Eng Int Syst (2026) 2: 221–233

Engineering
Intelligent Systems

Hierarchical Multi-Agent Deep Reinforcement Learning Architecture in Complex Industrial Production Scheduling

Haibo Peng[1,a], Guixiong Li[2,b*], Zhibo Zhang[1,c*] and Rong Zhou[2,d]

[1] *Yunnan Open University, Kunming 650500, Yunnan,China*
[2] *Baosight (Yunnan) Co., Ltd, Kunming 650500, Yunnan, China*

Complex industrial production scheduling problems are characterized by high dimensionality, dynamics, and multi-objectives. Existing scheduling methods are usually based on mathematical programming or a single-agent architecture, which have problems such as insufficient model complexity and dynamics, difficulty in dealing with high-dimensional problems, and poor scalability. In particular, they have limited performance when facing multi-task collaborative optimization and real-time environmental changes.Therefore, we designed a hierarchical multi-agent deep reinforcement learning (DRL) architecture that separates global task allocation from local execution through a hierarchical structure. The high-level deep Q-network (DQN) is used for resource allocation, while the low-level proximal policy optimization (PPO) is used to achieve fine scheduling. The centralized training with a decentralized execution (CTDE) framework is included to coordinate the behavior of multiple agents. At the same time, the graph neural network (GNN) is used to model task dependencies and design reward functions to balance short-term and long-term objectives, thereby improving scheduling efficiency, flexibility, and adaptability to uncertainty and providing an efficient and intelligent solution for complex industrial production scheduling. Experimental results show that the hierarchical multi-agent deep reinforcement learning system is significantly better than other algorithms in terms of key indicators such as task completion time, resource utilization, system delay rate, cost saving rate, and scheduling failure rate. In large-scale task scheduling, the average task completion time is reduced by 15%–25% compared with other methods, and the system delay rate is kept at around 5%. In a dynamic environment's equipment failure recovery scenario, the scheduling failure rate drops rapidly from 18% to below 10%, demonstrating its efficient global optimization and local adjustment capabilities. The research results show that the hierarchical multi-agent method provides an efficient and flexible solution for complex industrial production scheduling, which has important theoretical value and practical significance.

Keywords: Complex Industrial Production Scheduling, Deep Reinforcement Learning, Hierarchical Multi-agent, Centralized Training with Decentralized Execution, Graph Neural Network

1. INTRODUCTION

The problem of complex industrial production scheduling has become one of the core challenges in the development of modern industrial intelligence due to its high dimensionality, dynamism, and multi-objective characteristics. As Industry 4.0 and intelligent manufacturing are increasingly being implemented, the shortcomings of traditional scheduling methods based on mathematical programming or single agents have gradually emerged, including low model complexity, difficulty in solving high-dimensional problems, and poor scalability. When dealing with multi-task collaborative optimization, online environmental changes, and uncertain

*Corresponding Author. [a]Email: harpor1982@163.com, [b]18404072@masu.edu.cn, [c] 454734714@qq.com, [d]Email:8573746@qq.com

scenarios, traditional methods usually cannot flexibly meet the actual production scheduling needs. Therefore, it is necessary to build an intelligent system architecture to efficiently solve complex industrial scheduling problems, which has become a challenging problem that needs to be solved urgently in academia and industry.

With the increase in the amount of research on Industry 4.0 and intelligent manufacturing,the shortcomings of mathematical programming or single-agent methods based on traditional scheduling methods have gradually emerged. These shortcomings include low model complexity, difficulty in solving high-dimensional problems, and poor scalability. With the rapid development of artificial intelligence (AI) technology, new ideas for solving high-dimensional scheduling problems continue to be proposed. Lu Hong constructed a scheduling model that includes four decision-making links: machine allocation, processing sequence, personnel arrangement, and start-stop control. He proposed a dual-enhanced memetic algorithm (DMA) solution with a dual-enhanced local optimization strategy [1]. Liu Wei proposed a production scheduling model for manufacturing execution systems based on master-slave chains and edge computing, which executed cross-node and cross-workshop scheduling tasks, and implemented a decentralized production scheduling method to ensure data security and improve response speed [2]. Del Gallo explored the application of artificial intelligence in manufacturing systems. His study showed that particle swarm optimization, neural networks, and reinforcement learning are the most widely used technologies for solving scheduling problems [3]. Zhuang developed a network-based dynamic scheduling rule generation mechanism by applying complex network theory, extracting low-level heuristic rules from the perspective of system optimization, and transforming the automatic generation problem of heuristic rules into a multi-attribute decision-making problem [4]. Tamssaouet applied a preference model to solve the multi-objective complex shop scheduling problem, considering unavailable cycles and minimum time lag. The proposed method could provide a good solution for preferences [5]. However, existing methods still have shortcomings when dealing with complex dependencies and dynamic environments. There is an urgent need for a more efficient and flexible intelligent architecture to meet actual production needs.

Deep reinforcement learning (DRL), as a self-learning intelligent method, has shown great potential in solving complex scheduling problems. Waubert de Puiseau discussed the problem of production scheduling reliability obtained through DRL-based scheduling methods in industrial engineering, and determined the direction of current DRL research [6]. Liu proposed a method based on deep multi-agent reinforcement learning to solve the dynamic workshop job scheduling problem. His study showed that the proposed architecture significantly improved the learning efficiency [7]. Lee adopted deep reinforcement learning to handle the scheduling process for production planning, using a deep Q-network and proposed a new state, action, and reward method to optimize the scheduling strategy [8]. Zhang Ming used the DRL method to solve the dynamic scheduling problem with unexpected machine failures in a job shop manufacturing system. The method was verified in an actual dynamic production environment, and the results were better than those obtained by other methods [9]. Zhang Yi proposed a multi-agent manufacturing system based on deep reinforcement learning. The results showed that the system could obtain scheduling solutions that met various performance indicators and could efficiently and autonomously respond to resource or task disturbances [10]. These studies indicate that DRL has significant advantages in dealing with uncertainty, multi-task collaborative optimization, and real-time response in industrial scheduling, providing a new solution for complex industrial production scheduling.

Considering the high dimensionality, dynamics, and multi-objective characteristics of complex industrial production scheduling problems, in this paper, an intelligent scheduling system architecture is designed to handle multi-task collaborative optimization problems under environmental changes in real time. To this end, a hierarchical multi-agent deep reinforcement learning method is proposed. Global task allocation and local execution are divided into hierarchical structures. DQN is used in the high level for resource allocation to achieve global optimization. PPO is used at the low level to complete fine scheduling to improve local efficiency. At the same time, the CTDE framework is applied to coordinate the behaviors of multiple agents. GNN is used to model task dependencies and improve the adaptability of complex constraints. In addition, a multi-objective reward function is designed. Taking into consideration the effects of task delay, resource utilization, and cost saving factors, this function can improve system performance in the long term and balance the short-term task completion rate. The classic Job Shop Scheduling Problem (JSSP) dataset is used to complete experimental verification. This paper compares the performance of different algorithms and comprehensively evaluates the performance of the architecture in terms of task completion time, resource utilization, system delay rate, cost saving rate, and scheduling failure rate. The results show that the hierarchical multi-agent method has significant advantages in dynamic environments and provides an effective and flexible solution for complex industrial production scheduling.

2. METHODS

2.1 Hierarchical Structure

Traditional single-agent architectures have difficulty solving complex industrial production scheduling problems due to high dimensionality and dynamism. To solve this problem, the aim of this current research is to reduce the complexity of decision-making by designing a hierarchical structure to separate global task allocation and local execution, and then address the global and local problems separately to solve the complex industrial production scheduling problem. Unlike traditional methods, DQN designs a global optimization solution for global task allocation in assembly line production to maximize long-term cumulative returns. At the same time, DQN fully considers task priority, resource consumption, and task dependency. The low level uses the PPO algorithm to complete fine scheduling. Owing to the constraint mechanism

inherent in the PPO algorithm—specifically, its ability to limit the degree of deviation between the updated strategy and the original strategy during the policy optimization process—the stability and operational efficiency of local task execution are effectively safeguarded. PPO gradually optimizes the execution order and resource utilization of local tasks by dynamically adjusting the weight coefficient, thereby achieving efficient real-time scheduling within the work section.

At the high level, DQN is used for resource allocation, and its goal is to learn the global resource allocation strategy [11–12]. DQN selects the optimal action by approximating the state-action value function $Q*(s,a)$, which is formulated as:

$$Q*(s,a) = Er + \gamma \max_{a}, Q(s', a')|s, a \tag{1}$$

where s represents the current state; a represents the action; r is the immediate reward; γ is the discount factor; s' is the next state. By maximizing the long-term cumulative reward, DQN can generate a globally optimal task allocation strategy [13–14]. In practical applications, the high-level task allocation module is responsible for allocating tasks to different equipment or resources and optimizing them according to task priorities and resource constraints. In assembly line production, the high-level module can decide which tasks should be prioritized to which work sections based on order requirements and equipment state while considering the dependencies between tasks and time window constraints.

The low level uses PPO to achieve fine scheduling of local tasks [15–16]. PPO is a reinforcement learning algorithm based on policy gradient. Its core idea is to ensure the stability of updates by limiting the difference between the new policy and the old policy [17]. The objective function of PPO is:

$$L^{CLIP}(\theta) = E_t \min(r_t(\theta)A_t.clip(r_t(\theta), 1-\in, 1+\in)A_t) \tag{2}$$

where $r_t(\theta) = \frac{\pi\theta(a_t|s_t)}{\pi\theta_{old}(a_t|s_t)}$ is the probability ratio; A_t is the advantage function; $\in$ is the clipping range. By means of this formula, PPO can gradually optimize the execution order and resource utilization of local tasks while ensuring the stability of the strategy. In a specific work section, in order to ensure the maximum utilization of resources, the low-level modules can dynamically adjust the execution order of tasks according to the real-time state of the equipment and the urgency of the tasks.

The advantage of the hierarchical structure is that it can decompose complex scheduling problems into multiple sub-problems. The high-level task allocation module focuses on global optimization. The low-level task execution module focuses on local details. This division of labor not only reduces the computational burden of a single module, but also improves the overall scheduling efficiency. In addition, the hierarchical structure can better adapt to dynamic environments. In the event of equipment failure or order changes, the high-level module can quickly adjust the global task allocation strategy. The low-level module can flexibly adjust the task execution plan within the specific work section, thereby achieving an efficient response to the dynamic environment.

2.2 Multi-Agent Collaboration Mechanism

In complex industrial production scheduling, multi-task collaborative optimization requires efficient collaboration among multiple agents. To this end, this paper combines the CTDE framework and applies GNN to model task dependencies to enhance the robustness and collaboration capabilities of the system.

The core idea of the CTDE framework is to adopt a centralized information sharing mechanism in the training phase and distributed decision-making in the execution phase [18–19]. Specifically, during the training phase, each agent can not only access its own local observation o_i but also obtain information about other agents through a global state s. The joint reward function can be expressed as:

$$R_{joint} = \sum_{i=1}^{N} R_i \tag{3}$$

where R_i is the reward of the i-th agent. Through the joint reward function, the agent can learn to cooperate with other agents during the training process. In the execution phase, each agent only relies on its own local observation o_i to make decisions, which is formulated as:

$$a_i = \pi\theta(a_i|o_i) \tag{4}$$

This centralized training with a decentralized execution approach not only ensures the efficiency of training, but also improves the scalability of the system. In multi-workshop production scheduling, agents in different workshops can coordinate task allocation through the CTDE framework while maintaining independent decision-making during the execution phase, thus avoiding the communication bottleneck problem in a single-agent architecture.

This paper applies GNN to capture the dependencies between tasks. GNN models the correlation between tasks through node feature update rules [20–21]. The specific formula is:

$$h_v^{(l+1)} = \sigma(W * h_v^l) + \sum_{u} \in N(v) W_e * h_u^l \tag{5}$$

where h_v^l is the feature vector of node v in the l-th layer; $N(v)$ is the neighbor set of node v; W and W_e are learnable weight matrices; σ is the activation function. In this way, GNN can effectively represent the dependencies between tasks; for instance, some tasks must be started only after other tasks are completed [22–23]. In complex industrial production scheduling, GNN can effectively capture the dependencies between tasks by modeling tasks as nodes and process constraints as edges. Specifically, each task is regarded as a node in the graph, and the node features can include information such as task priority and processing time. The order or constraints between tasks are represented by edges. GNN uses node feature update rules to aggregate information from neighboring nodes through a multi-layer message-passing mechanism, thereby dynamically updating the feature vector of each node. For example, in the steelmaking scenario of iron and steel metallurgy, the steelmaking process in a converter must be completed before the casting process in

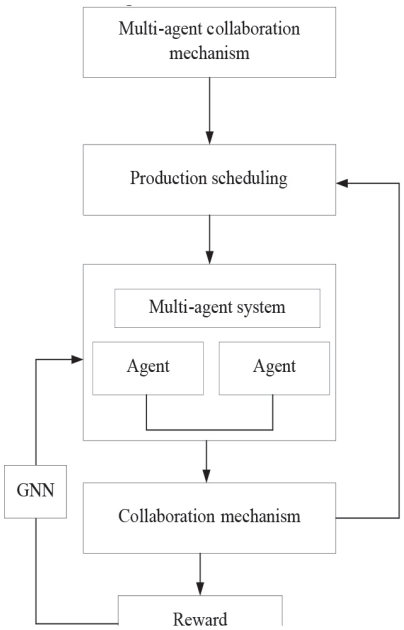

Figure 1 Multi-agent collaboration mechanism.

a continuous caster. This dependency can be represented by a directed edge connecting two task nodes. By learning this graph structure, GNN can extract complex dependencies between tasks and incorporate them into scheduling decisions.

Figure 1 presents the framework of the multi-agent collaboration mechanism.

The advantage of the multi-agent collaboration mechanism is that it can achieve distributed decision-making and efficient collaboration. In multi-workshop production scheduling, agents in different workshops can coordinate task allocation through the CTDE framework and use GNN to model task dependencies across workshops to ensure the consistency of global scheduling. In addition, it can improve the robustness of the system. When equipment failure occurs in a workshop, agents in other workshops can quickly adjust task allocation to ensure the continuity of overall production.

2.3 Reward Function Design

To balance the short-term task completion rate and long-term system performance optimization, in this research work, a multi-objective reward function is designed that comprehensively considers factors such as task delay, resource utilization, and cost savings [24–25].

The multi-objective reward function can be expressed as:

$$R = w_1 R_{delay} + w_2 R_{utilization} + w_3 R_{cost} \tag{6}$$

where R_{delay} represents the task delay penalty; $R_{utilization}$ represents the resource utilization reward; R_{cost} represents the cost saving reward; w_1, w_2, and w_3 are weight coefficients. Specifically, the task delay penalty can be defined as:

$$R_{delay} = -\sum_{i=1}^{N} \max(0, t_i - d_i) \tag{7}$$

where t_i is the actual completion time of task i; d_i is the expected completion time of task i. The resource utilization reward is defined as:

$$R_{utilization} = \frac{\sum_{j=1}^{M} u_j}{T} \tag{8}$$

where u_j is the usage time of resource j, and T is the total time. The cost saving reward can be defined as:

$$R_{cost} = \frac{C_{total} - C_{used}}{C_{total}} \tag{9}$$

where C_{total} is the total budget, and C_{used} is the actual cost consumed.

By designing a multi-objective reward function, a balance between short-term and long-term goals in a dynamic environment can be achieved. When an emergency order has to be inserted, the system can adjust the weight coefficients w_1, w_2, and w_3 to prioritize urgent tasks while taking into account resource utilization and cost savings [26]. In addition, the multi-objective reward function can improve the flexibility and adaptability of the system. In the event of equipment failure or order changes, the system can quickly respond to environmental changes by dynamically adjusting the reward weights, thereby ensuring the efficiency and rationality of the scheduling strategy.

2.4 Implementation of Deep Reinforcement Learning Algorithm

In complex industrial production scheduling, DRL, as a self-learning intelligent method, can gradually optimize the scheduling strategy by interacting with the environment [27–28]. However, traditional single-agent DRL methods often suffer from slow convergence and poor stability when faced with high-dimensional state spaces and dynamic environments. Hence, this paper applies various deep reinforcement learning techniques, including the priority experience replay mechanism and PPO, into the hierarchical multi-agent architecture, and improves the algorithm performance through innovative design [29–30].

The experience replay mechanism improves sample utilization efficiency by storing and sampling historical experience data. Specifically, quad (s_t, a_t, r_t, s_{t+1}) is stored in experience pool D, and samples are selected through the priority sampling mechanism. The sampling probability can be expressed as:

$$P(i) = \frac{p_i^\alpha}{\sum_k p_k^\alpha} \tag{10}$$

where $P(i)$ is the priority of sample i, and α is the parameter that controls the importance of priority. Through priority experience replay, the algorithm can learn high-value experience data more efficiently. In scenarios where tasks change frequently, priority experience replay can help the algorithm adapt to new task requirements more quickly, thereby improving the stability of the scheduling strategy.

The policy optimization algorithm uses the PPO algorithm to update policy parameters. PPO ensures the stability of the update by limiting the amount of difference between the new policy and the old policy. During the training phase, all agents share global state information and coordinate decisions through a joint reward function, $R_{joint} = \sum_{i=1}^{N} R_i$.

During the execution phase, each agent makes independent decisions based only on its local observation, o_i. This design not only improves training efficiency, but also increases the scalability of the system, enabling the system to flexibly respond to dynamically changing environments. The design of PPO must be combined with task dependency modeling to capture the complex relationships between tasks. In complex industrial production scheduling, there are often strong dependencies between tasks. To this end, this paper uses GNN to model task dependencies and uses the output of GNN as the input feature of PPO. By dynamically adjusting the weight coefficient, PPO can gradually optimize the scheduling strategy while ensuring its stability.

3. EXPERIMENTAL DESIGN

This paper selects the classic JSSP dataset in OR-Library as the test platform to further verify the effectiveness of the proposed hierarchical multi-agent deep reinforcement learning architecture in solving industrial production scheduling problems [31]. OR-Library is a public dataset library that is often used in research on scheduling problems. The JSSP dataset includes tasks of different scales and resource constraints at different levels from simple to complex. It is suitable for testing the performance of scheduling algorithms. There are many instances in the JSSP dataset. The task scale of the instances ranges from 6 to 500, and the number of machines ranges from 6 to 20. These instances adequately reflect the high dimensionality and dynamics of industrial production. To meet the needs of the hierarchical multi-agent architecture, in this paper, the raw data is preprocessed, key information such as task priority, processing time, and equipment state is extracted, and then converted into a format suitable for a deep reinforcement learning model input. In addition, to simulate a dynamic environment, random events are included in the experiment to test the algorithm's adaptability under uncertain conditions.

The experimental environment is built on a complex industrial production scheduling simulation platform, which can simulate actual scenarios such as complex industrial production, equipment operation state, and order management. By importing the JSSP dataset of OR-Library into the simulation platform, multiple dynamic scheduling scenarios are generated, including normal production process, equipment failure recovery, emergency order insertion, etc. These scenarios not only reflect the diversity of industrial production, but also increase the actual reference value of the experimental results. Table 1 lists the relevant data.

Task dependencies represent a sequence or constraint. T1 $\rightarrow$ T2 means that task T1 must be completed before task T2 starts.

In the experimental design, various representative comparison algorithms are selected so as to comprehensively evaluate the performance of the proposed hierarchical multi-agent deep reinforcement learning architecture. These algorithms cover traditional optimization methods, single-agent reinforcement learning algorithms, and existing advanced multi-agent scheduling models. Specifically, mixed-integer linear programming (MILP), as a classic mathematical programming method, can generate theoretically optimal scheduling solutions, so it is used as a benchmark comparison. The single-agent reinforcement learning method based on deep Q-network can cope with high-dimensional state space problems to a certain extent, so it is used to verify the advantages of the multi-agent architecture. The centralized scheduling method based on GNN generates high-quality scheduling strategies by capturing task dependencies, so it is selected for comparison. In addition, the multi-agent deep deterministic policy gradient (MADDPG) algorithm and the particle swarm optimization-based dynamic scheduling (PSO-DS) are advanced multi-agent reinforcement learning algorithms and swarm intelligence methods, respectively. The former performs well in continuous action space, and the latter has certain advantages in regard to real-time response. Finally, the DMA, a hybrid heuristic algorithm that integrates local optimization strategies, has shown superiority in the sustainable production scheduling problem and is therefore also included for the comparison. The performance of these algorithms is tested in order to verify the advantages of the proposed hierarchical multi-agent deep reinforcement learning architecture in terms of efficiency, flexibility, and adaptability from different perspectives and to provide a scientific and comprehensive reference for the experimental results.

4. RESULTS ANALYSIS

4.1 Performance Improvement Analysis

To demonstrate the performance advantages of the hierarchical multi-agent method in complex industrial production scheduling, this paper compares the average task completion time of different methods in small-scale, medium-scale, and large-scale task scenarios. The results are presented in Figure 2.

Table 1 Some data.

Instance name	Number of tasks (N)	Number of machines (M)	Processing time matrix	Task dependencies	Probability of equipment failure (%)	Order priority (1–5)
ft06	6	6	$\begin{bmatrix} 1 & 3 \\ 2 & 5 \\ 4 & 7 \\ 3 & 6 \\ 5 & 8 \\ 6 & 9 \end{bmatrix}$	T1 → T2, T3 → T5	2	3
ft10	10	10	$\begin{bmatrix} 2 & 8 \\ 3 & 5 \\ 4 & 10 \\ 5 & 10 \\ 6 & 10 \\ 7 & 4 \\ 8 & 6 \\ 9 & 8 \\ 10 & 10 \end{bmatrix}$	T1 → T4, T2 → T6	5	4
la01	10	5	$\begin{bmatrix} 5 & 6 \\ 4 & 8 \\ 3 & 7 \\ 2 & 9 \\ 1 & 10 \\ 6 & 5 \\ 7 & 4 \\ 8 & 6 \\ 9 & 8 \\ 10 & 7 \end{bmatrix}$	T3 → T7, T5 → T8	3	2
la02	10	5	$\begin{bmatrix} 6 & 5 \\ 5 & 7 \\ 4 & 6 \\ 3 & 8 \\ 2 & 8 \\ 2 & 9 \\ 1 & 10 \\ 7 & 5 \\ 8 & 6 \\ 9 & 7 \\ 10 & 8 \end{bmatrix}$	T2 → T4, T6 → T9	4	5
abz5	10	10	$\begin{bmatrix} 3 & 5 \\ 4 & 7 \\ 5 & 8 \\ 6 & 9 \\ 7 & 10 \\ 8 & 11 \\ 9 & 12 \\ 10 & 13 \\ 11 & 14 \\ 12 & 15 \end{bmatrix}$	T1 → T3, T4 → T7	6	1

Figure 2 shows the comparison curves for the average task completion time of different methods in small, medium, and large scales. The hierarchical multi-agent method has the lowest average task completion time for all scales of task scenarios, and this is also significant in large-scale task scheduling. In the small-scale instance ft06, the hierarchical multi-agent method reduces the task completion time by about 18% compared to MILP and by about 15% and 14% compared to single-agent DQN and centralized GNN, respectively. In the medium-scale instance la01, its task completion time is 18%–22% shorter than those of traditional methods. In the large-scale instance abz5, the hierarchical multi-agent

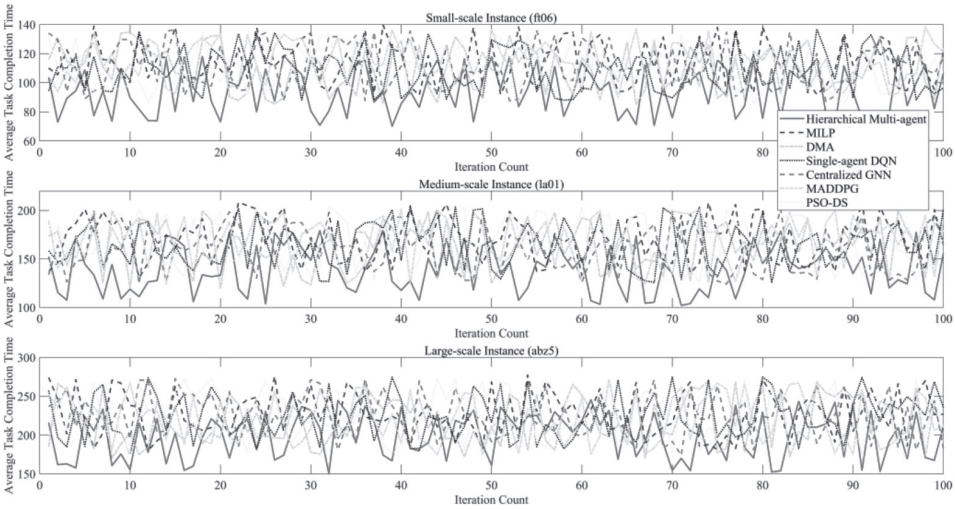

Figure 2 Completion time of tasks of different sizes.

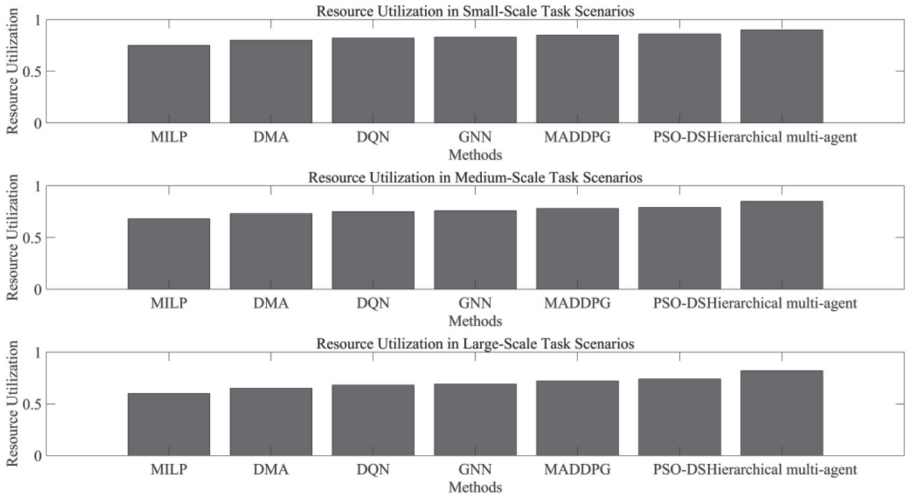

Figure 3 Resource utilization in different task scenarios.

method reduces the task completion time by 15%–25% compared to other methods. In addition, as the complexity of the task increases, the performance improvement of the hierarchical multi-agent method gradually increases, indicating its superiority in tackling high-dimensional and dynamic problems.

The hierarchical multi-agent method can perform well in terms of average task completion time mainly due to its unique hierarchical structure and multi-agent collaboration mechanism. First, the high-level DQN is used for global resource allocation, which can optimize task priority and resource utilization as a whole, avoiding the problem of local optimality in a single agent architecture. The low-level PPO is used for fine scheduling, which ensures that the efficiency of task execution in each work section is maximized. Second, through the CTDE framework, the proposed method can share global information during the training phase and learn to collaborate efficiently with other agents while maintaining independent decision-making during the execution phase, thereby improving the flexibility and scalability of the system. In addition, the GNN modeling of task dependencies further strengthens the robustness of the system, enabling it to better cope with complex constraints between tasks.

4.2 Resource Utilization Analysis

Resource utilization is an important indicator used to measure the performance of scheduling algorithms. Particularly in complex industrial production scheduling, efficient resource utilization can significantly reduce costs and improve system efficiency. Figure 3 compares the resource utilization of different methods in small-scale, medium-scale, and large-scale task scenarios.

Figure 3 shows the resource utilization comparison of different methods in small-scale, medium-scale, and large-scale task scenarios. The figure demonstrates that the hierarchical multi-agent system has high resource utilization in task scenarios of all scales, especially in large-scale task scheduling. On a large scale, the resource utilization of hierarchical multi-agent is 18% higher than that of MILP, 15% higher than that of single-agent DQN and 12% higher than that of centralized GNN. In addition, compared with MADDPG and PSO-DS, the resource utilization of hierarchical multi-agent is improved by 10% and 8%, respectively. Although the DMA method performs well in small and medium-scale tasks, its resource utilization is reduced in large-scale tasks due to the large computational overhead of global search.

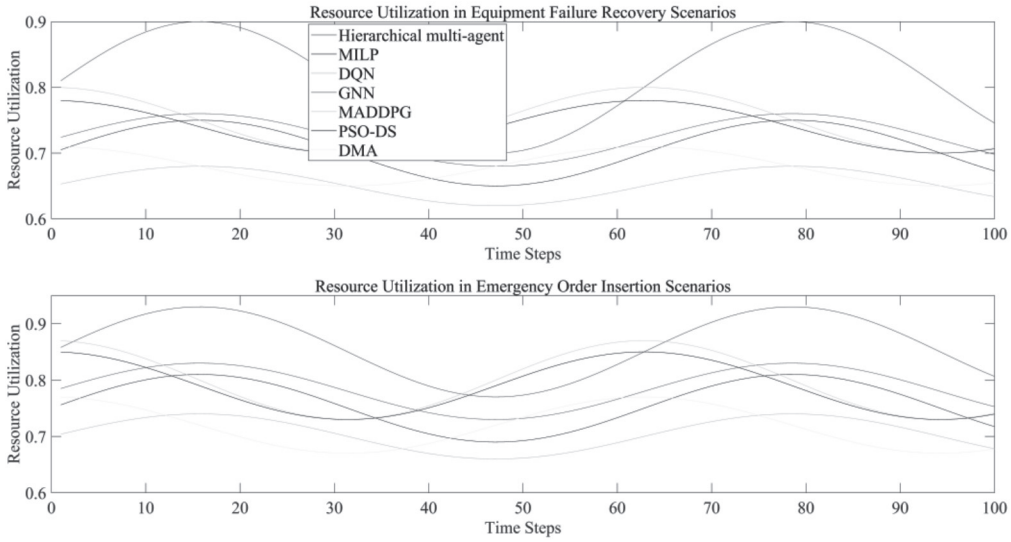

Figure 4 Resource utilization in a dynamic environment.

In a dynamic industrial production environment, uncertain events such as equipment failure recovery and emergency order insertion place higher demands on resource utilization. In this paper, two typical dynamic scenarios are simulated, and the resource utilization change curve is plotted in a dynamic environment to verify the adaptability of the hierarchical multi-agent in complex dynamic scenarios, as shown in Figure 4.

In Figure 4, in the scenarios of equipment failure recovery and emergency order insertion, the hierarchical multi-agent system performs particularly well, and its resource utilization rate always remains at a high level, indicating that the system has a strong ability to adapt to uncertainty. The resource utilization rates of MILP and DMA fluctuate significantly in dynamic environments. The single-agent DQN and centralized GNN cannot maintain stable resource utilization rates in complex scenarios due to the lack of multi-agent collaboration mechanisms.

4.3 System Delay Rate Analysis

The system delay rate is one of the important indicators used to measure the performance of the scheduling algorithm, indicating the proportion of tasks that fail to be completed on time during the scheduling process. A low system delay rate not only improves, production efficiency, but also significantly reduces the additional costs caused by task delays. Figure 5 shows the changes in system delay rates of different methods in a dynamic environment.

Figure 5 shows the system delay rate change curves of different methods in a dynamic environment, including two typical scenarios: equipment failure recovery and emergency order insertion. In the figure, the hierarchical multi-agent system shows the lowest system delay rate in all scenarios. In the equipment failure recovery scenario, the delay rate of this method remains at around 5%. In contrast, the delay rates of MILP and DMA fluctuate significantly, reaching 10% and 12%, respectively. In addition, in the emergency order insertion scenario, the delay rate of the hierarchical multi-agent is significantly lower than that of single-agent DQN and centralized GNN, indicating that it is more robust and adaptable in complex dynamic environments.

To demonstrate the delay rate of different methods, Table 2 lists the multi-indicator comparison data of different methods in small-scale, medium-scale, and large-scale task scenarios.

As shown in Table 2, there are obvious differences in performance in regard to the delay rate of different methods in small-scale, medium-scale, and large-scale systems. The hierarchical multi-agent has the lowest average delay rate, maximum delay rate, and delay rate standard deviation at all scales. It performs best in small-scale systems, where the average delay rate is only 5%. In contrast, the single-agent DQN has the highest delay rate indicators for all scales, with an average delay rate of up to 16% and a maximum delay rate of 21% in large-scale systems. It also has the largest standard deviation, reflecting its poor stability and adaptability. Other methods, such as MILP, DMA, centralized GNN, and PSO-DS, perform relatively similarly but are slightly inferior to the hierarchical multi-agent. Overall, the hierarchical multi-agent system has significant advantages in improving system performance and stability, and it is suitable for different scales of application scenarios.

4.4 Cost Saving Rate Analysis

Cost saving rate is one of the important measures used to determine the economic efficiency of scheduling algorithms. Especially in complex industrial production scheduling, reducing production costs can significantly improve the market competitiveness of enterprises. Figure 6 shows the cost saving rate comparison of different methods in small-scale, medium-scale, and large-scale task scenarios.

In Figure 6, the hierarchical multi-agent shows the highest cost saving rate for all scales of task scenarios. It has a significant advantage in large-scale task scheduling. On a large scale, the cost saving rate of the hierarchical multi-agent is 25% higher than that of MILP. In addition, compared with MADDPG and PSO-DS, the cost saving rate of the hierarchical multi-agent is increased by 15% and 12%, respectively.

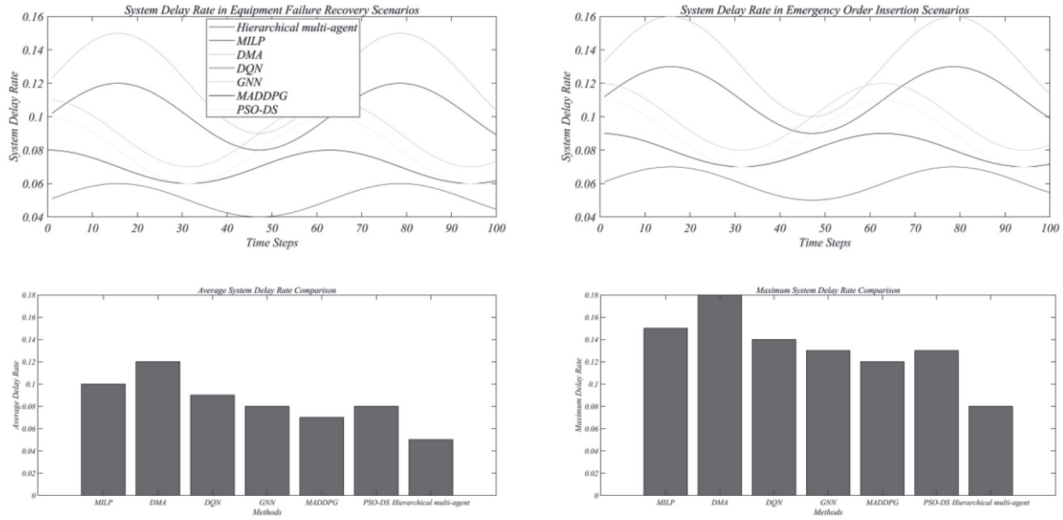

Figure 5 System delay rate changes in a dynamic environment.

Table 2 Comparison of multiple indicators of different methods.

	Small scale			Medium scale			Largescale		
	Average delay rate	Maximum delay rate	Standard deviation of delay rate	Average delay rate	Maximum delay rate	Standard deviation of delay rate	Average delay rate	Maximum delay rate	Standard deviation of delay rate
MILP	8%	12%	2%	10%	15%	3%	15%	20%	4%
DMA	7%	11%	1.8%	9%	14%	2.5%	14%	19%	3.5%
DQN	9%	13%	2.2%	11%	16%	3.2%	16%	21%	4.5%
GNN	8%	12%	2%	10%	15%	3%	14%	19%	3.5%
MADDPG	7%	11%	1.7%	9%	14%	2.4%	13%	18%	3%
PSO-DS	8%	12%	1.9%	10%	15%	2.8%	14%	19%	3.5%
Hierarchical multi-agent	5%	8%	1.5%	7%	12%	2%	10%	15%	2.5%

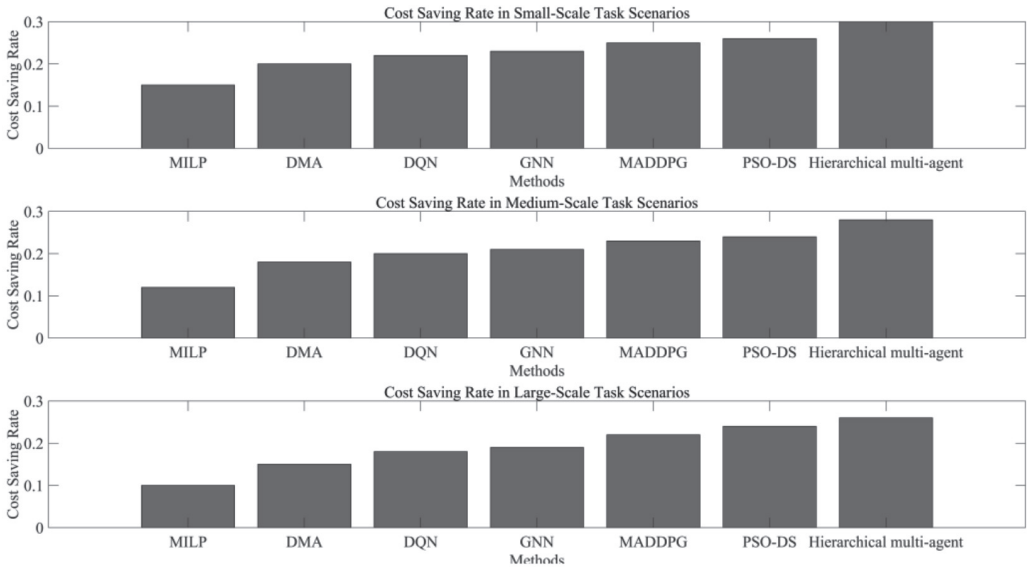

Figure 6 Cost saving rate.

Table 3 shows the comparison of cost saving rate data in a dynamic environment for two typical scenarios: equipment failure recovery and emergency order insertion.

In Table 3, there are obvious differences in the coping capabilities of various methods when facing two types of emergency scenarios: equipment failure recovery and emergency order insertion. Hierarchical multi-agent performs best in both scenarios, reaching 30% and 35%, respectively, showing its strong robustness and adaptability in handling complex dynamic events. In contrast, MILP and DMA

Table 3 Comparison of cost saving rate data in dynamic environment.

Method	Equipment failure recovery scenario (%)	Emergency order insertion scenario (%)
MILP	20	25
DMA	15	20
DQN	22	28
GNN	23	29
MADDPG	25	30
PSO-DS	24	28
Hierarchical multi-agent	30	35

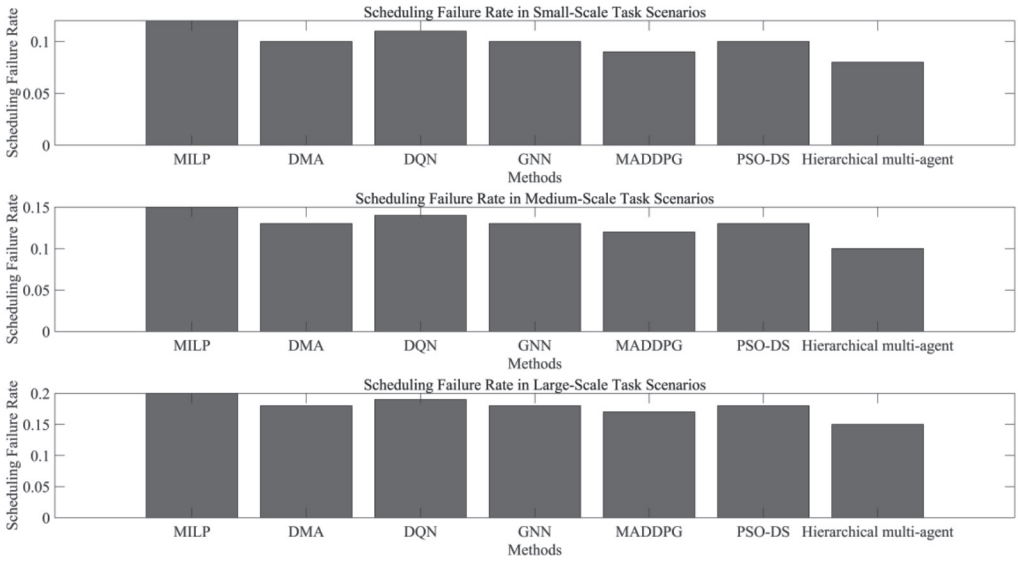

Figure 7 Scheduling failure rate of different algorithms.

methods have the weakest adaptability and low response capabilities in both scenarios, which may be due to their lack of flexible real-time decision-making mechanisms. Other methods, such as MADDPG, centralized GNN, and PSO-DS, perform moderately, although MADDPG also performs relatively well in both scenarios, second only to the hierarchical multi-agent. Overall, the hierarchical multi-agent method has significant advantages in regard to emergency handling, and it is more suitable for dynamic and complex production or scheduling environments.

4.5 Scheduling Failure Rate Analysis

The scheduling failure rate is one of the important indicators for measuring the robustness and stability of the scheduling algorithm. Particularly in complex industrial production scheduling, a low scheduling failure rate can effectively reduce scheduling interruptions caused by task conflicts, insufficient resources, or environmental uncertainties. Figure 7 shows the comparison of the scheduling failure rates of different methods in small-scale, medium-scale, and large-scale task scenarios.

In Figure 7, there are significant differences in the scheduling failure rates of different methods applied to three task scales. In general, as the task scale increases from small to large, the failure rates of all methods increase, reflecting the impact of task complexity on scheduling results. In small-scale tasks, the failure rate of each method is low, among which the hierarchical multi-agent method performs best, with only 0.08. In medium-scale tasks, this method still maintains the lowest failure rate, showing good adaptability and stability. In large-scale task scenarios, the hierarchical multi-agent still leads with the lowest failure rate of 0.15, indicating that it is more robust in complex scenarios. In contrast, the failure rates of MILP and PSO-DS increase significantly after the task scale is expanded, reaching 0.20 and 0.18, respectively, and their performance is relatively poor.

In Figure 8, the changes in the scheduling failure rate in two typical dynamic scenarios—equipment failure recovery and emergency order insertion— are simulated to verify the performance of the hierarchical multi-agent method in dealing with real-time environmental changes:

In the scheduling failure rate change curve in a dynamic environment, the performance of different algorithms varies significantly, reflecting their adaptability to real-time changes and uncertainties. The hierarchical multi-agent system shows the lowest scheduling failure rate in both equipment failure recovery and emergency order insertion scenarios, with small fluctuations and rapid recovery. In the equipment failure recovery scenario, its failure rate drops rapidly from 18% to below 10%, reflecting its efficient global optimization and local adjustment capabilities. In contrast, MILP has the largest fluctuation in failure rate and the slowest recovery, rising to 30% during equipment failure and gradually recovering to below 20% until step 40, indicating its poor adaptability to dynamic environments. DMA, DQN, and GNN have

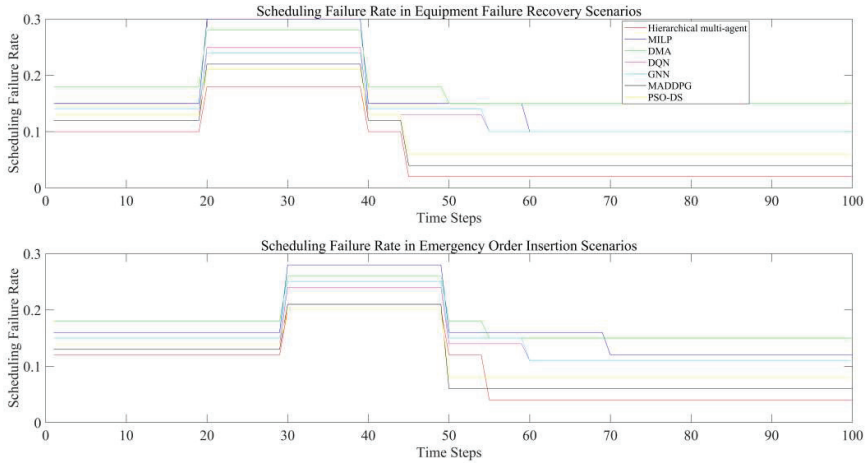

Figure 8 Scheduling failure rate in dynamic scenarios.

moderate fluctuations in failure rate and slightly faster recovery, but are limited by local optimization or insufficient dependency modeling. MADDPG and PSO-DS have a lower failure rate and faster recovery due to their advantages in continuous action space and swarm intelligence, but they are still inferior to the hierarchical multi-agent in regard to processing complex dependencies.

5. DISCUSSION

5.1 Performance Advantages and Architecture Correlation

The complex industrial production scheduling problem has always been a difficult research topic in both academia and industry due to its high dimensionality, dynamism, and multi-objective characteristics. The hierarchical multi-agent deep reinforcement learning architecture proposed in this paper significantly outperforms traditional methods and existing advanced algorithms, such as MILP, DMA, DQN, and MADDPG, in multiple performance indicators. The experimental results indicate that the hierarchical multi-agent method performs well in terms of task completion time, resource utilization, system delay rate, cost saving rate, and scheduling failure rate. In large-scale task scenarios and dynamic environments, its performance advantages are evident. In the scenarios of equipment failure recovery and emergency order insertion, the scheduling failure rate of the hierarchical multi-agent method remains below 10%, while other methods, such as MILP and DMA, have failure rates as high as around 30% and 20%, respectively.

Compared with other algorithms, the hierarchical multi-agent method has several advantages in many respects. Firstly, the hierarchical structure reduces decision complexity and improves overall system efficiency by separating the allocation of a global task from local execution. The high level adopts DQN for global resource allocation, which can optimize task priority and resource utilization as a whole. The use of PPO for fine scheduling at the low level ensures maximum task execution efficiency within each section. Secondly, the multi-agent collaboration mechanism combined with the CTDE framework enables the system to efficiently learn collaborative strategies during the training phase and maintain independent decision-making capabilities during the execution phase, thereby avoiding communication bottlenecks in a single-agent architecture. Additionally, GNN further enhances the system's ability to capture complex task dependencies, making it more adaptable in dynamic environments.

5.2 Shortcomings and Improvement Directions

However, the hierarchical multi-agent method is not perfect and still has several shortcomings that need to be addressed. Firstly, the computational overhead of the training phase is high, especially when the task scale is increased. Centralized training may lead to communication bottleneck problems and affect the scalability of the system. Secondly, the design of the reward function significantly impacts the algorithm's performance. The current selection of weight coefficients relies mainly on experience or heuristic methods and lacks an adaptive adjustment mechanism, which may lead to insufficient flexibility of the algorithm in some scenarios. In addition, although GNN can effectively capture the dependencies between tasks, the expressive ability of the model may be limited when the task dependencies are highly complex or dynamically changing, thereby affecting the accuracy of scheduling decisions. Finally, deep reinforcement learning methods are often regarded as "black box" models, and their decision-making process is difficult for humans to understand, which may cause credibility issues in actual industrial applications.

In view of the aforementioned shortcomings, several recommendations are made for future research directions. Firstly, a distributed computing framework or asynchronous update mechanism can be applied to reduce the computational burden of the training phase and improve scalability. Secondly, an adaptive reward function mechanism can be developed to enable the algorithm to adjust the weight coefficient in real time according to the dynamic changes of the environment so as to better balance short-term and long-term goals. In addition, combined with the graph attention network (GAT) or other advanced graph modeling technologies,

the processing ability of complex task dependencies can be further improved. Finally, combined with explainable artificial intelligence (XAI) technology, the transparency and credibility of the algorithm can be enhanced. Alternatively, leveraging the advancements in Large Language Models (LLMs) can improve the performance and generalization capability of production scheduling algorithms so that they can better meet industrial needs.

6. CONCLUSION

Complex industrial production scheduling poses severe challenges to traditional scheduling methods due to its high dimensionality, dynamics, and multi-objective characteristics. This paper proposes a hierarchical multi-agent deep reinforcement learning architecture, which separates global task allocation from local execution through a hierarchical structure, combines the CTDE framework to achieve efficient multi-agent collaboration, and uses GNN to model task dependencies and design multi-objective reward functions, which significantly improves scheduling efficiency and adaptability. Experimental results show that this method outperforms traditional methods and other advanced algorithms in key indicators such as task completion time, resource utilization, system delay rate, cost saving rate, and scheduling failure rate, especially in large-scale and dynamic scenarios. However, the training phase incurs high computational overhead, and there is still room for improvement in terms of reward function design and task dependency modeling. Future research could explore distributed computing frameworks, adaptive reward mechanisms, Large Language Models (LLMs), and more advanced graph modeling techniques to further improve algorithm performance and practical application potential. This study has important theoretical and practical significance as it provides an intelligent solution for the scheduling of complex industrial production.

REFERENCES

1. Lu Hong, et al. “Dual-enhanced Memetic Algorithm for Multi-task Scheduling in Sustainable Production”. Acta Automatica Sinica 50.4 (2024): 731–744.
2. Liu Wei, et al. “MES Production Scheduling Model Based on Master-Slave Chain and Edge Computing”. Journal of Zhengzhou University (Natural Science Edition) 55.3 (2023).
3. Del Gallo, Mateo, et al. “Artificial intelligence to solve production scheduling problems in real industrial settings: Systematic Literature Review”. Electronics 12.23 (2023): 4732.
4. Zhuang, Zilong, et al. “Network-based dynamic dispatching rule generation mechanism for real-time production scheduling problems with dynamic job arrivals”. Robotics and Computer-Integrated Manufacturing 73 (2022): 102261.
5. Tamssaouet, Karim, et al. “Multiobjective optimization for complex flexible job-shop scheduling problems”. European Journal of Operational Research 296.1 (2022): 87–100.
6. Waubert de Puiseau, Constantin, Richard Meyes, and Tobias Meisen. “On reliability of reinforcement learning based production scheduling systems: a comparative survey”. Journal of Intelligent Manufacturing 33.4 (2022): 911–927.
7. Liu, Renke, Rajesh Piplani, and Carlos Toro. “A deep multi-agent reinforcement learning approach to solve dynamic job shop scheduling problem”. Computers & Operations Research 159 (2023): 106294.
8. Lee, Young Hoon, and Seunghoon Lee. “Deep reinforcement learning based scheduling within production plan in semiconductor fabrication”. Expert Systems with Applications 191 (2022): 116222.
9. Zhang, Ming, et al. “Dynamic scheduling method for job-shop manufacturing systems by deep reinforcement learning with proximal policy optimization”. Sustainability 14.9 (2022): 5177.
10. Zhang, Yi, et al. “Dynamic job shop scheduling based on deep reinforcement learning for multi-agent manufacturing systems”. Robotics and Computer-Integrated Manufacturing 78 (2022): 102412.
11. Zhao Chen, Zhang Cheng, and Huang Yongming. “Research on Traffic-aware Intelligent Slicing Resource Allocation for Radio Access Network”. Journal of Signal Processing 40.4 (2024): 719–732.
12. Shang Xiaokai, Han Longlong, and Zhai Huipeng. “Research on elastic optical network resource allocation based on improved DQN reinforcement learning algorithm”. Optical Communication Technology 47.5 (2023): 12–15.
13. Lee, Insung, and Duk Kyung Kim. “Decentralized multi-agent DQN-based resource allocation for heterogeneous traffic in V2X communications”. IEEE Access 12 (2024): 3070–3084.
14. Yu, Li, et al. “A DQN-based joint spectrum and computing resource allocation algorithm for MEC networks”. GLOBECOM 2022–2022 IEEE Global Communications Conference. IEEE, 2022.
15. Garg, Shrishti, Shruti Gupta, and Vivek Srivastava. “Resource Allocation in IoT Networks via DQN and PPO Algorithm”. 2025 2nd International Conference on Computational Intelligence, Communication Technology and Networking (CICTN). IEEE, 2025.
16. Li, Han, et al. “Collaborative task offloading and resource allocation in small-cell MEC: A multi-agent PPO-based scheme”. IEEE Transactions on Mobile Computing (2024).
17. Li, Rongzhen, et al. “Proximal Policy Optimization (PPO)-Based Resource Allocation for Energy Harvesting Industrial Wireless Sensor”. (2023).
18. Johnson, Dazzle, Gang Chen, and Yuqian Lu. “Multi-agent reinforcement learning for real-time dynamic production scheduling in a robot assembly cell”. IEEE Robotics and Automation Letters 7.3 (2022): 7684–7691.
19. Y. Watanobe; Y. Yaguchi; K. Nakamura; T. Miyaji; R. Yamada; K. Naruse. Architecture and framework for data acquisition in cloud robotics. International Journal of Information Technology, Communications and Convergence, 2021 Vol.4 No.1, pp.1–25.
20. Hameed, Mohammed Sharafath Abdul, and Andreas Schwung. “Graph neural networks-based scheduler for production planning problems using reinforcement learning”. Journal of Manufacturing Systems 69 (2023): 91–102.
21. Song, Wen, et al. “Flexible job-shop scheduling via graph neural network and deep reinforcement learning”. IEEE Transactions on Industrial Informatics 19.2 (2022): 1600–1610.
22. Zhang, Zhen, et al. “A resource optimization scheduling model and algorithm for heterogeneous computing clusters based on GNN and RL”. The Journal of Supercomputing 80.16 (2024): 24138–24172.
23. Huang, Jiang-Ping, et al. “A novel priority dispatch rule generation method based on graph neural network and reinforcement learning for distributed job-shop scheduling”. Journal of Manufacturing Systems 69 (2023): 119–134.
24. Yu, Hao, Tarik Taleb, and Jiawei Zhang. “Deep reinforcement learning-based deterministic routing and scheduling for mixed-

criticality flows". IEEE Transactions on Industrial Informatics 19.8 (2022): 8806–8816.
25. Esteso, Ana, et al. "Reinforcement learning applied to production planning and control". International Journal of Production Research 61.16 (2023): 5772–5789.
26. Rinciog, Alexandru, and Anne Meyer. "Towards standardising reinforcement learning approaches for production scheduling problems". Procedia CIRP 107 (2022): 1112–1119.
27. Rigatos, G., Zervos, N., Siano, P., Abbaszadeh, M., Wira, P., & Onose, B. (2019). Nonlinear optimal control for DC industrial microgrids. Cyber-Physical Systems, 5(**??**), 231–253.
28. Che, Gelegen, et al. "A deep reinforcement learning based multi-objective optimization for the scheduling of oxygen production system in integrated iron and steel plants". Applied Energy 345 (2023): 121332.
29. Song, Wen, et al. "Stochastic economic lot scheduling via self-attention based deep reinforcement learning". IEEE Transactions on Automation Science and Engineering 21.2 (2023): 1457–1468.
30. Zhang, Lixiang, Yan Yan, and Yaoguang Hu. "Deep reinforcement learning for dynamic scheduling of energy-efficient automated guided vehicles." Journal of Intelligent Manufacturing 35.8 (2024): 3875–3888.
31. Beasley, J.E. "OR-Library: Job Shop Scheduling Problem." Brunel University London , 2023
32. Yan Yang. "Innovation mechanism of common technology collaboration in emerging industries within the context of big data and the Internet of Things". Engineering Intelligent Systems 33.1 (2025).

Haibo Peng, male, Workin Yunnan Open University. His main research interests are enterprise automation, informatization and intellectualization.He has won 2 second prizes of science and technology award of Yunnan Province and 1 third prize of science and technology award of China Iron and steel metallurgy Association. And he has successively applied for 11 invention patents, 17 utility model patents and 2 appearance patents, registered 8 software copyrights, published 13 papers and participated in the preparation of 1 industry standard.
Email:harpor1982@163.com

Guixiong Li, male, holds a bachelor's degree from Yunnan University. His main research interest is smart supply chain. Has won one new information consumption demonstration project from the Ministry of Industry and Information Technology of China. He has successively applied for 9 invention patents, 14 utility model patents and 1 appearance patents, registered 6 software copyrights, published 5 papers.
Email:18404072@masu.edu.cn

Zhibo Zhang,female,work in Yunnan Open University. Her main research interests are enterprise innovation and management of technology, metal material development and intelligent processing. She has successfully participated in one National Key R&D Program of China and a number of Major science and technology projects of Yunnan Province. She has successively applied for 32 invention patents, and 4 utility model patents , published 17 papers and participated in the preparation of 1 national standard and 1 local standard.
Email: 454734714@qq.com.

Rong Zhou, female, graduated from Kunming University of Science and Technology, mainly engaged in automation, obtained 5 patents and published 2 papers.
Email:8573746@qq.com