Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

A dynamic scheduling method with Conv-Dueling and generalized representation based on reinforcement learning

Minghao Xia[a], Haibin Liu[a*], Mingfei Li[a] and Long Wang[a]

[a]*Faculty of Materials and Manufacturing, Beijing University of Technology, Beijing 100124, China*

C H R O N I C L E

Article history:
Received March 1 2023
Received in Revised Format May 3 2023
Accepted June 29 2023
Available online June, 29 2023

Keywords:
Scheduling problem
Deep reinforcement learning
Dynamic disturbance
Conv-dueling network model

A B S T R A C T

In modern industrial manufacturing, there are uncertain dynamic disturbances between processing machines and jobs which will disrupt the original production plan. This research focuses on dynamic multi-objective flexible scheduling problems such as the multi-constraint relationship among machines, jobs, and uncertain disturbance events. The possible disturbance events include job insertion, machine breakdown, and processing time change. The paper proposes a conv-dueling network model, a multidimensional state representation of the job processing information, and multiple scheduling objectives for minimizing makespan and delay time, while maximizing the completion punctuality rate. We design a multidimensional state space that includes job and machine processing information, an efficient and complete intelligent agent scheduling action space, and a compound scheduling reward function that combines the main task and the branch task. The unsupervised training of the network model utilizes the dueling-double-deep Q-network (D3QN) algorithm. Finally, based on the multi-constraint and multi-disturbance production environment information, the multidimensional state representation matrix of the job is used as input and the optimal scheduling rules are output after the feature extraction of the conv-dueling network model and decision making. This study carries out simulation experiments on 50 test cases. The results show the proposed conv-dueling network model can quickly converge for DQN, DDQN, and D3QN algorithms, and has good stability and universality. The experimental results indicate that the scheduling algorithm proposed in this paper outperforms DQN, DDQN, and single scheduling algorithms in all three scheduling objectives. It also demonstrates high robustness and excellent comprehensive scheduling performance.

1. Introduction

The manufacturing industry is undergoing significant transformation due to the rapid advancements in science and technology. Traditional manufacturing is facing a crucial need for change. In a typical workshop scheduling problem, there is a single processing machine present (Brucker & Schlie, 1990). The primary objective of the scheduling problem is to determine the optimal sequence of job processing, aiming to minimize the makespan and expedite the scheduling tasks. Ouelhadj and Petrovic (2009) introduced the Flexible Job Shop Scheduling Problem (FJSP). as the flexible job-shop scheduling problem (FJSP), which includes multiple processing machines, multiple jobs and operations, and different operations which have different processing times. Gao et al. (2005) determined when each process should be dispatched to which machine for production and processing by studying various scheduling algorithms. To obtain better benefits in actual workshop production and processing, there has been a growing body of research on FJSP. Recently, most of the proposed solutions for the multi-objective dynamic flexible job-shop scheduling problem (MFJSP) in production workshops have primarily focused on static production environments. Here, the processing information, such as machines and jobs in the workshop, is completely known and the current solutions overlook the numerous disruptive factors in real production processes. However, in actual production scenarios, dynamic events occur, such as job insertion, machine breakdown and unavoidable disturbance factors (Garey et al.

* Corresponding author
E-mail: liuhb@bjut.edu.cn (H Liu)
ISSN 1923-2934 (Online) - ISSN 1923-2926 (Print)
2023 Growing Science Ltd.
doi: 10.5267/j.ijiec.2023.6.003

1976). These random disturbances lead to a serious deviation from the expected results when implementing the original static scheduling scheme, which greatly reduces the punctuality rate of task completion and production efficiency. The dynamic multi-objective flexible job scheduling problem (DMFJSP) aims to optimize scheduling for fast completion, on-time delivery, and minimal delays. The investigation of dynamic optimal scheduling solutions holds significant importance for the production and processing operations of modern manufacturing industries.

Current research shows that task scheduling algorithms can be roughly divided into three categories: single scheduling rule algorithms, metaheuristic scheduling algorithms, and neural network-based scheduling algorithms. The scheduling rule algorithms (Chao & LaPaugh, 1993, Davis & Wellings, 1995, Yang & Yan, 2007, Lee et al., 2011) include FIFO (First In First Out), round robin, etc. These scheduling rules respond to disturbances in the shortest time to complete the scheduling. However, not only can they not achieve the global optimal goal, but they also cannot guarantee even a local optimal state, and different scheduling scenarios require different scheduling rules. Metaheuristic algorithms, such as proposed by Wei and Zhao (2004), commonly divide dynamic scheduling problems into static sub-problems. These sub-problems are then optimized using intelligent algorithms such as genetic algorithms (Gonçalves et al., 2005, Rahmani et al., 2019, Zhang et al., 2020b) and particle swarm scheduling algorithms (Lin et al., 2010, Jianfang et al., 2014, Zhao et al., 2019) are utilized to address the rescheduling point and efficiently complete scheduling tasks. Although this can find solutions to the scheduling problem better than a single scheduling rule algorithm, it takes a long time and cannot meet the requirements of punctuality and timeliness in the production process and has poor correlation with dynamic scheduling problems.

There has been an increasing emphasis among scholars on task scheduling research that incorporates neural networks. This takes advantage of DRL in machine learning, improves the robustness of intelligent scheduling, and completes scheduling tasks efficiently. In the face of multiple processing information constraints such as jobs, machines, operations, and processing time, as well as uncertain disturbance events, the agent should comprehensively utilize the current production status information to make optimal scheduling decisions. To address the limitations of traditional machine learning in state and action space dimensions, deep reinforcement learning is employed to tackle complex control problems, offering a promising approach for DMFJSP. (Zhang et al., 2020a). Waschneck et al. (2018) used DQN to address the workshop production scheduling problem and employed a collaborative neural network scheduling algorithm. Dai et al. (2017) proposed a method using graph neural networks to solve scheduling problems. In this approach, a greedy policy is employed to handle the scheduling process and determine the actions to be taken, effectively addressing the scheduling problem. Liu et al. (2022) considered that in real-world workshop production processes, there is a need to enhance the ability to handle random disruptive events in order to meet requirements such as flexible manufacturing and rapid responsiveness. Chang et al. (2022) proposed a novel scheduling algorithm to address the dynamic disruption problem caused by the insertion of new jobs in the production process. However, limited research has addressed the dynamic disturbance events that arise in real production environments under uncertain conditions. Furthermore, few studies have taken into account the diverse objectives involved in production scheduling when tackling the disturbance problem. Therefore, effectively accomplishing the scheduling task while addressing these disturbances has been largely overlooked.

To better align with the actual workshop production process and make a valuable contribution to the research on future smart factories, this study focuses on a dynamic multi-objective flexible scheduling workshop problem. This problem incorporates three dynamic events and three specific objectives. The disturbances are job insertion, machine breakdown, and processing time change. The three objectives are makespan, delay time, and completion punctuality rate. We propose a dynamic multi-objective intelligent scheduling algorithm based on D3QN and conv-dueling network models.

In this paper, the DMFJSP is addressed by employing a deep reinforcement learning algorithm, taking into consideration the criteria of efficiency, punctuality, and low-latency scheduling. The main work is outlined as follows: (i) We propose to use a multidimensional matrix containing job and machine state information as a state representation, which can express fully the information on which the machine performs actions. This is conducive to the rapid training of the neural network and achieving better convergence results, making it easier for the machine to make optimal action decisions. (ii) A comprehensive scheduling reward function is proposed, which combines the main reward and the branch reward, to guide the neural network in learning the globally optimal scheduling strategy. (iii) A conv-dueling network model is proposed for selecting optimal scheduling rules at various rescheduling decision points. This model takes a multidimensional state matrix as input and generates the value of scheduling rules as output. By leveraging this approach, the model effectively determines the most efficient scheduling strategy for different rescheduling decision points. The network model consists of three parts: a feature extraction network, a state network, and an advantage network to achieve global optimal scheduling. The network model can be optimized effectively in both static and dynamic cases. (iv) Numerical experiments conducted under different workshop scheduling environment configurations demonstrate that the proposed conv-dueling scheduling algorithm based on D3QN outperforms other single scheduling rules as well as scheduling algorithms based on DQN and DDQN in terms of overall scheduling performance.

The remaining sections of this paper are outlined as follows. Section II briefly reviews heuristic and deep reinforcement learning scheduling methods. The third part introduces the principles of Q learning, DQN, and D3QN. The fourth part establishes the mathematical model of dynamic scheduling. The fifth part provides the specific content of the scheduling environment, network model, and deep reinforcement learning D3QN algorithm training. Section VI gives the experimental details and experimental results. Finally, the seventh part draws the conclusions. Table 1 presents a list of the key abbreviations utilized in this study.

Table 1
Key abbreviations in this study

Abbreviation	Description
JSP	Job-Shop Scheduling Problem
DJSP	Dynamic Job-Shop Scheduling Problem
FJSP	Flexible Job-Shop Scheduling Problem
MFJSP	Multi-objective Flexible Job-Shop Scheduling Problem
DFJSP	Dynamic Flexible Job-Shop Scheduling Problem
DMFJSP	Dynamic Multi-objective Flexible Job-Shop Scheduling Problem
DRL	Deep Reinforcement Learning
DQN	Deep Q-learning Network
Conv-Dueling	Convolutional Dueling Double Deep Q-learning Network

2. Literature review

The primary focus of this paper is to address the material scheduling problem in flexible workshops. To tackle this issue, we propose a dynamic scheduling method with Conv-Dueling and generalized representation based on reinforcement learning. We provide a brief overview of flexible workshop scheduling and the application of deep reinforcement learning in intelligent material production scheduling. The Flexible Job Shop Scheduling Problem (FJSP) is recognized as a more complex NP-hard problem compared to the Job Shop Scheduling Problem (JSP). Previous studies on reinforcement learning in the context of FJSP are summarized in Table 2. The table reveals that the consideration of dynamic random disturbance events and objectives in FJSP is quite extensive. The DMFJSP investigated in this paper is similarly complex.

Reinforcement learning captures the characteristics of sequential decision-making in job-shop scheduling problems through state values or action values. In the realm of reinforcement learning, the agent learns iteratively while interacting with the environment, enabling autonomous decision-making. This approach enables high-quality, dynamic, and rapid decision-making for complex problems. The repair-based scheduler of Wei and Dietterich (1995) starts from a critical path plan and repairs constraint violations incrementally with the goal of finding a short conflict-free plan. Aydin and Ztemel (2000) proposed a novel network training method aimed at enabling the agent to generate more autonomous scheduling plans. Wang and Usher(2004) conducted a simulation experiment designed to investigate the problem of selecting scheduling rules for a single production equipment. Yang and Yan (2007) proposed a scheduling algorithm as a solution to address the limitations of a single scheduling rule's local capability. Additionally, an adaptive scheduling control strategy was introduced to tackle the scheduling problem. Wei and Zhao (2004) proposed the concept of the estimated mean late rate of work and completed the scheduling through Q-learning algorithm training in the learning phase. The scheduling method proposed by Shahrabi et al. (2017) utilizes variable neighborhood search and incorporates reinforcement learning. This integration aims to address dynamic job-shop scheduling problems while simultaneously considering factors such as job insertions and machine failures. Qu et al. (2016) described the process of dealing with complex scheduling problems and designed a scheduling environment. Subsequently, they proposed a scheduling algorithm that can adaptively update the production plan based on real-time environmental conditions and random perturbation events during the execution process. Wang (2020) utilized multi-agent technology to address the dynamics and uncertainties of the production environment in the processing workshop. Additionally, it addresses the issue of random perturbations in the production environment of the manufacturing system. The system designed by Wang and Yan (2016) utilizes adaptive workshop scheduling rules based on reinforcement learning states and implements an interoperable knowledge scheduling system.

Traditional reinforcement learning is used to solve small-scale problems. The multidimensional state space often faced in real reinforcement learning task scheduling processes has infinitely many state characteristics and multi-constraint factors need to be considered, such as the remaining time of job processing and relative completion time which are all continuous values. Deep reinforcement learning Mnih et al. (2015) merged deep learning with RL, enabling comprehensive learning from perception to action in a single framework. Deep reinforcement learning leverages sensory input, such as vision, to directly generate actions, eliminating the need for manual feature extraction. The primary objective of deep reinforcement learning is to utilize deep neural networks to learn the environment transition function, enabling machines to learn and perform complex decision-making tasks through interaction with the environment.

Shiue et al. (2018) introduced a novel approach for incrementally maintaining a knowledge base, and proposed to use the MDR mechanism for reinforcement learning-based RTS by combining two mechanisms. Shi et al. (2020) developed a workshop scheduling simulation environment for deep reinforcement learning, which addressed the disruptions between early transfer time and the arrival of new orders. They applied intelligent scheduling algorithms to reentrant automated production lines. Han and Yang (2020) proposed an approach of disjunctive graph scheduling combined with the flexible advantages of a deep reinforcement learning convolutional neural network (CNN). This study directly learns behavioral policies from the input manufacturing state. Liu et al. (2021) combined deep reinforcement learning and designed a workshop scheduling algorithm to quickly and efficiently accomplish scheduling tasks by optimizing scheduling, thereby reducing the completion time in the production process. Wang et al. (2021) formalized the equipment and job constraint problem in workshop scheduling as a problem of selecting the processing sequence. He took into account the structure of JSSP by representing it with a graph and proposed a framework that utilizes a graph neural network to learn. Zhang et al. (2022) addressed the issue of random interference in a large-scale workshop scheduling scenario by integrating an intelligent agent's greedy strategy from deep reinforcement learning. They proposed a multi-agent manufacturing system and enhanced its overall scheduling performance. Tassel et al. (2021) introduced an efficient workshop scheduling environment to address problems that cannot be solved by combinatorial optimization within a given time limit. They designed a reward function that better guides the intelligent agent to learn how to complete scheduling tasks. Luo (2020) studied the production workshop scheduling problem with new job insertion. They extracted seven general state representation functions to simulate the production environment and trained a neural network using these functions. Additionally, they incorporated an improvement called soft target weight update into their approach. Burggräf et al. (2022) utilized the actor-critic algorithm, interpreting the production system as a multi-agent system. The work proposed in (Song et al., 2022) designed a graph neural network to train the entire scheduling process, utilizing graphs to represent states and simulate the relationships between jobs and equipment in the workshop.

Table 2
Current research based on Q learning algorithms

System	Symbol	Algorithm	Events	Objectives
Wei (2004)	JSP	Q-learning	\	Average job delay rate
Qu (2016)	DJSP	Q-learning	1	Makespan
Wang (2018)	DMJSP	Deep Q-learning	1	Earliness punishment Tardiness punishment
Waschneck (2018)	DFJSP	Deep Q-learning	1	Uptime utilization
Altenmüller (2020)	DFJSP	Deep Q-learning	2	Count of violation events
Luo (2021)	DMFJSP	Deep Q-learning	1	Total tardiness Average machine utilization rate
OURS (2023)	**DMFJSP**	**Deep Q-learning**	**3**	**Makespan** **Job completion on time rate** **Total tardiness**

3. Principle on Q-learning and deep Q-learning

3.1 Reinforcement Learning

Reinforcement Learning (RL) is represented as having 5-tuples denoted as (S, A, P, γ, R) follows a specific strategy π to interact with the surrounding environment. The RL agent's objective is to maximize the cumulative return rewards. This process entails selecting actions in a given state and subsequently adhering to a specific policy. The process is defined by Eq. (1).

$$Q_{\pi^*}(s,a) = \max_{\pi} E\,[\,r_t + \gamma\, r_{t+1} + \gamma\, 2\, r_{t+2} + \cdots \mid s_t = s, a_t = a, \pi] \tag{0}$$

Here, discount factor is a parameter that determines the relative significance of short-term rewards compared to cumulative return rewards. The Q function is commonly denoted as $Q_\pi(s,a)$. Bellman (1957) proved that the Bellman optimality equation can be used to obtain the optimal value of the action-value function, which is represented by formula (2):

$$Q_{\pi^*}(s,a) = \sum_{s'} p(s'|s,a)\left[r(s,a,s') + \gamma \max_{a'} Q_{\pi^*}(s',a')\right] \tag{2}$$

By applying the Bellman optimality equation, we can derive a standard Q-learning algorithm, as described in the work of (Sutton & Barto, 2018).

3.2 DDQN and DDDQN

Deep Neural Networks (DNNs) have made remarkable advancements in various artificial intelligence tasks, primarily attributed to their remarkable data learning capability and general approximation ability (Silver et al., 2016). One intriguing approach is the integration of Reinforcement Learning (RL) with DNNs. Double Deep Q-Networks (DDQNs) are widely utilized for addressing high-dimensional problems. Both architectures comprise two DNN structures; however, the distinction lies in the fact that DQN employs identical network structures and incorporates a stable target to mitigate the explosive dimensionality calculation of an unstable target. Nevertheless, during the maximization operation, the agent often tends to favor higher estimates that exceed the actual values.

This paper primarily concentrates on the material scheduling problem in flexible workshops and presents a deep reinforcement learning approach to tackle issue. We utilized an advanced framework called D3QN algorithm for an intelligent material scheduling model, which combines the benefits of DDQN and dueling DQN. The framework consists of two dueling neural networks with identical structures. As mentioned earlier, dual network structure offers the advantage of providing stable objectives to mitigate the computational explosion associated with non-stationary objectives. Fig. 1. illustrates the architecture of the dueling network. The total value function can be written as in Eq. (3) (Wang et al., 2016).

$$Q(s,a;\,\theta_t,\alpha,\beta) = V(s;\,\theta_t,\alpha) + A(s,a;\,\theta_t,\beta) \tag{2}$$

However, in practical engineering applications, there is a need to enhance the identification of this function. To achieve this, it is beneficial to focus on the advantage function, allowing us to express the value function typically as Eq. (4) (Zhang et al., 2021).

$$Q(s,a\,;\theta_t,\alpha,\beta) = V(s\,;\ \theta_t,\alpha) + \left(A(s,a\,;\theta_t,\beta) - \frac{1}{A}\sum_{a'\in A} A(s,a'\,;\theta\,t,\beta) \right) \tag{3}$$

At each step t, the agent selects action a_t based on policy $\pi(a|s)$, then transitions from state s_t to state s_{t+1} in the environment, and calculates the reward value r_{t+1}. These signals are stored in the experience replay pool. These tuples are then used as training data, enabling the computation of the corresponding action-value functions. The target value for network $\widehat{Q}$ is determined by selecting the action with the highest function value from the Q network, just as described in Eq. (5).

$$Y_t^{D3QN} \ = \ R_{t+1} + \tau\hat{Q}_t(S_{t+1}, argmaxQ\ (\ S_{t+1}, a\,; \theta_t^-\,, \alpha, \beta\ )\,; \theta_t^-\,, \alpha, \beta) \tag{4}$$

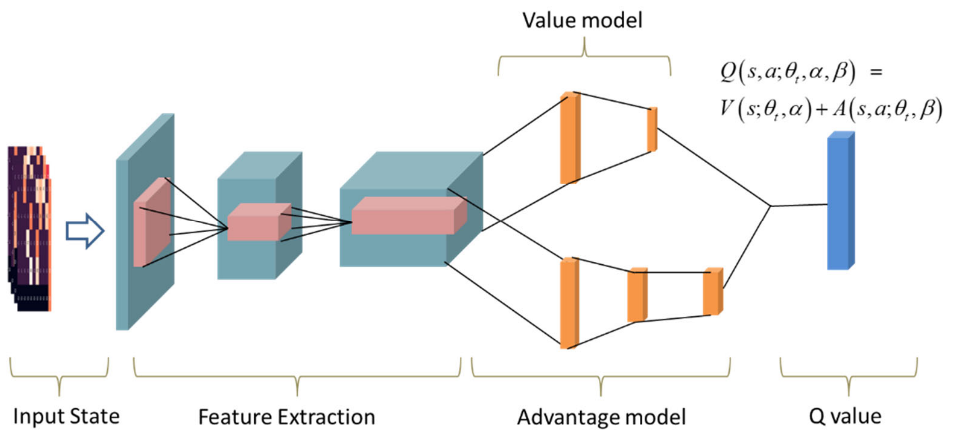
Value model
$Q(s,a;\theta_t,\alpha,\beta) =$
$V(s;\theta_t,\alpha)+A(s,a;\theta_t,\beta)$
Input State
Feature Extraction
Advantage model
Q value

Fig. 1. Dueling network architecture.

Here, θ_t^- and $\theta_t^{-\prime}$ represent the synaptic weights. This objective can be mathematically represented by Eq. (6).

$$L^{D3QN}(\theta_t^-) \ = \ E\left[\left(Y_t^{D3QN} - Q_t\,(\,S_t, a'\,; \theta_t^- \ \,, \alpha, \beta)\right)^2\right] \tag{5}$$

This study validates the effectiveness and robustness of scheduling by utilizing the D3QN algorithm as a robust framework. It possesses several noteworthy advantages: (i) In the context of implementing dynamic multi-objective job shop scheduling problems, a dueling network is utilized to effectively balance the advantages and disadvantages of different elements, particularly when they have similar values. (ii) D3QN exhibits exceptional generalization and nonlinear fitting capabilities, enabling thorough exploration of potential relationships within multi-constraint production scheduling environments.

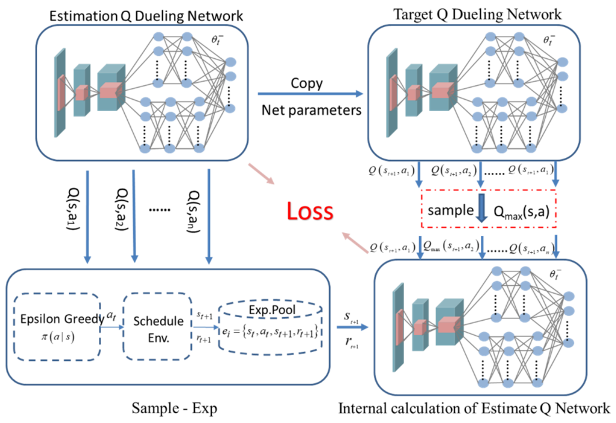

Fig. 2. Learning procedure of Conv-Dueling scheduling algorithm

4. Problem formulation

The initial step involves establishing the logical scheduling formula for Discrete Multi-Flow Job Shop Problem (DMFJSP), where lowercase letters denote indexes and uppercase letters denote collections. In the flexible workshop scheduling problem, let's assume there are $J = \{J_1, J_2, \dots, J_j\}$ jobs and $M = \{M_1, M_2, \dots, M_m\}$ machines. Each Job J_j comprises multiple processing operations. consists of one or more processing operations, denoted as $O = \{O_1, O_2, \dots, O_i\}$, such as turning, milling, grinding, welding, etc. Jobs must be processed in a predetermined order. All operations can be executed on multiple machines, each with its own distinct processing time denoted as P_{ji}. The scheduling task involves allocating jobs to machines for processing in a manner that minimizes makespan and delay time, while maximizing the completion punctuality rate. In Table 3, the task deadline (DDT) is an indicator of task urgency. For a time, point T_i there are n_i jobs J_i of operations. The cut-off time D_i can be calculated as $D_i = A_i + \left(\sum_{j_{j=1}}^{n_i} \bar{t}_{i,j}\right) \bullet DDT$.

This research aims to deal with unpredictable dynamic disturbance events such as job insertion, machine breakdown, and processing time change in the process of production scheduling when there are multi-constraint relationships between processing machines and jobs. It assigns each process $O_{i,j}$ of the job to the appropriate processing machine M_m for processing at an appropriate time, In order to efficiently complete all scheduling tasks and achieve superior comprehensive scheduling performance in terms of both on-time completion rate and makespan. The entire process of production scheduling processing is shown in Fig. 3. The details of the dynamic disturbance event processing shown in the figure are as follows.

Job insertion refers to the need to add new jobs to complete the production tasks due to insufficient completeness or new task requirements other than the initial planned tasks of the workshop production scheduling. In this paper, once a new job is inserted in the scheduling process, except for the jobs being processed by each machine, the rest of the jobs and the newly inserted jobs are re-scheduled and calculated to generate a new scheduling plan. Machine breakdown is an unavoidable and random disturbance event in the actual production process. There are many types of machine failure, and each has a different maintenance time. It is important to deal with different failure events, minimize workshop losses, and better ensure the punctuality rate of scheduling and other goals. In this paper, operations are immediately halted when a machine fails., and then the machine enters the unavailable maintenance state. The interrupted job can be rescheduled to be processed by other machines or continue processing after the machine maintenance is completed. Processing time change refers to the situation that during the production process, due to factors such as different proficiency of workers in operating a machine or machine problems, the processing task cannot be completed according to the specified processing time, and processing is completed in advance or delayed. In the processing process for this paper, changes in man-hours occur randomly, while other tasks are rescheduled and adjusted based on the current status information of the jobs. The following are scheduling constraints:

(i) After the machine has processed one job, it can process the next job

(ii) The job must be machined according to the machining sequence

(iii) Negligible transport times and job loading and unloading times

(iv) Job-to-job unconstrained relationships

(v) Each job has a different processing time on different equipment

(vi) When a machine breaks down, no job can be processed

(vii) Disturbance factors can occur on any machine or job

Table 3
Symbolic representation of production information

Notation	Description
$\mathbf{n}$	The number of jobs
$\mathbf{m}$	The number of machines
$\mathbf{j_i}$	The ith job
$\mathbf{M_k}$	The kth machine
$\mathbf{O_{i,j}}$	The jth operation of job J_i
$\mathbf{M_{i,j}}$	Machine set that can process operation $O_{i,j}$
$\mathbf{n_i}$	Number of operations of job J_i
$\mathbf{t_{i,j,k}}$	Processing time of operation $O_{i,j}$on machine M_k
$\mathbf{t_{i,j}}$	Processing time of operation $O_{i,j}$
$\mathbf{C_{i,j}}$	Completion time of operation $O_{i,j}$
$\mathbf{A_i}$	The arrival time of ith job
$\mathbf{D_i}$	The deadline time of ith job
$\mathbf{B_k}$	The failure of kth machine
$\mathbf{ID_k}$	Number of idle time intervals of machine k

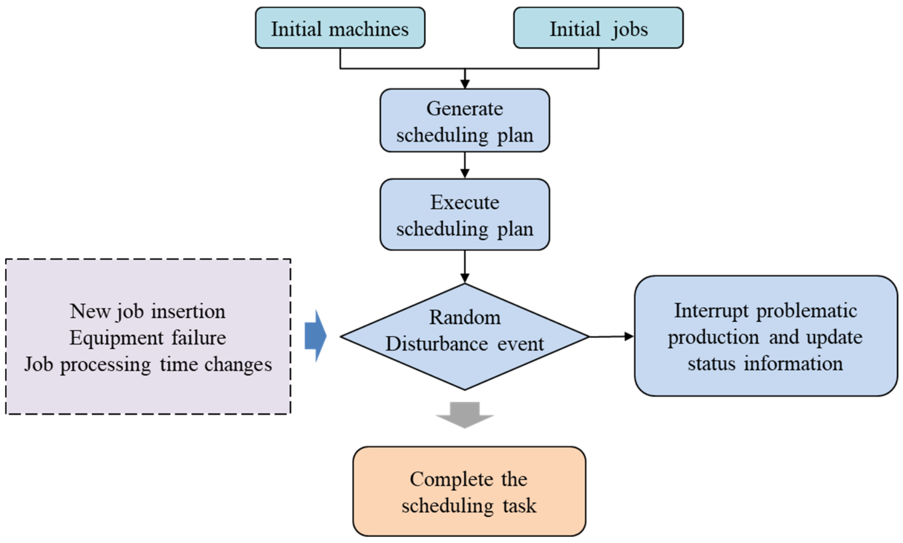

Fig. 3. Scheduling flow chart.

5. Scheduling problem transformation

This paper employs the D3QN algorithm for unsupervised training of the network model. Finally, according to the multi-constraint and multi-disturbance production environment information, the multidimensional state representation matrix of the job is used as input and the optimal scheduling rules are output after the feature extraction and decision-making of the conv-dueling network model, so as to realize the optimal scheduling of tasks.

5.1 State feature

The state space represents all possible environments within the workshop. In less complex environments, the state space can be described using two-dimensional vectors or tables. However, job shop scheduling problems often involve high-dimensional state spaces, which makes it infeasible for RL algorithms to explore all states. Using these information indicators directly as state features can significantly alter the algorithm's input, leading to a substantial decrease in training speed and the generality of D3QN without training.

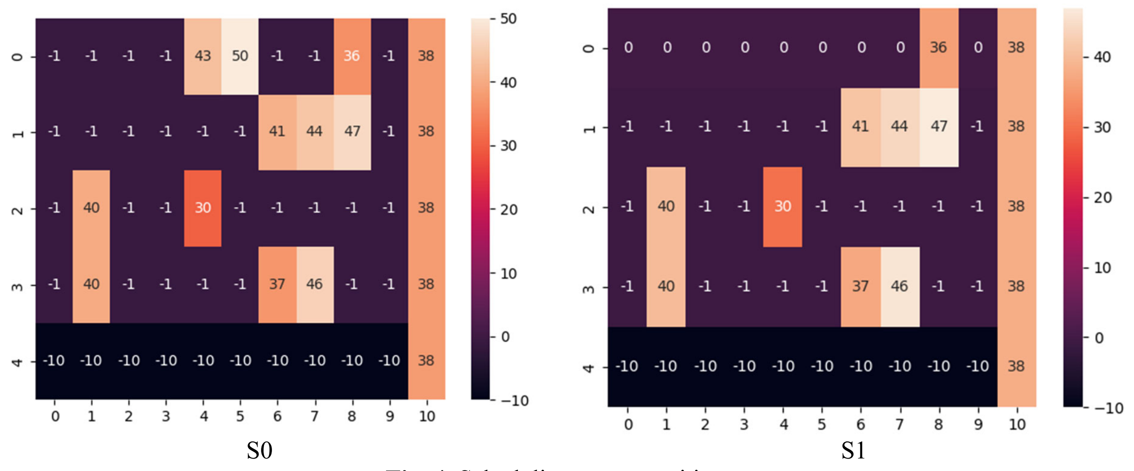

Fig. 4. Scheduling state transition.

In order to leverage the power of deep learning for feature extraction from the original input, this paper proposes a state space consisting of a multidimensional matrix that incorporates both job and machine state information. This matrix enables better handling of the relationship between state features and the action space. This can fully express the information on which the machine executes the action, and also facilitates the rapid training of the neural network and better convergence results, making it easier for the machine to make optimal action decisions. The multidimensional state matrix uses different scheduling feature information as different channels of the image, and each channel has the length of the machine serial number, the width of the process sequence and the height of the job quantity. The scheduling characteristics considered here include information such as job, operation, machine, processing time, deadline, and current time. Each element in the matrix is normalized. If a machine is processing a task, the machine is in a busy state, and the corresponding value represents the remaining unprocessed time for that task on the machine. The other values in the same row are set to zero, indicating that there are no other operations being processed on that machine now. The processing time channel on the far right of the image represents the value calculated by weighing the processing time, deadline, and current time, and expresses the multi-time information more completely.

5.2 Dispatching rules

There is no scheduling rule that can be universally applicable to all production objectives and workshop environments. Selecting different scheduling rules based on different environmental states is of great significance for improving production efficiency. The action in this context is to choose the appropriate jobs for priority scheduling based on the scheduling rules. The action space comprises a range of heuristic scheduling rules. In order to overcome the problem that a single scheduling rule is not applicable to multiple scheduling scenarios, we select appropriate scheduling rules through deep reinforcement learning for different environmental states. Too few scheduling rules will easily result in the agent being unable to solve the global optimal scheduling task in the face of complex and diverse environments. Having too many scheduling rules can indeed impact the efficiency of the agent in learning the optimal scheduling strategy, which in turn hinders the goal of achieving real-time and efficient scheduling. Therefore, we comprehensively consider processing time, job completion rate, waiting time, deadline, arrival time, idle time, and other information factors to design nine kinds of better scheduling rules, as shown in Table 4.

5.3 Definition of rewards

The reward function provides immediate feedback signals to the agent to guide its learning and decision-making process. By appropriately designing a composite reward function, it is possible to guide the equipment to learn adaptive and efficient behavioral strategies when faced with different workpiece selection problems. However, designing an appropriate reward function is a challenging task that requires balancing exploration and exploitation, addressing reward sparsity, and navigating the challenges of guiding learning. The objective is to take into account the minimum makespan, minimum delay time, and maximum completion punctuality rate in a comprehensive manner. To achieve this, we employ a compound reward approach that combines the main task with branch tasks. We design the branch reward to guide the agent in learning the optimal actions. By using this method, we aim to optimize the scheduling process by considering multiple performance metrics simultaneously. The main line rewards give positive or negative feedback of success or failure when a training session is completed, which solves the problems that sparse rewards affect convergence and dense rewards easily cause local optima. The reward function of the main line is given in formula (8):

$$reward1 = \begin{cases} -R \text{ if d\_r>t\_r ; if j\_t>max\_t} \\ R + R\_B * c\_t \text{ if c\_t>=}(1-r) \end{cases} \tag{6}$$

Here, R and R_B are the reward values set after multiple experiments, c_t is the probability of on-time completion, d_r is the failure rate, j_t is the training step, max_t is the step threshold, and r is the target completion rate indicator. The branch rewards are shown in formula (8):

$$reward2 = -(j\_l / m\_s) * \mu \tag{7}$$

where j_l represents the information about overdue tasks in the job, m_s is total machine information, and μ is weight coefficient. The total reward is shown in formula (9), where α is the weight factor.

$$reward = reward1 + \alpha * reward2 \tag{8}$$

Table 4
Actions

Rule	Description
SPT	Minimum processing time
LPT	Maximum processing time
SCRJ	Minimum completion rate
SMWT	Minimum remaining waiting time
CCMT	Minimum product of the completion rate and minimum waiting time
EDF	Earliest cut-off time
FIFO	Earliest time of arrival
LMWT	Maximum waiting time
SIT	Minimum idle time

6. Experimentation

6.1 Training details

Table 5 displays the data range employed in training our deep reinforcement learning algorithm. To ensure that the training data encompasses all uncertain events, We designed the initial training data for each training process, which includes information about the jobs and machines, as well as randomly occurring perturbation events. The parameters associated with the jobs and machines are generated randomly within the ranges specified in Table 5.

6.2 Training details

In this experiment, we utilize randomly generated data instances that adhere to the constraints presented in Table 6. At the start of production, there are 20 jobs, and the number of machines can be either 10 or 20. The processing capabilities of the machines are also randomly assigned. Additionally, there are 30, 50, and 80 dynamic disturbance events, each with a random type. Similar to the training phase, the job and machine parameters are randomized. In total, there are 30 combinations of test cases, each with 50 runs.

Table 5
Training data details

Parameter type	Range
Total number of machines	1-50
Number of available machines of each operation	0-50
Number of initial jobs at beginning	0-50
Total number new inserted jobs	0-100
Total number of fault types	0-100
Repair time of failure	1-99
Due date tightness (DDT)	1-5
Number of operations belonging to a job	0-10
Processing time of an operation on an available machine	0-50
Average value of exponential distribution between two successive new job arrivals (Eave)	0-100

6.2.1 Sensitivity study/hyperparameter design of control parameters

After conducting numerous experiments, we have determined the optimal hyperparameters, as presented in Table 7, which result in optimal training performance. Among these hyperparameters, the learning rate is particularly influential. We utilize the gradient descent method to enhance the algorithm's performance, necessitating the selection of an appropriate learning rate value. An excessively large learning rate can cause the neural network to oscillate around the global optimum during training. Conversely, a learning rate that is too small may lead to slow convergence or convergence to a local optimum. Therefore, selecting the appropriate learning rate is crucial for training the model effectively. In this paper, we employ a piecewise constant approach to gradually decrease the learning rate as the number of training iterations increases. The learning rate is decreased exponentially with increasing training iterations, and the decreasing step sizes are defined as (200, 500, 800).

6.2.2 Comparison of the results of DQN, DDQN, and D3QN

The D3QN algorithm achieved reward convergence within the first 2000 training rounds, as depicted in Fig. 5. As the training progresses, the rewards of all three algorithms converge to the maximum value. The post-convergence oscillation is mainly attributed to the occasional random action selection with a low probability. Among the three algorithms, the DQN algorithm exhibits the poorest performance, lowest learning efficiency, and slowest convergence speed. DDQN shows significant improvement and achieves faster convergence compared to the DQN algorithm, but it still falls short of the D3QN algorithm. The D3QN algorithm stands out with the best convergence and stability. The superiority of the D3QN algorithm can be attributed to its utilization of the dueling network architecture and the integration of the multi-dimensional state space of the Deep Double Q-Network. These design elements help alleviate the issue of action value overestimation. Furthermore, the convergence results of the completion punctuality rate for the different algorithms are displayed in Figure 6. The results demonstrate that all the dueling-double-deep reinforcement learning algorithms achieve convergence, with respective completion punctuality rates of 0.85, 0.8, and 0.7. It is evident that the D3QN algorithm exhibits more stable convergence at a higher reward value compared to the other two algorithms. The overall scheduling performance indicates that these algorithms have not effectively learned improved scheduling rules at the rescheduling point. leading to a lower job completion punctuality rate. These findings emphasize that in scenarios involving multiple constraints and disturbances, the D3QN-based dynamic multi-objective intelligent scheduling algorithm proposed in the study has successfully acquired more efficient scheduling rules.

Table 6
Test data details

Parameter	Value
Total number of machines	[10,20]
Number of available machines of each operation	Unif [0,M]
Number of initial jobs	20
Total number new inserted jobs	[30,50,80]
Total number of fault types	[0-10]
Repair time of failure	[10-55]
Due date tightness (DDT)	2.0
Number of operations belonging to a job	Unif[2,5]
Processing time of an operation on an available machine	Unif[0,50]
Average value of exponential distribution between two successive new job arrivals (Eave)	{50, 75,100}

Table 7
Hyperparameters

Hyperparameters
MAX_STEP = 500
R = 100
R_B = 1
TEST_RATE = 25
TEST_EPOCH_NUM = 1
GAMMA = 0.99
EPOCH_NUM = 2000
LR= 0.01
LR_DECREMENT = 0.1
E_GREEDY = 0.8
E_GREEDY_DECREMENT =0.0005
MAX_SIZE = 5000
BATCH_SIZE = 32
WARM_UP_SIZE = 1000
UPDATE_TARGET_MODEL_RATE = 100

6.2.3 Comparison with other scheduling rules

To evaluate the efficiency and adaptability of the suggested dynamic multi-objective intelligent scheduling algorithm, a comparative experiment was conducted to evaluate its performance against single scheduling rules. Multiple sets of experimental data were generated randomly, simulating various task scheduling scenarios in the production process, with different parameter settings for m, n_{add}, and E_{ave}. For each set of experimental data, the proposed scheduling algorithm and each individual scheduling rule were repeated 50 times. Tables 8-10 present the average and standard deviation values of makespan, delay time, and completion punctuality rate obtained by each method. The superior outcomes are indicated in bold.

The study conducted multiple experiments and calculated the average values and standard deviations. These results provide insights into the impact of our designed scheduling algorithm and other scheduling rules on the overall scheduling performance under different experimental parameter configurations. As anticipated, when there is a higher number of newly arriving jobs and more frequent arrivals, the delay time and completion time increase, while the job completion rate decreases. Furthermore, it can be observed that as the number of machines increases, the job delay time decreases, and the job completion rate improves. These results highlight the influence of parameter settings on scheduling performance and offer valuable insights into the behavior of the proposed algorithm and alternative scheduling rules in varying conditions. Comparatively, our proposed scheduling algorithm outperforms single scheduling rules by achieving remarkable improvements in minimizing makespan and delay time while maximizing the completion punctuality rate. The experimental results indicate that the scheduling algorithm is capable of making decisions that lead to globally optimal scheduling strategies when facing different scheduling scenarios. In a dynamic production environment, no single scheduling rule can achieve optimal scheduling performance as effectively as our proposed algorithm. These results provide evidence that our algorithm exhibits strong generality and performs well in various untrained situations. It showcases the algorithm's ability to adapt and make effective scheduling decisions in changing and unpredictable production environments. The scheduling method proposed in this study can select the optimal scheduling strategy based on the current workshop production environment and scheduling task information. It aims to achieve more efficient scheduling and demonstrates comprehensive performance. This adaptability and decision-making capability make the scheduling algorithm more effective and versatile compared to relying on a single scheduling rule.

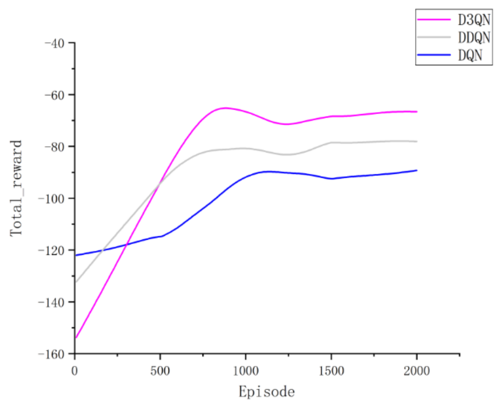

Fig. 5. Comparison of the complete rate of three algorithms

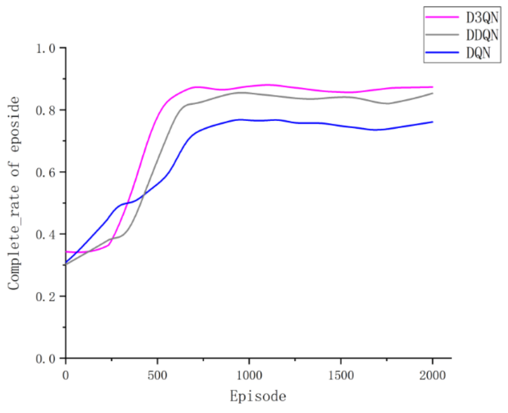

Fig. 6. Comparison of the total reward of three algorithms

Table 8
Comparison of average and standard deviation value of makespan after 50 runs of single rule scheduling algorithm and ours

m	n_{add}	E_{ave}	Rule1	Rule2	Rule3	Rule4	Rule5	Rule6	Rule7	Rule8	Rule9	**Ours**
10	30	50	7.31e+02/5.77e+01	9.50e+02/6.70e+01	8.43e+02/7.18e+01	7.12e+02/7.03e+01	6.72e+02/4.53e+01	6.89e+02/5.05e+01	8.09e+02/6.12e+01	7.40e+02/5.58e+01	6.92e+02/5.37e+01	**5.38e+02/3.67e+01**
		75	7.69e+02/5.96e+01	9.64e+02/6.34e+01	8.05e+02/7.11e+01	6.97e+02/7.37e+01	6.81e+02/4.24e+01	6.93e+02/5.86e+01	8.21e+02/6.01e+01	7.50e+02/5.57e+01	6.88e+02/5.97e+01	**5.57e+02/3.81e+01**
		100	7.44e+02/6.13e+01	9.60e+02/5.51e+01	8.03e+02/6.41e+01	7.16e+02/6.86e+01	7.18e+02/4.32e+01	7.09e+02/5.53e+01	8.32e+02/6.15e+01	7.51e+02/5.20e+01	6.80e+02/5.27e+01	**5.40e+02/3.96e+01**
	50	50	1.00e+03/5.63e+01	1.26e+03/6.88e+01	1.07e+03/8.71e+01	9.92e+02/7.94e+01	9.47e+02/5.99e+01	9.75e+03/7.05e+01	1.07e+03/5.90e+01	1.02e+03/6.87e+01	9.90e+02/6.68e+01	**8.12e+02/5.75e+01**
		75	1.35e+03/7.26e+01	1.65e+03/7.35e+01	1.42e+03/9.20e+01	1.35e+03/7.51e+01	1.27e+03/6.33e+01	1.31e+03/6.26e+01	1.43e+03/6.56e+01	1.34e+03/7.41e+01	1.25e+03/6.91e+01	**8.56e+02/5.11e+01**
		100	2.06e+03/7.24e+01	2.17e+03/7.37e+01	1.44e+03/9.95e+01	1.33e+03/7.05e+01	1.30e+03/4.21e+01	1.27e+03/7.89e+01	1.38e+03/8.39e+01	1.31e+03/5.26e+01	1.35e+03/5.11e+01	**8.71e+02/5.99e+01**
	80	50	1.43e+03/6.98e+01	1.77e+03/7.43e+01	1.48e+03/9.96e+01	1.40e+03/8.30e+01	1.33e+03/5.19e+01	1.39e+03/6.38e+01	1.46e+03/8.47e+01	1.39e+03/6.88e+01	1.35e+03/6.64e+01	**9.76e+02/6.14e+01**
		75	1.44e+03/9.50e+01	1.75e+03/7.72e+01	1.47e+03/8.32e+01	1.37e+03/8.87e+01	1.37e+03/6.74e+01	1.36e+03/6.56e+01	1.43e+03/6.85e+01	1.48e+03/6.50e+01	1.32e+03/6.17e+01	**9.94e+02/6.31e+01**
		100	1.40e+03/6.96e+01	1.74e+03/8.03e+01	1.51e+03/9.11e+01	1.40e+03/8.35e+01	1.32e+03/7.27e+01	1.37e+03/6.79e+01	1.49e+03/8.19e+01	1.43e+03/7.49e+01	1.36e+03/8.19e+01	**1.03e+03/6.91e+01**
20	30	50	4.20e+02/5.02e+01	5.33e+02/6.46e+01	4.44e+02/5.28e+01	3.82e+02/5.88e+01	3.81e+02/4.12e+01	3.73e+02/6.07e+01	4.33e+02/4.74e+01	4.54e+02/6.32e+01	3.65e+02/5.81e+01	**2.84e+02/4.51e+01**
		75	4.82e+02/7.55e+01	5.39e+02/6.71e+01	4.59e+02/6.20e+01	3.88e+02/6.41e+01	4.64e+02/5.42e+01	3.81e+02/6.79e+01	4.50e+02/5.74e+01	4.44e+02/5.78e+01	3.94e+02/6.19e+01	**3.01e+02/4.91e+01**
		100	5.39e+02/9.03e+01	5.41e+02/7.03e+01	5.33e+02/7.84e+01	4.34e+02/7.94e+01	4.39e+02/6.43e+01	4.26e+02/8.39e+01	4.72e+02/6.58e+01	4.99e+02/8.17e+01	4.09e+02/6.47e+01	**3.17e+02/5.21e+01**
	50	50	5.66e+02/5.13e+01	7.36e+02/5.82e+01	6.21e+02/5.73e+01	4.84e+02/5.33e+01	4.69e+02/4.52e+01	4.91e+02/4.81e+01	5.64e+02/4.59e+01	5.32e+02/4.82e+01	4.72e+02/4.86e+01	**3.64e+02/3.98e+01**
		75	5.94e+02/5.95e+01	7.13e+02/6.38e+01	6.52e+02/6.66e+01	5.09e+02/5.73e+01	4.94e+02/5.07e+01	5.16e+02/6.04e+01	5.69e+02/5.85e+01	5.16e+02/5.07e+01	4.80e+02/5.14e+01	**3.78e+02/4.59e+01**
		100	5.78e+02/7.48e+01	7.28e+02/6.18e+01	6.74e+02/7.50e+01	5.55e+02/6.91e+01	4.90e+02/5.63e+01	5.22e+02/6.04e+01	5.95e+02/6.98e+01	5.61e+02/6.28e+01	5.23e+02/7.29e+01	**3.97e+02/5.61e+01**
	80	50	7.55e+02/5.70e+01	9.91e+02/6.05e+01	7.94e+02/5.57e+01	7.02e+02/5.45e+01	6.83e+02/5.22e+01	6.73e+02/5.05e+01	7.87e+02/4.79e+01	7.22e+02/4.59e+01	6.67e+02/4.29e+01	**5.69e+02/4.11e+01**
		75	7.45e+02/5.58e+01	9.58e+02/6.86e+01	7.63e+02/6.28e+01	6.81e+02/5.59e+01	6.66e+02/5.32e+01	8.02e+02/5.88e+01	8.02e+02/5.88e+01	8.02e+02/5.88e+01	6.70e+02/4.69e+01	**5.84e+02/4.95e+01**
		100	8.03e+02/6.94e+01	1.02e+03/6.16e+01	8.06e+02/5.93e+01	7.10e+02/4.38e+01	6.75e+02/6.05e+01	6.84e+02/5.79e+01	8.17e+02/6.41e+01	7.29e+02/5.47e+01	7.09e+02/7.29e+01	**6.07e+02/5.74e+01**

Table 9
Comparison of average and standard deviation value of Tr_{ave} after 50 runs of single rule scheduling algorithm and ours

m	n_{add}	E_{ave}	Rule1	Rule2	Rule3	Rule4	Rule5	Rule6	Rule7	Rule8	Rule9	**Ours**
10	30	50	3.66e-01/2.81e-02	8.04e-02/3.31e-02	2.80e-01/3.23e-02	1.96e-01/3.02e-02	1.85e-01/2.77e-02	1.95e-01/3.62e-02	1.24e-01/3.96e-02	1.42e-01/3.24e-02	4.80e-03/8.54e-03	**7.04e-01/2.41e-02**
		75	3.77e-01/2.81e-02	1.02e-01/3.57e-02	2.97e-01/3.77e-02	1.94e-01/3.02e-02	1.84e-01/2.69e-02	2.07e-01/3.22e-02	1.35e-01/4.13e-02	1.55e-01/3.36e-02	9.60e-03/1.28e-02	**7.26e-01/2.58e-02**
		100	5.36e-01/4.61e-02	2.79e-01/5.49e-02	4.52e-01/5.53e-02	3.52e-01/4.64e-02	4.36e-01/5.54e-02	3.40e-01/4.12e-02	3.17e-01/5.77e-02	4.39e-01/5.95e-02	1.55e-01/5.33e-02	**7.39e-01/2.72e-02**
	50	50	2.64e-01/1.78e-02	3.97e-02/1.79e-02	1.92e-01/2.02e-02	1.18e-01/2.71e-02	1.24e-01/2.16e-02	1.17e-01/2.88e-02	6.28e-02/2.53e-02	8.82e-02/2.35e-02	5.71e-04/2.79e-03	**6.15e-01/1.01e-02**
		75	1.93e-01/1.34e-02	2.46e-02/1.46e-02	1.42e-01/1.82e-02	7.93e-02/1.60e-02	9.26e-02/1.41e-02	7.01e-02/1.91e-02	3.87e-02/1.93e-02	5.43e-02/1.67e-02	3.65e-02/1.24e-02	**6.24e-01/1.45e-02**
		100	1.55e-01/1.21e-02	1.24e-02/9.31e-03	1.36e-01/7.45e-03	6.16e-02/1.21e-02	8.51e-02/4.95e-03	7.32e-02/1.69e-02	4.48e-02/1.02e-02	4.21e-02/2.16e-02	1.24e-02/4.54e-03	**6.39e-01/1.28e-02**
	80	50	1.89e-01/1.40e-02	2.06e-02/1.43e-02	1.45e-01/1.97e-02	7.34e-02/1.77e-02	8.97e-02/1.32e-02	6.19e-02/2.03e-02	3.23e-02/1.81e-02	6.13e-02/1.94e-02	1.48e-03/3.53e-03	**3.11e-01/9.92e-01**
		75	1.38e-01/1.68e-02	7.32e-02/1.48e-02	1.38e-01/1.68e-02	7.32e-02/1.48e-02	8.84e-02/1.40e-02	6.22e-02/1.76e-02	3.28e-02/1.58e-02	5.18e-02/1.81e-02	1.48e-03/1.14e-03	**3.29e-01/1.26e-02**
		100	1.94e-01/1.72e-02	3.06e-02/1.87e-02	1.53e-01/1.84e-02	7.61e-02/2.27e-02	9.11e-02/1.50e-02	5.66e-02/2.09e-02	3.87e-02/1.74e-02	6.05e-02/1.72e-02	1.20e-03/3.84e-03	**3.42e-01/1.42e-02**
20	30	50	5.64e-01/3.11e-02	3.99e-01/4.64e-02	5.61e-01/5.21e-02	4.81e-01/3.91e-02	5.36e-01/4.40e-02	3.88e-01/3.87e-02	4.28e-01/6.76e-02	6.14e-01/4.88e-02	5.02e-01/7.02e-02	**8.65e-01/2.15e-02**
		75	5.93e-01/4.02e-02	4.44e-01/5.72e-02	5.97e-01/4.39e-02	4.91e-01/4.96e-02	5.67e-01/5.73e-02	4.23e-01/4.06e-02	4.75e-01/6.69e-02	6.08e-01/5.27e-02	5.56e-01/8.19e-02	**8.83e-01/2.65e-02**
		100	5.95e-01/4.70e-02	5.22e-01/7.22e-02	6.08e-01/4.76e-02	5.26e-01/4.29e-02	5.92e-01/6.21e-02	4.63e-01/4.85e-02	5.21e-01/6.66e-02	6.40e-01/4.97e-02	6.22e-01/6.38e-02	**9.02e-01/2.91e-02**
	50	50	4.02e-01/3.58e-02	1.82e-01/3.63e-02	3.89e-01/3.11e-02	3.25e-01/2.88e-02	3.33e-01/2.20e-02	2.58e-01/3.29e-02	2.01e-01/4.03e-02	2.05e-01/5.06e-02	6.28e-02/3.07e-02	**7.55e-01/2.90e-02**
		75	4.29e-01/3.81e-02	2.22e-01/3.45e-02	4.26e-01/3.22e-02	3.39e-01/2.89e-02	3.57e-01/3.14e-02	2.79e-01/3.83e-02	2.12e-01/3.75e-02	2.52e-01/5.44e-02	1.24e-01/4.38e-02	**7.69e-01/3.19e-02**
		100	4.42e-01/4.42e-02	2.39e-01/4.59e-02	4.43e-01/3.21e-02	3.72e-01/3.28e-02	3.97e-01/4.06e-02	3.15e-01/3.21e-02	2.61e-01/4.25e-02	2.95e-01/6.65e-02	1.50e-01/5.95e-02	**7.93e-01/3.54e-02**
	80	50	2.72e-01/2.13e-02	7.64e-02/2.36e-02	2.59e-01/2.43e-02	2.09e-01/2.54e-02	2.16e-01/1.67e-02	1.68e-01/1.91e-02	1.13e-01/2.54e-02	9.24e-02/2.23e-02	2.83e-03/6.01e-03	**5.29e-01/1.81e-02**
		75	2.98e-01/2.18e-02	2.98e-01/2.18e-02	2.89e-01/2.03e-02	2.20e-01/2.30e-02	2.26e-01/2.20e-02	1.81e-01/3.28e-02	1.17e-01/2.59e-02	1.20e-01/2.71e-02	1.44e-02/1.26e-02	**5.54e-01/2.06e-02**
		100	3.17e-01/3.05e-02	1.16e-01/2.81e-02	3.03e-01/2.71e-02	2.26e-01/2.86e-02	2.42e-01/2.23e-02	1.91e-01/2.87e-02	1.38e-01/2.71e-02	1.41e-01/3.23e-02	3.12e-02/1.92e-02	**5.85e-01/2.61e-02**

Table 10

Comparison of average and standard deviation value of $Tard_{ave}$ after 50 runs of single rule scheduling algorithm and ours

m	n_{add}	E_{ave}	Rule1	Rule2	Rule3	Rule4	Rule5	Rule6	Rule7	Rule8	Rule9	**Ours**
10	30	50	6.02e+03/6.34e+02	1.15e+04/1.03e+03	7.03e+03/8.08e+02	9.75e+03/6.27e+02	1.15e+04/1.10e+03	1.09e+04/7.70e+02	8.28e+03/9.04e+02	6.79e+03/6.81e+02	1.08e+04/1.22e+03	**1.71e+03/3.13e+02**
		75	5.47e+03/6.92e+02	1.13e+04/1.02e+03	6.36e+03/7.37e+02	9.32e+03/6.60e+02	1.11e+04/1.12e+03	1.06e+04/8.61e+02	8.08e+03/7.23e+02	6.84e+03/7.54e+02	1.02e+04/9.49e+02	**1.53e+03/3.64e+02**
		100	1.18e+03/3.79e+02	4.91e+03/8.13e+02	2.86e+03/5.93e+02	5.90e+03/4.43e+02	3.54e+03/6.50e+02	6.79e+03/5.38e+02	3.97e+03/5.99e+02	1.21e+03/2.85e+02	2.83e+03/6.46e+02	**8.12e+02/9.84e+01**
	50	50	1.67e+04/1.33e+03	2.88e+04/2.18e+03	1.79e+04/1.50e+03	2.37e+04/1.29e+03	3.03e+04/2.64e+03	2.56e+04/1.79e+03	2.05e+04/1.36e+03	1.89e+04/1.35e+03	2.77e+04/2.62e+03	**8.44e+03/5.62e+02**
		75	3.75e+04/2.28e+03	6.10e+04/3.11e+03	3.92e+04/2.25e+03	4.97e+04/2.27e+03	6.62e+04/4.25e+03	5.25e+04/2.29e+03	4.45e+04/2.28e+03	4.03e+04/2.51e+03	6.07e+04/3.68e+03	**9.15e+03/7.36e+02**
		100	6.61e+04/3.07e+03	1.06e+05/5.87e+03	4.52e+04/3.61e+03	5.50e+04/1.62e+03	7.67e+04/2.61e+03	5.75e+04/2.41e+03	5.08e+04/2.35e+03	4.45e+04/1.59e+03	7.33e+04/2.27e+03	**9.87e+03/8.31e+02**
	80	50	4.11e+04/2.71e+03	6.96e+04/4.01e+03	4.32e+04/2.67e+03	5.43e+04/2.80e+03	7.27e+04/3.36e+03	5.83e+04/2.53e+03	4.97e+04/2.38e+03	4.75e+04/2.12e+03	6.83e+04/3.87e+03	**2.64e+04/4.54e+02**
		75	4.28e+04/2.86e+03	7.04e+04/3.54e+03	4.49e+04/1.84e+03	5.56e+04/2.46e+03	7.67e+04/4.85e+03	5.85e+04/2.58e+03	4.94e+04/2.20e+03	4.69e+04/2.64e+03	6.78e+04/3.98e+03	**2.31e+04/5.61e+02**
		100	3.94e+04/2.71e+03	6.65e+04/3.52e+03	4.19e+04/2.27e+03	5.28e+04/2.27e+03	7.04e+04/3.95e+03	5.64e+04/2.44e+03	4.93e+04/8.65e+03	4.80e+04/2.34e+03	6.72e+04/4.84e+03	**2.41e+04/5.71e+02**
20	30	50	2.44e+02/1.47e+02	1.12e+03/2.95e+02	5.57e+02/1.78e+02	1.52e+03/2.23e+02	8.74e+02/2.52e+02	2.56e+03/2.61e+02	8.55e+02/2.33e+02	4.86e+01/5.01e+01	1.56e+02/8.10e+01	**2.74e+01/6.92e+01**
		75	1.33e+02/9.54e+01	8.76e+02/3.69e+02	4.83e+02/1.92e+02	1.32e+03/2.69e+02	6.35e+02/2.71e+02	2.04e+03/3.78e+02	7.51e+02/2.41e+02	5.26e+01/5.99e+01	8.49e+01/8.29e+01	**2.96e+01/7.65e+01**
		100	5.96+01/7.608e+01	5.40e+02/2.48e+02	1.98e+02/1.36e+02	8.79e+02/2.72e+02	4.07e+02/2.28e+02	1.60e+03/3.97e+02	4.89e+02/2.15e+02	1.32e+01/2.26e+01	2.77e+01/3.28e+01	**0.80e+01/4.30e+01**
	50	50	2.50e+03/5.28e+02	6.83e+03/9.38e+02	3.39e+03/5.36e+02	5.85e+03/4.65e+02	5.84e+03/6.49e+02	7.68e+03/4.53e+02	4.95e+03/7.38e+02	2.15e+03/3.78e+02	4.07e+03/6.60e+02	**9.76e+02/1.69e+02**
		75	1.53e+03/4.73e+02	5.62e+03/8.18e+02	2.78e+03/5.15e+02	5.32e+03/6.37e+02	4.84e+03/6.38e+02	6.93e+03/6.76e+02	4.46e+03/6.87e+02	1.60e+03/3.99e+02	3.07e+03/7.15e+02	**8.14e+02/8.71e+01**
		100	1.17e+03/3.99e+02	5.14e+03/9.95e+02	2.23e+03/5.80e+02	4.55e+03/7.15e+02	4.09e+03/7.71e+02	6.36e+03/8.10e+02	3.73e+03/6.69e+02	1.33e+03/5.41e+02	2.70e+03/6.85e+02	**6.76r+02/6.69e+01**
	80	50	1.10e+04/1.14e+03	2.34e+04/1.62e+03	1.27e+04/1.28e+03	1.78e+04/8.87e+02	2.19e+04/1.65e+03	2.11e+04/9.24e+02	1.64e+04/1.12e+03	1.25e+04/1.05e+03	1.98e+04/1.69e+03	**7.26e+03/7.49e+02**
		75	9.62e+03/1.43e+03	2.13e+04/1.97e+03	1.12e+04/7.54e+02	1.65e+04/8.38e+02	1.98e+04/1.66e+03	2.06e+04/9.27e+02	1.62e+04/1.37e+03	1.21e+04/8.86e+02	1.85e+04/2.05e+03	**5.66e+03/5.63e+02**
		100	8.22e+03/1.15e+03	1.97e+04/1.71e+03	1.03e+04/1.12e+03	1.54e+04/9.88e+02	1.74e+04/1.49e+03	1.90e+04/1.16e+03	1.49e+04/1.02e+03	1.08e+04/1.05e+03	1.62e+04/1.67e+03	**3.31e+03/3.18e+02**

Table 11

Comparison of average and standard deviation value of result after 50 runs of single rule scheduling algorithm and ours

m	n_{add}	E_{ave}	DDQN (makespan)	**Ours （makespan）**	DDQN (Trave)	**Ours (Trave)**	DDQN ($Tard_{ave}$)	**Ours ($Tard_{ave}$)**
10	30	50	6.86e+02/4.62e+01	**5.38e+02/3.67e+01**	6.32e-01/4.15e-02	**7.04e-01/2.41e-02**	2.98e+03/5.62e+02	**1.71e+03/3.13e+02**
		75	7.25e+02/4.99e+01	**5.57e+02/3.81e+01**	6.51e-01/4.92e-02	**7.26e-01/2.58e-02**	2.69e+03/6.37e+02	**1.53e+03/3.64e+02**
		100	6.94e+02/5.76e+01	**5.40e+02/3.96e+01**	6.61e-01/5.92e-02	**7.39e-01/2.72e-02**	2.48e+03/8.13e+02	**8.12e+02/9.84e+01**
	50	50	9.12e+02/6.94e+01	**8.12e+02/5.75e+01**	4.13e-01/3.61e-02	**6.15e-01/1.01e-02**	9.60e+03/8.79e+02	**8.44e+03/5.62e+02**
		75	9.73e+02/7.31e+01	**8.56e+02/5.11e+01**	4.55e-01/3.91e-02	**6.24e-01/1.45e-02**	9.76e+03/9.41e+02	**9.15e+03/7.36e+02**
		100	1.06e+03/7.35e+01	**8.71e+02/5.99e+01**	4.62e-01/4.15e-02	**6.39e-01/1.28e-02**	1.13e+04/1.08e+03	**9.87e+03/8.31e+02**
	80	50	1.39e+03/7.60e+01	**9.76e+02/6.14e+01**	2.15e-01/1.68e-02	**3.11e-01/9.92e-01**	3.32e+04/1.63e+03	**2.64e+04/4.54e+02**
		75	1.49e+03/8.38e+01	**9.94e+02/6.31e+01**	2.23e-01/2.13e-02	**3.29e-01/1.26e-02**	3.24e+04/1.34e+03	**2.31e+04/5.61e+02**
		100	1.64e+03/8.86e+01	**1.03e+03/6.91e+01**	2.94e-01/2.63e-02	**3.42e-01/1.42e-02**	3.11e+04/1.56e+03	**2.41e+04/5.71e+02**
20	30	50	2.98e+02/5.98e+01	**2.84e+02/4.51e+01**	7.12e-01/3.61e-02	**8.65e-01/2.15e-02**	4.64e+01/1.31e+02	**2.74e+01/6.92e+01**
		75	3.61e+02/6.74e+01	**3.01e+02/4.91e+01**	7.33e-01/3.99e-02	**8.83e-01/2.65e-02**	4.31e+01/1.22e+02	**2.96e+01/7.65e+01**
		100	3.79e+02/7.31e+01	**3.17e+02/5.21e+01**	7.65e-01/4.92e-02	**9.02e-01/2.91e-02**	2.61e+01/9.35e+01	**0.80e+01/4.30e+01**
	50	50	3.96e+02/5.21e+01	**3.64e+02/3.98e+01**	5.68e-01/4.01e-02	**7.55e-01/2.90e-02**	1.62e+03/8.11e+02	**9.76e+02/1.69e+02**
		75	4.36e+02/6.62e+01	**3.78e+02/4.59e+01**	5.99e-01/4.68e-02	**7.69e-01/3.19e-02**	1.01e+03/2.37e+02	**8.14e+02/8.71e+01**
		100	4.67e+02/7.77e+01	**3.97e+02/5.61e+01**	6.16e-01/5.24e-02	**7.93e-01/3.54e-02**	8.64e+02/3.94e+02	**6.76r+02/6.69e+01**
	80	50	6.46e+02/6.95e+01	**5.69e+02/4.11e+01**	3.92e-01/3.62e-02	**5.29e-01/1.81e-02**	8.62e+03/1.62e+03	**7.26e+03/7.49e+02**
		75	6.66e+02/7.61e+01	**5.84e+02/4.95e+01**	4.04e-01/4.05e-02	**5.54e-01/2.06e-02**	7.06e+03/1.13e+03	**5.66e+03/5.63e+02**
		100	6.98e+02/8.99e+01	**6.07e+02/5.74e+01**	4.31e-01/4.87e-02	**5.85e-01/2.61e-02**	6.33e+03/9.31e+02	**3.31e+03/3.18e+02**

6.4 Comparison of double deep reinforcement learning algorithms

To demonstrate the superior comprehensive scheduling performance of the proposed dynamic scheduling method based on reinforcement learning with Conv-Dueling and generalized representation compared to the DDQN-based scheduling algorithm, the following experiments have been designed. To ensure fairness between the different algorithms, the DDQN algorithm utilizes the action and reward functions of our proposed scheduling algorithm. We discretized the state features of DDQN into nine discrete states. Table 11 displays the scheduling results for the two trained scheduling algorithms across the three scheduling objectives. It is evident from the table that the scheduling algorithm based on D3QN outperforms the DDQN-based algorithm in terms of mean 7. and standard deviation values for overall scheduling performance and achieving multiple scheduling objectives.

7. Conclusion

This paper proposes the design of a dynamic multi-objective flexible job-shop scheduling environment, which includes three dynamic disturbance events and three scheduling optimization objectives. We propose a dynamic scheduling method utilizing Conv-Dueling and a generalized representation based on reinforcement learning. This approach effectively addresses the challenges posed by multi-constraint, multi-disturbance, and multi-objective task scheduling problems in workshop settings. Our method leverages a dueling-double-deep Q-learning network, which enables more accurate learning of optimal scheduling rules in different scenarios. It generates globally optimal scheduling plans each time a disturbance event occurs. Furthermore, we introduce a multidimensional state representation matrix that encompasses job and machine state information. This comprehensive representation captures the necessary information for machine actions, facilitating rapid neural network training and yielding improved convergence results.

To validate the efficiency and superior comprehensive scheduling performance of the proposed dynamic scheduling method with Conv-Dueling and generalized representation based on reinforcement learning, several sets of experiments were conducted using various configurations of production environment parameters. The results clearly demonstrate that our proposed scheduling algorithm outperforms a single scheduling rule by a significant margin, regardless of whether the experimental parameter configurations were trained or untrained. In addition, we compared the scheduling algorithm based on D3QN with the DDQN scheduling algorithm. This comparison confirmed the effectiveness and robustness of our scheduling algorithm in addressing multi-constraints and multi-disturbances. Through these experiments, we have successfully verified the superior performance of our proposed conv-dueling neural network model. Its ability to outperform single scheduling rules and its effectiveness in handling complex scenarios involving multiple constraints and disturbances have been clearly established. In future research, we intend to investigate the effects of supplementary disruptive elements on job-shop scheduling. Specifically, we will investigate preemptive order insertion, energy consumption, loading and unloading time of materials, and other relevant factors. Furthermore, it is important to acknowledge that the reinforcement learning D3QN algorithm employed in this paper primarily focuses on value-based methods, which are not capable of directly optimizing the scheduling strategy. Consequently, we intend to delve into policy-based methods, such as A3C, TD3, TRPO, and other algorithms. By doing so, we can compare these approaches with our proposed scheduling algorithm based on D3QN and assess their effectiveness.

Acknowledgment

This work was supported by National Key Research and Development Program of China (2021YFB1716200) and Research Funds for Leading Talents Program (048000514122549).

References

Aydin, M. E., & Öztemel, E. (2000). Dynamic job-shop scheduling using reinforcement learning agents. *Robotics and Autonomous Systems*, *33*(2-3), 169-178.

Bellman, R. (1957). A Markovian decision process. *Journal of mathematics and mechanics*, *6*(5), 679-684.

Brucker, P., & Schlie, R. (1990). Job-shop scheduling with multipurpose machines. *Computing*.

Burggräf, P., Wagner, J., Saßmannshausen, T., Ohrndorf, D. & subramani, K. (2022). Multi-agent-based deep reinforcement learning for dynamic flexible job shop scheduling. Procedia *CIRP, 112*, 57-62.

Chang, J., Yu, D., Hu, Y., He, W., & Yu, H. (2022). Deep reinforcement learning for dynamic flexible job shop scheduling with random job arrival. *Processes*, *10*(4), 760.

Chao, L.-F. & Lapaugh, A. (1993). Rotation scheduling: A loop pipelining algorithm. *Proceedings of the 30th international Design Automation Conference*, 566-572.

Gao, Y., Rong, H., & Huang, J. Z. (2005). Adaptive grid job scheduling with genetic algorithms. *Future Generation Computer Systems*, *21*(1), 151-161.

Garey, M. R., Johnson, D. S., & Sethi, R. (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of operations research*, *1*(2), 117-129.

Gonçalves, J. F., de Magalhães Mendes, J. J., & Resende, M. G. (2005). A hybrid genetic algorithm for the job shop scheduling problem. *European journal of operational research*, *167*(1), 77-95.

Han, B. A., & Yang, J. J. (2020). Research on adaptive job shop scheduling problems based on dueling double DQN. *IEEE Access*, *8*, 186474-186495.

Jianfang, C., Junjie, C., & Qingshan, Z. (2014). An optimized scheduling algorithm on a cloud workflow using a discrete particle swarm. *Cybernetics and Information Technologies*, *14*(1), 25-39.

Lee, Z., Wang, Y., & Zhou, W. (2011, August). A dynamic priority scheduling algorithm on service request scheduling in cloud computing. In *Proceedings of 2011 International Conference on Electronic & Mechanical Engineering and Information Technology* (Vol. 9, pp. 4665-4669). IEEE.

Liu, R., Piplani, R., & Toro, C. (2022). Deep reinforcement learning for dynamic scheduling of a flexible job shop. *International Journal of Production Research*, *60*(13), 4049-4069.

Liu, Z., Chen, W., Zhang, C., Yang, C., & Cheng, Q. (2021). Intelligent scheduling of a feature-process-machine tool super network based on digital twin workshop. *Journal of manufacturing systems*, *58*, 157-167.

Lin, T. L., Horng, S. J., Kao, T. W., Chen, Y. H., Run, R. S., Chen, R. J., ... & Kuo, I. H. (2010). An efficient job-shop scheduling algorithm based on particle swarm optimization. *Expert Systems with Applications*, *37*(3), 2629-2636.

Luo, S. (2020). Dynamic scheduling for flexible job shop with new job insertions by deep reinforcement learning. *Applied Soft Computing*, *91*, 106208.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529-533.

Ouelhadj, D., & Petrovic, S. (2009). A survey of dynamic scheduling in manufacturing systems. *Journal of scheduling*, *12*, 417-431.

Qu, S., Wang, J., & Shivani, G. (2016, September). Learning adaptive dispatching rules for a manufacturing process system by using reinforcement learning approach. In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)* (pp. 1-8). IEEE.

Rahmani Hosseinabadi, A. A., Vahidi, J., Saemi, B., Sangaiah, A. K., & Elhoseny, M. (2019). Extended genetic algorithm for solving open-shop scheduling problem. *Soft computing*, *23*, 5099-5116.

Shahrabi, J., Adibi, M. A., & Mahootchi, M. (2017). A reinforcement learning approach to parameter estimation in dynamic job shop scheduling. *Computers & Industrial Engineering*, *110*, 75-82.

Shi, D., Fan, W., Xiao, Y., Lin, T., & Xing, C. (2020). Intelligent scheduling of discrete automated production line via deep reinforcement learning. *International journal of production research*, *58*(11), 3362-3380.

Shiue, Y. R., Lee, K. C., & Su, C. T. (2018). Real-time scheduling for a smart factory using a reinforcement learning approach. *Computers & Industrial Engineering*, *125*, 604-614.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, *529*(7587), 484-489.

Song, W., Chen, X., Li, Q., & Cao, Z. (2022). Flexible Job-Shop Scheduling via Graph Neural Network and Deep Reinforcement Learning. *IEEE Transactions on Industrial Informatics*, *19*(2), 1600-1610.

Sutton, R. S. & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Tassel, P., Gebser, M., & Schekotihin, K. (2021). A reinforcement learning environment for job-shop scheduling. *arXiv preprint arXiv:2104.03760*.

Wang, H. X., & Yan, H. S. (2016). An interoperable adaptive scheduling strategy for knowledgeable manufacturing based on SMGWQ-learning. *Journal of Intelligent Manufacturing*, *27*, 1085-1095.

Wang, L., Hu, X., Wang, Y., Xu, S., Ma, S., Yang, K., ... & Wang, W. (2021). Dynamic job-shop scheduling in smart manufacturing using deep reinforcement learning. *Computer Networks*, *190*, 107969.

Wang, Y. C., & Usher, J. M. (2004). Learning policies for single machine job dispatching. *Robotics and Computer-Integrated Manufacturing*, *20*(6), 553-562.

Wang, Y. F. (2020). Adaptive job shop scheduling strategy based on weighted Q-learning algorithm. *Journal of Intelligent Manufacturing*, *31*(2), 417-432.

Wang, Z., Schaul, T., Hessel, M., Hasselt, H., Lanctot, M., & Freitas, N. (2016, June). Dueling network architectures for deep reinforcement learning. In *International conference on machine learning* (pp. 1995-2003). PMLR.

Waschneck, B., Reichstaller, A., Belzner, L., Altenmüller, T., Bauernhansl, T., Knapp, A., & Kyek, A. (2018, April). Deep reinforcement learning for semiconductor production scheduling. In *2018 29th annual SEMI advanced semiconductor manufacturing conference (ASMC)* (pp. 301-306). IEEE.

Wei, Y., & Zhao, M. (2004, December). Composite rules selection using reinforcement learning for dynamic job-shop scheduling. In *IEEE Conference on Robotics, Automation and Mechatronics, 2004.* (Vol. 2, pp. 1083-1088). IEEE.

Zhang, W., & Dietterich, T. G. (1995, August). A reinforcement learning approach to job-shop scheduling. In IJCAI (Vol. 95, pp. 1114-1120).

Yang, H., & Yan, H. (2007, August). An adaptive policy of dynamic scheduling in knowledgeable manufacturing environment. In *2007 IEEE International Conference on Automation and Logistics* (pp. 835-840). IEEE.

Zhang, C., Song, W., Cao, Z., Zhang, J., Tan, P. S., & Chi, X. (2020a). Learning to dispatch for job shop scheduling via deep reinforcement learning. *Advances in Neural Information Processing Systems*, *33*, 1621-1632.

Zhang, G., Hu, Y., Sun, J., & Zhang, W. (2020b). An improved genetic algorithm for the flexible job shop scheduling problem with multiple time constraints. *Swarm and Evolutionary Computation*, *54*, 100664.

Zhang, S., Wu, Y., Ogai, H., Inujima, H., & Tateno, S. (2021). Tactical decision-making for autonomous driving using dueling double deep Q network with double attention. *IEEE Access*, *9*, 151983-151992.

Zhang, Y., Zhu, H., Tang, D., Zhou, T., & Gui, Y. (2022). Dynamic job shop scheduling based on deep reinforcement learning for multi-agent manufacturing systems. *Robotics and Computer-Integrated Manufacturing*, *78*, 102412.

Zhao, F., Qin, S., Yang, G., Ma, W., Zhang, C., & Song, H. (2019). A factorial based particle swarm optimization with a population adaptation mechanism for the no-wait flow shop scheduling problem with the makespan objective. *Expert Systems with Applications*, *126*, 41-53.