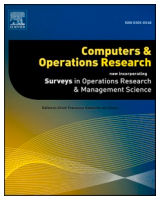
Computers & Operations Research 175 (2025) 106929

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor



Survey in Operations Research and Management Science

A literature review of reinforcement learning methods applied to job-shop scheduling problems☆

Xiehui Zhang , Guang-Yu Zhu *

School of Mechanical Engineering and Automation, Fuzhou University, Fuzhou, PR China

ARTICLE INFO

Keywords:
Reinforcement learning (RL)
Machine learning
Shop scheduling
Job-shop scheduling problem

ABSTRACT

The job-shop scheduling problem (JSP) is one of the most famous production scheduling problems, and it is an NP-hard problem. Reinforcement learning (RL), a machine learning method capable of feedback-based learning, holds great potential for solving shop scheduling problems. In this paper, the literature on applying RL to solve JSPs is taken as the review object and analyzed in terms of RL methods, the number of agents, and the agent upgrade strategy. We discuss three major issues faced by RL methods for solving JSPs: the curse of dimensionality, the generalizability and the training time. The interconnectedness of the three main issues is revealed and the main factors affecting them are identified. By discussing the current solutions to the above issues as well as other challenges that exist, suggestions for solving these problems are given, and future research trends are proposed.

1. Introduction

The job-shop scheduling problem (JSP), which has been considered a hard combinatorial optimization problem is one of the most classic nondeterministic polynomial-time hard (NP-hard) problems, and its application fields are extremely wide, involving industrial manufacturing, aerospace, logistics and transportation, medical and other fields (Chaudhry and Khan, 2016; Kayhan and Yildiz, 2023). Owing to the complexity and widespread existence of JSP, many scholars have attempted to optimize it via various algorithms, including mathematical approaches, heuristic approaches, and metaheuristic approaches (Chaudhry and Khan, 2016). While a simple structure and rapid convergence characterize these algorithms, they are often required to be manually provided with input algorithm parameters, and dynamic perturbations are not adapted. The interaction of agents with the environment and the autonomous decision-making process are the focal points of the reinforcement learning (RL) method. Therefore, RL methods are used to solve the shop scheduling problem because they are characterized by adaptive abilities and the effective handling of dynamic perturbations. Zhang and Dietterich (1995) were the first to use the RL method to solve the JSP problem; they applied the RL method to learn domain-specific heuristics for job shop scheduling. A few scholars subsequently used RL methods to solve JSP problems (Aydin and Öztemel, 2000; Wang and Usher, 2004). After a short literature review, Lihu and Holban (2009) noted that the RL method performed excellently in terms of the top five most promising algorithms for solving JSPs and that these JSPs have a large set of data with dynamic perturbations; the data are from a real automobile repair shop.

RL methods are categorized into single-agent RL methods and multi-agent reinforcement learning (MARL) methods based on the number of agents. The expansion of single-agent RL is carried out via MARL, where multiple agents are set up to work together to achieve the optimization objective. Strong robustness in solving complex problems and better performance in scheduling scenarios with multi-objective optimization and multidevice decision-making are demonstrated by MARL. RL methods can also be classified into value-based, policy-based, and combined value-based and policy-based methods, depending on how the policy is updated. The value function is the basis for action selection in value-based RL methods and is more suitable for solving discrete problems. The policy-based RL method is trained directly by the policy function to obtain the probability distribution of each action's selection in a specific state, and it has several advantages in solving continuity and large-scale problems. The combined value-based and policy-based RL method is a combination of the two methods mentioned above, in which a value-based method is introduced to evaluate the action while generating the action via a policy-based method. It combines the

☆ This work is supported in part by Natural Science Foundation of Fujian Province, under Grant 2023J01256, 2024J01349.
* Corresponding author at: Qi Shan Campus of Fuzhou University, No.2 Xue Yuan Road, Fuzhou City, Fujian Province, PR China.
E-mail address: zhugy@fzu.edu.cn (G.-Y. Zhu).

https://doi.org/10.1016/j.cor.2024.106929
Received 13 July 2024; Received in revised form 4 October 2024; Accepted 24 November 2024
Available online 27 November 2024

advantages of the two, which can not only solve the problem of continuity but also has a more stable and efficient training process.

Earlier research focused on applying Q-Learning (QL) to solve simple dynamic JSPs (Wei and Zhao, 2004), and QL achieves satisfactory results in addressing such problems. As research progresses, scholars have reported that the value-based RL method is affected by the curse of dimensionality as the number of factors considered by JSP continues to grow. The complexity of the problem increases, and the state and action spaces that need to be represented by the shop scheduling problems grow exponentially. The training process cannot converge, and enough experience is not learned by the agent to face various unexpected situations (de Witt et al., 2019). This bottleneck has stalled research on applying RL to shop scheduling problems for a long period. The resolution of the curse of dimensionality catastrophe through more in-depth research on RL methods and the popularity of techniques such as neural networks (NNs) led to the regained popularity of research on the use of RL to solve shop scheduling problems.

In addition to the curse of the dimensionality catastrophe problem, another issue of great interest in research is the generalization of scheduling methods. A lack of generalization in the scheduling approach can result in the desired results being achieved only when the shop scheduling model used for training is solved by trained agents (Hottung et al., 2021). In the context of an increasing number of customized products, scheduling methods lacking generalizability necessitate frequent retraining and do not meet intelligence requirements. Furthermore, the training time for agents should not be overlooked. The challenging generalization performance of most current scheduling systems makes it necessary to retrain trained agents before they can be directly applied to new environments. The shortening of training times enables the dispatch system to go into production faster in a new environment, leading to significant economic benefits.

In recent years, the use of RL methods to optimize combinatorial optimization problems or machine scheduling problems has become a major focus in these research fields. Mazyavkina et al. (2021) introduce three categories of RL methods to solve five typical combinatorial optimization problems, including traveling salesman problems, but exclude JSPs, and they make a comparison with traditional algorithms, indicating that RL models can become a promising direction for solving combinatorial problems. Kayhan and Yildiz (2023) analyzed 80 articles published from 1995 to 2020 on machine scheduling problems from different aspects, such as applied algorithms, machine environments, job and machine characteristics, objectives, and benchmark methods. They provide the most studied types of machine scheduling problems, potential research areas, etc. They reported that job shop scheduling, unrelated parallel machine scheduling, and single machine scheduling problems are the most studied types of problems.

As described in the following section, JSPs have more obvious complexity. Therefore, this paper aims to do the following. Taking RL methods for solving job shop scheduling problems as the research object, we analyze the main techniques proposed in the literature to solve JSPs from three aspects: RL methods, the number of agents, and the agent upgrade strategy. We summarize the thorny problems faced by RL methods in solving JSPs and provide insight for scholars, as well as future development trends. For this purpose, 93 papers on RL methods for solving JSPs from 2016 to 2023 are screened. Through the detailed analysis, this paper provides insight to scholars that it is necessary to focus on three problems: the curse of dimensionality, generalization and training time when RL methods are used to solve JSPs.

To facilitate the reader's grasp of the knowledge system of RL methods to solve JSPs and understand the connections between the sections of the paper, this review constructs the structure by answering the following four questions.

- Q1: How is the curse of dimensionality in value-based RL for solving JSPs addressed by current research?
- Q2: How can the generalizability of RL-based shop scheduling methods be improved by current research?
- Q3: How is the training time for RL agents being reduced by current research?
- Q4: What are the remaining challenges to researching the three issues mentioned above? How should these challenges be addressed in the future?

The remainder of this paper is organized as follows: Part 2 gives a brief introduction to JSP; Part 3 introduces RL, highlighting common element-setting methods and policy training algorithms in JSP; Part 4 provides a review of the screened literature; Part 5 first answers Q1, Q2, and Q3 based on the review section and then presents the challenges and further research directions through Q4; and finally, the paper concludes in Part 6.

2. Introduction of JSP

JSP can be described as n jobs to be machined on m machines, where each job has o processes, and the machining order of these o processes is determined, but there is no sequence between the processes of different jobs. In machining, a certain process of a job needs to be completed on a specific machine, and a certain amount of time is needed. Each machine can only process one part at a time and cannot be stopped until it is finished, so the machining sequence of all processes needs to be optimized to achieve certain optimization objectives (Xie et al., 2019).

The extensible directions of JSP include three aspects: constraints, dynamic perturbations, and optimization objectives.

Actual JSPs are often complex and require additional constraints to be added to the above constraints, depending on the characteristics of the workshop. The increase in constraints usually complicates the problem. For example, if multiple processing machines can be selected for a particular process of a workpiece, and the workpiece may be processed at different times on different machines, the job-shop scheduling problem becomes a flexible job-shop scheduling problem (FJSP).

Second, there are usually various dynamic perturbations in the actual shop floor, which may have a great impact on the scheduling results and therefore cannot be ignored in the scheduling system sink. Dynamic perturbations can be broadly categorized into two main types: internal and external perturbations. Internal perturbations include machine failures, deviations in process parameters, uncertainty in machining times, rework of defective products, etc. External perturbations include emergency order insertions, order cancellations, etc. JSPs that consider dynamic perturbations are uniformly called dynamic job-shop scheduling problems (DJSPs). The DJSP and the FJSP form a dynamic flexible job-shop scheduling problem (DFJSP).

In addition, in some workshops, factors such as energy consumption, environmental protection, production costs, and machine life need to be further considered in addition to completion time. For this type of problem multiple optimization objectives need to be set and solved, and different optimization objectives may have competing relationships, complicating the problem.

With the introduction of the above extensions, which make JSPs more complex, traditional algorithms often fall short in solving such problems. Therefore, different RL methods have been developed by scholars for solving these extended JSPs.

3. RL methods

A methodology for solving JSPs is provided by RL. The problem is first formulated as a suitable RL model, then the RL algorithm is introduced to train the agents, and the scheduling plan is generated using the trained agents. In addition, deep RL (DRL) is often considered a combination of deep learning and RL. This paper focuses on the performance of the scheduling system constructed using RL as a methodology, so not much expansion is provided for DRL.

3.1. Introduction of RL methods

The basic framework of RL to solve JSP is illustrated in Fig. 1, where a state is acquired by an agent through sensing the current environment, a reward is received as feedback for the last action performed, and then an action is chosen to change the environment based on a policy. The agent learns by repeating the above steps in a continuous trial-and-error manner with the goal of obtaining the best policy. The specific methods and steps used to train the agent are the various RL algorithms (Long et al., 2022).

The common mapping of various elements of the framework is presented as follows.

1) Agents: Entities that learn and make decisions are capable of having states sensed, actions performed, and rewards obtained. In JSP, agents are generally configured as transportation devices, processing machines, devices for task assignment, and decision-making components of the system.
2) Environment: The set of all states that can be perceived by an agent. In JSP, the environment generally refers to everything on the workshop that may impact the shop scheduling problem, such as dynamic perturbations and constraints.
3) State: A description of some feature of the environment is perceived by the agent. In JSP, this is usually represented as the current production state of the workshop that is acquired by the agent.
4) Action: The manipulation of the environment by the agent. In JSP, an action can be the assignment of a specific piece of work or a processing machine, the selection of a rule from a given set of scheduling rules, the adjustment of certain parameters of the system.
5) Reward: The reward that the agent receives is based on the action, and it is used to evaluate the agent's action, guiding the agent to acquire the best policy through learning. In JSP, the setting of rewards is closely related to the optimization objective; usually, a reward function related to the optimization objective is employed.
6) Policy: The action or probability distribution of individual actions selected by an agent in a given state. In JSP, policy refers to the probability that a certain machine, a certain product, or a certain processing rule should be selected based on the current state.

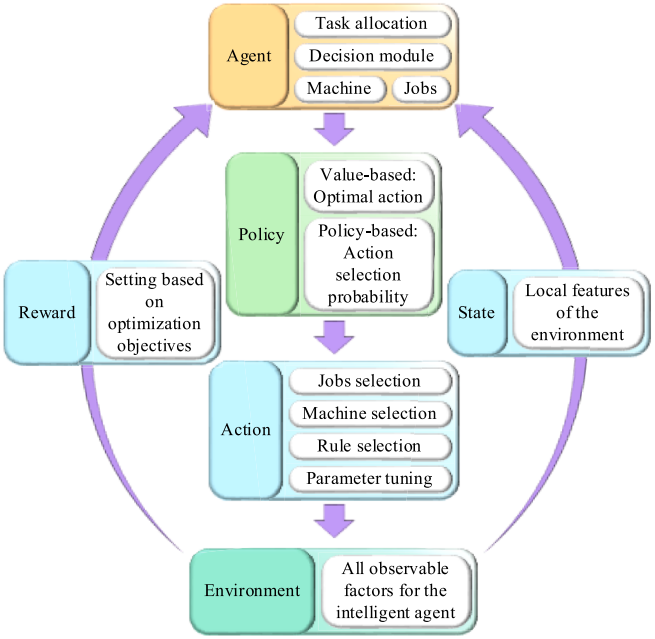


Fig. 1. RL Schematic.

3.2. Common RL algorithms in JSPs

The learning quality of the agent is directly determined by algorithms, which serve as the primary basis for naming RL methods. As shown in Fig. 2, RL algorithms can be divided into three categories: value-based, policy-based, and combined value-based and policy-based. In JSP, the commonly used value-based algorithms are QL, DQN, DDQN, etc.; the policy-based algorithms are PG, PPO, etc.; and the algorithms that combine the two, such as AC and DDPG.

The agent is trained by the value-based algorithm to obtain the optimal value function, and the agent's decision is guided by the value function to choose the action in a certain state that can yield the best long-term return.

1) Q-Learning (QL): During the training process in this value-based RL algorithm, the value function, recorded as discrete Q-values in a Q-table, is updated (Chen et al., 2020). Since discrete data can accurately record the long-term payoff of an action chosen in a given state, QL performs better in solving small-scale discrete problems but suffers from the curse of dimensionality when addressing complex problems. This issue causes the algorithm to fail to converge. Therefore, overcoming the curse of dimensionality in the state–action space is essential when the Q-Learning algorithm is used to solve complex JSPs.
2) Deep Q-Network (DQN): This is a modification of the Q-Learning algorithm to represent the Q-tables in terms of the Q-networks. The number of states to be represented is drastically reduced by the DQN, which introduces a neural network to approximate the Q-function. A target network is also introduced for evaluation to speed up convergence. Complex JSPs are addressed by the DQN, and a faster convergence speed is exhibited than that of the Q-Learning algorithm (Mnih et al., 2015). However, during value assessment, the maximum Q value of the target network is chosen by the DQN and may suffer from an overestimation of the value of the state and

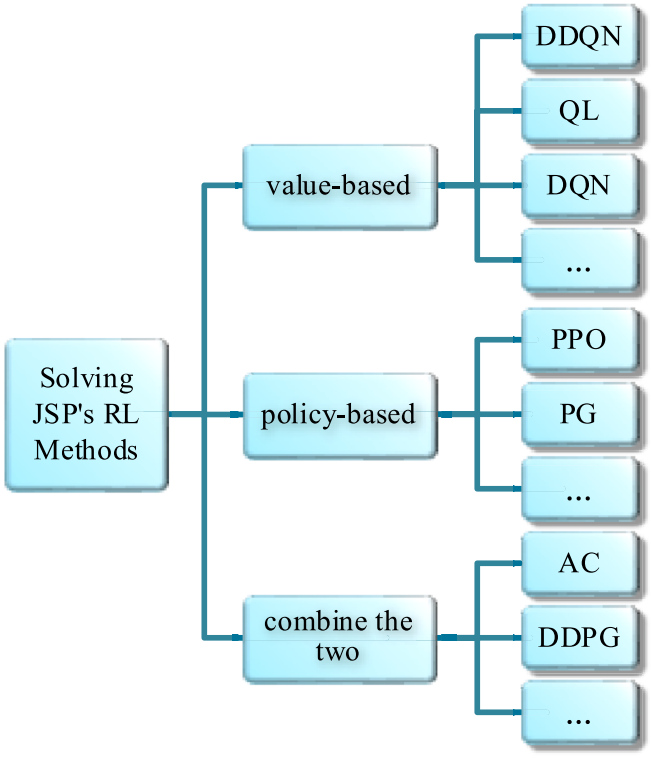


Fig. 2. RL methods for solving JSPs.

action, which can sometimes lead to convergence issues (Xu et al., 2023).

3) Double Deep Q-Network (DDQN): This improves the Q-Learning algorithm. The structures of the DDQN and DQN are the same. The primary distinction is that the largest Q value of the target network is not directly chosen for evaluation when updating the Q network in the DDQN. Instead, the best action in the Q network is selected as an input to the target network to obtain the Q value. The problem of overestimating the value of the state and action by the target network is avoided through this approach (Van Hasselt et al., 2016). Therefore, good overall performance is demonstrated by DDQN.

The agent's policy is trained directly via policy-based algorithms to obtain an optimal policy function. Actions are then performed directly by the agent based on the learned policy during its actions.

4) Policy Gradient (PG): PG is a policy-based RL algorithm that can effectively handle the continuity problem because of the direct optimization of the policy (Han and Yang, 2021). However, poor convergence performance is usually exhibited by it, with slow convergence speed and a tendency to fall into local optimal solutions. Improvement is typically needed for solving JSP.
5) Proximal Policy Optimization (PPO): This is an improvement on the PG. Although PPO is also based on the actor–critic framework, the value of the Critic output state during the training process assists the Actor in updating the policy rather than evaluating the goodness of the action. The overall focus is on policy updating, so it is categorized as a policy-based algorithm. In addition to being suitable for solving the continuity problem, the PPO algorithm limits the magnitude of policy changes at each update and has a more stable convergence process (Schulman et al., 2017). The superior comprehensive performance of the PPO algorithm has led to its widespread use in JSP solving in recent years.

A combination of value-based and policy-based algorithms combines both advantages.

6) Actor–Critic (AC): AC includes two components, the Actor and the Critic. The Actor is a policy-based algorithm responsible for generating actions, whereas the Critic is a value-based algorithm responsible for evaluating the Actor's actions (Huang et al., 2023). AC provides a good framework for such algorithms, but the basic AC algorithm for JSP solving is usually improved due to convergence difficulties.
7) Deep Deterministic Policy Gradient (DDPG): A DRL algorithm built on the framework of the AC algorithm (Lillicrap et al., 2015) allows not only complex JSPs to be effectively solved but also good convergence performance.

4. Literature review

This section reviews and analyzes the literature on RL methods for solving JSPs. First, the literature is systematically categorized, including literature selection and algorithm applications. After the literature is screened, the literature on the application of multi-agent RL (MARL) is first reviewed and analyzed in Section 4.2. There are 27 papers in this framework, accounting for 29 % of the total number of papers. Among these 27 papers, 22 used value-based RL methods, 4 used policy-based RL methods, and 1 used on combined value-based and policy-based RL methods. However, for MARL, we categorize and discuss it based on its application scenarios rather than discussing the training algorithms mentioned in these studies individually. The rest of the literature focuses on the single-agent RL method and its extensions, which are described in the literature of MARL in Section 4.3 to 4.5, depending on how the policy is updated. These methods are synthesized and analyzed via value-based RL methods in Section 4.3, policy-based RL methods in Section 4.4, and combined value-based and policy-based RL methods in Section 4.5. In Section 4.3, 37 papers (40 %) are analyzed. In Section 4.4, there are 23 papers (25 %). In Section 4.5, 6 papers (6 %) are analyzed.

4.1. Literature collation

4.1.1. Literature selection

We searched the Web of Science with the topics of "reinforcement learning" and "shop scheduling". For the content of the review to be in line with the current research interest, the literature from 2016 to 2023 was selected for review. Excluding review papers, we obtained a total of more than 200 papers on shop scheduling and RL. The application of RL methods in shop scheduling has only become topical again with the popularization of NNs in recent years, as revealed by our skimming of this literature. At the same time, articles whose research object is JSP have the highest frequency in this literature; therefore, the literature of JSP is taken as the review object in this paper. After adjusting for the above constraints, 93 papers on RL methods for solving JSPs from 2016 to 2023 are screened. The years in which the literature was published are shown in Fig. 3, which shows that the application of RL methods to JSPs gradually became an important topic in the years following 2019.

The number of documents from each publication is shown in Table 1. Solving JSP with RL essentially involves applying the relevant machine learning techniques in artificial intelligence to address problems in engineering practice. Therefore, relevant papers are included in journals across industry, artificial intelligence, computer technology, and others.

4.1.2. Algorithm applications

The commonly used RL algorithms for solving JSPs are introduced in Section 3.2. In this section, the frequency of use of various RL algorithms in the literature is summarized to more intuitively show the popularity of different algorithms for solving JSPs. Since some articles may use multiple algorithms, the sum of the frequency of algorithm use is greater than the number of documents.

As shown in Fig. 4, the value-based RL method is widely employed since the shop scheduling problem is a typical discrete combinatorial optimization problem. Although the problem of the curse of dimensionality arises in value-based RL methods as the problem is studied in depth, scholars are still trying to expand this method and address the curse of dimensionality in various ways in value-based RL methods. Therefore, the total number of applications of value-based RL methods remains high. On the other hand, owing to the excellent performance of the PPO algorithm and its advantages in solving complex JSPs, JSPs are currently being approached by an increasing number of scholars using PPO methods. In addition, JSP solving involves the application of other RL methods to explore the potential of RL methods further.

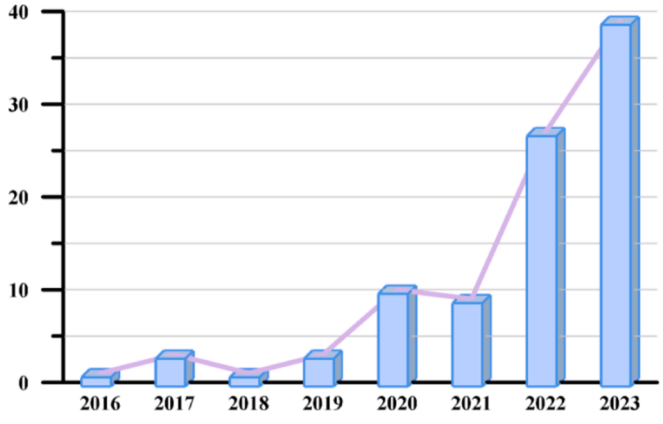


Fig. 3. Distribution of years of publication.

Table 1
Literature sources.

Title of Publication	Number
Computers & Industrial Engineering	8
Processes	5
Applied Soft Computing	4
IEEE Transactions on Automation science and Engineering	4
Journal of Manufacturing Systems	4
Expert Systems with Applications	3
IEEE Transactions on Industrial Informatics	3
International Journal of Production Research	3
IFAC Papers Online	3
Computers & Operations Research	2
Applied Sciences-Basel	2
International Journal of Simulation Modelling	2
IEEE Transactions on Systems Man Cybernetics-Systems	2
IEEE Transactions on Neural Networks and Learning Systems	2
IEEE Transactions on Evolutionary Computation	2
Journal of Intelligent Manufacturing	2
Machines	2
Procedia CIRP	2
Robotics and Computer-Integrated Manufacturing	2
Sustainability	2
Advanced Engineering Informatics	1
Advances in Neural Information Processing Systems	1
CMC-Computers Materials & Continua	1
Computer Networks	1
Evolutionary Intelligence	1
European Journal of Operational Research	1
Engineering Optimization	1
Frontiers in Environmental Science	1
Flexible Services and Manufacturing Journal	1
IEEE Access	1
International Journal of Advanced Manufacturing Technology	1
International Journal of Production Economics	1
IEEE Robotics and Automation Letters	1
Information Sciences	1
IEEJ Transactions on Electrical and Electronic Engineering	1
IEEE Transactions on Emerging Topics in Computational Intelligence	1
Journal of Computing and Information Science in Engineering	1
Journal of Intelligent & Fuzzy Systems	1
Journal of Computational Design and Engineering	1
Knowledge-Based Systems	1
Lecture Notes in Artificial Intelligence	1
Machine Learning and Knowledge Extraction	1
Proceedings of The Institution of Mechanical Engineers Part B-Journal of Engineering Manufacture	1
Science China-Technological Sciences	1
Other Conference Articles	10

4.2. The methods of multi-agent RL

Among the articles reviewed in this paper, 27 use multi-agent RL (MARL), 6 of which have hierarchical optimization objectives, and 9 use distributed RL. MARL for solving JSPs has recently become more popular; thus, a particular discussion of MARL and distributed RL is necessary. In JSP research, MARL is generally used when the following situations arise: (i) problems with a particular workshop structure, (ii) multi-objective problems, (iii) hierarchical problems, and (iv) distributed problems. In this section, the literature on MARL is categorized into points (i) and (ii), as well as other cases, such as general MARL, and (iii) and (iv) are discussed separately.

4.2.1. General multi-agent RL

One of the critical advantages of MARL is its ability to simplify complex problems to facilitate solutions. When the structure of the workshop is special, or there are multiple optimization objectives, difficulties in modeling arise with the single-agent RL method, and unsatisfactory optimization results may be caused by too many feasible solutions or interference between machines. However, these problems can be effectively solved by modeling with the MARL method and introducing suitable interaction mechanisms.

As shown in Table 2, the scheduling problem of automated guided vehicles (AGVs) was considered by Popper et al. (2021) in the JSP, where AGVs and processing machines are set up as separate agents that collaborate through a central coordination layer. The scheduling of two robots that could interfere was studied by Jungbluth et al. (2022); therefore, each of the two manipulators is set up as an agent to avoid interference by sharing the environment. MARL, while it is easier to model structure-specific scheduling problems, suffers from the following issues: (i) a global agent is usually needed, and the same problem of the curse of dimensionality is faced when complex problems are solved, and (ii) the interaction between swarm agents may lead to poor convergence performance and extended training time.

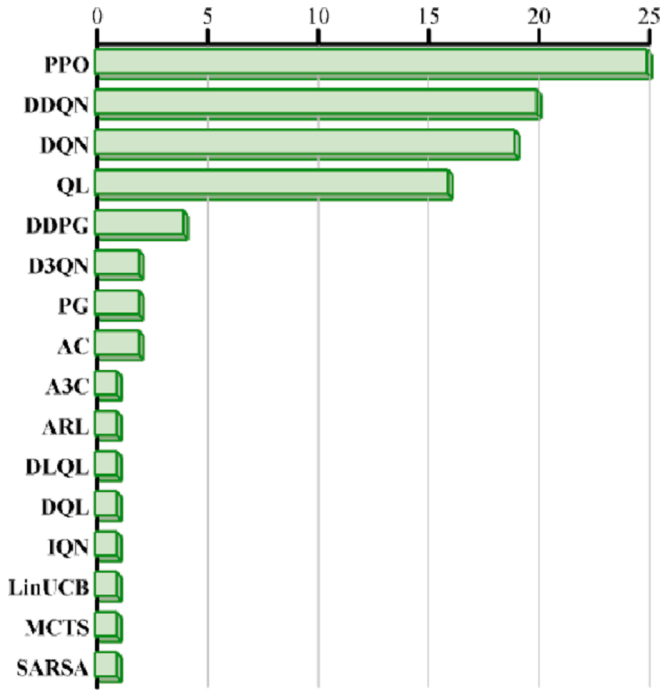


Fig. 4. Frequency of occurrence of algorithms.

4.2.2. Hierarchical RL

Hierarchical reinforcement learning is a framework that extends RL to multiple levels, where step-by-step learning is performed by RL agents to handle complex tasks efficiently. Although the single-agent RL method can also be used in hierarchical RL, MARL is adopted in the literature that uses the framework of hierarchical RL, as shown in Table 3; the compatibility of MARL is comparatively better.

The optimization objective is split and learned step by step via hierarchical RL, making the division of labor among the agents clearer, the interaction between the agents is effectively reduced, the convergence of the swarm agents is improved, and the training time is reduced. However, in the framework of hierarchical RL, a global objective still needs to be optimized by the global agent, and the curse of the dimensionality catastrophe problem may still be faced when training with value-based algorithms.

4.2.3. Distributed RL

Depending on the interaction mechanism, MARL methods can be further categorized into multi-agent joint methods and multi-agent independent methods. The MARLs of the previous two subsections fall into the former category, where the requirement for relatively complex interaction mechanisms between agents exists. The latter do not require complex interaction mechanisms, and independently learning agents are more focused on exploring the local environment, which is commonly implemented in JSP as distributed RL.

Problems in parallel are usually solved by distributed RL agents via a

Table 2
General MARL.

Reference	Problem	Objective	Algorithm	Description
Waschneck et al. (2018)	JSP	Machine average utilization maximization	DQN	A global agent is trained to optimize the global objective and influence the behavior of other agents.
Méndez-Hernández et al. (2019)	JSP	Makespan and total tardiness minimization	QL	Individual objectives are optimized by agents alone, and the global objective is collaborated on for optimization.
Baer et al. (2019)	JSP	Makespan minimization	DRL	Each product is controlled by an independent agent, and each agent acquires sparse global rewards during training.
Lang et al. (2020)	DJSP	Total earliness and tardiness minimization	QL	Four types of agents are set up to collaborate through a contract network negotiation mechanism.
Wang (2020)	FJSP	Makespan minimization	DQN	Each product is handled by an independent agent responsible for resource allocation and transportation, and the agents use indirect communication.
Pol et al. (2021)	FJSP	Makespan minimization	DDPG	Each artifact is set as an agent, and the collection of machines is considered as an environment, with the machines being selected by the agents through competition.
Wang et al. (2021b)	FJSP	Total tardiness minimization	PPO	Two types of agents, machine, and transportation, are set up, and information between the agents is obtained through a central coordination layer to prevent overlapping tasks.
Popper et al. (2021)	JSP	Makespan minimization	DQN	Two robots are set up as agents and collisions are prevented by sharing predicted trajectories.
Jungbluth et al. (2022)	FJSP	Makespan minimization	DQN	A collaborative MARL system is created where cooperative actions are taken by agents with their neighboring intelligence.
Zhang et al. (2023)	JSP	Makespan minimization	QL	Each job is considered an agent and is trained using the Q-Mixing networks (QMIX) algorithm.
Wang et al. (2022c)	DJSP	Total tardiness minimization	RL	A MARL system is proposed by RL, containing five collaborating agents with different objectives.
İnal et al. (2023)	DJSP	Total earliness and tardiness minimization	QL	Four types of agents are set up to collaborate through a contract network negotiation mechanism.

model of centralized training and decentralized execution of tasks, which can better split the solutions of complex JSPs (Johnson et al., 2022). Distributed RL methods are used in the literature, as shown in Table 4. As the size of the FJSP increases, the feasible solution space and the action space become large and are not suitable for solving via centralized RL methods. Combining independent learners (ILs) with an implicit quantile network (IQN) is also proposed. IQN, a distributed RL algorithm, has strong expressive power for distributed problems (Oh et al., 2022).

In addition, the learning speed of localized agents can be significantly improved by introducing experience-sharing when agents are centrally trained (Wang and Liao, 2024). Thus, the curse of dimensionality is effectively solved via distributed RL, which requires less training time than does general MARL. A hierarchical distributed architecture for dynamic FJSPs with continuously arriving jobs was proposed by Liu et al. (2022a), in which parameter-sharing techniques are used by agents to improve the training efficiency.

While solving large-scale problems with distributed RL avoids the curse of dimensionality and has better convergence performance, the solution accuracy may not be as good as that of general MARL. In this context, a combination of the distributed RL approach and the framework of hierarchical RL can be considered.

4.3. Value-based RL methods

Since value-based RL methods have been applied to JSP solving earlier, there are a wide variety of current approaches to value-based RL methods. The value-based RL methods are categorized as follows: (i) RL improvement of other algorithms, (ii) improvement of RL methods, and (iii) the remaining value-based RL methods.

4.3.1. RL improvement of other algorithms

The improvement of other algorithms by RL mainly refers to the application of RL to achieve the parameter adaptation of an algorithm, which is considered a simple application of the RL method. Research in this area is relatively mature, and this approach has been applied by many scholars to improve the performance of the proposed heuristic algorithms.

This category of usage falls into the literature, as shown in Table 5. The adaptive tuning of the parameters of the optimization process of the variable neighborhood search (VNS) algorithm at any rescheduling point to achieve dynamic scheduling was proposed by Shahrabi et al. (2017) via the QL algorithm. QL is introduced into their algorithms to guarantee the diversity of offspring by Long et al. (2022) and Li et al. (2022a), Li et al. (2023a). The remaining literature applied RL methods to improve the convergence performance of algorithms to obtain better solutions.

4.3.2. Improvement of RL methods

Most of the current applications of RL methods are realized in conjunction with NNs, so the most common RL improvement method is

Table 3
Hierarchical RL.

Reference	Problem	Objective	Algorithm	Description
Wang et al. (2021c)	JSP	Earliness minimization, machine average utilization maximization, balance machine load	DQL	Machine idleness and machine load balancing are optimized by the top-level agents, while lead times for all jobs are minimized by the bottom-level agents.
Luo et al. (2022a)	DFJSP	Total tardiness minimization, machine average utilization maximization, balance machine load	PPO	The job selection rules are determined by the top-level agent, and the machine assignment rules are determined by the bottom-level agent.
Luo et al. (2022b)	DFJSP	Total tardiness minimization, machine average utilization maximization	DDQN	Temporary objectives to be delegated to the bottom-level agent are determined by the top-level agent, and appropriate scheduling rules are selected by the bottom-level agent.
Chang et al. (2022a)	DFJSP	Machine average utilization maximization, total earliness and tardiness minimization	DDQN +D3QN	Proposing a hierarchical structure for DDQN and Dueling Double Deep Q-Network (D3QN). Temporary objectives for the bottom-level agent are determined by the top-level agent, and appropriate scheduling rules are selected by the bottom-level agent.
Lei et al. (2022)	DFJSP	Makespan minimization	DDQN +PPO	The dynamic problem is delegated to the bottom-level agent by the top-level agent through conversion to a static problem. The allocation of machines and jobs is the responsibility of the two bottom-level agents, respectively.
Wu et al. (2023)	DFJSP	Makespan minimization, total tardiness minimization	DDQN	The reward form of the optimization objective is selected by the high-level agent, and the appropriate scheduling rule is selected by the bottom-level agent.

Table 4
Distributed RL.

Reference	Problem	Objective	Algorithm	Description
Qu et al. (2016)	DJSP	On-time delivery, waiting products minimization.	ARL	A three-step framework, consisting of region decomposition, communication mechanism, and local RL, is proposed to train local agents with approximation reinforcement learning (ARL).
Bouazza et al. (2017)	FJSP	Minimize weighted job waiting time.	QL	The local environment is first obtained using the agents, and the most appropriate machine selection rules and product scheduling rules are selected using the QL agents.
Park et al. (2020)	FJSP	Makespan minimization.	DQN	Agents perform local optimization in a decentralized manner, and a network of policies that deals with changes in the number of machines is jointly learned.
Johnson et al. (2022)	FJSP	Makespan minimization.	DDQN	Agents are centrally trained and decentralized for localized tasks.
Oh et al. (2022)	FJSP	Makespan minimization.	IQN	A scheduling method is proposed that combines ILs with an IQN.
Liu et al. (2022a)	DFJSP	Total tardiness minimization.	DDQN	Agents undergo centralized training, and local tasks are executed decentralized. The training phase is simplified through parameter sharing.
Qin et al. (2023)	DJSP	Total tardiness minimization.	DDQN	The cooperation mechanism is established by the greedy-based strategy, realizing the collaboration mechanism between agents as partially distributed.
Zhu et al. (2023)	FJSP	Total tardiness minimization.	DDQN	Agents are centrally trained and decentralized for localized tasks.
Wang and Liao (2024)	FJSP	Average tardiness minimization.	PPO	A centralized agent is created to learn a generalized policy, and the learned policy is used for decision-making by localized agents.

the introduction of various NNs (Sun and Yang, 2023). Several studies have attempted to introduce other methods to improve the scheduling framework in RL, as shown in Table 6. The introduction of neural networks can effectively address the curse of dimensionality of QL in solving large-scale or dynamic shop scheduling problems. However, the trained intelligent descent still has difficulty to quickly generating new scheduling methods for dynamic perturbations, which is not conducive to solving JSPs with real-time scheduling requirements. Therefore, Chang et al. (2023) combined the DQN with digital twin (DT) technology to propose a new scheduling method that can be combined with workshop information more efficiently. This allows the trained agent to generate new scheduling plans quickly, enabling real-time scheduling.

In addition, improving the performance of RL through *meta*-heuristic

Table 5
RL Improvement of Other Algorithms.

Reference	Problem	Objective	Algorithm	Description
Shahrabi et al. (2017)	DJSP	Makespan minimization.	QL + VNS	The parameters of the optimization process of the variable neighborhood search (VNS) algorithm are tuned by QL for arbitrary rescheduling points.
Chen et al. (2020)	FJSP	Makespan minimization.	QL/ SARSA +GA	Parameters for genetic algorithm (GA) crossover and mutation operations are tuned by SARSA and QL.
Long et al. (2022)	FJSP	Makespan minimization.	QL + ABC	The artificial bee colony (ABC) population update dimension parameter is tuned by QL.
Li et al. (2022a)	FJSP	Makespan and total machine workload minimization.	QL+ MOEA/D	Population update parameters for the reference vector-guided multi-objective evolutionary algorithm based on decomposition (MOEA/D) are adjusted by QL.
Liu et al. (2023b)	FJSP	Makespan minimization.	QL + GA	Parameters for GA crossover and mutation operations are adjusted by QL.
Lin et al. (2022)	FJSP	Makespan minimization.	QL + GWO	The control parameters of the Gray Wolf Optimization (GWO) algorithm are adjusted by QL.
Li et al. (2023b)	FJSP	Makespan minimization.	QL + ABC	The best scheme of sublots for ABC is searched by QL.
Liu et al. (2023a)	JSP	Makespan, outsourcing costs and total energy consumption minimization.	QL + ISA	The exploration strategy of the Internal Exploration Algorithm (ISA) is selected by QL.
Li et al. (2023a)	FJSP	Makespan and total energy consumption minimization.	QL + MA	The size of the neighborhood parameter of the memetic algorithm (MA) is adjusted by QL.

algorithms is an interesting idea. A proposal that combines the DQN with the estimation of the distribution algorithm (EDA) to obtain better global optimal solutions through the powerful exploration capability of the EDA was made by Du et al. (2023). For the Q-Learning algorithm, however, the disadvantage of long training times is that it is more important than local convergence. Therefore, the hybrid genetic algorithm (HGA) was introduced by Chien and Lan (2021) as an optimizer to improve the efficiency and effectiveness of exploration by generating better quality solutions during each iteration, which in turn reduces the training time of the agent.

4.3.3. The remaining value-based RL methods

As shown in Table 7, value-based RL methods are relatively mature in JSPs, and high solution accuracy is achieved by them. In practice, however, it is often necessary to extend the existing algorithmic framework to make it more compatible with the characteristics of the JSP being solved, and objective optimization is better achieved. The algorithmic framework of the DQN for JSP was proposed to be extended by Lin et al. (2019), with edge computing in mind. Multiple output neurons are contained in the proposed multiclass deep Q-network (MDQN), enabling decision-making for multiple edge devices.

Table 6
Improvement of RL Methods.

Reference	Problem	Objective	Algorithm	Description
Drakaki and Tzionas (2017)	JSP	Makespan minimization.	QL	Colored timed petri nets (CTPNs) are introduced to model the dynamic behavior of flexible manufacturing systems.
Chien and Lan (2021)	DFJSP	Makespan minimization.	HGA+ DDQN	HGA is introduced as an optimizer.
Yan et al. (2022)	DFJSP	Makespan minimization.	DLQL	Digital twin (DT) technology is introduced, and a double-layer Q-learning (DLQL) is introduced as an optimization algorithm.
Du et al. (2023)	FJSP	Total electricity cost and Makespan minimization.	EDA+ DQN	The estimation of distribution algorithm (EDA) is introduced.
Chang et al. (2023)	DFJSP	Makespan minimization.	DQN	DT technology is introduced.

The curse of the dimensionality catastrophe problem can be effectively addressed by the approximate representation of state-action values through a neural network (Q-network) (Luo, 2020). The NN technique is employed in the current literature, applying QL as the main algorithm for solving JSPs. However, the training of NNs usually takes a long time, and attempts are made by several scholars to improve the process of exploration and utilization during agent training. The use of DDQN and soft target weights for the simultaneous updating of Q-networks for multi-objective FJSPs with crane transportation and installation times was proposed by Du et al. (2024). The use of soft target weights as an action selection policy for the DDQN algorithm for dynamic FJSPs with new job insertions was proposed by Luo et al. (2020). The probability of different actions being chosen by the algorithm in the early stages is made closer to equal, and the phenomenon that suboptimal actions may also be selected from greedy strategies in the later stages of learning is addressed. A DRL-based method was proposed by Chang et al. (2022b) to solve dynamic FJSPs with stochastic job arrivals, with the soft ε-greedy strategy being applied to balance exploration and exploitation rationally. The effectiveness of agent exploration is improved, and the agent training time is reduced by introducing an action selection policy.

4.4. Policy-based RL methods

The recording of state-action values is never discontinued in the training process of the value-based RL method, and even with the introduction of various technological tools, challenges in solving JSPs with many-dimensional state-action spaces continue to be faced. However, the policy-based RL method is trained directly without evaluating the value of the action and is better able to handle complex JSPs. As shown in Table 8, the use of policy-based algorithms to solve the JSP has been attempted by an increasing number of scholars. Policy-based algorithms are compared with value-based algorithms by Zhu et al. (2022a), who noted that policy-based algorithms have smoother search

Table 7
The Remaining Value-based RL Methods.

Reference	Problem	Objective	Algorithm	Description
Lin et al. (2019)	JSP	Makespan minimization.	DQN	The MDQN is proposed to address JSPs for multiple edge devices by adapting DQN using the Edge Computing Framework.
Seito and Munakata (2020)	JSP	Makespan minimization.	DRL	The DRL algorithm combined with graph convolutional neural networks (GCNN) is proposed to solve JSP.
Kardos et al. (2020)	JSP	Total earliness minimization.	DQN	The JSP is solved using a NN-based QL method.
Li et al. (2022b)	DFJSP	Makespan and total energy consumption minimization.	DQN	DFJSP is solved by combining three extensions to DQN to create a new algorithm.
Wang et al. (2022a)	DFJSP	Makespan and total earliness minimization, machine average utilization maximization.	DDQN	A real-time dynamic scheduling algorithm based on DDQN improvement is proposed to solve multi-objective DFJSP.
Luo (2020)	DFJSP	Total earliness minimization.	DDQN	A method based on DDQN and soft target weight update is proposed to solve DFJSP.
Xu et al. (2023)	DFJSP	Makespan minimization.	DDQN	A DDQN-based method is proposed to solve DFJSP.
Chang et al. (2022b)	DFJSP	Total earliness and tardiness minimization.	DDQN	A DDQN-based method for solving DFJSPs with random arrival of jobs is proposed.
Liu et al. (2022b)	JSP	Total waiting time and makespan minimization.	D3QN	A D3QN combined with a graph neural network (GNN) end-to-end framework is proposed to solve JSP.
Du et al. (2024)	FJSP	Makespan and total energy consumption minimization.	DDQN	A DDQN-based method for solving DFJSPs with transportation and installation time is proposed.
Bai and Lv (2022)	JSP	Makespan minimization.	DQN、QL	Remanufactured JSPs are addressed using QL and DQN, respectively.
Zhu et al. (2022b)	FJSP	Makespan minimization.	LinUCB	A Linear Upper Confidence Bounds (LinUCB) method is proposed to solve FJSP.
Yang et al. (2023)	DJSP	Balance machine load.	DDQN	A DDQN combined with the GNN method is proposed to solve DJSP.

rules and can be more efficiently used to solve many-dimensional problems. A policy-based RL method to train the policy network is proposed. A DRL framework based on the PG algorithm for complex FJSPs was proposed by Han and Yang (2021), which has strong generalizability.

Among the policy-based RL methods, PPO is increasingly used by scholars to solve JSPs because of its superior overall performance. The graph neural network (GNN) is usually introduced by the PPO approach for better extraction of JSP features, and NNs are used as policy networks that can effectively handle parameterized inputs (Liu and Huang, 2023). The curse of dimensionality can be better avoided by the policy-based RL method, and continuous state and action space problems are more suitable, as noted by Wang et al. (2021a). Furthermore, the PPO algorithm converges well by limiting the sensitivity of the parameters. A PPO-based DRL method was proposed by Zhang, Lu et al. (2022) to solve dynamic JSPs containing machine faults, and it was shown that a more stable learning process is achieved by the PPO algorithm than by the PG algorithm, and a faster convergence speed is achieved than that of trust region policy optimization (TRPO).

4.5. Combined value-based and policy-based RL methods

As shown in Table 9, the value-based RL method and the policy-based RL method have strengths in solving JSPs, while the advantages of both are possessed by the AC-framed RL method. When the problem size is small, a better value of state-action can be obtained by the Critic, which guides the Actor to generate a better policy than the policy-based RL method does, whereas when the problem size is large, although the Critic cannot obtain a better value of state-action, the policy can still be updated directly by the Actor, avoiding the curse of dimensionality that exists in the value-based RL method. In AC algorithms, two NNs usually represent the Actor and Critic; however, in the AC-based scheduling framework proposed by Elsayed et al. (2022) for dynamic FJSPs, a graph isomorphism network (GIN) is shared by the Actor and Critic, and the proposed model can be adapted for real-time scheduling with some generalizability. Many algorithms with superior performance, such as DDPG, are derived using AC as a reference framework. Liu et al. (2020) proposed solving dynamic FJSPs with the DDPG algorithm, which is shown to exhibit good performance in solving problems of the same size as the cases used for training. DDPG is used by Gui et al. (2023) to train a policy network for selecting appropriate weights to aggregate multiple single scheduling rules into better rules.

5. Discussion

This section will summarize and discuss the four problems raised in the preamble. The first three problems are trending issues in RL methods for solving JSPs and are discussed separately in this section. Finally, the fourth question is answered in the future trends section.

Fig. 5 illustrates the solution for the first three issues. As shown in the figure, methods for solving the curse of dimensionality include: (i) function approximation, (ii) problem splitting, (iii) reducing complexity, and (iv) other policy update approaches. Among these methods, all of them start from the perspective of RL element design and issue simplification, except for the use of policy-based RL methods which involve changing the intelligence training object. Methods to improve generalizability include: (i) feature representation, (ii) network sharing, (iii) the GNN, (iv) the pointer network, (v) curriculum learning, and (vi) the end-to-end model. Among these methods, feature representation aims to improve the generalizability by designing the intelligence state representation with curriculum learning as a special training strategy. In addition, the vast majority of means to improve generalization are related to NN techniques. Methods to reduce training time include: (i) interaction between agents, (ii) an improved algorithm framework, (iii) an exploration strategy, and (iv) the effectiveness of policy updates. Among these methods, with the exception of the improved algorithm framework, which improves the structure of the RL method, the remaining methods improve the training process of the agent. These specific methods are described in the following section.

5.1. Curse of dimensionality

The curse of dimensionality is addressed by implementing value-based RL methods to solve large-scale or dynamic JSPs. This problem is addressed through the combination of different technological approaches in a value-based RL framework, which is summarized follows.

1) Function approximation: Through function approximation, updating the Q-table is transformed into a function fitting problem, where similar states can receive similar output actions, significantly reducing the number of state-action values that need to be stored in the Q-table. The NN is most commonly used as a function approximator to approximate the value function to realize the solution of high-dimensional numbers of JSPs.

Table 8
Policy-based RL methods.

Reference	Problem	Objective	Algorithm	Description
Saqlain et al. (2023)	FJSP	Makespan minimization.	MCTS+ FIFO/ SJF/LJF	The method of Monte Carlo tree search (MCTS) combining three baseline scheduling techniques is proposed to solve FJSP.
Park et al. (2021)	JSP	Makespan minimization.	PPO + GNN	The problem features are proposed to be extracted with GNN, and the policy network is trained with PPO.
Zhang et al. (2020)	JSP	Makespan minimization.	PPO + GNN	A size-independent policy network is designed using GNN, and the policy network is trained using PPO.
Cunha et al. (2021)	JSP	Makespan minimization.	PPO + NN	A DRL method for training agents using PPO is proposed.
Han and Yan (2021)	FJSP	Makespan minimization.	RNN + PG	The decoder network is modeled with a recurrent neural networks (RNN), and its parameters are optimized using PG.
Wang et al. (2021a)	JSP	Makespan minimization.	PPO + NN	The DRL method for training agents using PPO is proposed.
Zhu et al. (2022a)	JSP	System idle time minimization.	NN + RL	A state-action value-based learning NN is designed, and a policy-based DRL algorithm is proposed to train the network.
Zhang et al. (2022)	JSP	Machine average utilization maximization.	PPO + DNN	The scheduling policy is parameterized with a deep neural network (DNN), and the policy network is trained by PPO.
Hammami et al. (2022)	JSP	Makespan and system idle time minimization.	PPO + GNN	The states and constraints of JSPs are proposed to be represented with GNN-extracted graphs, and the policy network is trained with PPO.
Wang et al. (2022b)	FJSP	Makespan and carbon emissions minimization.	PPO + DNN	The scheduling policy is parameterized with a DNN, and the policy network is trained by PPO.
Vivekanandan et al. (2023)	JSP	Makespan minimization.	PPO + NN	A DRL method using PPO to train agents is proposed, and an order swapping mechanism is introduced in the environment.
Wu and Yan (2023)	DJSP	Makespan minimization.	SPP-Net +PPO	A DRL model based on spatial pyramid pooling networks (SPP-Net) is proposed to train the policy network by PPO.
Zhao et al. (2023)	FJSP	Total tardiness minimization.	PPO + NN	A DRL method is proposed in which the policy network is trained using PPO.
Gan et al. (2023)	JSP	Makespan minimization.	PPO + GNN	A DT-based adaptive scheduling system framework is proposed to achieve the optimization of policies with improved PPO.
Liu and Huang, (2023)	JSP	Makespan minimization.	PPO + GNN	The policy of the JSP is proposed to be parameterized with GNN, and the policy network is trained with PPO.
Song et al. (2023)	FJSP	Makespan minimization.	PPO + GNN	A heterogeneous graph of FJSPs is proposed, along with a GNN specifically adapted to this heterogeneous graph, and the policy network is trained with PPO.
Hameed and Schwung (2023)	JSP	Makespan minimization.	GNN + RL	A framework for GraSP-RL based on RL and GNN is proposed, and curriculum learning is introduced in the framework.
Gebreyesus et al. (2023)	DJSP	Makespan minimization.	GAM + PPO	The gated-attention model (GAM) is proposed, and the policy network is trained with PPO.
Yuan et al. (2023)	JSP	Makespan minimization.	GIN + PPO	The DRL method using PPO to train the policy network is proposed, and the invalid action masking (IAM) technique is introduced.

2) Problem splitting: The large-scale problem is broken down into smaller problems by being split. Splitting JSPs needs to incorporate job shop features. Therefore, in practice, the problems associated with splitting are essentially solved via MARL. The center agent in many MARLs is set to obtain global information and may likewise suffer from the curse of dimensionality. This problem can be solved by setting up only collaborating agents with local perspectives and providing sparse rewards or by using distributed RL to achieve problem splitting.

Table 9
Combined value-based and policy-based RL methods.

Reference	Problem	Objective	Algorithm	Description
Elsayed et al. (2022)	FJSP	Makespan minimization.	GNN +AC	The problem features are extracted with GNN, and the policy network is trained with AC.
Ren et al. (2020)	JSP	Makespan minimization.	LSTM +A3C	Two long short-term memory (LSTM) encoding and decoding networks with the same structure are built; a pointer network is introduced to prioritize jobs, and the parameters of the pointer network are optimized using asynchronous advantage actor-critic (A3C).
Liu et al. (2020)	JSP	Makespan minimization.	CNN+ DDPG	To solve DJSP, the utilization of DDPG is proposed.
Huang et al. (2023)	JSP	Balance machine load.	GIN +AC	The problem features are extracted with GNN, and the policy network is trained with AC.
Chen et al. (2022)	JSP	Makespan minimization.	DNN+ DDPG	The scheduling policy is proposed to be parameterized with DNN, and the policy network is trained using DDPG.
Gui et al. (2023)	DFJSP	Makespan minimization.	DQN	DT technology is introduced.

3) Reducing complexity: Reducing complexity refers to modeling RL in a way that simplifies the representation of states in scheduling methods for complex problems. For example, sequence clustering can be used to reduce the state dimension of the JSP scheduling system.

4) Other policy update approaches: The policy-based RL methods are more suitable for continuous state and action space problems and can avoid the curse of dimensionality in the state–action space. Therefore, in this part, policy-based RL methods are used to solve the curse of dimensionality. In particular, superior performance in solving JSPs is achieved by the PPO algorithm because the sensitivity of the parameters is limited. In addition, the DDPG algorithm, which is framed by the AC algorithm, also generates policies directly through Actors, thus avoiding the curse of dimensionality.

The essential reason for the curse of the dimensionality catastrophe problem is that when the state in RL is designed, the dimension of the state features is too large, which makes the RL method difficult to train or unable to be trained normally. We believe that the function approximation approach is facilitated by the current popularity of NN technology, so the curse of the dimensionality catastrophe problem is most commonly solved by using the function approximation method; the problem splitting approach is direct and effective, and is widely used in MARL; the complexity reduction approach provides ideas for further solving the curse of dimensionality catastrophe, but the specific implementation method needs to be further explored; and the possibility of effectively solving JSPs that are more complex than the original method is offered by policy-based methods.

5.2. *Generalization*

In a workshop with complex and changing scenarios and a wide variety of products, the generalizability of the algorithms is improved, enabling the scheduling system to be trained with a certain model and then quickly generalized to untrained scheduling scenarios for application. Different methods have been proposed for improving generalizability.

1) Feature representation: The generalizability of the RL method for solving JSPs can be improved by a suitable representation of the states and actions of the workshop. The state characteristics of RL in JSP are usually defined as metrics such as the number of machines,

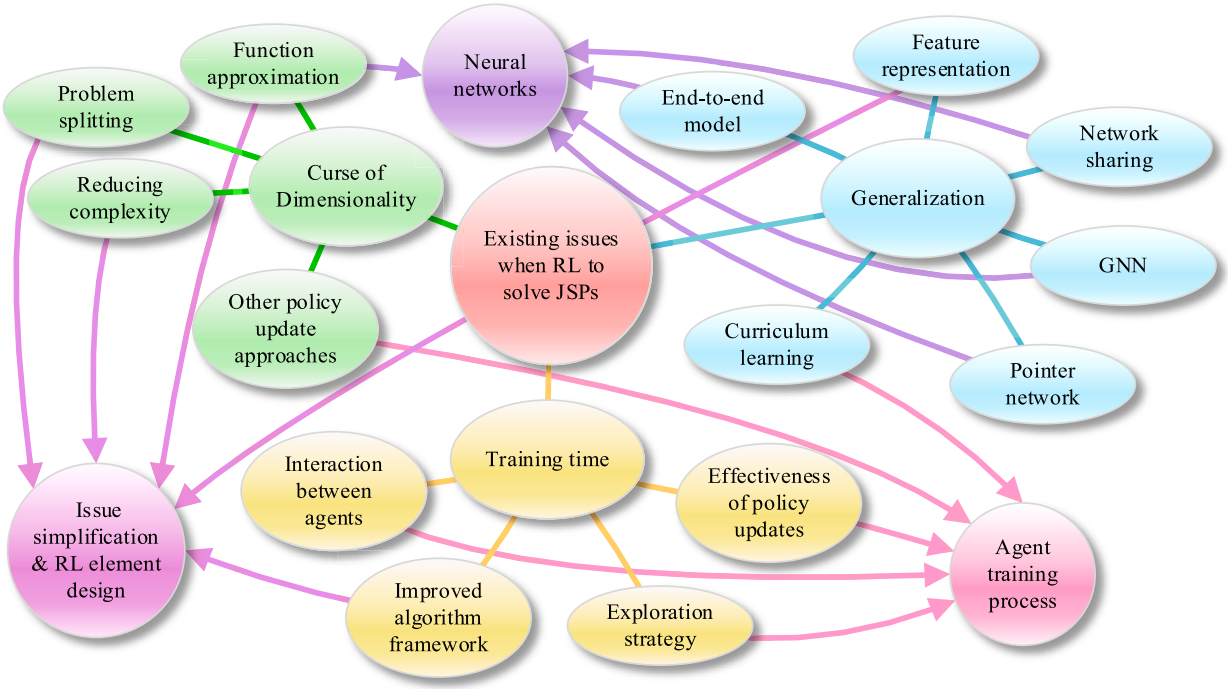


Fig. 5. Problems and solutions.

jobs, operations, etc. The number of these metrics is typically uncertain in different production environments. If these metrics are set directly to the state, the inputs to the system could change dramatically from one JSP to the next. The normalized design of some features as inputs allows changes in these metrics to be adapted to the proposed scheduling method. In addition, when instances are of the same size, the generalizability of scheduling methods is improved by introducing the OSM in the environment and constructing a space by aggregating multiple PDRs, which allows policies to be generalizable in terms of scale size.

2) Network sharing: The sharing of NNs between agents in MARL is trained in a way that adapts to changes in the number of machines. In this approach, each agent acts in a decentralized manner, and even if the number of machines changes, the Q-network can be adapted to solve the new scheduling problem without retraining.
3) GNN: The RL method combined with the GNN has strong generalizability because of the powerful feature extraction capability of the GNN. When a JSP is solved by the GNN, the JSP needs to be represented as a disjunctive graph, and then the disjunctive graph of the JSP is converted into node embeddings via the GNN. Since GNNs can handle graphs of different sizes and policy networks with node embeddings as inputs are also size-independent, trained policies can be used for JSPs of different sizes. The GIN is a specific GNN model. Using the GIN for feature extraction in shop scheduling problems can efficiently aggregate node features and other neighboring features, allowing the scheduling method to be effectively generalized to different scales without retraining or knowledge transfer.
4) Pointer network: A pointer layer can be set up in the scheduling framework to output solutions with variable lengths, which can be applied to decision-making for problems of different sizes, improving the generalizability of the scheduling system.
5) Curriculum learning: Curriculum learning is a training policy in which the agent is first trained with the simplest task, and the difficulty of the task increases as the training progresses. The gradual increase in task difficulty allows unknown events to be handled by the trained agent (Hameed and Schwung, 2023).
6) End-to-end model: Features can be automatically extracted without human intervention, with strong generalizability in the end-to-end model. In recent years, a new method for solving combinatorial optimization problems end-to-end has been produced by combining sequence-to-sequence models with DRL(Bonetta et al., 2023). A given problem instance is taken as input, and the method directly outputs a solution via trained DNNs. In particular, in the end-to-end model based on a matrix encoding network, matrix-based data are directly used as inputs, which can solve JSPs more effectively (Kwon et al., 2021). This searches in the solution space rather than in the rule space and has the advantages of high solution quality and strong generalization ability.

In summary, we suggest that by expressing state characteristics in general terms rather than in response to specific problems, more generalized state features can be obtained. Trained agents are naturally able to generalize. The realization of these methods and the provision of some convenience for the research are facilitated by many of the existing tools, such as the methods of GNN, pointer networks, and shared networks, which are based on the premise that most of the current RL methods are built on NNs.

5.3. Training time

The training time of the agents in MARL is affected by the interaction mechanism, and the convergence performance of the algorithm, without considering the number of agents, is closely related to the training time of the agents. Various methods for reducing the training time of an agent are described as follows.

1) Interaction between agents: For general MARL, the framework of hierarchical RL can be introduced, and a clear division of labor can reduce the interactions between the agents and the global agent's workload and speed up training. In addition, distributed RL can be introduced to adopt the model of centralized training and decentralized execution of tasks, and the sharing of experience in the centralized training phase can improve the training efficiency of the agent.
2) Improved algorithm framework: The algorithmic framework in value-based RL methods can be improved to increase training efficiency. For example, the training speed can be effectively improved in the DQN by introducing experience replay to store learned data, reducing the overestimation of the current optimum, and increasing the training speed through the use of double-layer Q-Learning (DLQL).
3) Exploration strategy: The agent training time can be reduced by the introduction of better exploration strategies in value-based RL methods. On the one hand, the efficiency of exploration can be improved by introducing other methods, such as the introduction of EDA in RL to discover more solution space or the introduction of HGA to increase the efficiency and effectiveness of exploration during the training process. On the other hand, different exploration strategies can be introduced, such as introducing "softmax" as the action selection strategy so that the algorithm tends to be explored in the early stage and tends to have better action selected in the later stage.
4) Effectiveness of policy updates: For policy-based RL methods, improving the effectiveness of policy updates can reduce the training time of the agent. For example, the step size of each update is restricted by the PPO algorithm, and the stability of exploration is dramatically improved, making the whole convergence process more stable and faster. Introducing an attention mechanism in the policy network allows fast access to effective information and thus an effective response to large-scale problems. In addition, the IAM technique can be introduced to reduce the exploration space during agent training and improve exploration efficiency. However, there is a contradiction between this method and generalization, which may lead to overfitting.

In summary, in terms of reducing training time for agents, the training time is considered based on being able to solve the curse of the dimensionality catastrophe problem, and the training time itself is determined by the training results. When the agent can meet the criteria for use after training, the training has been completed. The training time is increased by interactions between MARL agents, so designing better interaction mechanisms or using distributed RL are considered good solutions. To reduce the training time, the most commonly used method for value-based RL methods today is the exploration strategy of the agents; more attention should be given to the effectiveness of updating the strategy for policy-based RL methods. In addition, the excellent convergence performance of the PPO algorithm makes it the current mainstream algorithm.

5.4. Future trends

Question 4 is answered in this section, the shortcomings of the current research are discussed, and future research directions for solving the shop scheduling problem via RL are explored.

The curse of the dimensionality catastrophe problem is addressed by the mainstream adoption of NNs, but relying entirely on NNs to solve complex JSPs is associate with by high training costs. Therefore, when value-based RL methods are used to solve JSPs, more consideration can be given in the future to designing suitable representations of state and action to discretize the state and actions of complex problems and ensure that the Q-tables can converge in training. On the other hand, directly solving JSPs that employ analytic graph representations is a completely

new way of thinking, and the combination of GNNs and policy-based RL methods is currently a popular JSP solution. In this context, we suggest further research in the future on the potential of strategy-based RL methods and RL methods framed in terms of AC.

The powerful feature extraction capabilities of NNs rely on current scheduling methods to improve generalization, which is essentially an advantage of the development of NN technology. In this context, in the future, more attention should be given to the design of the features of RL to improve the generalizability of the model to improve the generalizability performance of the RL method itself in solving the shop scheduling problem. MARL is more scalable than single agent RL, but MARL itself is a very complex field, and future research on its generalizability could attempt more MARLs. In addition, methods that enhance generalizability, such as curriculum learning techniques, should be further explored.

The performance of these methods, but not the training time, currently discussed by many scholars. Only some of the articles specifically optimize training time for agents, but training time is closely related to economics and deserves further consideration. For the training time, further optimization can be achieved on agent exploration by introducing other methods or strategies to improve exploration efficiency and reduce the training time of the agent. For example, the application of attention mechanisms is currently in its infancy, and more in-depth research can be done in the future. There is sometimes a contradiction between reducing the training time of an agent and preventing it from falling into a local optimum. Therefore, the training time of the agent in practical applications needs to be reduced in conjunction with the solved problem while ensuring the optimization result.

In addition to the challenges raised regarding these three issues, many other scheduling methods deserve to be considered further.

JSPs containing regular constraints, a few dynamic perturbations, and a single objective optimization are addressed by most of the currently proposed RL methods. Further research should be conducted to enable the RL method to solve JSPs containing specific constraints, more dynamic perturbations, and multiple objective optimizations in the proposed scheduling method.

Real-time scheduling methods are required in some environments, and existing real-time RL scheduling methods are capable of handling real-time scheduling when the dynamic perturbations are known. However, for unknown perturbations, rescheduling may not result in a satisfactory scheduling solution. When unknown dynamic perturbations occur, it is clear that retraining agents to generate new solutions is not a reasonable approach. Therefore, the real-time scheduling capability of the RL method in the face of unknown perturbations should be improved in the future, and the framework of an adaptive scheduling system based on DTs shows promise as a direction to pursue.

Large-scale or complex workshops are more compatible with MARL, whereas many existing RL methods are designed for single-agent scenarios. Therefore, more attempts should be made to choose MARL in the future, especially to explore further the application of distributed RL combined with the experience sharing scheme in JSPs; combining the distributed RL method with the framework of hierarchical RL is indeed a good idea. In addition, owing to the complexity of JSP, the training time of multiple agents tends to be long. Therefore, improving the interaction mechanism between multiple agents to increase the training efficiency of MARL also deserves further investigation.

6. Conclusion

After analyzing the literature on RL solutions to shop scheduling problems in recent years, we reviewed the articles on RL algorithms for solving JSPs published after 2016. From the perspective of researching development frontiers, we propose the curse of dimensionality, generalizability, and training time problems, review the solutions to these problems. Some suggestions on the answers to these problems are given, and the trends of future research on RL algorithms for solving JSPs are proposed. The following conclusions can be drawn.

The three main problems proposed in this paper are closely related and, in some cases, contradictory. The essential reason for the curse of the dimensionality catastrophe problem is that when the state in RL is designed, the dimension of the state features is too large. More generalized state features can be obtained by expressing state characteristics in general terms, rather than in response to specific problems. The training time is considered based on being able to solve the curse of the dimensionality catastrophe problem, and the training time itself is determined by the training results. The reduction in training time is often positively correlated with the complexity of the state and action.

The design of the various elements of the agent, such as state design, action design, and reward design, has different impacts on the three main problems. One of the most important factors is the design of the state in RL. If the design of the state features is simple, the training time of the agent is short, the agent easily converges, and there is no dimensionality disaster. However, in this case, the features obtained by the agent may not be sufficient, resulting in the state obtained even after the agent is trained, which is not enough to obtain enough information to solve JSPs with high quality. If the state features are designed in greater detail, the dimension of the state will be too large, the training time will increase significantly, resulting in an inability to converge. Therefore, when designing the state in RL, it is necessary to select the appropriate dimensions expressed by state features. This will be an important research direction in the future, but it is also a difficult point.

Action design is expressed generally in the selection of scheduling rules or the process assignment of workpieces. When an action is a scheduling rule, the overall number of actions is usually small. At this time, if the state is properly designed, the training process more easily converges, and there is usually no dimensionality catastrophe or training time that is too long. This type of action usually has a certain degree of generality at different scales and types of JSPs. However, the performance of RL is strongly affected by the performance of scheduling rules, and the effectiveness of RL is more difficult to guarantee. The action design of this type of method is usually related to the specific problem itself, and the generalizability is poor. Owing to the complexity of JSPs, in the future, when RL is applied to solve JSPs, we should consider designing more general actions to make the algorithm more generalizable. Moreover, better scheduling rules should be selected or designed to ensure the effectiveness of the action.

When an agent is trained, the reward can guide the direction of the agent's movement. The setting of the reward usually needs to be consistent with the direction of the optimization objective of the JSPs so that the training of the agent is carried out in a direction that is favorable to the optimization objective. Rewards, as feedback for agents to perform actions, need to change constantly depending on the environment in many cases. Therefore, how to design more effective rewards when solving JSP is a problem that needs further research.

At present, the policy-based RL method has great potential for generating scheduling schemes, and it is possible to achieve a more "intelligent" scheduling approach. However, from the perspective of the advantages and disadvantages of generating scheduling schemes, the value-based RL method is more convenient. When a value-based RL method is directly applied to solve JSPs, designing and expressing state features directly or converging to an ideal Q-table is difficult. However, the *meta*-heuristic algorithm can explore a large solution space, which undoubtedly solves this problem well. By combining with *meta*-heuristic algorithms, agents can better design generalized features, and solve the curse of dimensionality catastrophe problem. In this case, the agent training time and the training sample required can usually be shorter. This is a seemingly simple and efficient way to apply RL to JSPs. We suggest that the combination of *meta*-heuristic algorithms and RL methods is also a direction worth exploring in future research.

CRediT authorship contribution statement

Xiehui Zhang: Writing – original draft, Visualization, Software, Investigation, Formal analysis, Data curation. **Guang-Yu Zhu:** Writing – review & editing, Supervision, Project administration, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Aydin, M.E., Öztemel, E., 2000. Dynamic job-shop scheduling using reinforcement learning agents. Robot. Auton. Syst. 33 (2), 169–178.

Baer, S., et al., 2019. Multi-agent reinforcement learning for job shop scheduling in flexible manufacturing systems, in Proc. Int. Conf. Artif. Intell. Ind. (AI4I), 22-25.

Bai, Y., Lv, Y.Q., 2022. Reinforcement learning-based job shop scheduling for remanufacturing production. IEEE Int. Conf. Ind. Eng. Eng. Manage. 246–251.

Bonetta, G., Zago, D., Cancelliere, R., et al., 2023. Job Shop Scheduling via Deep Reinforcement Learning: A Sequence to Sequence Approach. International Conference on Learning and Intelligent Optimization. Springer International Publishing, Cham, pp. 475–490.

Bouazza, W., Sallez, Y., Beldjilali, B., 2017. A distributed approach solving partially flexible job-shop scheduling problem with a Q-learning effect. IFAC-PapersOnLine 50 (1), 15890–15895.

Chang, X., et al., 2023. Digital twin and deep reinforcement learning enabled real-time scheduling for complex product flexible shop-floor. Proc. Inst. Mech. Eng. Part b: J. Eng. Manuf. 237 (8), 1254–1268.

Chang, J.R., Yu, D., Zhou, Z., et al., 2022a. Hierarchical Reinforcement Learning for Multi-Objective Real-Time Flexible Scheduling in a Smart Shop Floor. Machines 10 (12), 1195.

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

Chaudhry, I.A., Khan, A.A., 2016. A research survey: review of flexible job shop scheduling techniques. Int. Trans. Oper. Res. 23 (3), 551–591.

Chen, R.H., et al., 2020. A self-learning genetic algorithm based on reinforcement learning for flexible job-shop scheduling problem. Comput. Ind. Eng. 149, 106778.

Chen, R.Q., Li, W.X., Yang, H.B., 2022. A deep reinforcement learning framework based on an attention mechanism and disjunctive graph embedding for the job-shop scheduling problem. IEEE Trans. Ind. Inf 19 (2), 1322–1331.

Chien, C.F., Lan, Y.B., 2021. Agent-based approach integrating deep reinforcement learning and hybrid genetic algorithm for dynamic scheduling for Industry 3.5 smart production. Comput. Ind. Eng. 162, 107782.

Cunha, B., et al., 2021. Intelligent scheduling with reinforcement learning. Appl. Sci. 11 (8), 3710.

de Witt, C.A.S., Foerster, J.N., et al., 2019. Multi-agent common knowledge reinforcement learning. Adv. Neural Inf. Process. Syst. 32, 9924–9935.

Drakaki, M., Tzionas, P., 2017. Manufacturing scheduling using colored Petri nets and reinforcement learning. Appl. Sci. 7 (2), 136.

Du, Y., Li, J.Q., Chen, X.L., et al., 2023. Knowledge-based reinforcement learning and estimation of distribution algorithm for flexible job shop scheduling problem. IEEE Trans. Emerging Topics Comp. Intell. 7 (4), 1036–1050.

Du, Y., Li, J.Q., Li, C.D., et al., 2024. A reinforcement learning approach for flexible job shop scheduling problem with crane transportation and setup times. IEEE Trans. Neural Networks Learn. Sys. 35 (4), 5695–5709.

Elsayed, E.K., Elsayed, A.K., Eldahshan, K.A., 2022. Deep Reinforcement Learning-Based Job Shop Scheduling of Smart Manufacturing. Comput. Mater. Continua 73 (3), 5103–5120.

Gan, X.M., et al., 2023. Digital twin-enabled adaptive scheduling strategy based on deep reinforcement learning. Sci. China Technol. Sci. 66 (7), 1937–1951.

Gebreyesus, G., et al., 2023. Gated-Attention Model with Reinforcement Learning for Solving Dynamic Job Shop Scheduling Problem. IEEJ Trans. Electr. Electron. Eng. 18 (6), 932–944.

Gui, Y., et al., 2023. Dynamic scheduling for flexible job shop using a deep reinforcement learning approach. Comput. Ind. Eng. 180, 109255.

Hameed, M.S.A., Schwung, A., 2023. Graph neural networks-based scheduler for production planning problems using reinforcement learning. J. Manuf. Syst. 69, 91–102.

Hammami, N.E.H., et al., 2022. Job Shop Scheduling: A Novel DRL approach for continuous schedule-generation facing real-time job arrivals. IFAC-PapersOnLine 55 (10), 2493–2498.

Han, B., Yang, J., 2021. A deep reinforcement learning based solution for flexible job shop scheduling problem. Int. J. Simul. Model. 20 (2), 375–386.

Hottung, A., Kwon Y. D., Tierney K. 2021. Efficient active search for combinatorial optimization problems. arXiv preprint arXiv:2106.05126.

Huang, J.P., et al., 2023. A novel priority dispatch rule generation method based on graph neural network and reinforcement learning for distributed job-shop scheduling. J. Manuf. Syst. 69, 119–134.

İnal, A.F., et al., 2023. A Multi-Agent Reinforcement Learning Approach to the Dynamic Job Shop Scheduling Problem. Sustainability 15 (10), 8262.

Johnson, D., Chen, G., Lu, Y., 2022. Multi-agent reinforcement learning for real-time dynamic production scheduling in a robot assembly cell. IEEE Robot. Autom. 7 (3), 7684–7691.

Jungbluth, S., et al., 2022. Reinforcement Learning-based Scheduling of a Job-Shop Process with Distributedly Controlled Robotic Manipulators for Transport Operations. IFAC-PapersOnLine 55 (2), 156–162.

Kardos, C., et al., 2020. Dynamic scheduling in a job-shop production system with reinforcement learning. Procedia CIRP 97, 104–109.

Kayhan, B.M., Yildiz, G., 2023. Reinforcement learning applications to machine scheduling problems: a comprehensive literature review. J. Intell. Manuf. 34 (3), 905–929.

Kwon, Y.D., Choo, J., Yoon, I., et al., 2021. Matrix encoding networks for neural combinatorial optimization. Advances in Neural Information Processing Systems 34, 5138–5149.

Lang, S., et al., 2020. Integration of deep reinforcement learning and discrete-event simulation for real-time scheduling of a flexible job shop production. Proc. Winter Simul. Conf. (WSC) 3057–3068.

Lei, K., et al., 2022. An End-to-end Hierarchical Reinforcement Learning Framework for Large-scale Dynamic Flexible Job-shop Scheduling Problem. Proc. Int. Jt. Conf. Neural Networks (IJCNN) 1–8.

Li, Y.X., et al., 2022b. Real-time data-driven dynamic scheduling for flexible job shop with insufficient transportation resources using hybrid deep Q network. Rob. Comput. Integr. Manuf. 74, 102283.

Li, Y.B., et al., 2023b. A reinforcement learning-artificial bee colony algorithm for flexible job-shop scheduling problem with lot streaming. Appl. Soft Comput. 146, 110658.

Li, R., Gong, W.Y., Lu, C., 2022a. A reinforcement learning based RMOEA/D for bi-objective fuzzy flexible job shop scheduling. Expert Sys. Appl. 203, 117380.

Li, R., Gong, W.Y., Lu, C., Wang, L., 2023a. A learning-based memetic algorithm for energy-efficient flexible job shop scheduling with type-2 fuzzy processing time. IEEE Trans. Evol. Comput. 27 (3), 610–620.

Lihu, A., Holban, S., 2009. Top five most promising algorithms in scheduling. In Proceedings – 2009 5th international symposium on applied computational intelligence and informatics. SACI 2009, 397–404.

Lillicrap, T.P., et al., 2015. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971.

Lin, C.C., et al., 2019. Smart manufacturing scheduling with edge computing using multiclass deep Q network. IEEE Trans. Ind. Inf. 15 (7), 4276–4284.

Lin, C.R., Cao, Z.C., Zhou, M.C., 2022. Learning-based grey wolf optimizer for stochastic flexible job shop scheduling. IEEE Trans. Autom. Sci. Eng. 19 (4), 3659–3671.

Liu, C.L., Chang, C.C., Tseng, C.J., 2020. Actor-critic deep reinforcement learning for solving job shop scheduling problems. IEEE Access 8, 71752–71762.

Liu, C.L., Huang, T.H., 2023. Dynamic Job-Shop Scheduling Problems Using Graph Neural Network and Deep Reinforcement Learning. IEEE Trans. Syst. Man Cybern. Syst. 53 (11), 6836–6848.

Liu, X.Y., Liu, L., Jiang, T.H., 2023b. A self-learning interior search algorithm based on reinforcement learning for energy-aware job shop scheduling problem with outsourcing option. J. Intelligent Fuzzy Syst. 44 (6), 10085–10100.

Liu, R.K., Piplani, R., Toro, C., 2022a. Deep reinforcement learning for dynamic scheduling of a flexible job shop. Int. J. Prod. Res. 60 (13), 4049–4069.

Liu, J.J., Sun, B.F., Li, G.D., et al., 2023a. An integrated scheduling approach considering dispatching strategy and conflict-free route of AMRs in flexible job shop. Int. J. Adv. Manuf. Technol. 127 (3–4), 1979–2002.

Liu, Z.Y., Wang, Y., Liang, X.X., et al., 2022b. A graph neural networks-based deep Q-learning approach for job shop scheduling problems in traffic management. Inf. Sci. 607, 1211–1223.

Long, X.J., et al., 2022. A self-learning artificial bee colony algorithm based on reinforcement learning for a flexible job-shop scheduling problem. Concurr. Comput. Pract. Exper. 34 (4), e6658.

Luo, S., 2020. Dynamic scheduling for flexible job shop with new job insertions by deep reinforcement learning. Appl. Soft Comput. 91, 106208.

Luo, S., Zhang, L.X., Fan, Y.S., 2022a. Real-time scheduling for dynamic partial-no-wait multiobjective flexible job shop by deep reinforcement learning. IEEE Trans. Autom. Sci. Eng. 19 (4), 3020–3038.

Luo, S., Zhang, L.X., Fan, Y.S., 2022b. Dynamic multi-objective scheduling for flexible job shop by deep reinforcement learning. Comput. Ind. Eng. 159, 107489.

Mazyavkina, N., Sviridov, S., Ivanov, S., Burnaev, E., 2021. Reinforcement learning for combinatorial optimization: A survey. Comput. Oper. Res. 134, 105400.

Méndez-Hernández, B.M., et al., 2019. A multi-objective reinforcement learning algorithm for JSSP. Lect. Notes Comput. Sci. 11727, 567–584.

Mnih, V., et al., 2015. Human-level control through deep reinforcement learning. Nature 518 (7540), 529–533.

Oh, S.H., Cho, Y.I., Woo, J.H., 2022. Distributional reinforcement learning with the independent learners for flexible job shop scheduling problem with high variability. J. Comput. Des. Eng. 9 (4), 1157–1174.

Park, I.B., et al., 2020. A reinforcement learning approach to robust scheduling of semiconductor manufacturing facilities. IEEE Trans. Autom. Sci. Eng. 17 (3), 1420–1431.

Park, J., et al., 2021. Learning to schedule job-shop problems: representation and policy learning using graph neural network and reinforcement learning. Int. J. Prod. Res. 59 (11), 3360–3377.

Pol S. et al. 2021. Global Reward Design for Cooperative Agents to Achieve Flexible Production Control under Real-time Constraints, in International Conference on Enterprise Information Systems (ICEIS), 1,515-526.

Popper, J., Yfantis, V., Ruskowski, M., 2021. Simultaneous production and AGV scheduling using multi-agent deep reinforcement learning. Procedia CIRP 104, 1523–1528.

Qin, Z.J., Johnson, D., Lu, Y.Q., 2023. Dynamic production scheduling towards self-organizing mass personalization: A multi-agent dueling deep reinforcement learning approach. J. Manuf. Syst. 68, 242–257.

Qu, S.H., Wang, J., Shivani, G., 2016. Learning adaptive dispatching rules for a manufacturing process system by using reinforcement learning approach, in IEEE Int. Conf. Emerging Technol. Factory Autom. (ETFA) 1–8.

Ren, J.F., Ye, C.M., Yang, F., 2020. A novel solution to JSps based on long short-term memory and policy gradient algorithm. Int. J. Simul. Model. 19 (1), 157–168.

Saqlain, M., Ali, S., Lee, J.Y., 2023. A Monte-Carlo tree search algorithm for the flexible job-shop scheduling in manufacturing systems. Flexible Serv. Manuf. J. 35 (2), 548–571.

Schulman J., et al. 2017. Proximal policy optimization algorithms, arXiv preprint arXiv: 1707.06347.

Seito, T., Munakata, S., 2020. Production Scheduling based on Deep Reinforcement Learning using Graph Convolutional Neural Network, in ICAART - Proc. Int. Conf. Agents Artif. Intell. 2, 766–772.

Shahrabi, J., Adibi, M.A., Mahootchi, M., 2017. A reinforcement learning approach to parameter estimation in dynamic job shop scheduling. Comput. Ind. Eng. 110, 75–82.

Song, W., et al., 2023. Flexible job-shop scheduling via graph neural network and deep reinforcement learning. IEEE Trans. Ind. Inf. 19 (2), 1600–1610.

Sun, Z., Yang, Y., 2023. Difusco: Graph-based diffusion solvers for combinatorial optimization.“. Advances in Neural Information Processing Systems 36, 3706–3731.

Van Hasselt, H., Guez, A., Silver, D., 2016. Deep reinforcement learning with double q-learning, in 30th AAAI Conf. Artif. Intell. (AAAI) 2094–2100.

Vivekanandan, D., et al., 2023. A Reinforcement Learning Approach for Scheduling Problems with Improved Generalization through Order Swapping. Machine Learning and Knowledge Extraction 5 (2), 418–430.

Wang, Y.F., 2020. Adaptive job shop scheduling strategy based on weighted Q-learning algorithm. J. Intell. Manuf. 31 (2), 417–432.

Wang, H., Cheng, J.F., Liu, C., et al., 2022a. Multi-objective reinforcement learning framework for dynamic flexible job shop scheduling problem with uncertain events. Appl. Soft Comput. 131, 109717.

Wang S. Y., Li J.X., Luo Y. C. 2021. Smart Scheduling for Flexible and Hybrid Production with Multi-Agent Deep Reinforcement Learning, in Proc. IEEE Int. Conf. Inf. Technol., Big Data Artif. Intell. (ICIBA), 288-294.

Wang, L.B., Hu, X., Wang, Y., et al., 2021b. Dynamic job-shop scheduling in smart manufacturing using deep reinforcement learning. Comput. Netw. 190, 107969.

Wang, S.Y., Li, J.X., Tang, H., et al., 2022b. CEA-FJSP: Carbon emission-aware flexible job-shop scheduling based on deep reinforcement learning. Front. Env. Sci. 10, 1059451.

Wang, Z.Q., Liao, W.Z., 2024. Smart scheduling of dynamic job shop based on discrete event simulation and deep reinforcement learning. J. Intell. Manuf., Jun. 35, 2593–2610.

Wang, H.X., Sarker, B.R., et al., 2021a. Adaptive scheduling for assembly job shop with uncertain assembly times based on dual Q-learning. Int. J. Prod. Res. 59 (19), 5867–5883.

Wang, Y.C., Usher, J.M., 2004. Learning policies for single machine job dispatching. Rob. Comput. Integr. Manuf. 20 (6), 553–562.

Wang, X.H., Zhang, L., et al., 2022c. Solving job scheduling problems in a resource preemption environment with multi-agent reinforcement learning. Rob Comput Integr Manuf 77, 102324.

Waschneck, B., et al., 2018. Deep reinforcement learning for semiconductor production scheduling, in 29th Annu. SEMI Adv. Semicond. Manuf. Conf. (ASMC) 301–306.

Wei, Y., Zhao, M., 2004. Composite rules selection using reinforcement learning for dynamic job-shop scheduling. IEEE Conf. Rob. Autom. Mechatron. 1083–1088.

Wu, Z.F., et al., 2023. Efficient Multi-Objective Optimization on Dynamic Flexible Job Shop Scheduling Using Deep Reinforcement Learning Approach. Processes 11 (7), 2018.

Wu, X.Q., Yan, X.F., 2023. A spatial pyramid pooling-based deep reinforcement learning model for dynamic job-shop scheduling problem. Comput. Oper. Res. 160, 106401.

Xie, J., et al., 2019. Review on flexible job shop scheduling. IET Collab. Intell. Manuf. 1 (3), 67–77.

Xu, Z.Y., Chang, D.F., Luo, T., Gao, Y.P., 2023. Intelligent scheduling of double-deck traversable cranes based on deep reinforcement learning. Eng. Optim. 55 (12), 2034–2050.

Yan, Q., Wang, H.F., Wu, F., 2022. Digital twin-enabled dynamic scheduling with preventive maintenance using a double-layer Q-learning algorithm. Comput. Oper. Res. 144, 105823.

Yang, Z., Bi, L., Jiao, X.G., 2023. Combining Reinforcement Learning Algorithms with Graph Neural Networks to Solve Dynamic Job Shop Scheduling Problems. Processes 11 (5), 1571.

Yuan, E., et al., 2023. Solving job shop scheduling problems via deep reinforcement learning. Appl. Soft Comput. 143, 110436.

Zhang, W., Dietterich, T.G., 1995. A reinforcement learning approach to job-shop scheduling. In: In 1995 International Joint Conference on Artificial Intelligence, pp. 1114–1120.

Zhang, J.D., He, Z.X., et al., 2023. DeepMAG: Deep reinforcement learning with multi-agent graphs for flexible job shop scheduling. Knowl Based Syst 259, 110083.

Zhang, M., Lu, Y., Hu, Y.X., et al., 2022. Dynamic scheduling method for job-shop manufacturing systems by deep reinforcement learning with proximal policy optimization. Sustainability 14 (9), 5177.

Zhang, C., Son, W., Cao, Z.G., et al., 2020. Learning to dispatch for job shop scheduling via deep reinforcement learning,“ in 34th Intern. Confere. Neural Inf. Proces. Syst. 1621–1632.

Zhao, L.L., Fan, J.X., Zhang, C.J., et al., 2023. A DRL-Based Reactive Scheduling Policy for Flexible Job Shops With Random Job Arrivals,“. IEEE Trans. Autom. Sci. Eng. 1–12.

Zhu, H.H., Tao, S., Gui, Y., et al., 2022a. Research on an Adaptive Real-Time Scheduling Method of Dynamic Job-Shop Based on Reinforcement Learning. Machines 10 (11), 1078.

Zhu, X.F., Xu, J.Z., Ge, J.H., et al., 2023. Multi-Task Multi-Agent Reinforcement Learning for Real-Time Scheduling of a Dual-Resource Flexible Job Shop with Robots. Processes 11 (1), 267.

Zhu, H.H., Zhang, Y., Liu, C.C., 2022b. An Adaptive Reinforcement Learning-Based Scheduling Approach with Combination Rules for Mixed-Line Job Shop Production. Math. Probl, Eng, p. 1672166.