Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

Deep Q-Learning Based Cross-Cell Scheduling Problem Under Flexible Paths

Yingzhi Kong
School of Management
University of Shanghai for Science and Technology
Shanghai, China

Jing Ni
School of Management
University of Shanghai for Science and Technology
Shanghai , China

***Abstract*—For the problem that the genetic algorithm solution process is blind and it is difficult to guide the algorithm to optimize the ideal solution space, a deep reinforcement learning algorithm is designed to better solve the dual-objective scheduling problem composed of completion time and energy consumption. Firstly, based on the characteristics of the cross-cell scheduling problem, 9 state spaces, and 12 composite scheduling rules are designed as the underlying basis of the algorithm, and the number of cross-cell transportation is integrated into the agent state space and scheduling rules. Secondly, for the characteristics of large list dimension and poor generalization ability of deep reinforcement learning environment, the dual network layer neural network structure of action main network and target network is introduced. Then, a cumulative reward function that can express the target features is designed for the dual objectives of completion time and total energy consumption. This function helps the agent to obtain more accurate environmental reward values based on reflecting multiple objectives. Finally, the solution effect of the algorithm under different weights on the problem is analyzed by an example experiment, which shows that the designed algorithm can effectively solve the multi-objective scheduling problem proposed in this paper. It verifies that the algorithm can guide the agent to explore in the direction of optimization under various composite scheduling rules.**

***Keywords*—composite scheduling rule; cross-cell scheduling; deep Q-learning; dual-network deep Q-network; energy-saving optimization**

I. INTRODUCTION

With the continuous evolution of the industrial environment and the increasing demand for production flexibility, companies are increasingly focusing on multi-objective optimization in the production process to achieve efficient manufacturing. In this context, the cross-cell scheduling problem faces severe challenges due to its complex constraints and diverse optimization objectives. Traditional heuristic algorithms (e.g., genetic algorithms) rely on empirical rules and local searches, which make it difficult to effectively balance the dynamic trade-offs between the number of cross-cell transports and energy consumption in a dynamic production environment. In addition, most of the existing research focuses on single-objective or static scenarios and lacks real-time optimization capability for multi-objective dynamic scheduling.

In recent years, Deep Reinforcement Learning (DRL) has shown great potential in complex scheduling problems. DRL is capable of dynamic decision-making and global optimization through the interactive learning between intelligence and the environment. In related research, Han and Yang[1] proposed an adaptive shop floor scheduling method based on Dueling Double DQN , which optimizes the accuracy of action value assessment in complex decision-making scenarios by separating the state value function and action advantage function. Luo [2] designed a dynamic flexible scheduling framework based on DQN for the real-time requirement of new job insertion and proposed a â€˜flexible priorityâ€™ mechanism to dynamically adjust the scheduling order of new jobs and in-process tasks. Zhao et al. [3] used deep Q-networks to process high-dimensional state spaces and designed a cross-cell transport time prediction

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

module to optimize path selection to reduce transport energy consumption. Luo et al. [4] further improved the deep reinforcement learning framework by proposing a 'dynamic priority buffer' mechanism to prioritize the learning of critical scheduling events (e.g., device conflicts). Du et al. [5] incorporate crane transport time and equipment preparation time into the scheduling model, design an end-to-end DRL framework, and propose a 'joint time-energy reward function' to dynamically adjust the weights of transport and processing time. Xiao Pengfei et al. [7] studied the non-displacement flow shop scheduling problem based on deep reinforcement learning, proposed a multi-objective optimization framework, and achieved adaptive scheduling for complex production environments through deep Q-networks. Zhu et al. [8] proposed a deep reinforcement learning-based real-time scheduling optimization method for cloud manufacturing, which achieves efficient scheduling of large-scale production tasks through multi-intelligence collaboration.

However, most existing methods ignore the influence of flexible paths on state space and action selection in cross-cell scheduling and fail to effectively integrate the synergistic optimization of energy consumption and time objectives. Aiming at the above problems, this paper proposes a cross-cell scheduling method based on Deep Q-Learning (DQN), which comprehensively describes the dynamic characteristics of the production environment by designing a 9-dimensional state space and 12 kinds of composite scheduling rules. It combines the dual-network architecture with an adaptive rewarding mechanism to improve the algorithm's coordination ability in multi-objective conflicts. The experimental results show that the proposed method significantly outperforms the traditional heuristic algorithm in global energy optimization and real-time decision-making ability, which provides a new idea for energy-saving scheduling in flexible manufacturing systems.

II. PROBLEM DESCRIPTION AND MODELLING

A. Description of the problem

The cross-cell scheduling problem under flexible paths is defined as follows: the set of n workpieces to be machined i = {1,2,...,n}, the set of processes for workpiece i is j = {1,2,..., O}, the factory has a set of production cells u = {1,2,...,c}, and the cell has a set of equipment k = {1,2,...,m} and the equipment is capable of processing multiple types of workpieces, the scheduling process is as follows: the process j of the workpiece I needs to be scheduled to the equipment k of the production cell u to complete the processing, the initial start time of the workpiece set S_i is 0, and the completion time is determined by the completion time CK_k of the last machine. The workpieces have multiple flexible machining paths, which are composed of paths between numerous machines and between numerous cells. A complete scheduling process is represented when all parts in the set are scheduled.

B. Modelling

The optimization objectives in this chapter address the global completion time and total energy consumption as shown in Equations (1) and (2)

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

$$F_1 = \min CK_{max} \tag{1}$$

$$F_2 = \min E_t = \min(\sum_{k}^{m} E_{ku}^{pro} + \sum_{k}^{m} E_{ku}^{idle} + \sum_{u}^{c} E_{uu'}^{trans}) \tag{2}$$

Since the concept of the Pareto solution set is not introduced in the proposed deep Q-learning, the weighting method[5]is chosen to convert the bi-objective into a single objective for solving, as shown in Equation (3).

$$F = \min\{\varpi F_1 + (1-\varpi)F_2\} \tag{3}$$

Constraint Conditions:

$$C_{ij} = S_{ij} + p_{ijk} * x_{ijk} + t_{uu'} * z_{j,j+1}, S \geq 0, \forall i, j, k, u \tag{4}$$

$$CK_{ij} = SK_{ku} + t_{ijk} * x_{ijk}, SK_{ku} \geq 0, \forall i, j, k, u \tag{5}$$

$$S_{i,j+1} \geq CK_{ij}, \forall i, j \tag{6}$$

$$SK_{i,i+1} \geq C_i, \forall i, j \tag{7}$$

$$CK_{i,i+1,k} - CK_{ik} \geq p_{i+1,jk} * y_{i,i+1}, \forall i, j, k \tag{8}$$

$$x_{ijku} = 1, \forall i, j, k, u \tag{9}$$

$$y_{i,i+1} = 1, \forall i, j, k \tag{10}$$

Eq. (4) is the expression for the completion time of any workpiece, where C_{ij} is the completion time of process j of workpiece i, S_{ij} refers to the start time of the process, $t_{uu'}$ is the transport time of the workpiece between cells, p_{ijk} is the processing time of process j of workpiece i on equipment k, and x_{ijk} is a decision variable indicating that process j of workpiece i is assigned to equipment k. Eq. (5) is the expression for the completion time of any workpiece processed by any equipment, where $z_{j,j+1}$ expresses that the workpiece generates cross-cell transport, and SK_{ku} and CK_{ku} respectively represent the current start time and completion time of equipment. The purpose of Eq. (5) is to separate the constraints of the workpieces and the equipment explicitly so that the algorithmic decoding process can make judgment directly according to the constraints of the equipment; Eq. (6) denotes the sequential relationship between the workpieces of the same workpiece; and Eq. (7) represents the sequential relationship between the last two workpieces of any equipment. The sequence relationship between the two processes on any equipment; Equation (8) indicates that there may be idle time for the two processes before and after any equipment, where $CK_{i,k}$and $CK_{i+1,k}$ indicate the completion time of the two processes before and after equipment k; Equation (9) indicates that any process of any workpiece can only be selected to complete the machining of a cell of a piece of equipment at the same time; Equation (10) indicates that the front and back of the equipment at a certain moment in the process. Only one workpiece can be selected for processing.

III. SCHEDULING PROCESS BASED ON MARKOV DECISION

A. Markov decision process of cross-cell scheduling under flexible path

The operation logic of deep Q-learning is the interaction between the intelligent body and the environment, and in solving the scheduling problem, the problem is first transformed into a Markov Decision Process (MDP), which consists of five elements, S, A, P, R,γwhich are the state space, the action, the probability of state transfer, the value of the reward function, and the rate of loss of the Q-value, respectively, and the MDP in deep Q-learning can be specifically described as follows: the intelligent body has an initial state and selects an action according to the current state. State and select an action based on the current state, after which the intelligent body receives feedback from the environment about the reward value and the next state. It then uses its previous experience to update its policy, stores its experience in an experience playback area, and uses this experience to update its Q-function values. Where γ is with what probability the intelligent body receives the reward value.

In the cross-cell scheduling problem based on MDP, it can be viewed as an intelligent body scheduling a process to be processed on one machine within a cell based on the reward value and the environment feedback Q-value, after the selection, the intelligent body will update the Q-value based on the reward value of the environment feedback, and select the next action based on the updated Q-value, the current shop floor state is computed from the state-space function, and to make the operation of the MDP it is necessary to Define the shop floor state function, scheduling rules and target function reward value. Figure 1 shows the flowchart of MDP for cross-cell scheduling.

Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

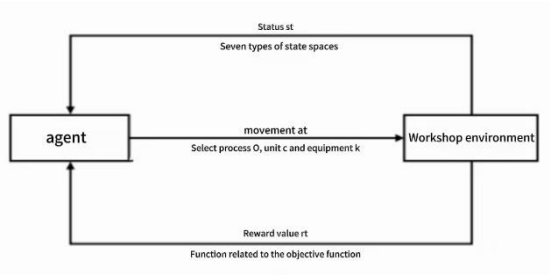


Figure 1 MDP for cross-cell scheduling

B. state space

State space is a mathematical representation of the characteristics of the scheduling environment. Accurate state space can reflect the specific situation of each object in the production environment at each time point. In the cross-cell scheduling problem under the flexible path, the environment includes three state objects: workpiece, cell, and equipment. The flexible path is an important feature of the scheduling problem. The number of cross-cells can reflect the flexible path between cells. Each cross-cell transportation will produce a period of transportation time, thereby increasing the transportation time and idle time, which will have an impact on the production system. For cell utilization and cross-cell times, this paper also designs two state functions. Combined with time and energy consumption factors, a total of 9 features are set to reflect the state of the scheduling process, of which (1) (2) (3) (4) is the feature of the job design, (5) (6) (7) (8) is the feature of the equipment, and (9) is the cell. The relevant symbol description is shown in Table Ⅰ.

Table Ⅰ Explanation of the meaning of symbols

digital character	implication
P_{ijk}	Processing time
n	Total number of workpieces
O	Total number of processes
n_i	Workpiece indexing
op_j	Process Index
m	Total number of equipment
k	Equipment Index
c	Total number of units
u	Unit Index
Opt_i	Total number of completed processes
Sop_{ij}	The start time of the workpiece
Cop_{ij}	The end time of the workpiece
R_{ij}	Number of cross-unit transactions
CV_k	Completion time of the equipment
SK_k	The start time of the equipment

(1) The average completion rate $CJ_{favg}(t)$ of the workpiece reflects the processing progress of the workpiece at t time.

$$CJ_{fin}(t)=\frac{Opt_i(t)}{n} \tag{11}$$

$$CJ_{favg}(t)=\frac{\sum_{i=1}^{n}CJ_{fin}(t)}{n_i} \tag{12}$$

(2) The variance $DJ_{iavg}(t)$ of the ratio of the idle time of the job to the total process time reflects the idle time of the job.

$$DJ_{idle}(t)=\frac{\sum_{j=2}^{O}Sop_{ij}-Cop_{i(j-1)}}{Cop_{i(j-1)}} \tag{13}$$

$$DJ_{iavg}(t)=\sqrt{\frac{1}{n}(\sum_{i=1}^{n}DJ_{idle}(t))^2} \tag{14}$$

(3) The ratio of the number of cross-cell transportation of the workpiece and the total completion process of the workpiece $TJ_{atrans}(t)$ reflects the workpiece in the single cell.

$$TJ_{trans}(t)=\frac{\sum_{j=2}^{n}R_{ij(j-1)}}{op_i} \tag{15}$$

$$TJ_{atrans}(t)=\frac{\sum_{i=1}^{n}TJ_{trans}}{n} \tag{16}$$

(4) The ratio of the current completion time of the workpiece to the total processing time of the workpiece, $CPJ_{avg}(t)$, mainly reflects the difference in completion time between workpieces. Equation (17) represents the average value of the machining time of all feasible equipment for a process, which is used to estimate the approximate machining time required for this process.

$$Aop_j=\frac{\sum_{om=1}^{Om}p_{om}}{Om_j} \tag{17}$$

$$CPJ_i(t)=\frac{CJ_{fin}(t)}{\sum_{j=1}^{O}Aop_j} \tag{18}$$

$$CPJ_{avg}(t)=\frac{\sum_{i=1}^{n}CPJ_i}{n} \tag{19}$$

(5) The average equipment utilization rate $V_{kavg}(t)$ reflects whether the equipment is fully utilized. When the utilization rate

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

of a device is large, the load of the device may be too high, which is at a bottleneck, while other devices have been idle. Therefore, balancing the utilization rate of the device can effectively reduce the completion time.

$$V_k(t)=\frac{\sum_{i=1}^{n}\sum_{j=1}^{O}p_{ij}}{CV_k(t)} \tag{20}$$

$$V_{kavg}(t)=\frac{\sum_{k=1}^{m}V_k(t)}{m} \tag{21}$$

(6) Compared with the mean value, the variance $V_{kvar}(t)$ of equipment utilization can better let the agent know whether the equipment is charged or not.

$$V_{kvar}(t)=\sqrt{\frac{\sum_{m=1}^{k}(V_k(t)-V_{kavg}(t))^2}{m}} \tag{22}$$

(7) The average machine idle time rate TV_{tavg} reflects the idle time difference of the equipment processing process. This state space function can more accurately reflect which equipment is idle at the current time than (5).

$$T_{idle}(t)=CV_k(t)-\sum_{i=1}^{n}\sum_{j=1}^{op_i(t)}p_{ijk}-SK_{ijk} \tag{23}$$

$$TV_{avg}(t)=\frac{\sum_{k=1}^{m}T_{idle}(t)}{m} \tag{24}$$

(8) The average maximum energy consumption ratio of the equipment reflects the difference between the energy consumption indicators of the equipment during processing.

$$VCE(t)=\sum_{i=1}^{n}\sum_{j=1}^{op_i(t)}(p_{ijk}pe_k)+\sum_{k=1}^{m_k(t)}(T_{idle}ie_k)+\sum_{i}^{n}(\sum_{k}^{m_k(t)}R_{ij}*t_u*tu_k) \tag{25}$$

$$VCE_{avg}(t)=\frac{\sum_{k=1}^{m}VCE(t)}{m}/max(VCE(t)) \tag{26}$$

(9) Cell utilization variance $U_{uavg}(t)$ reflects the utilization rate of all the equipment in the cell and helps the intelligent body to judge whether the cell is in a loaded state or not. In the cross-cell scheduling problem under the flexible path, the reasonableness cannot be observed only from the utilization rate of the equipment. When the utilization rate of the cell is too high, it means that the equipment in the cell has a strong processing capacity. It is possible that all the equipment in the cell is scheduled for multiple processes, while there is a lot of idle equipment in other cells. A cell utilization perspective gives a better picture of the global processing situation.

$$U_u(t)=\frac{\sum_{k=1}^{m}\sum_{i=1}^{n}\sum_{j=1}^{op_i(t)}p_{ijk}}{CT_u(t)} \tag{27}$$

$$U_{uavg}(t)=\frac{\sum_{u=1}^{c}U_u(t)}{c} \tag{28}$$

C. Action spaces

After selecting an action, the intelligent body selects the action based on the state and strategy of the current time step and sends the selected action to the environment. The environment performs the appropriate action based on the action and gives back to the intelligent body a message containing the reward value and the state of the next time step. The intelligence will update the policy and value function based on this information and select the action based on the updated policy and value function at the next time step. The scheduling rules designed in the scheduling problem enable the intelligent body to learn that it should select the specific artifacts and equipment, and the scheduling rules are considered an important stage in the trial and error phase of the intelligent body, which should satisfy all the constraints in the problem. Q Learning is where the intelligent body obtains the link between the current action for each goal by accumulating learning experience through the judgment of the state matrix. The scheduling rules in traditional shop floor scheduling problems involve only the selection of workpieces and equipment so that the intelligent body learns the advantages and disadvantages of the action at that moment in time, and in cross-cell scheduling the information of the cell also interacts with the environment, so the scheduling rules applied should contain three levels of workpieces, cells and equipment. Most of the researches use heuristic scheduling rules as actions, this choice is difficult to satisfy the complex scheduling problems, and the composite scheduling rules have better performance in complex scheduling environments[10]. Therefore, in this paper, we first design some heuristic scheduling rules for only a single level and combine them into composite scheduling rules. The logic of composite scheduling

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

rules is to select one from the workpiece scheduling rules, cell scheduling rules, and equipment scheduling rules for combination, so a total of 12 kinds of composite scheduling rules of 2*3*3 can be selected, for example, in the order of selecting (1) (2) (1), the composite scheduling rule selects the workpiece with the smallest average machining time and arranges it to the lowest utilization equipment in the lowest loaded cell. The logic diagram of the specific combination of scheduling rules is shown in Figure 2

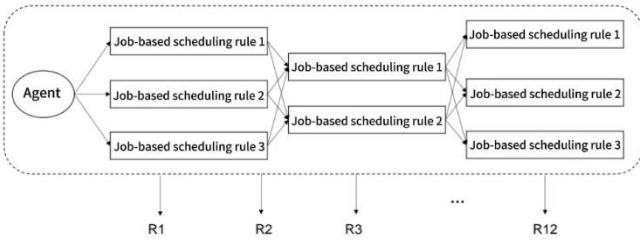


Fig 2 Scheduling rule selection Combinatorial logic diagram

Workpiece-based scheduling rules:

(1)Preference is given to the workpiece with the shortest average processing time for the next process, which allows for fast completion of all processes for some workpieces in the pre-scheduling phase.

$$J_i \leftarrow \arg\min \sum_{j=1}^{op_i(t)} Aop_j(t) \tag{29}$$

(2)Priority is given to workpieces with the smallest number of cross-cell transports, reflecting the current completion time of the workpiece based on the number of cross-cell transports; the lower the number of transports, the lower the resulting transport time, and the lower the possible completion time and lower the energy consumption.

$$J_i \leftarrow \arg\min \sum_{i=1}^{n} \sum_{j=1}^{op_i(t)} R_{ij}(t) \tag{30}$$

(3) Prioritize the workpiece with the largest number of remaining processes to avoid some of the workpieces always not being arranged.

$$J_i \leftarrow \arg\min(O_i - op_i) \tag{31}$$

Device-based scheduling rules :

(1) give priority to the selection of processing capacity of the strongest equipment, processing capacity of the equipment needs to ensure that it is fully.

$$V_i \leftarrow \arg\max(\sum_{i=1}^{n} \sum_{j=1}^{op_i(t)} Om_{ij}) \tag{32}$$

(2) give priority to choose the equipment utilization rate of the lowest equipment, avoid some equipment is always idle.

$$V_i \leftarrow \arg\min V_k(t) \tag{33}$$

(3) Prioritize the equipment with the smallest processing energy consumption. This rule allocates a smaller processing energy consumption each time.

$$V_i \leftarrow \arg\min \sum_{k=1}^{m} \sum_{i=1}^{n} \sum_{j=1}^{op_i(t)} (p_{ijk} * p_{ek}) \tag{34}$$

Cell-based scheduling rules :

(1) The equipment with the smallest cell load is preferred, and the small cell load represents the possibility of equipment in the cell.

$$V_i \leftarrow \arg\min U_u(t) \tag{35}$$

(2) Select the cell with the largest idle energy consumption, and reduce multiple devices from the perspective of the cell.

$$V_i \leftarrow \arg\min \sum_{u=1}^{c} \sum_{k=1}^{uk} (VCE_k(t) - \sum_{i=1}^{n} \sum_{j=1}^{op_i(t)} (R_{ij} * t_u * tu_k)) \tag{36}$$

D. Reward function

The setting of the reward function can reflect the objective function in the optimization problem, which is used to assess the goodness of the intelligent body's action selection, and the reward function is divided into the instant reward function and cumulative reward function [6], the instant reward function can quickly reflect the absolute gain of the current objective, but it is impossible to observe the historical reward, which makes the intelligent body blind to the backward selection, and the reward value is too high or too low, which will result in the algorithm gradient updating is too slow to affect the algorithm's Efficiency. Since the actual values of completion time and total energy consumption can only be calculated once each time, if the completion time and energy consumption values are used as the immediate rewards, they will become very sparse and make it difficult for the DQN algorithm to converge [10]. In this paper, we propose a cumulative reward function considering the completion time and energy consumption, which evaluates the

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

merit of action selection by weighting the two objective functions of the current time step and the previous time step, as shown in Equation (39), and the derivation process is shown in Equation (37) and (38).

$$R=\sum_{t=1}^{T} r_t=\sum_{t=1}^{T}(\varpi_1 r_t^{CK}+\varpi_2 r_t^{CE}) \tag{37}$$

$$\begin{aligned} R&=\sum_{t=1}^{T}\varpi_1(CK(t-1)-CK(t))+\sum_{t=1}^{T}\varpi_2(CE(t-1)-CE(t)) \\ &=\varpi_1(CK(1)-CK(t-1))+\varpi_2(CE(1)-CE(T+1)) \end{aligned} \tag{38}$$

Where ϖ_1 and ϖ_2 are the two weights of the current device completion time and energy consumption, CK is the completion time of the time step, and CE is the energy consumption of the time step. Since the smart body does not perform any action in the initial step, the initial values of CK(1) and CE(1) are both 0. Therefore, the R-value is further simplified to Equation (39):

$$R=-(\varpi_1 C_{max}+\varpi_2 E_{total}) \tag{39}$$

Through formula (39), it can be found that the cumulative reward value is the negative sum of completion time and total energy consumption at time t, so the reward function is negative.

IV. DUAL NETWORK DQN ALGORITHM

Deep Q Networks combines reinforcement learning algorithms for deep neural networks and Q learning. The neural network is used to approximate the Q function in reinforcement learning, taking the state and action space as neural network inputs and outputting the corresponding Q values. The deep network contains experience playback technology, which is used to store the historical experience of the intelligent body and randomly select the experience book for practice, which effectively improves the stability of the learning ability of the algorithm.

Q-learning belongs to a branch of reinforcement learning. During the scheduling process, the intelligent body Q puts different states and actions into a Q list, and the interaction process between the intelligent body and the environment will eventually output a Q function to select the optimal action.

The main idea of the DDQN algorithm is to replace the list of Q learning with a neural network, for large scale problems, the Q list increases massively in dimension with the scheduling process leading to computational inefficiency, the use of neural networks aims to avoid this problem. The basic DQN algorithm implements the decision making process based on a single Q network but still has limitations for large amounts of data [4], DDQN introduces an important improvement by using two Q networks, the first Q network is the master network which selects an action based on the current state and the network that estimates the Q value is called the target network which calculates the Q value based on the action selected by the action network .DDQN reduces the dimensionality disaster by separating it from the two networks. DDQN reduces the problems of dimensional catastrophe and overestimation of Q-values by separating them from the two networks, and the loss function expression for the target network is shown in Equation (40).

$$L(\theta)=E[(r+\gamma \max Q(s',a',\theta')-Q(s,a,\theta))^2] \tag{40}$$

For the cross-cell scheduling problem under the flexible path, this paper designs 9 scheduling states as inputs and 12 scheduling rules as outputs to the DDQN, and selects the appropriate composite scheduling rules according to the priorities of the rules for the workpieces, cells, and equipment, and arranges the work processes to specific cells and equipments. The neural network will feed the Q-value after each action to the intelligent body, and train the target network through the historical experience stored in the experience pool. Therefore, DDQN will fit the network according to different arithmetic cases to determine the optimal scheduling rules, and the optimal scheduling rules in the changing process, thus adapting to more scheduling environments and better real-time scheduling capability. The logic diagram of DDQN is shown in Fig.3.

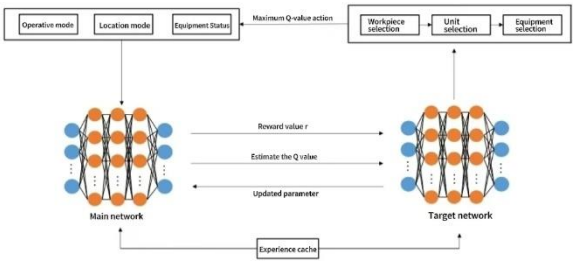


Fig. 3 DDQN logic diagram

A. Neural network

In this paper, we utilize two BP neural networks, with the same structure as the action network and the target network, respectively, as illustrated in Figure 3. Each network possesses a three-layer structure, comprising input, hidden, and output layers. The fundamental characteristic of BP neural networks is the treatment of each neuron as a non-linear function. The

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

combination of multiple neurons, contingent on the size of the hidden layer, facilitates the execution of forward and backpropagation. The features of the BP network are simple and obvious from input to output, and there is no need to extract additional features manually. In this paper, the input layer is 9, representing 9 kinds of state spaces; the output layer is 12, representing 12 kinds of composite scheduling rules. The Leaky ReLu function [10] is selected as the activation function of the neural network, a choice that has been demonstrated to effectively address the issue of gradient disappearance whilst enhancing the model's generalization capabilities for non-linear characteristics. Furthermore, the Leaky ReLu function circumvents the problem of neuron inactivity when the neuron input is 0. The specific formula of function is (41)

$$f(x)=\begin{cases} x\ ,\ x>0 \\ ax, x\le 0 \end{cases} \tag{41}$$

B. Discovery Strategy

A search policy in deep Q learning is a way to find the optimal Q value in the state space, specifically denoting the adoption of some scheduling rule to maximize the cumulative reward in a given state. In this paper, we choose the ε − greedy search policy function, ε − greedy selects actions according to the maximum Q value but also selects random actions with a certain probability to avoid choosing the same action every time, which can prevent the algorithm from falling into the local optimum to a certain extent, and the mathematical expression of the greedy policy is shown in (42).

$$Action=\begin{cases} \arg\max Q(a) & ,p=1-\varepsilon \\ Random\ Action, p=\varepsilon \end{cases} \tag{42}$$

C. Experience replay

The neural network framework is used in deep Q-learning, and the experience playback technique is an important training technique in neural networks, which can help the intelligent body to draw experience learning from the previous scheduling process and improve the generalization ability of the neural network model. The intelligent body stores the tuple data of the production state, scheduling rules, accumulated rewards, and the next state of each time step into the experience pool, and when training the main network, the intelligent body randomly draws a batch of data from the experience pool, calculates the loss function and updates the network parameters, making the training data more random and independent. The specific steps are: select a group of data from the experience pool, calculate the target network Q value of the data, and then update the network weights in a gradient descent manner. When the capacity of the experience pool reaches a threshold, the experience of the lowest time step is moved out of the experience pool to ensure that the subsequent experience is recorded.

D. Algorithm procedure

The flow chart of the DDQN algorithm proposed in this paper to solve the cross-cell scheduling problem under the flexible path is shown in Figure 4.

Step1: Initialise the algorithm parameters, including the number of algorithm iterations L, greedy probability ε, learning rate λ, experience pool size D, the number of samples selected Batch_ size, and the target network updating frequency, and lay out the action and target neural networks;

Step2: Initialize the environment state S and set the total number of processes OP;

Step3: Perform the first scheduling, initialize the first sequence of the intelligent body as well as preprocess the first state and save it to the experience pool D;

Step4: Perform scheduling rule selection based on ε greedy strategy;

Step5: Calculate the t-moment reward r_t for executing the action and save the resulting historical experience data to the experience pool;

Step6: Judge whether the size of historical experience in the experience pool exceeds D. If it exceeds, delete the first experience matrix that enters the pool according to the size of experience gained;

Step7:When the experience data reaches the learning threshold, randomly draw Batch_ size samples from the experience pool to perform the Q value calculation of the main network and the target network; if it does not reach it, then carry out new scheduling;

Step8: Train the main network and target network according to the loss function and learning rate λ;

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

Step9:When the number of learning times reaches the update frequency of the target network, the current main network is copied to the target network, and the target network is updated.

Step10: determine whether the number of scheduling processes reaches OP then determine the end of one schedule, the number of iterations L plus 1, and return to Step2; otherwise output the lowest value of the objective function and the current value of the reward function, stop the algorithm.

V. NUMERICAL EXPERIMENTS AND ANALYSIS

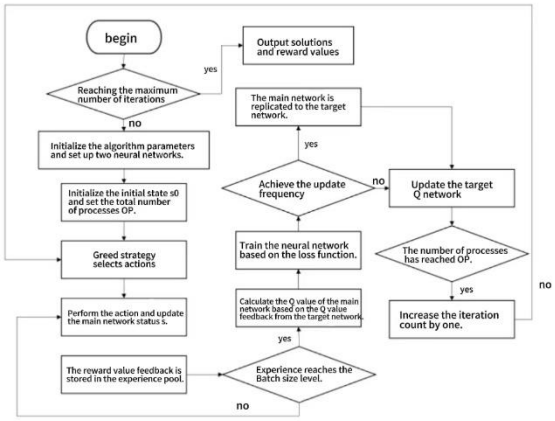


Fig 4. DDQN Flowchart

A. Example design

To verify the effectiveness of the DDQN algorithm in solving the dual-objective energy-saving flexible cross-cell scheduling problem, this chapter selects MK01-MK08 in the flexible job benchmark example library for simulation testing. The 8 examples are divided into new examples : MK01_3, MK02_2,MK03_3,MK04_3,MK05_2, MK06_4,MK07_4,MK08_5. Taking MK01_2 as an example, the scale of the example is 10*6*3, where 10 represents the number of workpieces, 6 represents the number of equipment, and 3 represents the number of cells. The equipment numbered 1,2,3 is placed with cell 1, the equipment numbered 4 is placed with cell 2, and the equipment numbered 5 and 6 is placed with cell 3. According to the simulation of the energy-saving production environment, the cell processing energy consumption pe∈[5,10]kw, the cell idle energy consumption ie∈[2,5]kw, and the cell transportation energy consumption te∈5kw.

B. Parameter selection

The experimental environment of this paper is PyCharm Python3.7, the computer hardware is Intel 1070ti GPU@8G, and the number of CUDA cores is 2432.The parameter settings of DDQN have a large impact on the performance of the algorithm, and after many experimental results, we select the reasonable parameters of the main framework parameters and neural network parameters, as shown in Table Ⅱ.

Table Ⅱ Parameter setting table

Main framework parameters	iterations $L-episode$	1000
	Greediness selection rate ε	0.001
	discount rate γ	0.6
	Size of the experience pool $Memory\_size$	2000
	Network update frequency Q_fre	200
Neural network parameters	input layer	9
	hidden layer	128*3
	output layer	12

C. Algorithm Performance Analysis

The DDQN algorithm is used to solve eight scale examples, and the specific reward values and objective functions that meet the conditions are obtained, as shown in Table Ⅲ. Secondly, the MK01_3 example is analyzed in detail. The convergence performance of the algorithm is illustrated by the changing trend of the 1000-generation reward value and the weighted target value. The effectiveness of the composite scheduling rule proposed in this paper is illustrated by 10 examples. Finally, the solution effect under three different weight selections in the objective function is analyzed.

Table Ⅲ The results of 8 groups of examples

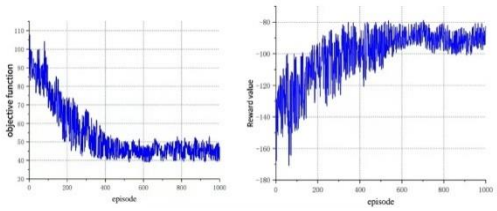
example	Weighted objective function value	Reward value	Maximum completion time	total energy consumption
MK01_3	39.725	-79.50	40.00	39.45
MK02_2	61.725	-123.85	50.00	73.45
MK03_3	162.420	-324.84	206.00	318.84
MK04_3	189.910	-379.23	178.00	201.82
MK05_2	103.925	-207.85	90.50	117.35
MK06_4	197.735	-395.25	236.00	159.47
MK07_4	1028.77	-2057.54	1180.00	877.54
MK08_5	1478.205	-2956.41	1348.00	1608.41

(1) Algorithm convergence analysis

The MK01_3 and MK03_3 algorithms are selected to illustrate the process of scheduling decision-making using the DDQN algorithm. Fig.5(a) shows the weighted objective function value of the MK01_3 algorithm after 1,000 iterations, and Fig.5(b) shows the direction of the reward value of the MK01_3 algorithm. Filtering the results of the algorithms makes the direction of the function values in each generation more clear . From Fig. 4(a), it can be seen that the algorithm gradually falls into convergence around episode=380, in which the fluctuation of the algorithm gradually decreases, indicating that the algorithm has been able to select a small number of reasonable

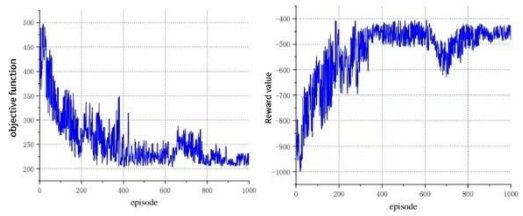
International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

scheduling rules based on the historical experience, and the algorithm goes to a stable region after episode=500. In the initial 200 generations, due to the influence of greedy strategy, the algorithm selects random scheduling action rules, so the upper limit value changes a lot, after accumulating some experience data, the scheduling intelligence can accurately select some scheduling rules suitable for the current state. Figure5(b) shows the convergence graph of the total reward value of the MK01_3 algorithm, through the reward value of chapter 2.2, the DDQN reward value of the flexible cross-cell scheduling problem is set as a negative value, which is opposite to the change of the objective function value, and the algorithm enters into the convergence gradually around 400 generations, but the reward value shows a certain downward trend after 800 generations, but the change is more stable . The algorithm gradually converges in about 400 generations, but after 800 generations, the reward value shows a certain downward trend, but the change is more stable, which corresponds to the trend of the objective function. According to the change of the reward value, the algorithm may maintain the optimal solution in one objective function but reduce the solution effect in another objective function at this stage, but the overall trend does not change much, and the algorithm shows better convergence in general.



(a)Objective function (b)Reward value

Fig.5 Iteration diagram for the MK01_3 algorithm



(a) Objective function (b) Reward value

Fig.6 Iteration diagram of MK03_3 example

Figure 6 shows the iterative graph of the MK03_3 algorithm. In Figure 6(a), the algorithm gradually goes to a steady state after episodes=400, while the reward value in Figure 6(b) shows a reverse trend in the 400th generation, and in the 600th to 800th generation, both the objective function value and the reward value fluctuate, and the algorithm may be overfitting at this time, but it returns to the convergence in the 800th generation. Convergence The weighted objective function and reward value change curves of the two sets of cases show that DDQN can form a good convergence when solving the two sets of problems, and can find a better solution set.

(2) Analysis of scheduling rule selection

Scheduling rules are an essential process of agent decision-making. Reasonable scheduling rules help agents to accurately select appropriate processes, cells, and equipment for the optimization process after accumulating specific experience. To analyze the performance of the composite scheduling rules, the scheduling rules selected by 54 processes in the MK01_3 example are counted. The total scale of the scheduling rules selected for 10 instances is 5400 times, and the distribution map of the number of uses of the composite scheduling rules is shown in Figure 7.

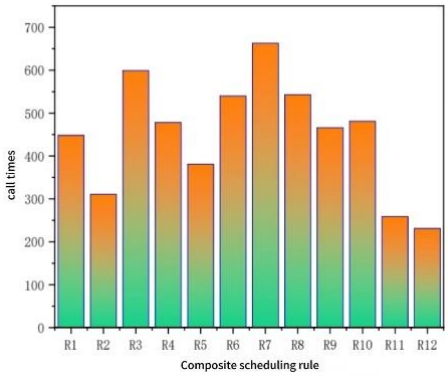


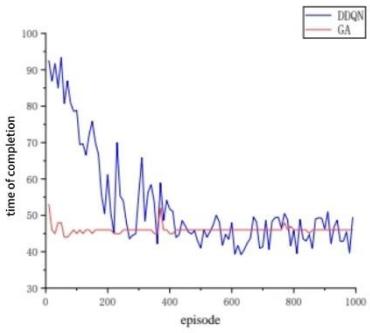
Fig.7 Composite scheduling rule distribution diagram

From Figure 7, it can be concluded that the selection times over 500 are R3, R6, R7, and R8. Among them, R3 and R8 are compound scheduling rules based on the number of times a process crosses cells, which indicates that the number of times a process crosses cells plays a strong guiding effect on the selection of DDQN intelligences when setting the workpiece scheduling rules in this paper. The R7 scheduling rule is the highest number of times an action is selected globally, and it is the cell utilization rate, The composite rule of average equipment utilization and workpiece idle time shows that the scheduling model can be solved effectively by reducing the idle time in the production environment and increasing the cell equipment utilization.

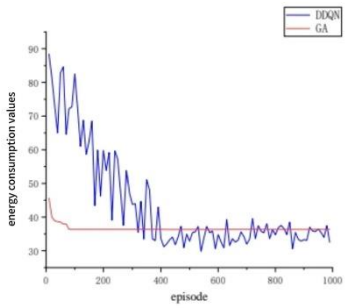
International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

D. Comparison of DDQN and heuristic algorithm

The genetic algorithm is a classical heuristic algorithm for shop scheduling problems. Due to the randomness of the crossover and mutation process of the algorithm, the iterative process is relatively blind. DDQN analyzes the results of each scheduling. From these previous scheduling results, the agent selects the next scheduling rule and has learning ability. To verify that DDQN can find the global optimal solution of completion time and energy consumption targets more accurately, this paper compares it with a genetic algorithm. The specific parameters of the genetic algorithm are set as follows: crossover rate p_c= 0.9, mutation rate p_m = 0.15, initial population pop = 500, parent population generation p_i = 200, the weight value of an objective function of the two algorithms is 0.5, and the maximum number of iterations set is episode = 1000 generations. The comparison results are shown in Figure 8. The weight values of the objective functions of the two algorithms are selected as 0.5. The comparison results are shown in Figure 8.



(a) Completion time iteration diagram



(b) Total energy consumption iteration diagram

Fig 8. Comparison of DDQN and GA

According to Fig. 8(a), it can be seen that the GA is trapped in a certain local optimum, although the iteration curve is relatively smooth, the GA found the solution set with better completion time in about 70 generations, but it did not find the solution again in the subsequent iterations, which indicates that the algorithm's optimization process is limited to a certain local optimum solution space, while the DDQN algorithm, although it does not generate an optimal solution set in the initial generation, can find a better solution set for the case according to its learning ability. Although the DDQN algorithm does not generate an optimal solution set in the initial generation, it can find a better solution set for this algorithm based on its learning ability. According to Fig. 8(b), it can be seen that the gap between the two in finding the optimal solution for total energy consumption is not too big, and the GA also enters the optimal convergence state but still fails to achieve the effect of the optimal solution set generated by DDQN. The comparison of the algorithm iterations for the two objectives shows that DDQN effectively avoids local optimums through its learning ability, and has an advantage over heuristic algorithms such as GA in solving the cross-cell scheduling problem under flexible paths.

In order to verify the performance comparison between the DDQN proposed in this chapter and the IVNSGA-II proposed in Chapter 3, the MK01 and MK03 algorithms are selected to compare the completion time and global energy consumption of the DDQN and the IVNSGA-II.

Since the MK01_3 algorithm in this chapter is divided into three cells, but the MK01 algorithm in Chapter 3 is divided into two cells, where devices 1, 2, and 3 are assigned to cell 1, and devices 4, 5 and 6 are assigned to cell 2, this paragraph selects the DDQN and IVNSGA-II algorithms for the comparison of DDQN and IVNSGA-II, to compare the DDQN and IVNSGA-II algorithms in two sets of the same size of algorithm and to obtain a more objective solving effect. And IVNSGA-II are compared with the two sets of examples MK01_2 and MK03_3, which are set up in Chapter 3, in this paragraph. The two algorithms are each run 5 times, and the number of iterations is 1000 to get the optimal mean value of completion time and the optimal mean value of global energy consumption, in which the parameters of DDQN are the same as those in Chapter 4.2, and the value of weight ϖ is set to 0.5, which means that the completion time is the same as that of global energy consumption, and the results are as shown in Table Ⅳ

Table Ⅳ MK01 _ 2 and MK03 _ 3 double targets

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

example	scale	Comparison indicators	DDQN	IVNSGA-II
MK01_2	10×6×2	ΔC_{max}	48.00	**40.00**
		ΔE_{total}	**39.112**	42.325
MK03_3	15×8×3	ΔC_{max}	208.60	**206.60**
		ΔE_{total}	**318.168**	327.58

From the comparison of the mean values of completion time and global energy consumption in Table Ⅳ, it can be seen that the DDQN algorithm's mean values of global energy consumption are better than that of the IVNSGA-II algorithm in both sets of cases, while the opposite is true in terms of completion time. From the analysis of Chapter 3 and this chapter, it can be concluded that in MK01_2, due to the small number of cells, the scheduling rule in DDQN that selects a low number of cross-cells cannot show its advantage in this algorithm, but in the optimization of the global energy consumption, DDQN, due to the high usage rate of its multiple scheduling rules based on the reduction of the idle energy consumption, makes its idle energy consumption less than that of the IVNSGA-II in the optimization of the idle energy consumption. Optimization. For IVNSGA-II, due to the existence of the variable neighborhood structure and the specificity of the model, it can achieve a very good effect on the optimization of the completion time, and therefore the mean value of the completion time of IVNSGA-II is much better than that of DDQN in the MK01_2 algorithm In MK03_3, due to the increase in the number of workpieces, DDQN can select the scheduling rule based on cell and equipment utilization several times during the higher iteration process, and the difference between IVNSGA-II and IVNSGA-II is smaller, although the completion time is still inferior to IVNSGA-II, the global energy consumption is still better than IVNSGA-II, which indicates that the learning process of DDQN algorithm based on the historical experience is conducive to finding a better solution set for the global energy consumption after several runs.

E. Analysis of DDQN solution set under different weights

The objective function proposed in this chapter encompasses the selection of weights ϖ. Three distinct selections of ϖ for the MK01_3 algorithm are employed to ascertain the impact on the objective function, and the trend of the results obtained in every 10th of the 1000th generation is utilized as an explanatory graph for the selection of weights and objectives, as illustrated in Fig.9. As demonstrated in Fig.9, the varying values of ϖ are indicative of the varying degrees of focus on the completion time and total energy consumption in the objective function. These varying degrees of focus result in different benefits for the algorithm's solution results. When ϖ=0.3, the upper limit of the weighted objective function is elevated, signifying that the selection of the ϖ as the focal point of the scheduling does not result in the global optimum. Typically, a superior solution can be attained for one objective function, while a different solution can be obtained for another. ϖ = 0.5 signifies that the DDQN is equivalent to the two optimization objectives, and a more pronounced degree of convergence can be achieved at this juncture. When ϖ = 0.7, the objective function value in the region of 150-200 generations can find some low-weighted solutions, in the 200 to 400 generations the effect is less different from ϖ = 0.5, and after 400 generations the stability is higher than ϖ = 0.5. DDQN can effectively assist the decision maker in identifying an optimal scheduling outcome, particularly in the context of concerns regarding completion time and energy consumption.

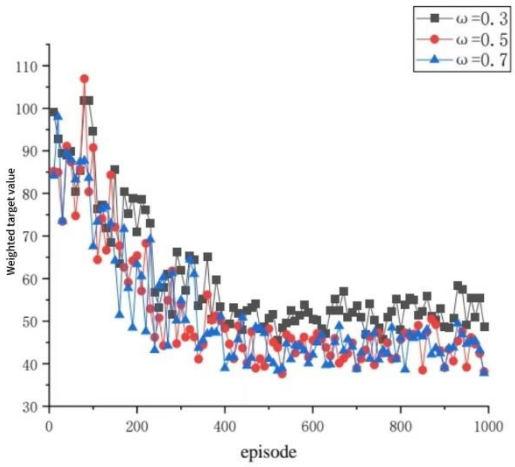


Fig 9. Convergence curves under different weights

In order to analyze the specific solution of completion time and total energy consumption under different weight values, this chapter lists the optimal worst completion time and the optimal worst total energy consumption under three weights, as shown in Table Ⅴ.

Table Ⅴ Optimal worst case values for completion time and total energy consumption

weight ϖ	Optimal completion time	The worst completion time	Optimal total energy consumption	The worst total energy consumption
0.3	42	108	38.23	121.87
0.5	38	103	39.89	121.93
0.7	36	98	42.58	123.08

In the form of a table, it is more intuitive to see the effect of different weights. When the weight ϖ is 0.7, the optimal scheduling result of the completion time is obtained, in which the DDQN pays more attention to the completion time than to

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 14 Issue 03, March-2025

the energy consumption, and when ϖ = 0.3, the optimal value of the total energy consumption can be obtained, but it is not possible to arrive at the globally optimal completion time. With a weight ϖ of 0.5, based on the worst completion time and the worst total energy consumption, it can be seen that the gap between the optimal worst is the smallest and the stability of the algorithm is the best with this weight. Therefore, for the MK01_ example, if you are concerned about the lowest value of energy consumption, you can choose a weight of 0.3, and the loss of completion time is very much; if you want to maximize the completion time, you can choose ϖ = 0.7, and you need to ignore the determination of the value of energy consumption, and if the decision maker is looking for a compromise solution, then you can set ϖ to 0.5.

VI. SUMMARY OF THIS CHAPTER

In this chapter, we seek a deep learning algorithm with better real-time performance than heuristic algorithms for the problem of multi-objective cross-cell scheduling under flexible paths and design a DDQN algorithm with a two-layer neural network. Through the coordination ability of cross-cell times for completion time and global energy consumption in Chapter 3, the idea is integrated into the state space expression and scheduling rule selection in DDQN. In the experimental part, DDQN is used to solve 10 new cases after cell division, and the analysis of MK01_3 and MK03_3 shows the effectiveness of the algorithm in solving the multi-objective IESCFR problem, and it can give the corresponding scheduling scheme under different weights of the decision-makers, and based on the comparison between the solving results of MK01_3 and the genetic algorithm, it shows that it is more suitable to solve this problem through the learning process of the intelligent body. Based on the comparison between the solution results on the MK01_3 algorithm and the genetic algorithm, it shows that the learning process through intelligence is more suitable for solving this problem. In the future, the neural network part of DDQN can be further improved to construct a better neural network model and set a more suitable search strategy and intelligent body selection action method to cope with the cross-cell scheduling problem in different environments.

REFERENCE

[1] Han B A,Yang J J. Research on Adaptive Job Shop Scheduling Problems Based on Dueling Double DQN[J]. IEEE ACCESS, 8:186474-186495, 2020.

[2] Luo S. Dynamic scheduling for flexible job shop with new job insertions by deep reinforcement learning[J]. Applied Soft Computing, 91: 106208, 2020.

[3] Zhao YJ, Wang YH, Tan YY, Zhang J, Yu H. Dynamic jobshop scheduling algorithm based on deep q network[J]. IEEE Access, 9: 122995-123011, 2021.

[4] Bin Luo, Sibao Wang, Bo Yang, Lili Yi, An improved deep reinforcement learning approach for the dynamic job shop scheduling problem with random job arrivals[C]//Journal of Physics: Conference Series. IOP Publishing, 1848(1): 012029, 2021.

[5] Du Y, Li JQ, Li CD, Duan PY. A reinforcement learning approach for flexible job shop scheduling problem with crane transportation and setup times[J]. IEEE Transactions on Neural Networks and Learning Systems, 2022.

[6] Zhong JW, Shi YQ. DQN-based intelligent factory job shop scheduling [J]. Modern Manufacturing Engineering, (09) : 17-23 + 93, 2021.

[7] Xiao PF, Zhang CY, Meng LL, Hong H, Dai W. Non-permutation flow shop scheduling problem based on deep reinforcement learning [J]. CIMS, 27 (01) : 192-205, 2021.

[8] Zhu HY, Li MR, Tang Y, Sun YF. A deep-reinforcement-learning-based optimization approach for real-time scheduling in cloud manufacturing[J]. IEEE Access, 8: 9987-9997, 2020.

[9] Qiao DP, Duan LQ, Li HL, Xiao YQ.. Job-shop scheduling problem optimization based on deep reinforcement learning [J].Manufacturing technology and machine tools, (04) : 148-155, 2023.

[10]Wang GP, Duan Meng, Niu CY. Stochastic gradient descent algorithm based on convolutional neural network [J]. Computer Engineering and Design, 39 (02) : 441-445 + 462, 2018.