J. COMBIN. MATH. COMBIN. COMPUT. 127a (2025) 6141--6161

Journal of Combinatorial Mathematics and Combinatorial Computing
www.combinatorialpress.com/jcmcc

Combinatorial Press

# Research on the Impact of Multi-Agent Based Systems on the Development of Intelligent Manufacturing

Haonan Wu[1,✉]

[1] *School of Future Technologies, Xi'an Jiaotong University, Xi'an, Shaanxi, 710049, China*

ABSTRACT

Agent technology is widely used in intelligent manufacturing and digital workshop as a new method to solve complex, dynamic and distributed artificial intelligence application problems. This paper firstly summarizes the application steps of Agent technology in 3 aspects of modeling, simulation and monitoring of intelligent manufacturing system on the basis of a brief description of multi-agent system. Then, based on reinforcement learning theory, a multi-agent collaborative algorithm SRL_M3DDPG based on state representation learning is proposed.Finally, the algorithm model is tested and applied to the smart shop scheduling problem. The learning curve of the SRL_M3DDPG algorithm in the example remains relatively stable after the 3400th round, and the maximum completion time of the scheduling is 29. Comparing with other composite scheduling rules, the delay rate of this paper's algorithmic model is the lowest, which is only 15.47%, which indicates that the algorithm is able to significantly reduce the delay rate of the workpiece. In addition, this paper's algorithm achieves better results in adaptive intelligent manufacturing workshop scheduling, finding the shortest machining completion time of 221 unit time, which can adapt to the dynamic intelligent manufacturing workshop environment.

*Keywords:* multi-agent system, reinforcement learning, SRL_M3DDPG algorithm, intelligent manufacturing

## 1. Introduction

Multi-Agent system is a collection of multiple Agents, which is a problem solving network formed by the coordination and cooperation of multiple Agents in order to solve complex problems that cannot be solved by a single Agent. In a multi-agent system, the master agent can develop new planning or solution methods to deal with incomplete and uncertain knowledge through communication with other agents [1-4]. Through inter-Agent cooperation, multi-agent systems not only improve the basic capabilities of each agent, but also further understand the social behavior from the interaction of

✉ Corresponding author.
*E-mail address:* wuhaonan@stu.xjtu.edu.cn (H. Wu).
Received 12 January 2024; Revised 05 March 2024; Accepted 25 December 2024; Published Online 15 April 2025.
DOI: 10.61091/jcmcc127a-338

agents, which can have an important impact on the development of intelligent manufacturing [5-7].

The development of modern industry cannot be separated from the advancement of different stages such as mechanization, automation, informatization and intelligence. Intelligent manufacturing, as an important part of the current manufacturing technology revolution, is an inevitable choice for realizing the high-quality development of the manufacturing industry and enhancing its competitiveness [8-11]. First of all, intelligent manufacturing can improve production efficiency and product quality. Through the application of intelligent equipment and systems, the work process can be automated and intelligent, which can improve production efficiency and reduce costs [12-14]. At the same time, intelligent manufacturing can also realize the comprehensive monitoring and control of the production process, reduce the interference of human factors, and improve the consistency and traceability of products [15-16]. Secondly, intelligent manufacturing is of great significance to the transformation and upgrading of traditional manufacturing industry. Traditional manufacturing industry in the face of the pressure of market competition, must be transformed and upgraded in order to maintain competitiveness. The application of intelligent manufacturing can make the traditional manufacturing industry realize the transformation of production mode, so as to adapt to the rapid changes in market demand [17-20]. At the same time, intelligent manufacturing can also improve the production environment, improve work efficiency, reduce energy consumption and resource waste. And the multi-agent based smart manufacturing system realizes the flexibility, adaptivity and high efficiency of smart manufacturing by introducing intelligences as autonomous decision-making and interacting entities [21-24].

Literature [25] examined intelligent manufacturing in the valve industry, pointed out the deficiencies in the industry, and illustrated that relying on intelligent equipment is an effective path for the industry to improve productivity, ensure quality management, and effectively control costs. Literature [26] explored the security of intelligent manufacturing, by pointing out its vulnerabilities, potential risks, and the inadequacy of existing measures, and revealed the important role of security in intelligent manufacturing systems. Literature [27] examined the impact of smart manufacturing on the high-quality development of enterprises, which effectively promotes the high-quality development of enterprises by reducing operating costs, improving capacity utilization, and promoting technological innovation. Literature [28] explored the role of big data in supporting smart manufacturing, and proposed a conceptual framework for supporting big data by describing a historical view of the data lifecycle in the manufacturing industry and providing a narrative of typical application scenarios of the framework. Literature [29] discusses the components of intelligent design, intelligent processing, and intelligent control in information-physical production systems, showing that these technologies can play an important role in optimizing efficiency, reducing errors, and saving costs. Literature [30] reviewed the current state of manufacturing automation standards, noting that current automation standards provide a foundation for developing smart manufacturing solutions. It is argued that smart manufacturing automation can be implemented faster by disseminating, adopting and improving relevant standards in a demand-driven manner. The above studies illustrate the impact of smart manufacturing on the high quality development of enterprises, including improving productivity and product quality, reducing costs, and promoting technological innovation. However, these research results do not show the impact of multi-agent systems on the development of smart manufacturing. However, this does not mean that the research in the field of intelligent manufacturing "multi-agent systems" has not been paid attention to, but rather a lot, especially in the application of intelligent manufacturing systems. Literature [31] proposes an intelligent manufacturing system based on multi-agent systems and reinforcement learning as well as machine learning with intelligent agents to make better decisions. It reveals that this intelligent manufacturing system has a competitive advantage in a dynamic environment. Literature [32] proposes multi-agent approaches that can assist in the design of sustainable smart manufacturing systems that focus on identifying manufacturing components and designing and integrating sustainability-oriented mechanisms in system specifications to provide specific development guidelines. Literature [33] outlines a multi-agent decision-making model that allows for the formation of interactive social networks of multidimensional physical states and intelligences making decisions. The effect of different network topologies on the collective performance of cyber-physical systems is also investigated. Literature [34] reviews current multi-agent based manufacturing approaches and provides a general vision of current trends.

In this paper, based on the technical principles of multi-agent systems, the modeling, simulation and monitoring of intelligent manufacturing systems are implemented. Meanwhile, integrating reinforcement learning and multi-agent systems, the SRL_M3DDPG algorithm based on state representation learning is proposed to realize the coordinated scheduling of the intelligent manufacturing workshop through the collaboration between multi-agents. In order to verify the effectiveness of the algorithmic model, scheduling examples are selected to test the performance of the algorithm and compare it with other composite scheduling rules. Finally, the algorithm is applied to adaptive dynamic scheduling, which is further tested and studied through the arithmetic examples.

## 2. Multi-agent based intelligent manufacturing system modeling and simulation

The multi-agent system theory in distributed artificial intelligence provides feasible technical support for the realization of intelligent manufacturing system. In this paper, on the basis of generalizing and summarizing the technical principles of multi-agent systems, the steps of establishing a multi-agent system simulation model are proposed.

### *2.1. Technical Principles of Multi-Agent Systems*

A multi-agent system (MAS) is an organic collection of multiple computable agents [35]. The system can coordinate a set of autonomous Agent behaviors (knowledge, goals, methods, and planning) to solve a problem with a common goal or action.

2.1.1. Concept of Agent and its modeling. Although the concept of Agent is applied very frequently, there is still no unified concept. It is generally recognized that an Agent is an entity that can perform a certain task continuously and autonomously in a non-decision-making environment. It has its own knowledge, goals and independent processing capabilities. According to this definitional model, an expert, a technician, an intelligent machining center, an independent machining shop, a manufacturing subsystem, or a functional department in a manufacturing system can be regarded as an Agent in a manufacturing system.

2.1.2. Agent Structure Type and Functional Hierarchy Analysis. According to the current research on Agents, they can be categorized into the following seven types according to their intelligence:
1) Passive Agent. It is the least intelligent class among all types of Agents and performs necessary actions only when other objects call them by means of communication.
2) Reactive Agent. It has one more function than Passive Agent: the ability to monitor the environment proactively. Once the state parameters of the environment change, it can make the necessary response. The most typical application of Reactive Agent is robots, especially Brookes type of insect robots.
3) CBS Agent. Its main feature is its strong knowledge representation ability. As an imitation of domain experts, CBS Agent has two types of knowledge, beliefs and strategies, which represent the Agent's static and dynamic knowledge, or basic knowledge and meta-knowledge, respectively.
4) BDI Agent. This is the most typical intelligent agent, or autonomous agent, in the current research on agents, which has beliefs (i.e., knowledge), desires (i.e., tasks), and intentions (i.e., planning for the realization of desires).A typical application of the BDI Agent is the software agent that collects information for its owner on the Inter-net.The more advanced robots are also of the BDI Agent type. The more advanced robots are also of the BDI Agent type.

5) Social Agent. It lives in a society composed of multiple Agents. The functions of this type of Agent include collaboration and competition. Office automation Agent is a typical example of collaboration. Multiple manufacturing company Agents competing for the right to contract tasks are typical examples of competition.
6) Evolutionary Agent. It is the Agent that has the ability to learn and improve itself. Multi-Agent systems that simulate biological societies (bees, ants) are typical examples of Evolutionary Agents.
7) Personality Agent. Agents that not only have thoughts, but also have emotions.This type of Agent is less studied, but very promising. The story task Agent in story comprehension research is a typical personified Agent.

After analyzing and researching, this paper believes that any one of the above Agents is not suitable to be used as the only expression of all kinds of manufacturing units. Each manufacturing unit Agent should be an autonomy that can be responsible for solving a certain type of problem. For accomplishing this task, the Passive Agent lacks sound thinking and reasoning capabilities.The BDI Agent has strong mental capabilities, but most of the underlying Manufacturing Agents do not need to have such strong capabilities. However, the competitive ability of the Social Agent and the learning ability of the Evolutionary Agent are again needed by the Manufacturing Unit Agent. Personality Agents are also used in the simulation of social phenomena using Manufacturing Agents.Therefore, this paper argues that Intelligent Manufacturing Systems (IMS) need to use almost all of the above Agent types.

2.1.3. Synthesis Mechanism of Multi-Agent Systems in Smart Manufacturing. Agents in an intelligent manufacturing environment are connected through a communication network to form an agent-based distributed manufacturing environment, i.e., a multi-agent system structure. Generally speaking, each Agent is a self-regulating, independent of each other single figure, but the Agent can also be established with a number of Agents “friend element” relationship, composed of relatively close contact. Agent-based intelligent manufacturing environment, with plug-and-play characteristics, can be flexibly reorganized through the network and increase or decrease at any time, in order to adapt to the rapid changes in the manufacturing environment and resources. The granularity of Agent can be specified according to the idea of hierarchical design to form a multi-level manufacturing environment.

In addition, borrowing from the socio-economic environment, the interest-driven factors that cause market competition and lead to the combination of social differentiation, in the smart manufacturing environment, the economic interests as a link, through the bidding-bidding-winning method in the competition, to drive the dynamic combination of the Agents, to realize the dynamic reorganization of the manufacturing system, in order to meet the market requirements of the requirements of manufacturing agility and adaptability. It is worth noting that in the absence of production orders, the manufacturing environment is simply a static organizational structure. When an opportunity arises in the market or an Agent has a production order that it is not capable of (or does not need to) take up in full, it sends out a task proposal to other Agents in the manufacturing environment. Agents that receive the bid decide whether or at what price to compete in the bidding process, based on the benefits and costs they will receive. After receiving the bids from the relevant Agents, the tenderer determines the winner from the point of view of whether or not it can obtain the maximum benefit, and signs a contract with it to clarify the rights and interests of both parties and issues the task order, in which the Agent with closer ties to the tenderer will be given priority. After the winning party gets the order, it can also tender to other Agents in the same way, and finally establish a contract-based dynamic contract network. With the completion of the order, the contract is terminated and this temporary dynamic manufacturing organization is eliminated. When a new market opportunity arises, this reorganization process starts again, and the dynamic reorganization process of the system is shown in Figure 1.

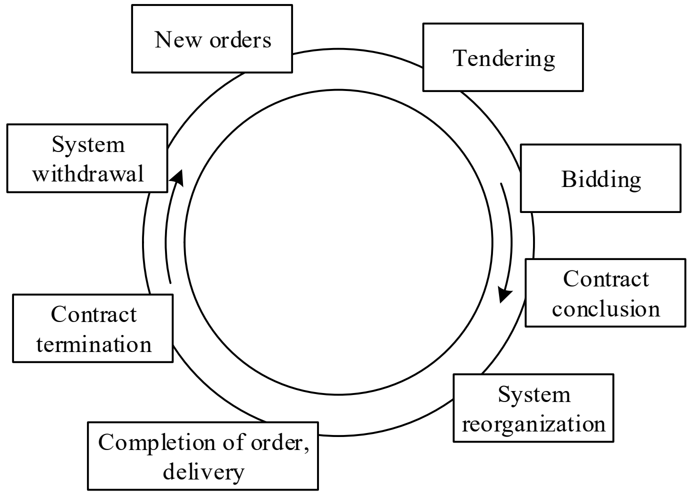


**Fig. 1.** Dynamic reorganization process of the system

## *2.2. Virtual and real manufacturing systems*

Constructing a manufacturing system that combines reality and reality is the key to realizing the integration of physical space and information space. The integration of physical space and information space can be divided into three stages: mapping from physical space to information space, feedback from information space to physical space, and interaction and communion between physical space and information space.The three stages correspond to three major contents, i.e., modeling, simulation, and monitoring. The relationship between the virtual-real manufacturing system and modeling, simulation, and monitoring in the digital workshop is shown in Figure 2.

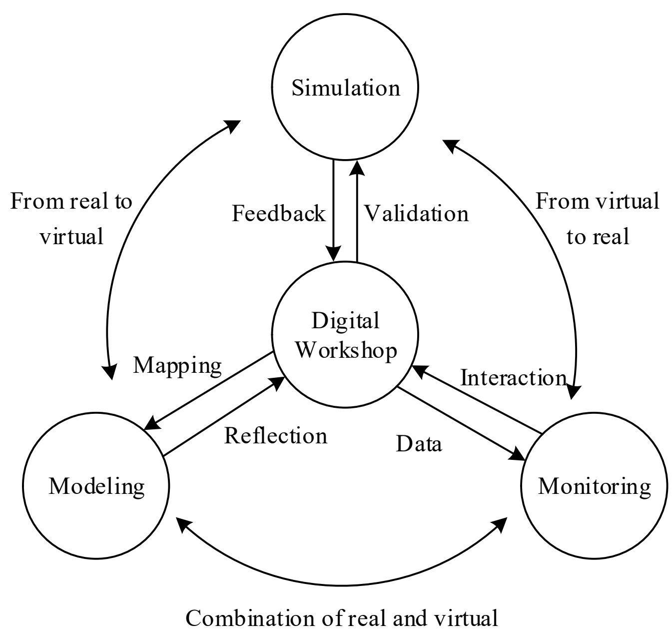


**Fig. 2.** Relationships between system and modeling, simulation and monitoring

## *2.3. Agent-based modeling of manufacturing systems*

Modeling is the foundation of digital workshop simulation and monitoring. Abstracting the physical and logical entities of the real physical space into an Agent model and truly reflecting its autonomy, proactivity, sociality and reactivity are the problems that need to be solved by modeling.

The modeling process of Agent in manufacturing system is shown in Fig. 3, which can be roughly

summarized as:
1) Discover Agent: Abstract workshop physical entities (processing equipment, logistics equipment, production resources, etc.) and logical entities (orders, production plans, management systems, etc.) into Agents.
2) Design Agent: Classify the Agent according to its attributes and select the appropriate Agent structure.
3) Agent behavior modeling. According to the structure of the Agent, the sensor, information processing unit, and actuator of the Agent are realized.
4) Determine the communication method between Agents, the main communication methods used are Knowledge Interchange Format (KIF), ontology and Knowledge Query Language (KQML).

Agent modeling ideas can be considered as the development and extension of object-oriented ideas, is a further understanding of the object and refinement. Object-oriented thinking with attributes and methods to express a class of entities, Agent in the object on the basis of the introduction of perception, reasoning and decision-making, endowed with the object "mind", from the static, passive object into a dynamic, active Agent, with the ability to perceive changes in the external environment, and according to changes in the external environment to make the appropriate The ability to perceive changes in the external environment and make corresponding decisions according to changes in the external environment to approach the target. In order to describe the properties of Agent such as self-awareness, self-decision making, and self-adaptation, researchers generally use an extension of the object-oriented approach to express the Agent. Common Agent modeling methods are extended UML (Unified Modeling Language) Object Oriented Petri nets and Finite State Machines (FSM).

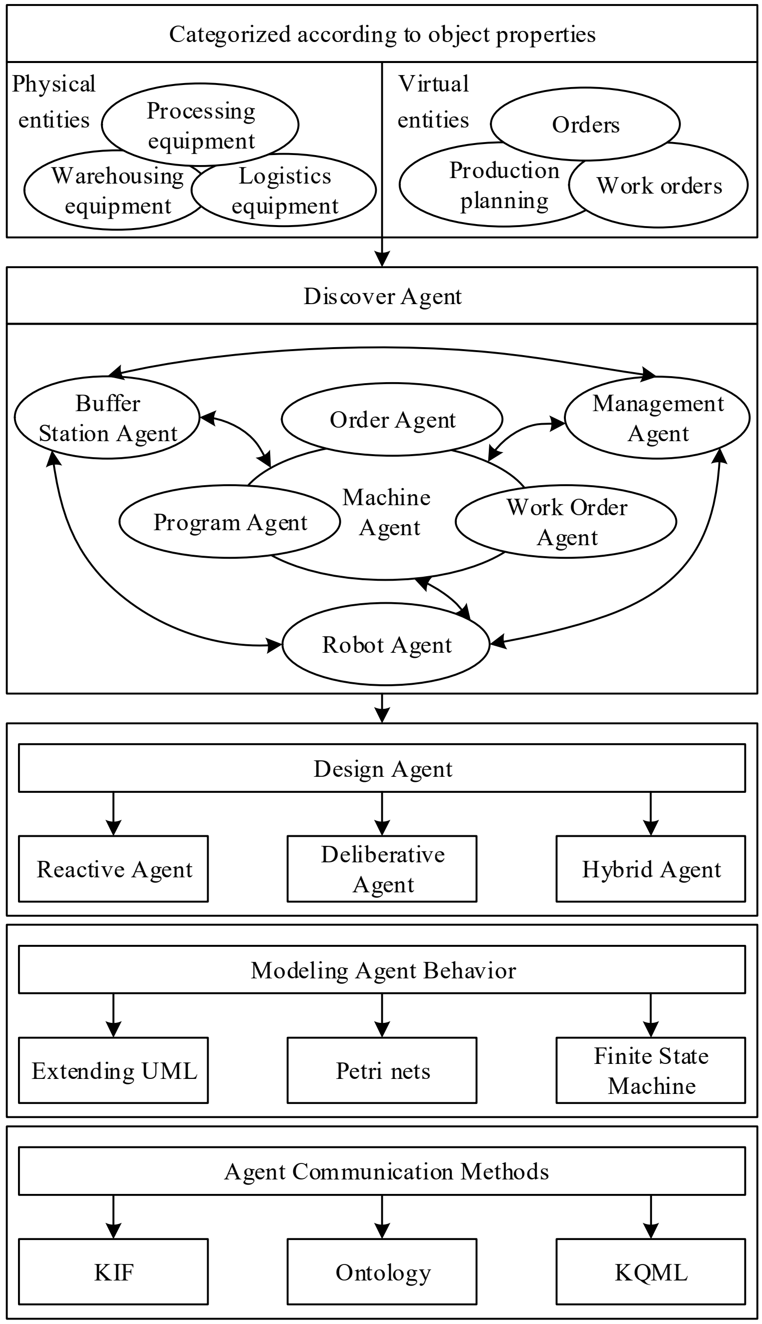


**Fig. 3.** Process of Agent-based modeling

### *2.4. Agent-based manufacturing system simulation*

A discrete manufacturing system is a complex system with multiple factors and perturbations, where events occur suddenly and randomly, and its internal behavior is unpredictable. In the flow of production activities, the production demand information is passed to the advanced planning and scheduling (APS) to generate the operation plan and scheduling, and in the face of changes in demand (such as additional orders, cancel orders, change orders, emergency orders) and uncontrollable factors in the production process (such as equipment failures) and so on how to quickly adjust the production plan, and validate it effectively in order to assist in the decision-making process, the need for simulation of the production plan. the APS sends the Production tasks issued to the production plant, the formation of logistics requirements, how to effectively organize a variety of logistics equipment to ensure that materials are quickly and accurately delivered to the designated location, to eliminate logistics bottlenecks, to achieve production balance, it is necessary to carry out simulation of production logistics. In the case of production flexibility, while ensuring the stability and efficiency of the production system, check the correctness of the control logic, it is necessary to simulate the production process. Simulation content mainly includes production planning simulation, production logistics simulation and production process simulation.

Agent-based simulation is completely different from the traditional top-down, static hierarchy-based simulation method, the Agent model can better describe the dynamic, stochastic and complex systems for the distributed network structure of the manufacturing system under the scheduling, production logistics and production logic to provide simulation verification and decision-making support.

### *2.5. Agent-based Manufacturing System Monitoring*

Real-time and effective monitoring is one of the goals of digitalization construction, and it is a concentrated expression of the combination of reality and reality. The hierarchical model of manufacturing system is shown in Figure 4, which can be divided into equipment level monitoring, unit level monitoring and workshop level monitoring.

The research and application of Agent technology in manufacturing system monitoring can be divided into three levels: equipment, unit and workshop, because the advantage of Agent lies in the study of complex, dynamic and distributed systems, so there is more research on Agent-based MES. At the same time, it can be seen that the convergence of physical information has prompted the traditional MES to shift from a pyramidal hierarchical structure to a network structure of distributed operating entities, and Agent-based MES has become a trend in the study of distributed network structure. In addition, the integrated application of multiple key technologies (e.g., RFID, OPCUA, etc.) to realize the monitoring and control of manufacturing systems is also one of the trends.

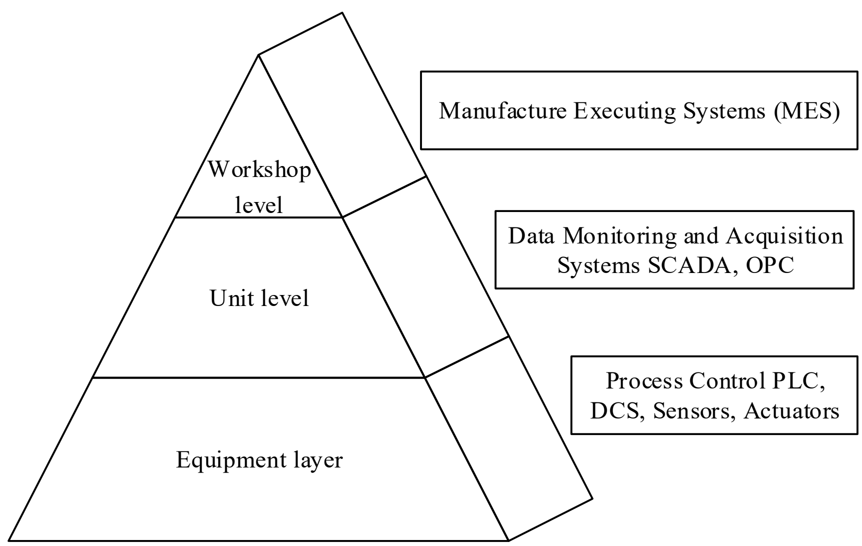


**Fig. 4.** Layered model of production system

## 3. Reinforcement Learning Based Collaborative Modeling for Multi-Agent Scheduling

In order to optimize the performance of the multi-agent based intelligent manufacturing system, this paper combines the reinforcement learning theory and designs the SRL_M3DDPG multi-agent collaboration algorithm based on state representation learning, which realizes the intelligent scheduling in the manufacturing process through the collaboration of multi-agents.

### *3.1. Enhanced learning*

Reinforcement learning (RL) is a machine learning approach to artificial intelligence that involves the creation of problematic computer programs capable of solving problems that require intelligence.RL is unique in that it learns by trial and error and derives its information from feedback that is continuous, evaluative, and sampled through the use of powerful nonlinear function approximations.

3.1.1. Components of Enhanced Learning. Intelligentsia: the ontology of reinforcement learning.

Environment: the external world where the intelligences are located and interact with.

State: (global state, local state) The state represents the reality of the environment and contains all the necessary information needed to accomplish the task.

Observations: Observations are the information actually received by the intelligence, which may be noisy or incomplete.

Actions: (Discrete Actions, Continuous Actions) A data representing the environment, and the state set is all possible states in the environment.

Reward values: (global rewards, local rewards) An intelligent body receives rewards based on the feedback provided by the environment. A reward is an evaluation of an intelligent body's action, indicating the effectiveness of that action in achieving a goal.

Strategy: reinforcement learning is learning from a mapping of environment states to actions, call the mapping relationship a strategy. It represents the thought process of how an intelligent body chooses an action.

Goal: Intelligentsia automatically find the optimal strategy in a continuous time series, i.e., maximize long-term cumulative rewards.

Reinforcement learning is actually the process by which an intelligent body learns to make optimal decisions while interacting with the environment. An intelligent body interacts with the environment through states, actions, and rewards. Suppose the environment is currently in state $s_t$ at moment $t$, and the intelligent body performs a certain action $a_t$ in the environment, which changes the original state of the environment and causes the intelligent body to arrive at a new state $s_{t+1}$ at moment $t+1$, and in the new state the environment generates feedback rewards $r_t$ to the intelligent body. The intelligent body performs a new action $a_{i+1}$ based on the new state and the feedback reward, and so on iteratively interacts with the environment through feedback signals. The ultimate goal of the above process is for the intelligent body to maximize the cumulative rewards, formulated as cumulative rewards $G = r_1 + r_2 + \ldots + r_n$. In addition, the rules for how to select actions based on states and rewards are called policies $\pi$, and the optimization objective of reinforcement learning policy optimization can be expressed as:

$$\max mize \sum_{t}^{\infty} r_t \tag{1}$$

3.1.2. Markov decision process. Markov Decision Processes (MDPs), mainly based on Markov processes and dynamic programming theory, provide a mathematical framework for sequential decision making to represent and deal with decision problems with uncertainty and delayed feedback [36]. This modeling approach allows reinforcement learning algorithms to learn optimal decision strategies by interacting with the environment. A Markov decision process generally consists of the following quintuple:

$S$: The set of all possible states of the environment, also known as the state space.

$A$: The set of all possible actions of the intelligences, also known as the action space.

$R: S \times A \times S \to \mathbb{R}$ : The reward value function of the environment.

$T: S \times A \times S \to [0,1]$: The state transfer probability function of the environment.

$\rho_0$ denotes the initial state distribution $\rho_0 : S \to \mathbb{R}_{\in[0,1]}$.

$\gamma$ : Discount factor.

The intelligent body interacts with the environment (MDP) at discrete time steps by executing strategy $\pi : S \to P(A)$. The strategy $\pi$ is defined as a probability distribution over the action space conditional on each state: $S \times A \to [0,1]$. The intelligent body learns the strategy with the objective of the total reward that can be obtained from a given state under a given strategy. The formula is as follows:

$$J(\pi) = \mathrm{E}_{A_0,\pi,T}\left[\sum_{t=0}^{\infty}\gamma^t r_t\right] \tag{2}$$

where the state of the environment is initialized to $s_0$ at the beginning of training and is sampled from the initial state distribution: $s_0 \sim \rho_0(s_0)$, and at each time step, depending on the state of the environment: $s_{t+1} \sim T(\cdot \mid s_t, a_t)$, the intelligent body selects the corresponding action $a_t$, where $a_t \sim \pi(s_t)$, $s_{t+1}$. After the above transfer computation is completed, the reward value is obtained $r_t = R(s_t, a_t)$, and $\gamma$ is a discount factor, with a value between 0 and 1, which is used to denote the current value of the future reward. A central concept of reinforcement learning is the training value function, which defines the expectation of a state or state action with respect to the long-term gain obtained by following strategy $\pi$. Specifically, the state value function $V^x(s)$ and the action value function $Q^\pi(s,a)$ are defined as the expectation of the long-term gain obtained by following strategy $\pi$, described as follows:

$$V^\pi(s) = \mathrm{E}_\pi\left[\sum_{t=0}^{\infty}\gamma' r_{t+1} \mid s_0 = s\right]\forall s \in S \tag{3}$$

$$Q^\pi(s,a) = \mathrm{E}_\pi\left[\sum_{t=0}^{\infty}\gamma' r_{t+1} \mid s_0 = s, a_0 = a\right]\forall s \in S, a \in A \tag{4}$$

The learning objective of reinforcement learning in a Markov decision process is to find the optimal maximized value function of strategy $\pi_*$, i.e., $\pi_* = \arg\max_\pi V^\pi(s) = \arg\max_\pi Q^\pi(s,a)$.

3.1.3. Semi-Markov decision processes. A semi-Markov decision process is a special kind of Markov decision process characterized by the fact that the state transfer does not occur uniformly at each time step but follows a specific probability distribution. Based on the concepts of temporal abstraction and and semi-Markov process, hierarchical reinforcement learning works to decompose a complex problem into a number of sub-problems, and solves the sub-problems one by one by a divide-and-conquer approach thus finally solving the long-term problem. There are two approaches to subproblem decomposition:

1) All subproblems work together to solve the decomposed task.

2) Continuously adding the results of the previous subproblem to the next subproblem solution. First the concept of time abstract action is formally defined as an option, an option $o \in O$ represents a triad $\{I_o, \pi_o, \beta_o\}$, where $I_o \in \mathrm{S}$ is the initial state set, $\pi_o$ is the option internal policy, and $\beta_o : I_o \to [0,1]$ is the termination function of the option for deciding the probability $s \in I_0$ that the option $o$ will terminate in state $s$.

Equipping a set of options in a Markov decision process becomes a semi-Markov decision process, while the optimal option value function on the included options is used to select the optimal option to execute the option internal policy. The formal definition of the option value function $Q_o^\pi(s,o)$ is as follows:

$$Q_o^\pi(s,o) = \mathrm{E}_\pi\left[\sum_{t=0}^{\infty}\gamma' r_{r+1} \mid s_0 = s, o_0 = o\right]\forall s,o \in S \times O \tag{5}$$

The above equation represents an estimate of the long-term cumulative return obtained by following strategy $\pi$, in state $s$ option $o$. where $\pi$ is an option-based strategy. The option learning framework uses the call-and-return execution mechanism, specifically, the intelligent body selects an option $O$ according to its option value function $Q_o^{\pi}(s,o)$ and follows the option internal policy $\pi_o$ to select the original action and the environment returns to the next state, at which time the intelligent body decides whether to terminate the option according to the termination function of option $O$, and if it does, it selects a new option according to the option value function $Q_o^{\pi}(s,o)$ and repeats the The above process is repeated.

3.1.4. Actor-Critic based algorithms. The Actor-Critic based reinforcement learning algorithm (Actor-Critic), which combines a value function based reinforcement learning approach and a policy based reinforcement learning approach, uses Actor to directly optimize the policy parameters while also using the Critic estimation of the value function $V(s)$ for evaluating the performance $\hat{Q}(s_t,a_t)=r_t+\gamma V(s_t)$ of policy $q^{\pi_\phi}(s,a)$ [37].

Unlike stochastic policy gradient algorithms that output policies that are probabilistic distributions over the action space, deterministic policy gradient algorithms (DPGs) optimize deterministic policies $\mu_\phi$ with the objective of maximizing a $Q$ function. For example, in the Deep Deterministic Policy Gradient algorithm (DDPG), the policy update formula is as follows:

$$\nabla_\phi J(\phi)=\mathrm{E}_{\mu_\phi}\left[\nabla_\phi \mu_\phi(s)\nabla_a Q^{\mu_\phi}(s,a)\Big|_{a=\mu_\phi(s)}\right] \tag{6}$$

### *3.2. Multi-Agent Collaboration Algorithm Based on State Representation Learning*

Multi-agent reinforcement learning method is an end-to-end learning method. However, the inputs for end-to-end learning are more features of the data than the original data. Especially in the case of high dimensionality of the data, such as image problems, the high dimensionality of the data leads to a high dimensional catastrophe. In order to better extract the features of the data, this paper proposes a method for state representation learning using neural networks, so that the network itself can capture the features well and be more adaptable to the data, and associating the method with the M3DDPG algorithm produces the SRL_M3DDPG algorithm.

3.2.1. MADDPG algorithm. The MADDPG algorithm framework mainly embodies the idea of centralized training and decentralized execution [38]. It separates the training of Critic network and Actor network, in which the training of Critic network not only needs the information of local Agent, but also needs the state information of other Agents in the system, while the Actor network only needs to be trained according to the information of local Agent, in this way for the behavior of local Agent, Critic network Critic network will reward the local Agent based on the overall, global information, which is conducive to the realization of collaborative intelligent behavior of the Agent group. This algorithm brings multi-agent collaboration algorithms to a new level.

The MADDPG algorithm allows Agents to learn to cooperate and compete. For each Agent, a small batch of $S$ samples are randomly drawn $D$ from the response buffer $S\left(x^j,a^j,r^j,x^j\right)$ to the Critic network and the output $Q$ values are fed back to the Actor whose objective function is:

$$y^j=r_i^j+\gamma Q_i^{\mu'}\left(x'^j,a_1',\ldots,a_N'\right)\Big|_{a_k'=\mu_k'\left(o_k^j\right)} \tag{7}$$

The Critic network loss formula is:

$$\mathrm{L}\left(\theta_i\right)=\frac{1}{s}\sum_j\left(y^j-Q_i^{\mu}\left(x'^j,a_1',\ldots,a_N'\right)\right)^2 \tag{8}$$

After Actor receives the value of $Q$, it optimizes the parameters by calculating the policy parameters using the local Agent's network. The optimization formula is:

$$\nabla_{\theta_i} J \approx \frac{1}{s}\sum_j \nabla_{\theta_{\mu\mu}}\left(o_i^j\right)\nabla_{a_i} Q_i^{\mu}\left(x^j, a_1^j, \ldots, a_N^j\right)\Big|_{a_i=\mu_i\left(o_i^j\right)} \tag{9}$$

3.2.2. M3DDPG Algorithm. The M3DDPG algorithm is an improved algorithm of the MADDPG algorithm.The M3DDPG algorithm was proposed to address the problem that Agents trained by deep reinforcement learning tend to be fragile and sensitive to the training environment, especially in multi-agent scenarios [39].

In order to learn robust strategies, the researchers suggest updating the strategy when considering the worst case scenario: optimizing the cumulative reward of each Agent under the assumption that all other Agents behave in the opposite way this produces the very large minima learning objective $\max_{\theta_i} J_M\left(\theta_i\right)$:

$$J_M\left(\theta_i\right) = E_{s^o \sim \rho}\left[\min_{\alpha_{j\neq 1}^o} Q_{M,j}^{\mu}\left(s^0, a_1^0, \cdots, a_N^0\right)\Big|_{a_i^o=\mu\left(o_i^o\right)}\right] \tag{10}$$

Update Critic by minimizing the loss. The loss formula is:

$$\mathrm{L}\left(\theta_i\right) = \frac{1}{s}\Sigma_k\left(y^k - Q_{Mi}^{\mu}\left(X^k, a_1^k, \ldots, a_N^k\right)\right)^2 \tag{11}$$

The Actor is updated using the sampled strategic gradient and the optimization formula is:

$$\nabla_{\theta_i} J \approx \frac{1}{S}\sum_k \nabla_{\theta_1}\mu_i\left(o_i^k\right)\nabla_{a_i} Q_{M,i}^{\mu}\left(x^k, a_1^*, \ldots, a_i, \ldots, a_N^*\right)\Big|_{a_i=\mu_i} \tag{12}$$

Update the target network parameters for each Agent:

$$\theta_i' \leftarrow \tau\theta_i + (1-\tau)\theta_i' \tag{13}$$

In the M3DDPG algorithm, a Multi-Agent Adversarial Learning (MAAL) approach is used to deal with the embedded minimization problem. The inner loop minimization is replaced by gradient descent method by approximating the nonlinear $Q$ function by a local linear function. The researcher defines a set of disturbance values ò that interfere with the value of the action that reduces the maximum $Q$ value of $a^*$. By linearizing the Q-function $Q_{M,i}^{\mu}\left(x, a, \ldots, a_N\right)$, we seek a disturbance value $\grave{o}_j$ that can be locally approximated in the direction of a small gradient $Q_{M,i}^{\mu}\left(x, a', \ldots, a_N'\right) w.r.t.a_j'$. The researcher then uses this localization to derive an approximation $\hat{\grave{o}}_j$ in the worst case by perturbing it by a small gradient step:

$$\forall j \neq i, \hat{\grave{o}}_j = -\alpha\nabla_a Q_{M,i}^i\left(X, a_1^i, \ldots, a_N'\right) \tag{14}$$

In the public notice (14) $\alpha$ is an adjustable coefficient representing the perturbation rate. It can also be understood as the step size of the gradient descent step: when $\alpha$ is too small, the local approximation error is smaller but due to the relatively small perturbation, the learning strategy may be far away from the very small and very large goal, i.e., the goal we need; when $\alpha$ is too large, the approximation error may cause too much trouble, and the whole learning process Agents may learn the strategy from failure frequently.

3.2.3. SRL_M3DDPG algorithm:

1) State Representation Learning. State representation learning is an important approach to the study of interaction problems, and its goal is to find a mapping from general observations that enables the selection of the correct behavior. However, reinforcement learning also has the concepts of state, behavior, and observation, which provides a prerequisite for using state representation learning in reinforcement learning algorithms. State representation learning can also use machine learning

methods. In this paper, using deep neural networks, we were able to establish a mapping relationship from state values to observations, and then the predicted observations were used as input parameters to the M3DDPG algorithm, which obtained the corresponding actions as well as the rewards, which established the predicted observation-behavior mapping. This use of state representation learning method to extract relevant information from high dimensional observations reduces the processing of high dimensional data as well as reduces the dependence on specific knowledge.

In the field of reinforcement learning, define an environment $\varepsilon$ in which the actions performed by Agenti at step $t$ are $a_i$ Each action causes the Agent to switch from the true state $\tilde{s}$ and the Agent obtains an observation $O_t$ from the sensor at step $t$ Normally, the Agent can be rewarded $r_t$ by performing the action in a given state at step t. The reward is given by a reward function that the Agent tries to maximize, and the reward equation is designed to indicate the quality of the Agenti action.

In the DRL process, which requires a lot of experimentation, especially in the automation of complex behaviors, an autoencoder is used to compress the observer into a low-dimensional state vector and train it to construct a mapping of observed states.

State Representation Learning (SRL) establishes a mapping from observation $O_t$ to the true state $\tilde{s}$ via deep neural networks (DNNs). The purpose of the state prior approach in this section is to establish a state-to-observation mapping relationship. The observed states are observations obtained from the environment. Given the initial state of each Agent $i$, the Agent's next observation and reward are predicted by training, and the reward helps to evaluate the behavior.

2) SRL_M3DDPG algorithm. The SRL_M3DDPG algorithm consists of three DNNs (SRL Model Network, Critic Network and Actor Network). In the first Deep Neural Networks (DNNs), the SRL model with a priori is constructed with the number of input neurons equal to the number of columns of the state matrix. The number of output neurons is equal to the number of columns of the observation matrix, and the hidden layer has 20 neurons. The rest of the DNNs parameters are consistent with the M3DDPG algorithm.

The SRL model defines a loss function $\mathrm{L}$ applied to a set of observations $(o_1,\ldots,o_n)$ minimized under $C$ conditions, where the loss function is:

$$loss = \mathrm{L}_{prior}\left(o_1,\ldots,o_n;\theta_\varphi \mid C\right) \tag{15}$$

The SRL model loss function equation is:

$$\mathrm{L}_{SRL} = -\sum_k y_k \log x_k \tag{16}$$

In step $t$, the real state $s_t$ and the actual observations $o_t$ are obtained and after the first DNNs the predicted values are obtained $\tilde{o}_t$. In the first DNNs, the number of input neurons is equal to the number of columns in the state matrix. The number of output neurons is equal to the number of columns in the observation matrix, and the hidden layer has 20 neurons. In the remaining DNNs (Critic network and Actor network), the Actor network selects an action matrix based on the two obtained parameters $\tilde{o}_t$ and $s_t$ and the M3DDPG's policy $a_t$, and the Critic network evaluates the matrix based on the state information and other information (e.g., information about the actions of the other Agents) $a_t$, and selects an optimal action $a_t$. When the Agent completes the action $a_t^*$ it is rewarded $r_t$ and reaches the next state $S_{t+1}$.

In the experiments, the SRL network model is first trained. The first DNNs are trained for 60,000 times, and the adelta optimizer is used to reduce the loss and adjust the weights by backpropagation. When the first layer network weights are fixed,the Actor and Critic network weights are updated.

# 4. Model application and analysis

## *4.1. Algorithm Solving and Verification*

This section focuses on the validation of the effectiveness of the collaborative model integrating reinforcement learning and multi-agent by building the scheduling environment and constructing the model of the reinforcement learning Agent in an object-oriented form using Python 3.10, and training the Agent decision module with the SRL_M3DDPG algorithm and applying it to the Dynamic Flexible Job Shop Scheduling Problem (DFJSP).

4.1.1. Algorithm performance testing:

1) Case Parameter Setting. In order to show the superiority and versatility of integrated reinforcement learning and multi-agent collaborative modeling, the parameters of the scheduling environment as shown in Table 1 are set to randomly generate the required simulation cases, with arrival times between workpieces obeying an exponential distribution of $X \sim Exp(5)$ and expiration times of $Di = Ai + \varphi \sum_{j=1}^{n} \overline{T}_{ij}$. Where $\overline{T}_{ij}$ denotes the cumulative average processing time of all operations on the job, and the tightness of the delivery date is denoted in $\varphi$ as a random coefficient between 1 and 2.

**Table 1.** Scheduling environment parameters

| Parameters | Value |
|---|---|
| Number of equipment | {5,10,15} |
| Number of jobs | {5,15,25} |
| The operand in the job | Rand [1,5] |
| The processing time of an operation on a computer | Rand [0,10] |
| The delivery factor of the workpiece | U [1,2] |

The scheduling environment parameters shown in Table 1 are used to randomly generate the arithmetic data, in which 5 randomly arriving workpieces with workpiece arrival times of 0, 0, 0, 4, 8, and 10, and the workpiece delivery urgency coefficient is set to 0.8.

2) Training hyperparameters. The SRL_M3DDPG algorithm is used in a random generation algorithm to train the decision module of the scheduling intelligences. The training hyperparameters and their values are shown in Table 2.

**Table 2.** SRL_M3DDPG algorithm parameters

| Algorithm parameter | Value |
|---|---|
| Learning rate of policy network | 0.0004 |
| Routing policy Indicates the structure of the network | [6,64,64,4] |
| The structure of the sorting strategy network | [6,64,64,5] |
| Learning rate of value network | 0.0004 |
| Structure of the value network | [6,64,64,2] |
| Strategy loss | 1 |
| Loss of value | 0.5 |
| Discount factor r | 0.97 |
| Crop parameter ε | 0.21 |
| Number of iterations | 5000 |
| Batch size | 90 |
| Optimizer | Adam |

3) Algorithm performance analysis. Example calculations can be generated based on the parameters in the table. The arrival of a new job or the completion of an operation is defined as a system event that triggers rescheduling. To further analyze the effectiveness of the proposed multi-agent based reinforcement learning scheduling approach for solving the DFJSP problem, the convergence performance of the algorithm is analyzed with the generation of examples of calculations, where the algorithm is trained for a simulated flexible job shop. The global delay penalty for the first 5000 rounds computed by the proposed SRL_M3DDPG algorithm is shown in Fig. 5. As can be seen from the curves, the objective value smoothly decreases and the volatility gradually decreases with the increase of the training step. After the 3400th round, the learning curve remains relatively stable. This indicates that the scheduling intelligences learn appropriate scheduling rules according to the changes in the production state, and this self-learning improves the adaptability of solving the DFJSP problem.

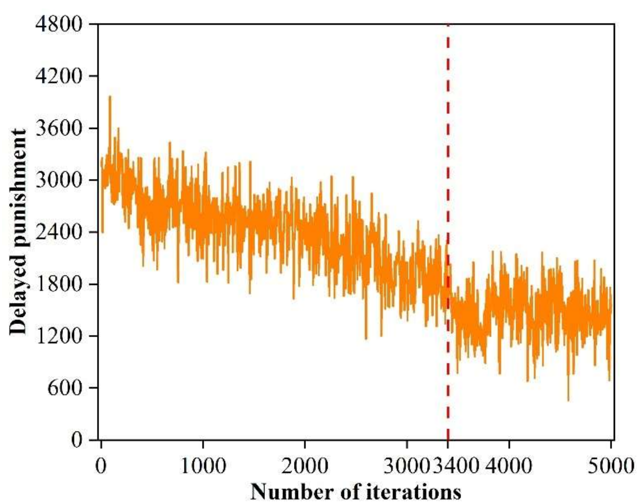


**Fig. 5.** SRL_M3DDPG iteration diagram

The scheduling of the randomly generated arithmetic example using the above methodology, the resulting Gantt chart of the scheduling result is shown in Fig. 6, with A1 to A5 denoting five different workshops. From Fig. 6, the maximum completion time for the scheduling of this arithmetic example is 29.

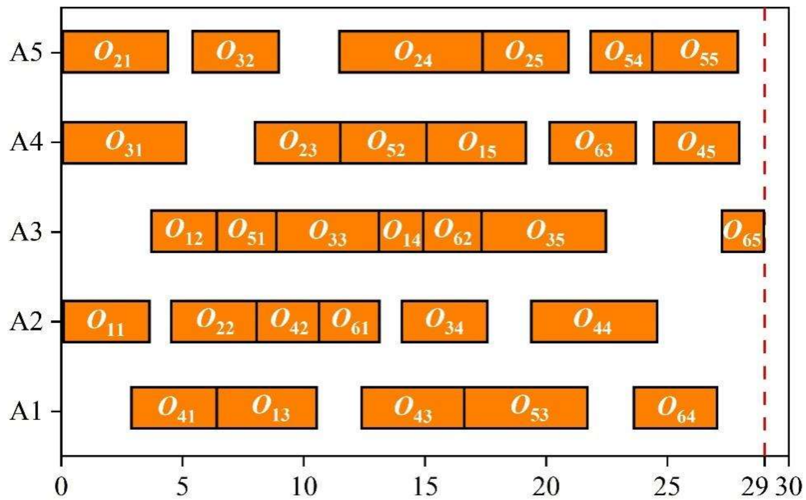


**Fig. 6.** Scheduling Gantt chart

4.1.2. Comparison with other methods. In order to verify the integration effect of the two intelligences in the scheduling collaboration model based on the SRL_M3DDPG algorithm, this paper runs them in parallel to solve the DFJSP problem. In order to show its effect, the results of testing the scheduling rules in this paper, two routing rules (CT and ET) and three sequencing rules (ATC, EDD, and MOD) of the highly advantageous priority-based scheduling rules are selected as the baseline to create six composite scheduling rules to compare and analyze with the method proposed in this paper, and the results of the latency rate of this algorithm comparing with the other composite rules are shown in Fig. 7.

Multi-agent based two-tier scheduling is where each intelligent performs decentralized scheduling decisions based on local observations. The results show that the model delay rate of this paper's algorithm is only 15.47%, which is lower than all the composite scheduling rules, and has significant superiority in reducing the artifact delay rate.

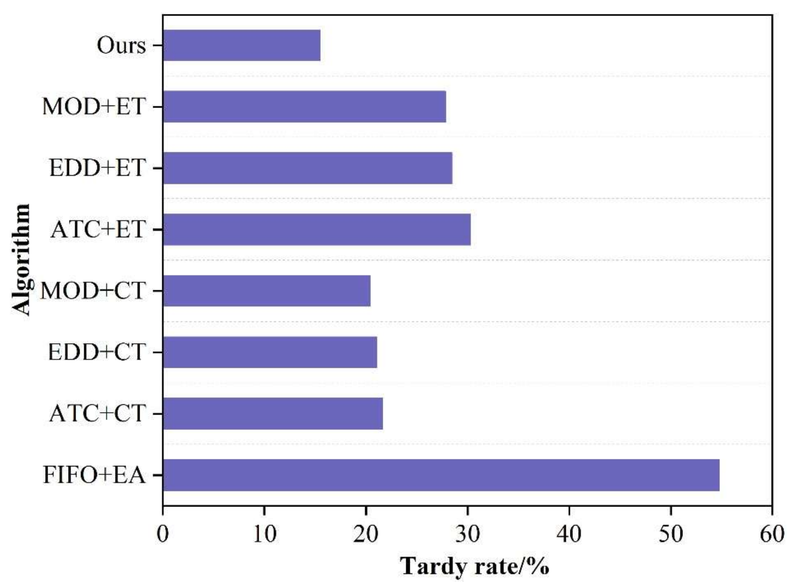


**Fig. 7.** Latency compared with other composite rules

## 4.2. Multi-Agent Dynamic Scheduling Example

Under the guidance of state representation learning, the default intra-unit scheduling method is used when the system is in steady state, and the original scheduling scheme is adjusted by dynamic scheduling when the system is in non-steady state, and the specific adjustment process is a quantitative calculation process. In order to verify the effectiveness and feasibility of the multi-agent collaborative algorithm based on state representation learning (SRL_M3DDPG algorithm) in this paper, this section will take the case of inserting an urgent task as an example to demonstrate the process of adaptive scheduling using the SRL_M3DDPG algorithm.

The process information of the flow operation of 6 workpieces with 10 processes is shown in Table 3.

**Table 3.** Processing information

| | Job piece 1 | Job piece 2 | Job piece 3 | Job piece 4 | Job piece 5 | Job piece 6 |
|---|---|---|---|---|---|---|
| Procedure 1 | 9 | 13 | 16 | 12 | 12 | 5 |
| Procedure 2 | 5 | 9 | 13 | 11 | 14 | 7 |
| Procedure 3 | 14 | 12 | 6 | 16 | 10 | 18 |
| Procedure 4 | 11 | 15 | 20 | 7 | 12 | 6 |
| Procedure 5 | 16 | 20 | 14 | 7 | 14 | 21 |
| Procedure 6 | 12 | 8 | 18 | 15 | 18 | 14 |
| Procedure 7 | 17 | 18 | 12 | 17 | 12 | 8 |
| Procedure 8 | 24 | 21 | 18 | 11 | 13 | 14 |
| Procedure 9 | 12 | 15 | 22 | 10 | 19 | 20 |
| Procedure 10 | 13 | 18 | 15 | 10 | 12 | 22 |

The scheduling Gantt chart obtained according to the in-unit scheduling method is shown in Fig. 8. Let the number of tasks to be periodically removed from the queue to be processed be N=6, and the shortest processing completion time is 230 units of time according to the SRL_M3DDPG algorithm, and the processing sequence is 3-6-1-4-2-5.

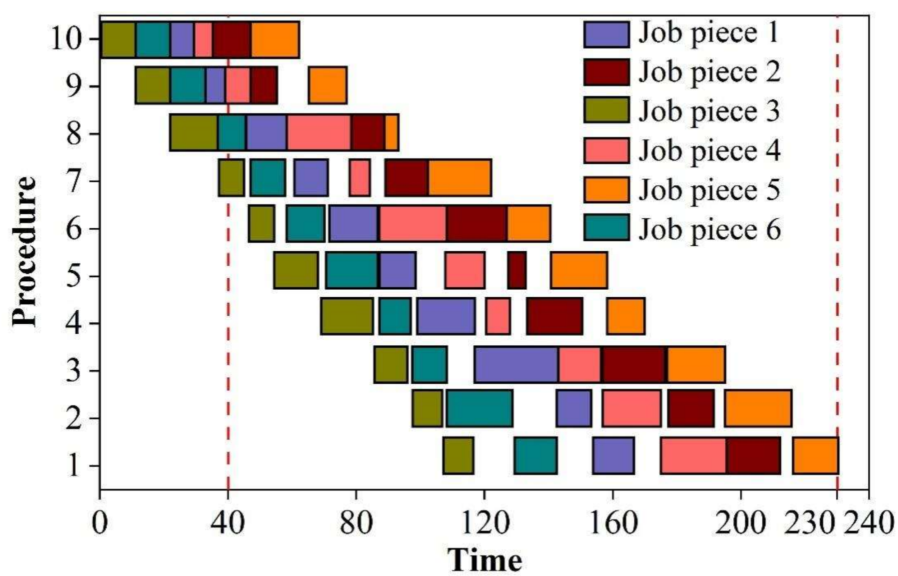


**Fig. 8.** Gantt chart for initial scheduling

Suppose that at moment point 40 the production system requires an urgent insertion of a workpiece task, and the workpiece task processing information is shown in Table 4.

**Table 4.** Emergency workpiece processing information

| Procedure | P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 | P9 | P10 |
|---|---|---|---|---|---|---|---|---|---|---|
| Emergency job piece | 17 | 11 | 9 | 12 | 18 | 10 | 15 | 17 | 16 | 14 |

According to the adaptive scheduling rules set the highest priority of the workpiece and directly into the processing queue, the processing tasks after the 40th moment in accordance with the SRL_M3DDPG algorithm for rescheduling to get the adjusted scheduling scheme shown in Figure 9.

If the arrival time of the emergency workpiece is just in the processing time of workpiece 2, the emergency workpiece should be placed at the top of the queue of the next periodic take-out task, and prioritized for processing. From the adjusted scheduling scheme, it can be seen that the insertion of the emergency workpiece is equivalent to the reordering in the processing queue according to the priority, but it will not disturb the relative processing order of the other workpieces in the queue, which is considered to ensure the stability of the production system to some extent. In this paper, SRL_M3DDPG algorithm is used for adaptive scheduling, although it cannot get the global optimal scheduling scheme, it can better adapt to the dynamic intelligent manufacturing workshop environment due to its good real-time performance.

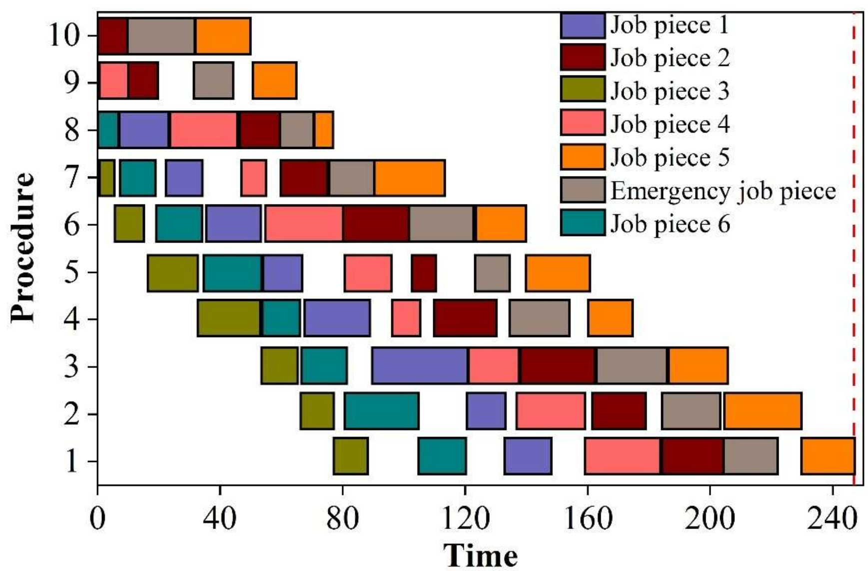


**Fig. 9.** Scheduling plan after the adjustment

## 5. Conclusion

In this paper, based on the multi-agent system, the steps and processes related to modeling, simulation and monitoring of intelligent manufacturing system are explored, and the SRL_M3DDPG algorithm based on state representation learning is proposed to realize adaptive dynamic scheduling of intelligent manufacturing workshops.

For the generated simulation arithmetic, the SRL_M3DDPG algorithm is utilized for training, and the obtained global delay penalty curves show that the target value decreases smoothly and the volatility decreases gradually with the increase of the training step. However, the learning curve remains relatively stable after the 3400th round, and the maximum completion time for the scheduling of this algorithm is 29. After the training of the SRL_M3DDPG algorithm, the scheduling intelligences are able to learn the scheduling rules according to the changes of the production state, and this self-learning approach improves the adaptability of solving the Dynamic Flexible Job Shop Scheduling Problem (DFJSP) problem. Meanwhile, comparing with other composite scheduling rules, the model delay rate of this paper's algorithm is only 15.47%, which is significantly superior in reducing the delay rate of the workpiece.

In addition, using the SRL_M3DDPG algorithm to solve the adaptive dynamic scheduling problem, its adapted scheduling scheme has high real-time performance and can ensure the stability of the production system, which is suitable for the dynamic intelligent manufacturing workshop environment.

## References

[1] Dorri, A., Kanhere, S. S., & Jurdak, R. (2018). Multi-agent systems: A survey. Ieee Access, 6, 28573-28593.

[2] Xie, J., & Liu, C. C. (2017). Multi-agent systems and their applications. Journal of International Council on Electrical Engineering, 7(1), 188-197.

[3] Chen, F., & Ren, W. (2019). On the control of multi-agent systems: A survey. Foundations and Trends® in Systems and Control, 6(4), 339-499.

[4] Uhrmacher, A. M., & Weyns, D. (Eds.). (2018). Multi-Agent systems: Simulation and applications. CRC press.

[5] Santos, G., Pinto, T., Praça, I., & Vale, Z. (2016). MASCEM: Optimizing the performance of a multi-agent system. Energy, 111, 513-524.

[6] Julian, V., & Botti, V. (2019). Multi-agent systems. Applied Sciences, 9(7), 1402.

[7] Amirkhani, A., & Barshooi, A. H. (2022). Consensus in multi-agent systems: a review. Artificial Intelligence Review, 55(5), 3897-3935.

[8] Li, H. X., & Si, H. (2017). Control for intelligent manufacturing: a multiscale challenge. Engineering, 3(5), 608-615.

[9] Zhou, D., Xu, K., Lv, Z., Yang, J., Li, M., He, F., & Xu, G. (2022). Intelligent manufacturing technology in the steel industry of China: a review. Sensors, 22(21), 8194.

[10] Cheng, G. J., Liu, L. T., Qiang, X. J., & Liu, Y. (2016, June). Industry 4.0 development and application of intelligent manufacturing. In 2016 international conference on information system and artificial intelligence (ISAI) (pp. 407-410). IEEE.

[11] Day, C. P. (2018). Robotics in industry-their role in intelligent manufacturing. Engineering, 4(4), 440-445.

[12] Egger, J., & Masood, T. (2020). Augmented reality in support of intelligent manufacturing–a systematic literature review. Computers & Industrial Engineering, 140, 106195.

[13] Randhawa, J. S., & Sethi, A. S. (2017). An empirical study to examine the role smart manufacturing in improving productivity and accelerating innovation. International Journal of Engineering and Management Research (IJEMR), 7(3), 607-615.

[14] Sahoo, S., & Lo, C. Y. (2022). Smart manufacturing powered by recent technological advancements: A review. Journal of Manufacturing Systems, 64, 236-250.

[15] Zhang, L., Zhou, L., Ren, L., & Laili, Y. (2019). Modeling and simulation in intelligent manufacturing. Computers in Industry, 112, 103123.

[16] Yang, T., Yi, X., Lu, S., Johansson, K. H., & Chai, T. (2021). Intelligent manufacturing for the process industry driven by industrial artificial intelligence. Engineering, 7(9), 1224-1230.

[17] Liang, S., Rajora, M., Liu, X., Yue, C., Zou, P., & Wang, L. (2018). Intelligent manufacturing systems: a review. International Journal of Mechanical Engineering and Robotics Research, 7(3), 324-330.

[18] Nguyen, H. D., & Tran, K. P. (2023). Artificial intelligence for smart manufacturing in industry 5.0: Methods, applications, and challenges. Artificial Intelligence for Smart Manufacturing: Methods, Applications, and Challenges, 5-33.

[19] Gao, L., Shen, W., & Li, X. (2019). New trends in intelligent manufacturing. Engineering, 5(4), 619-620.

[20] Wang, L. (2019). From intelligence science to intelligent manufacturing. Engineering, 5(4), 615-618.

[21] Shen, Y., & Zhang, X. (2023). Intelligent manufacturing, green technological innovation and environmental pollution. Journal of Innovation & Knowledge, 8(3), 100384.

[22] Li, C., Chen, Y., & Shang, Y. (2022). A review of industrial big data for decision making in intelligent manufacturing. Engineering Science and Technology, an International Journal, 29, 101021.

[23] Shan, S., Wen, X., Wei, Y., Wang, Z., & Chen, Y. (2020). Intelligent manufacturing in industry 4.0: A case study of Sany heavy industry. Systems Research and Behavioral Science, 37(4), 679-690.

[24] Sajjad, A., Ahmad, W., Hussain, S., & Mehmood, R. M. (2021). Development of innovative operational flexibility measurement model for smart systems in industry 4.0 paradigm. Ieee Access, 10, 6760-6774.

[25] Chang, V., Xu, Y. K., Zhang, J., & Xu, Q. (2021). Research on intelligent manufacturing development approach for China's local valve industry. Smart and Sustainable Built Environment, 10(2), 293-321.

[26] Tuptuk, N., & Hailes, S. (2018). Security of smart manufacturing systems. Journal of manufacturing systems, 47, 93-106.

[27] Zhao, D., & Wang, M. (2024). Smart Manufacturing Promotes High-Quality Development of Enterprises in China. Sustainability, 16(23), 10431.

[28] Tao, F., Qi, Q., Liu, A., & Kusiak, A. (2018). Data-driven smart manufacturing. Journal of Manufacturing Systems, 48, 157-169.

[29] Namjoshi, J., & Rawat, M. (2022). Role of smart manufacturing in industry 4.0. Materials Today: Proceedings, 63, 475-478.

[30] Lu, Y., Xu, X., & Wang, L. (2020). Smart manufacturing process and system automation–a critical review of the standards and envisioned scenarios. Journal of Manufacturing Systems, 56, 312-325

[31] Kim, Y. G., Lee, S., Son, J., Bae, H., & Do Chung, B. (2020). Multi-agent system and reinforcement learning approach for distributed intelligence in a flexible smart manufacturing system. Journal of Manufacturing Systems, 57, 440-450.

[32] Giret, A., Trentesaux, D., Salido, M. A., Garcia, E., & Adam, E. (2017). A holonic multi-agent methodology to design sustainable intelligent manufacturing control systems. Journal of cleaner production, 167, 1370-1386.

[33] Hsiao, J. H., & Chen, K. C. (2019, August). Network analysis of collaborative cyber-physical multi-agent smart manufacturing systems. In 2019 IEEE/CIC International Conference on Communications in China (ICCC) (pp. 219-224). IEEE.

[34] Pulikottil, T., Estrada-Jimenez, L. A., Rehman, H. U., Barata, J., Nikghadam-Hojjati, S., & Zarzycki, L. (2021, September). Multi-agent based manufacturing: current trends and challenges. In 2021 26th IEEE international conference on emerging technologies and factory automation (ETFA) (pp. 1-7). IEEE.

[35] Mingde Huang, Yiming Wu & Qiuxia Huang. (2025). Resilient and privacy-preserving consensus for multi-agent systems. Information Sciences121843-121843.

[36] Vipul Bansal, Yong Chen & Shiyu Zhou. (2025). Component-wise Markov decision process for solving condition-based maintenance of large multi-component systems with economic dependence. IISE Transactions(2),158-171.

[37] Fuxing Yao, Chao Sun, Bing Lu, Bo Wang & Haiyang Yu. (2025). Mixture of Experts Framework Based on Soft Actor-Critic Algorithm for Highway Decision-Making of Connected and Automated Vehicles. Chinese Journal of Mechanical Engineering(1),1-1.

[38] Ling Ai,Shaozhen Tang & Jie Yu. (2024). Multi-agent cooperative encirclement based on improved MADDPG algorithm. Journal of Physics: Conference Series(1),012033-012033.

[39] Yaozhong Zhang, Meiyan Ding, Jiandong Zhang, Qiming Yang, Guoqing Shi, Meiqu Lu & Frank Jiang. (2024). Multi-UAV pursuit-evasion gaming based on PSO-M3DDPG schemes. Complex & Intelligent Systems(5),6867-6883.