Research on the job shop scheduling problem based on digital twin and proximal policy optimization

Lilan Liu
Kai Guo
Zenggui Gao (gaozg@shu.edu.cn)
Shanghai University https://orcid.org/0000-0001-6216-9387
Jiaying Li

Research Article

Keywords: Job shop scheduling, Deep reinforcement learning, Smart manufacturing, Digital twin

Posted Date: February 22nd, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1355780/v1

Abstract

Job shop scheduling plays an important role in intelligent manufacturing. Effectively solving the scheduling problem is the key to realizing intelligent manufacturing. In the actual production process, there are various dynamic events, such as the problem of inserting orders when new orders arrive, which seriously affect the efficiency of scheduling execution. The traditional job shop scheduling algorithm has poor real-time response ability. When encountering emergencies, it needs to run again, and the time complexity is high. A traditional scheduling scheme cannot solve this problem well. To solve the above problems, we use the digital twin (DT) workshop to accurately capture the abnormal events in the production process in real time. Combined with the fast response ability of deep reinforcement learning (DRL), we propose a method to establish a digital twin workshop combined with deep reinforcement learning. In this method, we use a disjunctive graph to represent the state in reinforcement learning, a graph neural network (GNN) as the network structure, and the proximal policy optimization (PPO) algorithm to find the optimal scheduling scheme. We compare the scheduling results with heuristic scheduling rules and a meta heuristic algorithm to verify the feasibility of the algorithm. Finally, in the laboratory environment, aiming at the common order insertion problem in the actual production process, we verify the effectiveness of this job shop scheduling method.

1. Introduction

With the rapid development of computer and information technology, governments are actively promoting the transformation of their manufacturing industry to intelligence, for example, the American industrial internet made in China in 2025 [1] and German industry 4.0 [2]. At the same time, artificial intelligence, the Internet of Things, big data and others have become the key technologies for realizing intelligent manufacturing. Using these new technologies, the production efficiency of the workshop is greatly improved. The job shop scheduling problem is a combinatorial optimization problem that plays an important role in traditional manufacturing. It has been proven to be a nondeterministic-polynomial-time (NP) problem. Solving the job shop scheduling problem is the key to realizing intelligent manufacturing.

In recent decades, research on the job shop scheduling problem has focused on static environments and metaheuristic algorithms including genetic algorithms (GAs) [3] and particle swarm optimization (PSO) [4]. With the continuous updating of various technologies, job shop scheduling needs to meet the following requirements under the background of intelligent manufacturing. (1) Due to the large number of production types, strong continuity and frequent changes in the environment, the failure of a part may affect the overall operation of modern enterprises. Therefore, dealing with dynamic events in workshop scheduling in time for real-time production is an important problem. (2) The actual production process is often accompanied by various uncertain events such as order insertion and equipment failure that affect realization of the production plan. Therefore, uncertain events should be considered in the production process to specify the dynamic scheduling strategy.

To solve the above two challenges, a new job shop scheduling method needs to be proposed. In recent years, with the emergence of digital twins [5] and deep reinforcement learning [6], relying on the real-time decision-making advantages of digital twin real-time mapping and deep reinforcement learning, the new method provides a new idea to solve the job shop scheduling problem in intelligent manufacturing. In this paper, we model the job shop scheduling problem as a Markov decision process (MDP) and solve the dynamic job shop scheduling problem by deep reinforcement learning combined with digital twins. We use the proximal policy optimization (PPO) algorithm to deal with the high-dimensional disaster problem in the state and action space, the disjunctive graph to represent the state of the job shop scheduling problem, and the graph neural network (GNN) to embed the state in the scheduling process. The scheduling results are verified in our physical workshop and digital twin workshop.

The rest of this paper is organized as follows. The second part introduces the related work of digital twin workshops and reinforcement learning for job shop scheduling. The third part introduces the problem description and modelling. The fourth part introduces the algorithm used. The fifth part presents the experimental results. The conclusion is presented in the sixth part.

2. Related Works

2.1. Digital twin workshop

The concept of "digital twins" was first proposed by Professor Grieves at the University of Michigan. Definitions including physical entities, virtual models, and information systems connecting the two were introduced in later articles [7]. In recent years, with the rapid development of information technologies such as communication, computer, artificial intelligence technology and digital twin technology, the concept has attracted increasing attention and has become a hot research direction. Especially in the field of intelligent manufacturing, the application range and depth of digital twin technology are expanding, including from the product design stage to the workshop manufacturing and product service stage.

The digital twin workshop is a new operation mode of the future workshop proposed by Professor Tao Fei [8] in 2017. Tao Fei introduced the concept of the digital twin workshop and discussed its key technologies and implementation methods. A digital twin workshop is the product of a new generation of information technology and manufacturing technology. It realizes the iterative operation of workshop production planning, control and management through the real-time mapping of physical workshops and virtual workshops and is driven by workshop real-time data. Subsequently, Tao [9]–[12] explored the application of digital twinning in different directions and pointed out that digital twinning, as a key technology for the interconnection of everything in the intelligent workshop, is expected to use digital twins to break through the bottleneck of intelligent manufacturing. In 2021, Zhuang et al. [13] introduced a five-dimensional model of a digital twin workshop and a real-time monitoring method of the workshop operation state. The realistic real-time synchronous mapping of the digital twin workshop to the real

physical workshop has brought great research potential to the production and manufacturing optimization of the workshop.

Aiming at the data imbalance caused by the lack of fault data in the industrial workshop scene, Guo [14] and others used the digital twin workshop to simulate a large number of balanced datasets and established a real-time fault diagnosis system combined with transfer learning. Liu et al. [15] established the digital twin scene of the wheel production workshop combined with a super network model, used a digital twin workshop to simulate and optimize a workshop scheduling scheme and effectively realized the intellectualization of workshop scheduling. Fang et al. [16] established a digital twin workshop to sense the arrival of new orders. This system can respond to dynamic events in a physical workshop in real time and reschedule unexecuted orders and new orders. Through continuous closed-loop iteration, the scheduling results become increasingly accurate. Wang et al. [17] combined a digital twin workshop with production scheduling technology to control all aspects of uncertain factors in the production process. This provided a new idea for the real-time dynamic scheduling of digital twin workshops. The use of digital twin technology can capture the real-time data of the workshop in the production process and make the scheduling more intelligent to deal with unexpected problems in actual production and to be closer to the real environment. However, there has been limited research on the combined scheduling problem of digital twin workshops.

2.2. Application of Reinforcement Learning to Workshop Scheduling

Workshop scheduling is essentially a combinatorial optimization problem that can be solved using reinforcement learning. Due to its high adaptability to the environment, reinforcement learning can respond to dynamic events in real time once the model is trained and can also generalize the learned model to unknown scheduling problems. Therefore, reinforcement learning has great potential to solve the job shop scheduling problem [18]. However, at present, the research work of reinforcement learning application in the field of job shop scheduling is still in the exploratory stage, and the related research results are still relatively few.

To minimize the average delay, Wang et al. [19] let each machine use a Q-learning algorithm to select appropriate rules from the given scheduling rules to formulate policies. Paterna Arboleda et al. [20] used the reinforcement learning method to find a dynamic control strategy and a neural network to approximate the value function in reinforcement learning. Zhang et al. [21] used a Q-learning algorithm and multiple heuristic rules as actions to solve the parallel machine scheduling problem, and experiments proved the superiority of the reinforcement learning algorithm. Yang et al. [22] proposed an adaptive scheduling control strategy that can effectively obtain dynamic knowledge. For different emergencies, this method is more effective than a single scheduling rule.

Recently, with the increasing popularity of deep learning and to overcome the problem of dimension explosion when traditional reinforcement learning solves complex problems, the concept of deep reinforcement learning was proposed [23]. Han et al. [24] used a disjunctive graph to represent the

reinforcement learning state and combined a convolutional neural network (CNN) and reinforcement learning (RL). The proposed method is more suitable for complex production environments and order-oriented manufacturing scenarios. Lin et al. [25] used an edge computing framework in a semiconductor intelligent factory to adjust the deep Q network, and the simulation results showed better performance than the method using only one scheduling rule. Park et al. [26] used a graph neural network (GNN) to represent the state in reinforcement learning and used the near end strategy optimization (PPO) algorithm to solve the job shop scheduling problem end-to-end. Finally, the results obtained on each benchmark are better than a single scheduling rule. Wang et al. [27] examined the dynamics, machine failure, job rework and other uncertainties in the actual workshop operation process. The proximal strategy optimization (PPO) algorithm is used to find the optimal scheduling strategy and can realize adaptive and real-time production scheduling. Liu et al. [28] regarded the job shop scheduling problem as a sequential decision problem, used the deep deterministic policy gradient (DDPG) algorithm to train the network in parallel in the multiagent environment, and applied it to the dynamic job shop scheduling problem.

From the above references, we can see that using digital twin technology can establish real-time mapping of physical workshops and capture dynamic real-time events. Using reinforcement learning to solve the job shop scheduling problem can solve various real-time dynamic events in the actual production process, such as the order insertion problem. Compared with a metaheuristic algorithm, reinforcement learning has higher efficiency and a lower time dimension. However, there is no effective combination of digital twins and deep reinforcement learning to solve the job shop problem. This is the research value of this paper.

3. Problem Description And Modelling

The job shop scheduling problem is a classical NP-hard problem. It can be described as follows: Suppose there are n jobs $\mathrm{J} = \left\{J_1, J_2, J_3, \cdots, J_i, \cdots, J_n\right\} (i = 1,2, \ldots, n)$, m machines $\mathrm{M} = \left\{M_1, M_2, M_3, \cdots, M_j, \cdots, M_m\right\} (j = 1,2, \ldots, m)$. Each job has l process $J_i = \left\{O_{i1}, O_{i2}, O_{i3}, \cdots, O_{il}\right\} (J_i \in \mathrm{J}, l = 1,2, \ldots, m)$, and each operation must pass through l of m machines according to specific requirements. The processing time $\mathrm{T} = \left\{t_{il}\right\} (i = 1,2, \ldots, n, l = 1,2, \ldots, m)$ of each process is known. The size of the job shop scheduling problem instance is recorded as $|\mathrm{J}| \times |\mathrm{M}|$. The job shop scheduling problem needs to consider the following assumptions:

- Each process is processed on the designated machine, and the processing can be started only after the previous process is completed;
- One machine can only process one job at a time;
- Each operation can only be processed once on one machine;

- The process sequence and processing time of each operation are known and do not change with changes in the processing sequence;
- All machines and jobs are available at the initial time;
- No process cannot be interrupted during processing.

The objective function in the scheduling process is makespan, which minimizes the maximum completion time (1) under constraints (2)–(7):

$$\mathrm{P} = min\left\{\max \mathrm{C}_i\right\}(i = 1,2, \ldots, n)$$

1

$$C_{il} - t_{il} \geq C_{i(l-1)}$$

2

$$\sum_{i=1}^{n} Y_{ijt} \leq 1 \forall j \in [1, m], \forall t \in T$$

3

$$\sum_{j=1}^{m} Y_{ijt} \leq 1 \forall i \in [1, n], \forall t \in T$$

4

$$\sum_{j=1}^{m} X_{ij} \geq 1 \forall i \in [1, n]$$

5

C_i represents the completion time of job J_i. C_{il} represents the completion time of the l-th process of operation J_i. When job J_i is assigned to machine M_j at time t, $Y_{ijt} = 1$; otherwise, $Y_{ijt} = 0$. When job J_i is assigned to machine M_j, $X_{ij} = 1$; otherwise, $X_{ij} = 0$.

4. Deep Reinforcement Learning For Job Shop Scheduling

4.1. Markov decision process formulation

The Markov property means that the state of the current time is only related to the state and action of the previous time and has nothing to do with the actions and states of other times. Set

$h_t = \{s_1, s_2, s_3, ..., s_t\}$ (h_t includes all states before time t). Then, the Markov property can be expressed as shown in Formula (6). If the state transition of the multistage decision problem satisfies the Markov property, then the decision problem is a Markov decision problem (MDP). The Markov decision process can be described as 5 tuples, as shown in Formula (7). S represents the state space and is a description of the environment. $A(s)$ represents the action space that can be selected in state s. The R agent represents the reward given by the environment after making the action. $\pi(a \mid s)$ represents the probability distribution of output action a at state s.

$$p\left(s_{t+1} \mid s_t, a_t\right) = p\left(s_{t+1} \mid h_t, a_t\right)$$

6

$$E = \{S, A(s), R, P, \pi(a \mid s)\}$$

7

State. In this paper, we use a disjunctive graph to represent the current state. The disjunctive graph of the current decision state s can be expressed as $G(s) = (O, C(s), D(s))$. O represents the set of all nodes, and each node is an operation. C represents the set of all conjunctive arcs, and the conjunctive arcs represent the execution sequence between different processes of the same job. D represents the set of all disjunctive arcs, which represent the execution sequence between different processes on the same machine. As shown in Fig. 1 (a), each node represents a process, S represents the initial node, T represents the end node, and a number around the node indicating the processing time is required. Finally, when all disjunctive arcs have directions, this is a solution to the JSSP problem, as shown in Fig. 1 (b). The longest path from the start node to the end node is called the critical path, and the length is equivalent to the maximum completion time of the JSSP problem solution, that is, the makespan.

(a) JSSP instance (b) Complete solution

Figure 1 Disjunctive graph representation.

Action. An action $a \in A(s)$ refers to the operation that can be performed in state s. In reinforcement learning, the definition of action needs to express the essence of the selected action as much as possible, and the scheduling rule is an expression of the essence of the scheduling object. In this paper, our actions are composed of different heuristic scheduling rules. We select four common scheduling rules and meet the diversification principle as much as possible. A description of each rule is listed in Table 1.

Table 1
Job actions

Symbol	Rule	Description
a_1	Shortest Processing Time (SPT)	Select the job with the shortest processing time
a_2	Longest Processing Time (LPT)	Select the job with the longest processing time
a_3	Shortest processing time of subsequent operation (SSO)	Select the job with the shortest processing time of subsequent operation
a_4	Long processing time of subsequent operation (LSO)	Select the job with the Longest processing time of subsequent operation

Reward. The learning goal of reinforcement learning is to maximize the accumulated reward. The scheduling goal of this paper is to minimize the maximum completion time. In the scheduling process, the machine utilization is closely related to the maximum completion time. We believe that the higher the machine utilization, the smaller the maximum completion time. Machine utilization is defined as the ratio of the total processing time of all scheduled tasks to the current completion time in state s, where J and M are the scales of the scheduling problem, as shown in Formula (8). The reward function is shown in Formulas (9)−(10). $C_{max}(s)$ is the makespan value in state s.

$$U = \frac{P}{J * M * C_{max}} \tag{8}$$

$$r_s = U_s - U_{s-1} \tag{9}$$

$$R = \sum_{s=1}^{S} r_s = \sum_{s=1}^{S} U(s) - U(s-1) = U(k) - U(0) = U(s) = \frac{P}{NM \cdot C_{max}(s)} \tag{10}$$

4.2. Training Details

Graph embedding contains sufficient relationship information of each node in the graph. In this paper, we use the embedding layer to calculate graph embedding, which makes it easier to perform various tasks. We use a graph isomorphism network (GIN) [29] to learn the states represented by disjunctive graphs in reinforcement learning. GIN has proven to have a simple structure and strong expression ability. Given a graph $G = (V, E)$, where V represents the set of nodes and E represents the set of edges. In GIN, the feature update of a node is the aggregation of the adjacent node features of this node and the GIN

activation value of the previous layer of this node. Then, we put the aggregated features into the multilayer perceptron. The node update in GIN can be expressed as

$$h_v^{(k)} = MLP_{\theta_k}^{(k)}\left(\left(1+\epsilon^{(k)}\right)\cdot h_v^{(k-1)} + \sum_{u\in \mathrm{N}(v)} h_u^{(k-1)}\right)$$

11

where $h_v^{(k)}$ represents the activation value of node v in layer k, $\mathrm{N}(v)$ is all adjacent nodes of node v, and ϵ is a scalar parameter that can be learned. MLP is a multilayer perceptron function. The parameter is θ_k.

$$h_{\mathrm{G}}^{(k)} = READOUT\left(\left\{h_v^{(k)} \mid v \in G\right\}\right)$$

12

After obtaining the representation of all node features in the GIN, the READOUT function is used to convert the node features into graph features. The commonly used readout functions include sum (SUM), (AVG) and (MAX). According to [29], SUM has the strongest representation ability. Therefore, the SUM function is selected in this paper, and the formula is shown in (12). $h_{\mathrm{G}}^{(k)}$ represents the representation vector of graph G on layer k.

The proximal policy optimization (PPO) algorithm is based on the actor critical (AC) framework proposed by Openai in 2017. Compared with TRPO (trust region policy optimization), the PPO algorithm is easier to calculate, has strong adaptability and stable training, and its performance is not degraded due to the updating range being too large in the training process. This has been used as a benchmark algorithm in the field of reinforcement learning. The detailed training process of the algorithm is shown in Algorithm 1.

Algorithm 1: Pseudocode of PPO Algorithm

1: Input: actor network π_θ and $\pi_{\theta_{\text{old}}}$, critic network v_φ, and initialize the number of training steps N, discounting factor γ, update epoch U, coefficient c_p, c_v, c_e, ratio ϵ.

2: for $n = 1,2, \ldots, N$ **do**

3: generate K JSSP examples;

4. for episode $k = 1,2, \ldots, K$ **do**

5. while the episode is not terminated do

6. for $t = 1,2, \ldots$ **do**

7. observe $s_{k,t}$;

8. if $s_{k,t}$ is terminal **then**

9. break;

10. end

11. perform $a_{k,t}$ through $\pi_{\theta_{\text{old}}}\left(\cdot \mid s_{k,t}\right)$;

12. get reward $r_{k,t}$ and next state $s_{k,t+1}$;

13. compute advantage function $\hat{A}_{k,t} = \sum_0^t \gamma^t r_{k,t} - v_\varphi\left(s_{k,t}\right)$;

14. $r_{k,t}(\theta) = \dfrac{\pi_\theta\left(a_{k,t} \mid s_{k,t}\right)}{\pi_{\theta_{\text{old}}}\left(a_{k,t} \mid s_{k,t}\right)}$

15. end

16. end

17. $L_k^{CLIP}(\theta) = \sum_0^t min\left(r_{k,t}(\theta)\hat{A}_{k,t}, \text{clip}\left(r_{k,t}(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_{k,t}\right)$

18. $L_k^{VF}(\varphi) = \sum_0^t \left(v_\varphi\left(s_{k,t}\right) - \hat{A}_{k,t}\right)^2$

19. $L_k^S(\theta) = \sum_0^t S\left(\pi_\theta\left(a_{k,t} \mid s_{k,t}\right)\right)$

20. $L_k(\theta, \varphi) = c_p L_k^{CLIP}(\theta) - c_v L_k^{VF}(\varphi) + c_e L_k^S(\theta)$

21. end

22. for $u = 1,2, \ldots, U$ **do**

23. update θ, φ;

Algorithm 1: Pseudocode of PPO Algorithm

24. end

25. update $\theta_{\text{old}} \leftarrow \theta$

26.end

5. Experiment

5.1. Simulation results

In this section, we randomly generate some instances (5 * 5, 6 * 6, 8 * 8, 10 * 10, 15 * 15, 20 * 20). For each size, we randomly generate 100 instances, and the corresponding processing time of each job is an integer in the range of 1–99 to train and test the proposed algorithm. Finally, the trained algorithm is applied to ft [30], La [31] and taillard' [32] examples to compare the performance of the algorithm. During the training, we used PyTorch to implement the PPO algorithm. PyTorch runs on the Windows 10 64-bit operating system, and the hardware used is an Intel Xeon bronze 3106 CPU with NVIDIA Quadro p6000 GPU.

In deep reinforcement learning, the DRL algorithm is very sensitive to superparameters, and the training results and convergence are different under different superparameters. Therefore, taking the parameter adjustment in the training process of our randomly generated (6 * 6) example as an example, this paper analyses the influence of different clip values and network structure learning rates on the training process of the reinforcement learning algorithm. As shown in Fig. 2, the abscissa of each point indicates the training times, and the ordinate indicates the makespan. The training times are set to 10,000 during the training.

In the PPO algorithm, the clip value means cutting the advantage function and keeping it in the trust domain to prevent overly large changes in the update process from affecting the effect of the algorithm. This also prevents the accumulation of a large number of errors in the process of reusing the same batch of samples and cuts off the gradient of some samples that make the output of the current strategy change too much. The clip value usually takes a positive number greater than 0. For the same input sample, the ratio of policy output change caused by each parameter update is $1 - \epsilon$ and $1 + \epsilon$. The smaller the ϵ, the more stable the training; and the larger the ϵ, the more samples that are saved. In this paper, we select three values of 0.1, 0.2 and 0.3 for analysis. As shown in Fig. 2 (a), when clip = 0.1, the convergence effect of the algorithm is the best and the makespan is the smallest. Therefore, we select clip = 0.1.

Similarly, the network structure has a great impact on the algorithm training process. In this paper, we set up the actor network π_θ and critic network v_φ. The GIN layer is shared. The GIN layer has two MLPs. There are four layers of MLP in π_θ and v_φ. As shown in Fig. 2 (c), by adjusting the dimension of each layer, we compare four different network structures 64, 128, 256 and 512. When the dimension is equal to

128 and 256, the performance is the best, but an excessively high dimension affects the training efficiency of the algorithm. Therefore, the dimension selected in this paper is 128.

Figure 2 (b) shows the impact of different learning rates on the algorithm performance. When lr = 1e-3, the algorithm performance is the best.

Table 2
Algorithm Parameters

Parameters	Values
Proximal Policy Optimization	
Learning rate	1e-3
Clip	0.1
Entropy coefficient	0.1
Value function coefficient	2
Policy loss coefficient	2
Number of steps per update	32
Discount factor	0.95
Genetic Algorithm	
Iteration	5000
Crossover probability	0.8

To further evaluate the performance of the algorithm, we use a public example to verify it. We compare the algorithm to a genetic algorithm and various scheduling rules. The genetic algorithm is the most classical metaheuristic algorithm in the field of job shop scheduling. It has good performance, and the effect of solving examples is very good, but the solution time is relatively long. The parameter values of the PPO and GA algorithms are listed in Table 2.

Finally, the results of each algorithm and scheduling rule in the open example are shown in Table 3 and Fig. 3. The results indicate that the maximum completion time of the methods used in this paper is shorter than the maximum completion time of the heuristic rules, and 80% of the examples are better than or equal to the metaheuristic algorithm GA, while the DRL algorithm is much better than the metaheuristic algorithm with regard to solving speed. Therefore, the overall effect of the DRL method is better than the heuristic scheduling rules and metaheuristic algorithm. Compared with the optimal solution, the relative error of DRL is very small and is close to the theoretical optimal solution. The DRL method can obtain the theoretical optimal solution on 60% of the open examples, especially for examples with a small number of machines, and DRL can converge well to obtain the optimal solution.

Table 3
Experimental results

Instance	Size	OPT	SPT	LPT	SSO	LSO	GA	PPO
La01.txt	(10×5)	666	920	889	844	806	706	**666**
La02.txt	(10×5)	655	901	894	952	815	732	**655**
La03.txt	(10×5)	597	770	748	904	793	680	**597**
La04.txt	(10×5)	590	916	848	937	868	**637**	639
La05.txt	(10×5)	593	827	787	894	633	**593**	**593**
La06.txt	(15×5)	926	1369	1105	1201	1009	948	**926**
La07.txt	(15×5)	890	1128	1145	1187	1104	985	**894**
La08.txt	(15×5)	863	1168	1061	1190	1131	932	**863**
La09.txt	(15×5)	951	1289	1105	1319	1174	1001	**951**
La10.txt	(15×5)	958	1345	1136	1604	976	**958**	**958**
La11.txt	(20×5)	1222	1654	1476	1470	1289	1304	**1222**
La12.txt	(20×5)	1039	1352	1222	1324	1173	1087	**1047**
La13.txt	(20×5)	1150	1747	1298	1592	1350	1207	**1150**
La14.txt	(20×5)	1292	1757	1360	1895	1341	**1292**	**1292**
La15.txt	(20×5)	1207	1476	1510	1627	1411	1403	**1212**
La16.txt	(10×10)	945	1588	1238	1565	1274	**979**	980
La17.txt	(10×10)	784	1094	1157	1135	1060	809	**794**
La18.txt	(10×10)	848	1259	1264	1232	1090	903	**851**
La19.txt	(10×10)	842	1339	1140	1225	1134	878	**872**
La20.txt	(10×10)	902	1331	1293	1453	1219	964	**938**
Ft06.txt	(6×6)	55	87	73	88	61	**55**	**55**
Ta01.txt	(15×15)	1231	1872	1812	2148	1957	1419	**1315**
Ta11.txt	(20×15)	1357	2273	2117	2243	2216	**1633**	1788
Ta21.txt	(20×20)	1642	2443	2691	2610	2647	**1991**	1952
Ta31.txt	(30×15)	1764	2670	2589	2916	2478	2226	**1986**

5.2. Case study

To further verify the effect of the proposed method in the actual scene, we take the personalized customized car assembly line in our laboratory as an example. As shown in Fig. 4, we establish the framework of the digital twin workshop according to the five-dimensional model proposed by Tao et al. [8] to include a physical layer, virtual layer, service layer, DT data and connection (CN). The lower part of Fig. 4 introduces the production content of our laboratory. Our laboratory is a workshop for producing eight models of customized cars. The production of each model includes eight processes: chassis, front wheel, rear wheel, frame, left door, right door, front cover and logo. These are assembled on eight different robot arms, and the assembly times are different. The lower right part of Fig. 4 shows the customized ordering system we developed. We can select the model and quantity to be customized through the software. After confirming the order, the order is transmitted to the database. When a batch of orders is entered, we use the PPO algorithm described above to solve it. According to the obtained results, the automated guided vehicle (AGV) transports the materials. After the AGV runs to the designated station, the robot arm program is manually started to assemble the trolley. The job processing time in this paper includes the sum of the AGV transportation time and robot arm assembly time.

The solution process of the job shop scheduling problem based on the digital twin is shown in Fig. 5. First, we establish a digital twin scenario and then schedule tasks according to the PPO algorithm described above. We simulate and interact the scheduling results in the digital twin scenario. If the simulation results meet our preset requirements, then we feed back the results to the physical workshop for execution. If the personalized customized ordering system is continuously used for task distribution during the solution process, when the order insertion occurs, due to the perception of the DT workshop, the rescheduling mechanism is triggered after the DT data changes. When rescheduling occurs, the newly inserted order and original unfinished order are solved together, and the final result is simulated and verified in the digital twin workshop. If it meets the requirements, then it is fed back to the physical workshop for task assembly.

Next, we take 8 models in our laboratory as examples for verification. We train the algorithm with a randomly generated example of 8 * 8. The makespan during the operation of the algorithm is shown in Fig. 6. Then, the trained algorithm is used to solve it. Take 8 jobs as an example, as shown in Formula (13). The first parameter is the equipment number that can be processed, which here refers to the robot arm, and the second parameter is the required time (s).

$$J_1 = \begin{bmatrix} (3,38) & (5,67) & (1,62) & (7,54) & (8,59) & (6,73) & (4,65) & (2,49) \\ (8,49) & (4,71) & (5,71) & (7,41) & (1,51) & (2,49) & (3,62) & (6,45) \\ (1,48) & (4,63) & (6,51) & (5,59) & (3,42) & (7,43) & (2,61) & (8,52) \\ (4,60) & (8,53) & (3,63) & (1,72) & (2,76) & (7,69) & (6,59) & (5,69) \\ (6,68) & (8,74) & (4,66) & (5,69) & (2,42) & (3,70) & (1,49) & (7,52) \\ (7,57) & (2,48) & (8,49) & (3,66) & (1,68) & (6,79) & (4,56) & (5,41) \\ (1,48) & (5,79) & (6,83) & (2,85) & (8,73) & (7,52) & (4,43) & (3,62) \\ (2,50) & (6,86) & (3,74) & (4,53) & (1,63) & (8,71) & (5,66) & (7,88) \end{bmatrix}$$

13

The generated initial scheduling plan is solved by the PPO algorithm, as shown in Fig. 9. The abscissa represents the time (s), and the ordinate represents the equipment, which refers to the number of robot arms. The first parameter in the figure is the model, and the second parameter is the corresponding process. Taking 7 − 3 as an example, the parameter represents the third process of the seventh model. The maximum completion time obtained is 730 s.

Assuming that a dynamic event occurs at t = 500 s, we issue Model 2 and Model 8 through the personalized customization order system. Relying on the real-time mapping of the digital twin workshop, one can obtain the order insertion information in the DT data layer of the digital twin workshop, start the rescheduling solution and then simulate the rescheduling scheme in the digital twin workshop. If the requirements meet our preset, it is fed back to the physical workshop for execution. The obtained rescheduling scheme is shown in Fig. 14.

Assuming that a dynamic event occurs at t = 500 s, we issue Model 2 and Model 8 through the personalized customization order system. Relying on the real-time mapping of the digital twin workshop, one can obtain the order insertion information in the DT data layer of the digital twin workshop, start the rescheduling solution and then simulate the rescheduling scheme in the digital twin workshop. If the requirements meet our preset, it is fed back to the physical workshop for execution. The obtained rescheduling scheme is shown in Fig. 10.

Finally, we compare the results with heuristic rules and a genetic algorithm. The settings of various parameters are shown in Section 5.1. The running time and minimum maximum completion time of the scheduling results are listed in Table 4. It can be seen from the table that the running time of the heuristic rules and PPO algorithm is much shorter than that of the genetic algorithm, and the optimal value can be obtained in a very short time after training the DRL model. Therefore, we think the results obtained by DRL are impressive.

	SPT	LPT	SSO	LSO	GA	DRL
Execution time(s)	2.1	2.1	2.1	2.1	89.4	1.6
Makespan	992	1011	992	885	810	730

6. Conclusion

We studied the job shop scheduling problem in intelligent manufacturing. A dynamic adaptive algorithm based on deep reinforcement learning was proposed. The state in reinforcement learning is represented by a disjunctive graph, and the graph neural network is used for learning. To evaluate the performance of the proposed algorithm, we used heuristic scheduling rules and a metaheuristic algorithm to prove the superiority of the proposed algorithm. Furthermore, in the actual production process, various dynamic events in the workshop affected the execution of the initial scheduling plan. To solve the dynamic scheduling problem in the actual production process, we proposed a method combining digital twin and deep reinforcement learning. This method uses a digital twin workshop to capture the dynamic disturbance in the actual production process in real time. When dynamic events are captured, we use deep reinforcement learning to reschedule, which avoids the disadvantage of the long running time of the metaheuristic algorithm. Finally, we carried out experimental verification in a laboratory environment. The results show that the method based on the digital twin and deep reinforcement learning can effectively deal with sudden situations in the actual production process.

Declarations

Acknowledgements

The work was supported in part by the National Key Research and Development Program of China under the Grant No. 2021YFB3300503, in part by the National Defense Fundamental Research Foundation of China under the Grant No. JCKY2020413C002.

Availability of data and materials

The data that support the findings of this study are available from the corresponding author on reasonable request.

Ethical approval The authors declare that there is no ethical issue applied to this article.

Consent for publication The authors declare that all authors agree to sign the transfer of copyright for the publisher to publish this article upon on acceptance.

Competing interests 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.

References

1. Sue B (2016) Innovation Design: Made in China 2025. dmi Rev 27(1):53–58
2. Mosterman PJ, Zander J (2016) Industry 4.0 as a Cyber-Physical System study. Softw Syst Model 15:17–29. doi: 10.1007/s10270-015-0493-x
3. Zhan Y, Qiu C (2008) Genetic algorithm application to the hybrid flow shop scheduling problem. 10.1109/icma.2008.4798833
4. Liu L-L, Zhao G-P, Ou'Yang S-S, Yang Y-J (2011) Integrating theory of constraints and particle swarm optimization in order planning and scheduling for machine tool production. Int J Adv Manuf Technol 57:285–296. doi: 10.1007/s00170-011-3294-6
5. Lu Y, Liu C, Wang KI-K et al (2020) Digital Twin-driven smart manufacturing: Connotation, reference model, applications and research issues. Robot Comput Integr Manuf 61:101837. doi: 10.1016/j.rcim.2019.101837
6. Silver D, Schrittwieser J, Simonyan K et al (2017) Mastering the game of Go without human knowledge. Nature 550:354–359. doi: 10.1038/nature24270
7. Grieves M, Vickers J, Digital, Twin (2017) Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems[M]. Springer International Publishing, pp 85–113
8. Tao F, Zhang M (2017) Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing. IEEE Access 5:20418–20427
9. Tao F, Zhang H, Liu A, Nee AYC (2019) Digital Twin in Industry: State-of-the-Art. IEEE Trans Industr Inf 15:2405–2415. doi: 10.1109/tii.2018.2873186
10. Tao F, Cheng J, Qi Q et al (2018) Digital twin-driven product design, manufacturing and service with big data. Int J Adv Manuf Technol 94:3563–3576. doi: 10.1007/s00170-017-0233-1
11. Qi Q, Tao F (2018) IEEE Access 6:3585–3593. doi: 10.1109/access.2018.2793265. Digital Twin and Big Data Towards Smart Manufacturing and Industry 4.0:360 Degree Comparison
12. Tao F, Sui F, Liu A et al (2019) Digital twin-driven product design framework. Int J Prod Res 57:3935–3953. doi: 10.1080/00207543.2018.1443229
13. Zhuang C, Miao T (2021) The connotation of digital twin, and the construction and application method of shop-floor digital twin. Robot Com-Int Manuf 68:102075. https://doi.org/10.1016/j.rcim.2020.102075
14. Guo K, Wan X, Liu L, Gao Z, Yang M (2021) Fault Diagnosis of Intelligent Production Line Based on Digital Twin and Improved Random Forest. Appl Sci 11:7733
15. Liu Z, Chen W, Zhang C, Yang C, Cheng Q (2020) Intelligent scheduling of a feature-process-machine tool supernetwork based on digital twin workshop. J Manuf Syst. https://doi.org/10.1016/j.jmsy.2020.07.016
16. Fang Y, Peng C, Lou P et al (2019) Digital-Twin-Based Job Shop Scheduling Toward Smart Manufacturing. IEEE Trans Industr Inf 15:6425–6435. doi: 10.1109/tii.2019.2938572

17. Wang Y, Wu Z (2020) Model construction of planning and scheduling system based on digital twin. Int J Adv Manuf Technol 109:2189–2203. doi: 10.1007/s00170-020-05779-9
18. Ren J, Ye C, Yang F (2021) Solving flow-shop scheduling problem with a reinforcement learning algorithm that generalizes the value function with neural network. Alexandria Eng J 60:2787–2800. doi: 10.1016/j.aej.2021.01.030
19. Wang Y-C, Usher JM (2004) Learning policies for single machine job dispatching Robot. Computer-Integrated Manuf 20:553–562. doi: 10.1016/j.rcim.2004.07.003
20. Paternina-Arboleda CD, Das TK (2005) A multi-agent reinforcement learning approach to obtaining dynamic control policies for stochastic lot scheduling problem. Simul Model Pract Theory 13:389–406. doi: 10.1016/j.simpat.2004.12.003
21. Zhang Z, Zheng L, Weng MX (2007) Dynamic parallel machine scheduling with mean weighted tardiness objective by Q-Learning. Int J Adv Manuf Technol 34:968–980. doi: 10.1007/s00170-006-0662-8
22. Yang H-B, Yan H-S (2009) An adaptive approach to dynamic scheduling in knowledgeable manufacturing cell. Int J Adv Manuf Technol 42:312–320. doi: 10.1007/s00170-008-1588-0
23. Mnih V, Kavukcuoglu K, Silver D et al (2015) Human-level control through deep reinforcement learning. Nature 518:529–533. doi: 10.1038/nature14236
24. Han B-A, Yang J-J (2020) Research on Adaptive Job Shop Scheduling Problems Based on Dueling Double DQN. IEEE Access 8:186474–186495. doi: 10.1109/access.2020.3029868
25. Lin C-C, Deng D-J, Chih Y-L, Chiu H-T (2019) Smart Manufacturing Scheduling With Edge Computing Using Multiclass Deep Q Network. IEEE Trans Industr Inf 15:4276–4284. doi: 10.1109/tii.2019.2908210
26. Park J, Chun J, Kim SH et al (2021) Learning to schedule job-shop problems: representation and policy learning using graph neural network and reinforcement learning. Int J Prod Res 59:3360–3377. doi: 10.1080/00207543.2020.1870013
27. Wang L, Hu X, Wang Y et al (2021) Dynamic job-shop scheduling in smart manufacturing using deep reinforcement learning. Comput Netw 190:107969. doi: 10.1016/j.comnet.2021.107969
28. 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. doi: 10.1109/access.2020.2987820
29. Keyulu Xu W, Hu (2018) Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In International Conference on Learning Representations,
30. Fisher H, Thompson G (1963) In: Industrial Scheduling JF, Muth, Thompson GL (eds) “Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules,”. Prentice-Hall, vol., Upper Saddle River, NJ, USA, pp 225–251
31. Li LS (1984) “Resouce constrained project scheduling: An experimental investigation of heuristic scheduling techniques (supplement),” School of Ind. Admin., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. ORNL/Sub-7654/1,

32. Taillard E (1993) Benchmarks for basic scheduling problems. Eur J Oper Res 64:278–285. doi: 10.1016/0377-2217(93)90182-m

Figures

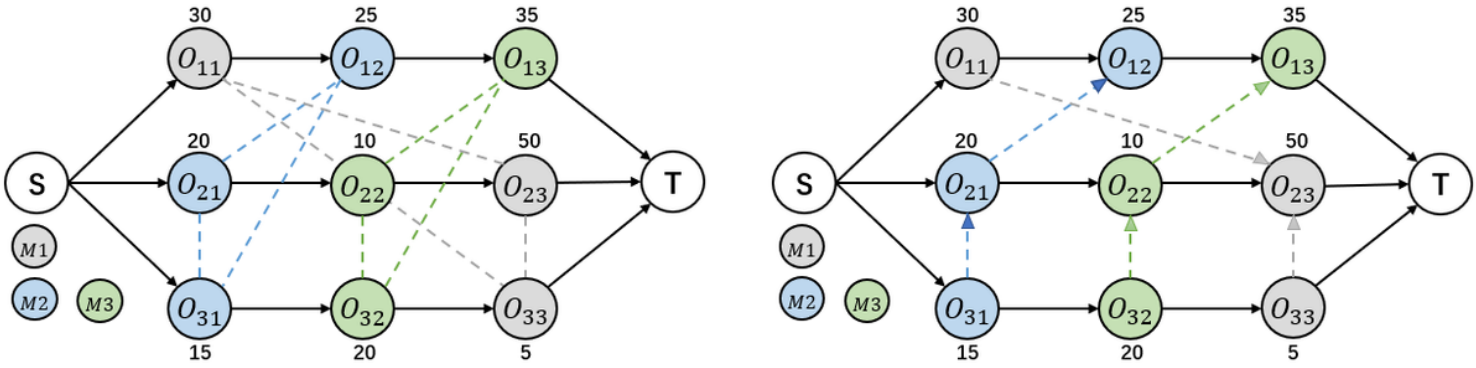

Figure 1

Disjunctive graph representation.

(a) JSSP instance (b) Complete solution

Figure 2

Verification results for each hyperparameter

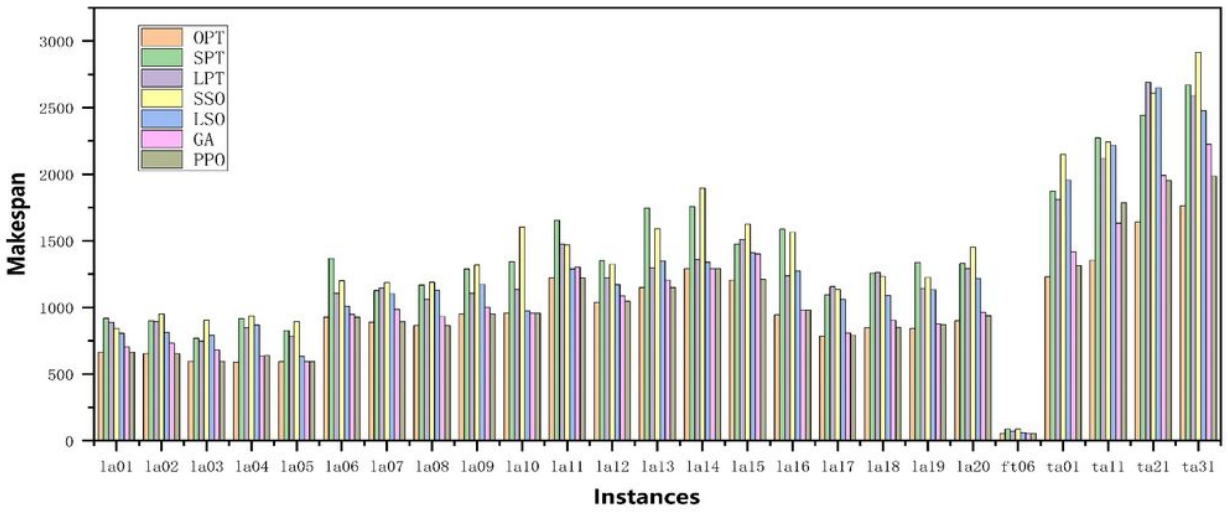

Figure 3

Makespan with different algorithms

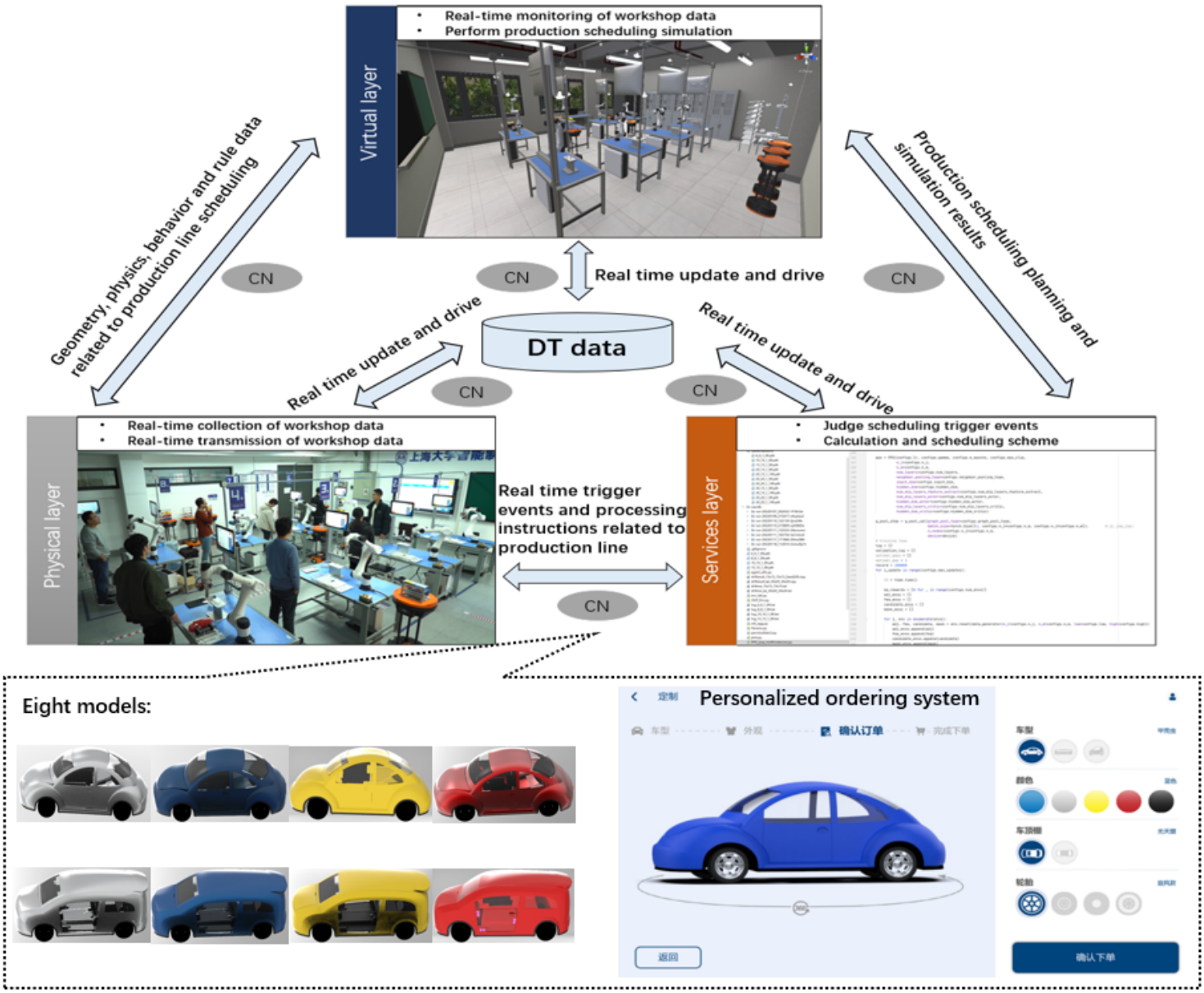

Figure 4

Overall framework of digital twin workshop

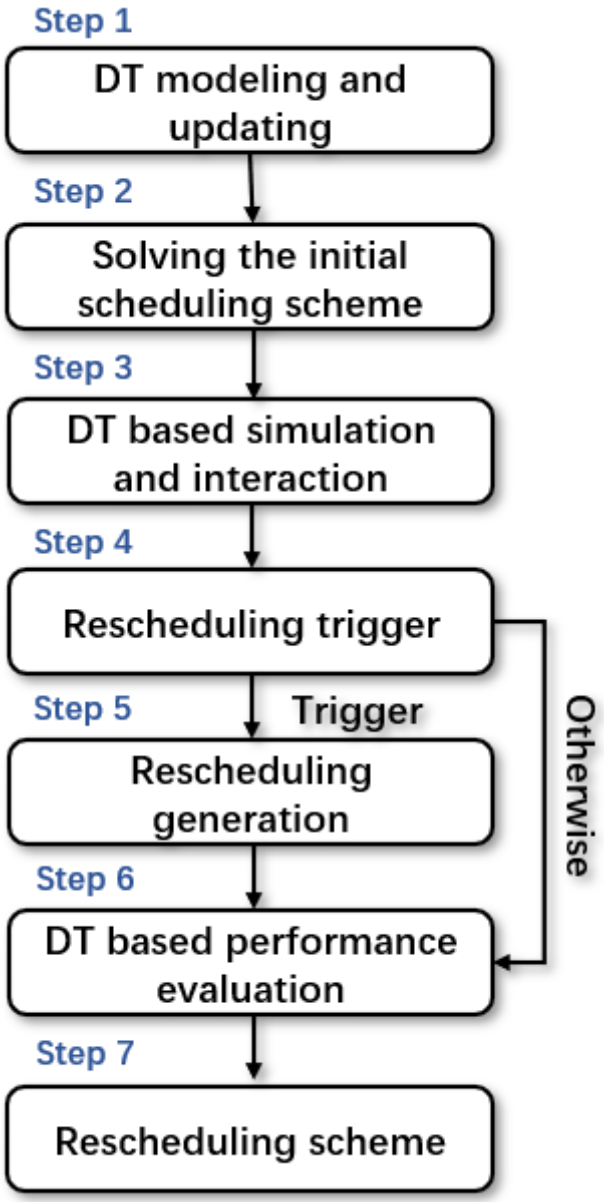

Figure 5

Makespan with different algorithms

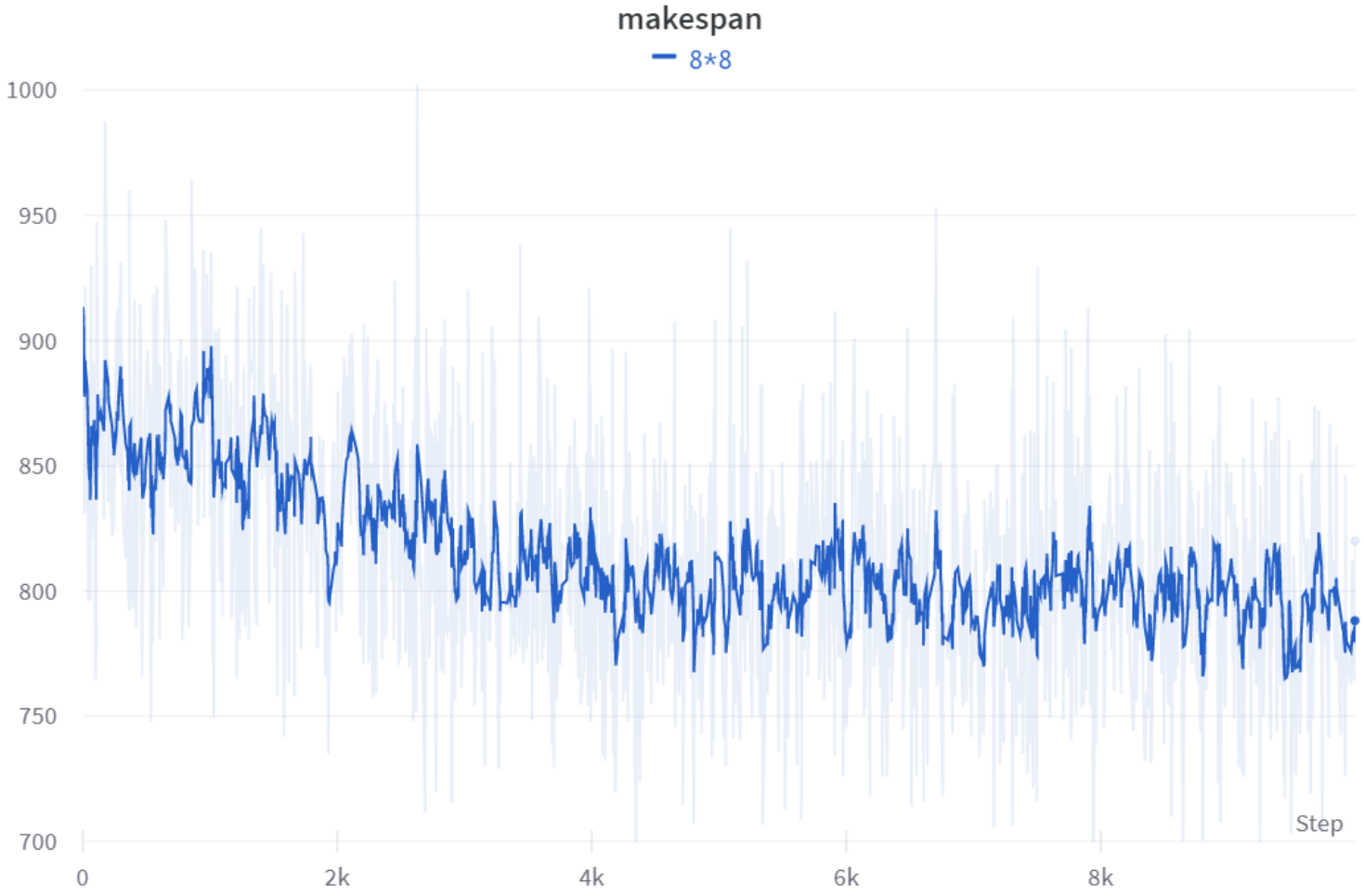

Figure 6

Makespan value during training

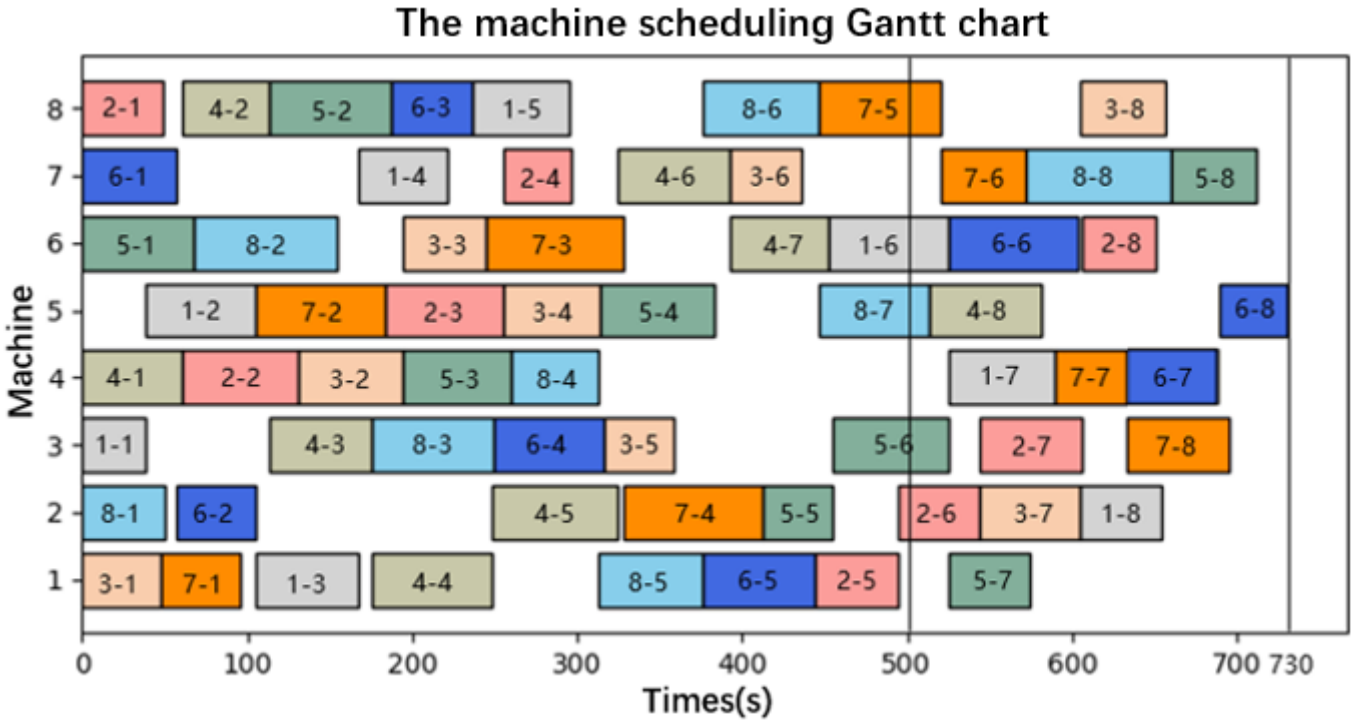

Figure 7

Initial scheduling plan

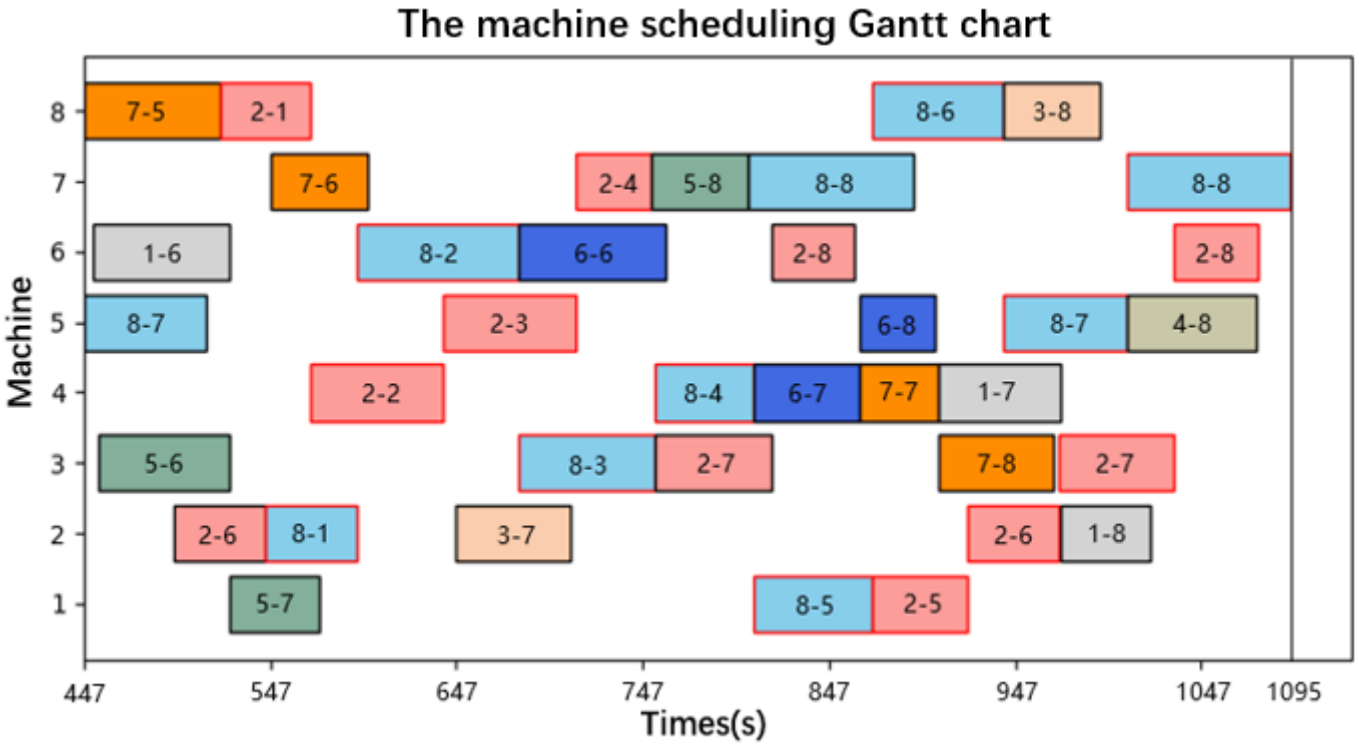

Figure 8

Rescheduling plan after new order arrives