948.067	918.455
MK-2-12	**4.824**	696.596	1247.932	4750.929	1931.157
MK-2-13	**23.590**	361.226	2397.251	7700.891	2636.949
MK-2-14	**3.198**	559.035	3805.628	10950.066	10860.940
MK-2-15	**0.610**	1311.067	3161.046	10340.673	5665.504
MK-3-01	**8.172**	70.151	82.388	62.633	72.585
MK-3-02	**4.471**	39.680	106.610	77.532	124.279

(Continued)

Table 6 (continued)

	Q-MOGWO	MA	NSGA-II	MOEA/D	MOGWO
MK-3-03	**103.530**	233.989	345.571	193.757	674.889
MK-3-04	**40.625**	268.067	169.470	88.779	167.414
MK-3-05	**10.514**	223.988	230.011	183.460	254.496
MK-3-06	**30.177**	101.561	308.865	141.637	441.956
MK-3-07	**62.999**	274.331	372.838	80.358	517.304
MK-3-08	**129.913**	311.758	1341.182	611.138	1960.633
MK-3-09	**27.876**	333.318	1357.865	1185.981	1883.171
MK-3-10	**218.080**	647.407	1280.364	1227.488	1905.381
MK-3-11	**0.809**	410.461	703.069	491.003	720.391
MK-3-12	**0.246**	553.592	1719.839	727.915	2128.851
MK-3-13	**27.351**	392.819	1533.313	1133.513	2903.875
MK-3-14	**0.054**	1037.839	2777.246	1459.384	3462.166
MK-3-15	**10.475**	513.426	3586.269	1919.109	4692.594
MK-4-01	**9.910**	66.681	72.205	113.673	102.103
MK-4-02	**5.134**	40.364	117.182	97.689	148.117
MK-4-03	**65.560**	203.737	554.247	1278.462	619.975
MK-4-04	**43.605**	165.024	86.888	273.664	199.358
MK-4-05	**12.311**	195.211	240.223	465.922	313.644
MK-4-06	**87.032**	97.620	434.910	864.321	513.480
MK-4-07	**13.815**	162.190	246.298	514.142	390.048
MK-4-08	**16.888**	403.572	1165.692	2597.720	1388.311
MK-4-09	**10.787**	406.639	1777.226	4007.228	2050.637
MK-4-10	**5.239**	503.998	836.156	2440.588	957.855
MK-4-11	**34.409**	414.475	604.326	1742.325	739.629
MK-4-12	**24.853**	530.439	1658.010	3358.178	2171.534
MK-4-13	**19.425**	468.675	1948.452	7030.645	2519.904
MK-4-14	**10.042**	913.890	3554.755	8028.150	4735.741
MK-4-15	**13.663**	682.696	4287.531	9764.774	4645.277

Table 7: The average values of the HV metric for 45 instances

	Q-MOGWO	MA	MOEA/D	NSGA-II	MOGWO
MK-2-01	**0.416**	0.299	0.325	0.360	0.402
MK-2-02	**0.265**	0.195	0.177	0.232	0.224
MK-2-03	**0.222**	0.121	0.102	0.196	0.162
MK-2-04	**0.376**	0.210	0.318	0.358	0.330
MK-2-05	**0.335**	0.221	0.181	0.280	0.292
MK-2-06	**0.142**	0.086	0.063	0.116	0.107
MK-2-07	**0.314**	0.187	0.165	0.288	0.263
MK-2-08	**0.293**	0.235	0.085	0.228	0.216

(Continued)

Table 7 (continued)

	Q-MOGWO	MA	MOEA/D	NSGA-II	MOGWO
MK-2-09	**0.267**	0.210	0.044	0.181	0.172
MK-2-10	**0.298**	0.231	0.066	0.209	0.209
MK-2-11	**0.380**	0.297	0.099	0.277	0.269
MK-2-12	**0.287**	0.215	0.066	0.232	0.209
MK-2-13	**0.297**	0.243	0.062	0.218	0.211
MK-2-14	**0.258**	0.190	0.050	0.198	0.167
MK-2-15	**0.284**	0.231	0.062	0.203	0.193
MK-3-01	**0.436**	0.299	0.325	0.360	0.371
MK-3-02	**0.504**	0.438	0.391	0.413	0.394
MK-3-03	**0.279**	0.171	0.212	0.244	0.198
MK-3-04	**0.376**	0.210	0.318	0.350	0.330
MK-3-05	**0.373**	0.235	0.249	0.321	0.319
MK-3-06	**0.182**	0.120	0.151	0.144	0.135
MK-3-07	**0.418**	0.275	0.372	0.368	0.325
MK-3-08	**0.260**	0.175	0.152	0.181	0.172
MK-3-09	**0.301**	0.232	0.179	0.235	0.200
MK-3-10	**0.436**	0.367	0.301	0.312	0.373
MK-3-11	**0.309**	0.178	0.173	0.195	0.196
MK-3-12	**0.252**	0.150	0.146	0.179	0.171
MK-3-13	**0.263**	0.179	0.156	0.218	0.165
MK-3-14	**0.211**	0.112	0.116	0.143	0.142
MK-3-15	**0.226**	0.147	0.122	0.144	0.125
MK-4-01	**0.475**	0.345	0.346	0.416	0.385
MK-4-02	**0.405**	0.330	0.264	0.314	0.294
MK-4-03	**0.336**	0.194	0.206	0.287	0.271
MK-4-04	**0.362**	0.228	0.251	0.354	0.301
MK-4-05	**0.460**	0.348	0.290	0.400	0.369
MK-4-06	**0.160**	0.093	0.088	0.126	0.119
MK-4-07	**0.393**	0.250	0.225	0.343	0.304
MK-4-08	**0.389**	0.297	0.134	0.296	0.297
MK-4-09	**0.387**	0.316	0.165	0.293	0.282
MK-4-10	**0.425**	0.343	0.135	0.326	0.319
MK-4-11	**0.487**	0.386	0.168	0.393	0.385
MK-4-12	**0.377**	0.283	0.136	0.303	0.284
MK-4-13	**0.390**	0.309	0.147	0.319	0.299
MK-4-14	**0.333**	0.248	0.097	0.261	0.250
MK-4-15	**0.364**	0.291	0.116	0.275	0.266

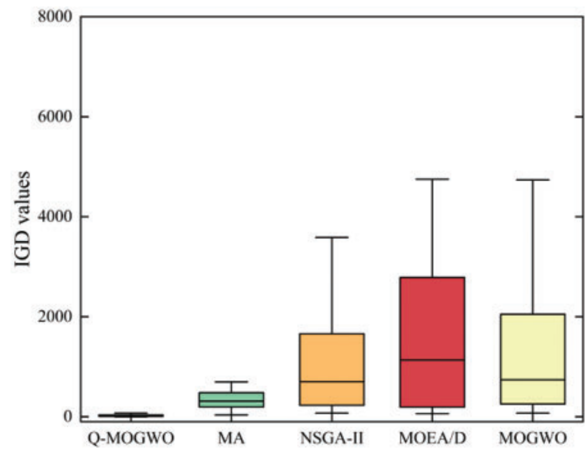

Figure 9: Experimental results of all algorithms for boxplot on the HV values

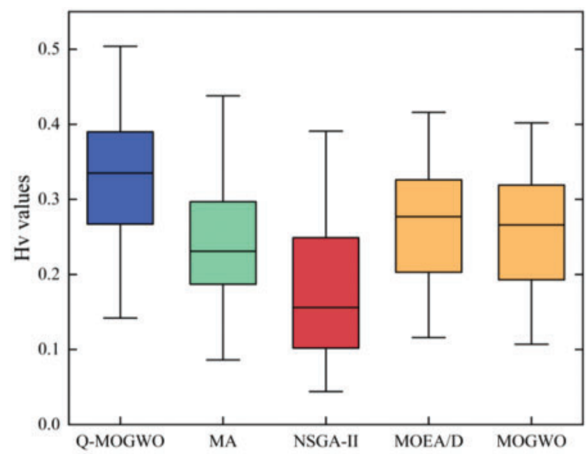

Figure 10: Experimental results of all algorithms for boxplot on the HV values

To visualize the performance of Q-MOGWO, NSGA-II, MOEA/D, MA and MOGWO, 6 instances (Mk-3-01, Mk-3-08, Mk-3-15, Mk-4-01, Mk-4-08, Mk-4-15) with different scales are selected, and the Pareto front obtained from one run of each algorithm for each selected instance is shown in Fig. 11. It can be observed that the Pareto front of Q-MOGWO is closer to the coordinate axis than that of compared algorithms, which indicates that the Pareto front of Q-MOGWO has better quality than that of compared algorithms.

In addition, to further prove the effectiveness of Q-MOGWO, the IGD and HV values in Tables 6 and 7 are analyzed by Friedman's statistical test with 95% confidence intervals, and the results are shown in Table 8. The better results are highlighted in bold for each instance. Table 8 indicates that the mean, standard deviation, minimum and maximum values of IGD and HV for Q-MOGWO surpass those of the compared algorithms. For a significance level of 0.05, the obtained p-value is 0, which proves that the performance of Q-MOGWO is significantly different from that of compared algorithms.

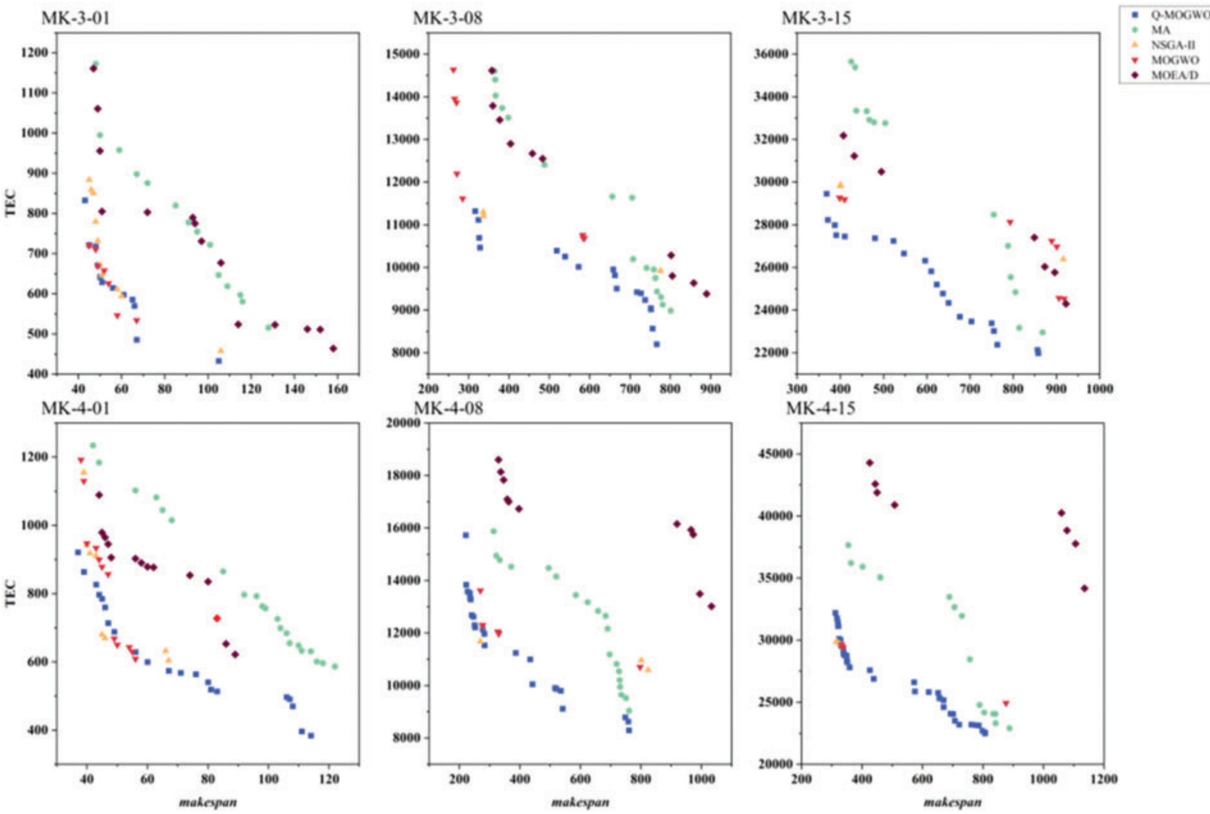

Figure 11: The Pareto fronts of selected instances for Q-MOGWO and compared algorithms

Table 8: Friedman test of IGD and HV on Q-MOGWO and compared algorithms

Metrics	Algorithms	Rank	N	Mean	Std.	Min	Max
IGD	Q-MOGWO	**1.000**	45.000	**30.933**	**40.791**	**0.054**	**218.080**
	MA	2.380	45.000	361.984	267.675	35.413	1311.067
	MOEA/D	3.360	45.000	1126.209	1154.538	72.205	4287.531
	NSGA-II	4.090	45.000	2279.091	2985.190	62.633	10950.070
	MOGWO	4.180	45.000	1598.153	2014.227	72.585	10860.940
	p-value	0.000					
HV	Algorithms	Rank	N	Mean	Std.	Min	Max
	Q-MOGWO	**5.000**	45.000	**0.333**	0.087	**0.142**	**0.504**
	MA	2.480	45.000	0.238	0.080	0.086	0.438
	MOEA/D	1.380	45.000	0.178	**0.095**	0.044	0.391
	NSGA-II	3.500	45.000	0.269	0.082	0.116	0.416
	MOGWO	2.640	45.000	0.255	0.083	0.107	0.402
	p-value	0.000					

The experimental results show that Q-MOGWO outperforms the compared algorithms. The main reasons are as follows: (1) The hybrid population initialization strategy generates high-quality initial population and enhances global exploration of Q-MOGWO; (2) The active decoding strategy that effectively uses the public idle time of machines and workers decodes solutions to high-quality scheduling schemes; (3) According to the characteristics of the problem, two kinds of wolf predation strategies are designed to effectively explore the search space of solutions and increase the population diversity; (4) The Q-learning-based local search strategy enhances the local search capability and efficiency of Q-MOGWO, leading to accelerated convergence.

6 Conclusions and Future Work

In this paper, Q-MOGWO is proposed to solve the DFJSP-DRC with the objectives of minimizing makespan and total energy consumption. In Q-MOGWO, three scheduling rules are used to generate high-quality initial solutions, and an active decoding strategy converts solutions into reasonable scheduling schemes. Two predation strategies are designed to explore the unknown regions of solution space in the wolf predation phase. To improve the local search capability of Q-MOGWO, two kinds of neighborhood structures based on critical factories are designed. Through the effectiveness analysis, it can be found that the factory selection based on Q-learning significantly enhances the performance of Q-MOGWO. Especially when solving large-scale problems, Q-MOGWO is superior to the compared algorithms and has better non-dominated solutions.

The problem studied in this paper does not consider the impact of dynamic events on the scheduling schemes. Although worker resource is introduced, worker fatigue is not considered. Therefore, in future work, dynamic events such as machine failure and emergency order insertion will be considered, and worker fatigue will be introduced into the optimization objectives. In addition, some learning mechanisms will be introduced into the framework of Q-MOGWO to obtain stronger adaptability.

Acknowledgement: Thanks to three anonymous reviewers and the editors of this journal for providing valuable suggestions for the paper.

Funding Statement: This work was supported by the Natural Science Foundation of Anhui Province (Grant Number 2208085MG181); the Science Research Project of Higher Education Institutions in Anhui Province, Philosophy and Social Sciences (Grant Number 2023AH051063); and the Open Fund of Key Laboratory of Anhui Higher Education Institutes (Grant Number CS2021-ZD01).

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: Hongliang Zhang, Yi Chen and Gongjie Xu; data collection: Yuteng Zhang; analysis and interpretation of results: Hongliang Zhang, Yi Chen and Gongjie Xu; draft manuscript preparation: Hongliang Zhang, Yi Chen. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: The data generated in this paper are available from the corresponding author on reasonable request.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Yu F, Lu C, Zhou J, Yin L. Mathematical model and knowledge-based iterated greedy algorithm for distributed assembly hybrid flow shop scheduling problem with dual-resource constraints. Expert Syst Appl. 2024;239:122434. doi:10.1016/j.eswa.2023.122434.
2. He L, Chiong R, Li W, Dhakal S, Cao Y, Zhang Y. Multiobjective optimization of energy-efficient job-shop scheduling with dynamic reference point-based fuzzy relative entropy. IEEE Trans Industr Inform. 2022;18(1):600–10. doi:10.1109/tii.2021.3056425.
3. Wang GG, Gao D, Pedrycz W. Solving multiobjective fuzzy job-shop scheduling problem by a hybrid adaptive differential evolution algorithm. IEEE Trans Industr Inform. 2022;18(12):8519–28. doi:10.1109/tii.2022.3165636.
4. Meng LL, Zhang CY, Ren YP, Zhang B, Lv C. Mixed-integer linear programming and constraint programming formulations for solving distributed flexible job shop scheduling problem. Comput Ind Eng. 2020;142:106347. doi:10.1016/j.cie.2020.106347.
5. Sang YW, Tan JP. Intelligent factory many-objective distributed flexible job shop collaborative scheduling method. Comput Ind Eng. 2022;164:107884. doi:10.1016/j.cie.2021.107884.
6. Tang H, Fang B, Liu R, Li Y, Guo S. A hybrid teaching and learning-based optimization algorithm for distributed sand casting job-shop scheduling problem. Appl Soft Comput. 2022;120:108694. doi:10.1016/j.asoc.2022.108694.
7. Zhang ZQ, Wu FC, Qian B, Hu R, Wang L, Jin HP. A Q-learning-based hyper-heuristic evolutionary algorithm for the distributed flexible job-shop scheduling problem with crane transportation. Expert Syst Appl. 2023;234:121050. doi:10.1016/j.eswa.2023.121050.
8. Luo Q, Deng QW, Gong GL, Zhang LK, Han WW, Li KX. An efficient memetic algorithm for distributed flexible job shop scheduling problem with transfers. Expert Syst Appl. 2020;160:113721. doi:10.1016/j.eswa.2020.113721.
9. Du Y, Li JQ, Luo C, Meng LL. A hybrid estimation of distribution algorithm for distributed flexible job shop scheduling with crane transportations. Swarm Evol Comput. 2021;62:100861. doi:10.1016/j.swevo.2021.100861.
10. Xu WX, Hu YW, Luo W, Wang L, Wu R. A multi-objective scheduling method for distributed and flexible job shop based on hybrid genetic algorithm and tabu search considering operation outsourcing and carbon emission. Comput Ind Eng. 2021;157:107318. doi:10.1016/j.cie.2021.107318.
11. Li R, Gong WY, Wang L, Lu C, Jiang SN. Two-stage knowledge-driven evolutionary algorithm for distributed green flexible job shop scheduling with type-2 fuzzy processing time. Swarm Evol Comput. 2022b;74:101139. doi:10.1016/j.swevo.2022.101139.
12. Gong GL, Chiong R, Deng QW, Gong X. A hybrid artificial bee colony algorithm for flexible job shop scheduling with worker flexibility. Int J Prod Res. 2020;58(14):4406–20. doi:10.1080/00207543.2019.1653504.
13. Tan WH, Yuan XF, Wang JL, Zhang XZ. A fatigue-conscious dual resource constrained flexible job shop scheduling problem by enhanced NSGA-II: An application from casting workshop. Comput Ind Eng. 2021;160:107557. doi:10.1016/j.cie.2021.107557.
14. Zhao P, Zhang H, Tang HT, Feng Y, Yin W. Research on flexible job-shop scheduling problem in green sustainable manufacturing based on learning effect. J Intell Manuf. 2022;33(6):1725–46. doi:10.1007/s10845-020-01713-8.
15. Shi JX, Chen MZ, Ma YM, Qiao F. A new boredom-aware dual-resource constrained flexible job shop scheduling problem using a two-stage multi-objective particle swarm optimization algorithm. Inf Sci. 2023;643:119141. doi:10.1016/j.ins.2023.119141.
16. Luo Q, Deng QW, Gong GL, Guo X, Liu X. A distributed flexible job shop scheduling problem considering worker arrangement using an improved memetic algorithm. Expert Syst Appl. 2022;207:117984. doi:10.1016/j.eswa.2022.117984.

17. Meng LL, Zhang CY, Zhang B, Ren YP. Mathematical modeling and optimization of energy-conscious flexible job shop scheduling problem with worker flexibility. IEEE Access. 2019;7:68043–59. doi:10.1109/access.2019.2916468.
18. He Z, Tang B, Luan F. An improved african vulture optimization algorithm for dual-resource constrained multi-objective flexible job shop scheduling problems. Sensors. 2023;23(1):90. doi:10.3390/s23010090.
19. Lin CS, Lee IL, Wu MC. Merits of using chromosome representations and shadow chromosomes in genetic algorithms for solving scheduling problems. Robot Comput Integr Manuf. 2019;58:196–207. doi:10.1016/j.rcim.2019.01.005.
20. Li XY, Xie J, Ma QJ, Gao L, Li PG. Improved gray wolf optimizer for distributed flexible job shop scheduling problem. Sci China Technol Sci. 2022;65(9):2105–15. doi:10.1007/s11431-022-2096-6.
21. Xie J, Li X, Gao L, Gui L. A hybrid genetic tabu search algorithm for distributed flexible job shop scheduling problems. J Manuf Syst. 2023;71:82–94. doi:10.1016/j.jmsy.2023.09.002.
22. Li R, Gong WY, Wang L, Lu C, Zhuang XY. Surprisingly popular-based adaptive memetic algorithm for energy-efficient distributed flexible job shop scheduling. IEEE Trans Cybern. 2023;53(12):8013–23. doi:10.1109/tcyb.2023.3280175.
23. Zhu N, Gong GL, Lu D, Huang D, Peng NT, Qi H. An effective reformative memetic algorithm for distributed flexible job-shop scheduling problem with order cancellation. Expert Syst Appl. 2024;237:121205. doi:10.1016/j.eswa.2023.121205.
24. Karimi-Mamaghana M, Mohammadi M, Meyer P, Karimi-Mamaghanb AM, Talbi E-G. Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: A state-of-the-art. Eur J Oper Res. 2022;296:393–422. doi:10.1016/j.ejor.2021.04.032.
25. Cao ZC, Lin CR, Zhou MC, Huang R. Scheduling semiconductor testing facility by using cuckoo search algorithm with reinforcement learning and surrogate modeling. IEEE Trans Autom Sci Eng. 2019;16(2):825–37. doi:10.1109/tase.2018.2862380.
26. Chen RH, Yang B, Li S, Wang SL. A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem. Comput Ind Eng. 2020;149:106778. doi:10.1016/j.cie.2020.106778.
27. Cao ZC, Lin CR, Zhou MC. A knowledge-based cuckoo search algorithm to schedule a flexible job shop with sequencing flexibility. IEEE Trans Autom Sci Eng. 2021;18(1):56–69. doi:10.1109/tase.2019.2945717.
28. Li R, Gong W, Lu C. A reinforcement learning based RMOEA/D for bi-objective fuzzy flexible job shop scheduling. Expert Syst Appl. 2022;203:117380. doi:10.1016/j.eswa.2022.117380.
29. Li HX, Gao KZ, Duan PY, Li JQ, Zhang L. An improved artificial bee colony algorithm with Q-learning for solving permutation flow shop scheduling problems. IEEE Trans Syst Man Cybern. 2023;53(5):2684–93. doi:10.1109/tsmc.2022.3219380.
30. Lu C, Gao L, Pan Q, Li X, Zheng J. A multi-objective cellular grey wolf optimizer for hybrid flowshop scheduling problem considering noise pollution. Appl Soft Comput. 2019;75:728–49. doi:10.1016/j.asoc.2018.11.043.
31. Lu C, Xiao S, Li X, Gao L. An effective multi-objective discrete grey wolf optimizer for a real-world scheduling problem in welding production. Adv Eng Softw. 2016;99:161–76. doi:10.1016/j.advengsoft.2016.06.004.
32. Mirjalili S, Saremi S, Mirjalili SM, Coelho LdS. Multi-objective grey wolf optimizer: A novel algorithm for multi-criterion optimization. Expert Syst Appl. 2016;47:106–19. doi:10.1016/j.eswa.2015.10.039.
33. Kacem I, Hammadi S, Borne P. Pareto-optimality approach for flexible job-shop scheduling problems: Hybridization of evolutionary algorithms and fuzzy logic. Math Comput Simul. 2002;60(3):245–76. doi:10.1016/S0378-4754(02)00019-8.
34. Lu C, Gao L, Li XY, Xiao SQ. A hybrid multi-objective grey wolf optimizer for dynamic scheduling in a real-world welding industry. Eng Appl Artif Intell. 2017;57:61–79. doi:10.1016/j.engappai.2016.10.013.

35. Zhang GH, Shao XY, Li PG, Gao L. An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. Comput Ind Eng. 2009;56(4):1309–18. doi:10.1016/j.cie.2008.07.021.
36. Czyzżak P, Jaszkiewicz A. Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. J Multi-Criteria Dec. 1998;7(1):34–47. doi:10.1002/(SICI)1099-1360(199801)7:13.0.CO;2-6.
37. Zhang GH, Xing KY, Cao F. Discrete differential evolution algorithm for distributed blocking flowshop scheduling with makespan criterion. Eng Appl Artif Intell. 2018;76:96–107. doi:10.1016/j.engappai.2018.09.005.
38. Zhang QF, Li H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evol Comput. 2007;11(6):712–31. doi:10.1109/tevc.2007.892759.
39. Kalyanmoy D, Associate M, Sameer A, Meyarivan T. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput. 2002;6(2):182–97. doi:10.1109/4235.996017.
40. Mencía R, Sierra MR, Mencía C, Varela R. Memetic algorithms for the job shop scheduling problem with operators. Appl Soft Comput. 2015;34:94–105. doi:10.1016/j.asoc.2015.05.004.
41. Zitzler E, Thiele L. Multiobjective evolutionary algorithms a comparative case study and the strength Pareto approach. IEEE Trans Evol Comput. 1999;3(4):257–71. doi:10.1109/4235.797969.
42. Brandimarte P. Routing and scheduling in a flexible job shop by tabu search. Ann Oper Res. 1993;41:157–83. doi:10.1007/BF02023073.
43. Cao XZ, Yang ZH. An improved genetic algorithm for dual-resource constrained flexible job shop scheduling. In: Fourth Int Conf Intell Comput Technol Automat. Shenzhen, China; 2011. p. 42–5. doi:10.1109/ICICTA.2011.18.
44. Yuan MH, Li YD, Zhang LZ, Pei FQ. Research on intelligent workshop resource scheduling method based on improved NSGA-II algorithm. Robot Comput Integr Manuf. 2021;71:102141. doi:10.1016/j.rcim.2021.102141.