System Modeling, Control, and Intelligent Manufacturing Systems - Research Article

Advances in Mechanical Engineering

Advances in Mechanical Engineering
2022, Vol. 14(3) 1–19
© The Author(s) 2022
DOI: 10.1177/16878132221086120
journals.sagepub.com/home/ade
SAGE

Reinforcement learning for online optimization of job-shop scheduling in a smart manufacturing factory

Tong Zhou, Haihua Zhu, Dunbing Tang, Changchun Liu, Qixiang Cai, Wei Shi and Yong Gui

Abstract

The job-shop scheduling problem (JSSP) is a complex combinatorial problem, especially in dynamic environments. Low-volume-high-mix orders contain various design specifications that bring a large number of uncertainties to manufacturing systems. Traditional scheduling methods are limited in handling diverse manufacturing resources in a dynamic environment. In recent years, artificial intelligence (AI) arouses the interests of researchers in solving dynamic scheduling problems. However, it is difficult to optimize the scheduling policies for online decision making while considering multiple objectives. Therefore, this paper proposes a smart scheduler to handle real-time jobs and unexpected events in smart manufacturing factories. New composite reward functions are formulated to improve the decision-making abilities and learning efficiency of the smart scheduler. Based on deep reinforcement learning (RL), the smart scheduler autonomously learns to schedule manufacturing resources in real time and improve its decision-making abilities dynamically. We evaluate and validate the proposed scheduling model with a series of experiments on a smart factory testbed. Experimental results show that the smart scheduler not only achieves good learning and scheduling performances by optimizing the composite reward functions, but also copes with unexpected events (e.g. urgent or simultaneous orders, machine failures) and balances between efficiency and profits.

Keywords

Job shop, online scheduling, multi-objective optimization, composite reward, reinforcement learning

Date received: 27 October 2021; accepted: 18 February 2022

Handling Editor: Chenhui Liang

Introduction

In the last three decades, the world has experienced three times of industrial revolutions, enabling industry with automated machines and information systems to support rapid and innovative product design. The manufacturing industry plays an important role in the development processes of the economy and technology. Today, manufacturing companies face a large number of challenges in a turbulent environment originating from saturated markets, abrupt and unprecedented changes in market demands, an increasing number of product variants, and smaller lot sizes.[1,2] This trend is intensified by production modes such as mass customization and individualization, which guarantee the creation of unique products to meet the requirements of almost every customer.[3] The transformation of

College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, China

Corresponding author:
Haihua Zhu, College of Mechanical and Electrical Engineering, Nanjing University of Aeronautics and Astronautics, 29 Yudao Street, Nanjing 210016, China.
Email: h.zhu@nuaa.edu.cn

traditional industrial markets inevitably leads to the evolution of manufacturing modes.

Production scheduling is a complex combinatorial problem, more specifically, a non-polynomial-hard (NP-hard) problem, which coordinates manufacturing resources to get optimal sequences and combinations. It is traditionally elaborated in a centralized manner using offline methods and considering the problems are static and deterministic.[4] However, initial production plans often become invalid due to many unexpected events that come from internal (e.g. machine failures, operator absence), external (e.g. rush orders, unavailability of raw materials), or composite factors (e.g. time fluctuation, resource uncertainty). Therefore, scheduling algorithms are required to modify the original schedules for eliminating the disturbances. Early studies focus on rescheduling resources offline to achieve new schedules, which may suspend the manufacturing processes and increase the lead time.[5] Simulation technologies (e.g. multi-agent system) build scheduling algorithms with empirical rules and historical data, but they are limited in achieving optimal schedules in a changeable manufacturing environment.[6] Recently, artificial intelligence (AI) algorithms such as reinforcement learning (RL) are adopted to industrial manufacturing with regard to online scheduling. However, most RL-based scheduling algorithms are inefficient at learning to optimize the decision-making policies when multiple objectives are considered in a smart manufacturing factory.[7] The overall performances of manufacturing systems are influenced by many factors such as order requirements, machine properties, and supply chain profits, which can be transformed into composite reward functions in RL-based scheduling systems. This paper realizes online scheduling based on RL with composite reward functions, which makes manufacturing systems to be more efficient and robust. Scheduling decisions are made online according to the operation attributes and machine states together. Composite rewards are given to the smart scheduler at all action moments, equipping smart manufacturing systems with self-learning and adaptation capabilities.

The remaining sections of this paper are organized as follows. Section 2 shows an overview of existing manufacturing scheduling approaches. Based on RL, Section 3 invites composite rewards and online methods to develop a system with self-learning abilities for job-shop scheduling. Section 4 designs an experimental testbed to conduct case studies on online scheduling. Section 5 expresses detailed experimental results and proposes the general relationships between composite rewards and manufacturing scheduling. Finally, Section 6 rounds up the paper with conclusions and prospects.

Literature review

The production scheduling problem has been widely studied, mainly due to its dynamic nature, highly combinatorial aspects, and applicability in manufacturing systems.[8] In past years, researchers proposed many scheduling methods such as mathematical programming,[9,10] rule-based dispatching,[11–13] expert systems,[14,15] heuristic search,[16,17] or biomimetic algorithms.[18] For example, Hou et al.[19] formulated an integrated distributed production and distribution problem with a mixed integer programming model and developed an enhanced brain storm optimization algorithm to solve the model.

Fu et al.[20] proposed an evolutionary algorithm and a local search method to solve a dual-objective stochastic hybrid flow shop deteriorating scheduling problem considering job deterioration and uncertain job processing time. However, these methods handle jobs or uncertainties with continuous iterations or periodic optimization, which takes extra computing time to adjust the initial plans offline and obtain newly optimal schedules. In addition, duplicated calculations for rescheduling resources may cause the whole system to shut down when the environmental states change frequently.[21] When dealing with scheduling and rescheduling problems, some researchers used swarm intelligent algorithms[22] to obtain optimal solutions from a large action space. In dynamic and uncertain environments, the optimized schedule produced by traditional scheduling methods can quickly become unacceptable, so dynamic rescheduling is required as fast as possible.

The fourth industrial revolution marked by intelligent manufacturing attracts researchers to a new research area of intelligent scheduling.[23] Agent technologies provide a natural way to conduct production scheduling in distributed manufacturing environments.[24–27] Many socialized interactive mechanisms, such as contract net protocol,[28] game theory,[29] auction mechanism,[30] are used to coordinate machines in multi-agent systems, where resources can acquire rational dynamic schedules autonomously based on specific rules.[31] In traditional multi-agent systems, machines participate in negotiation and interaction with each other according to specific rules. Such agent-based or holonic[4] scheduling systems implement distributed manufacturing, but they require the support of sophisticated rules and have poor adaptability. Rule-based scheduling methods provide a platform for machines to realize real-time job-shop scheduling; however, historical data and future circumstances cannot be stored and anticipated. Hence, some evaluation factors are used to record operation logs such as capability, self-confidence, activity, and robustness. These factors provide guidance for prospective decisions, but they can only mark down the participation times of machines in the decision platform. Thus, such improvements are weak in providing comprehensive and rapid decision mechanisms for machine interaction, and it is difficult to improve the scheduling performance substantially. Distributed scheduling integrates discrete units or factories and enables them to operate independently, which improves the operating efficiency of the whole manufacturing system.[32] However, the distributed manufacturing manner may cause environmental fluctuation and local optimum.

Currently, innovative online scheduling methods are needed to follow the trend of mass customization of products. These methods have one or more typical features such as self-organized operation, self-adaptation adjustment, and self-learning optimization. With the success of Alpha Go,[33,34] RL aroused the interest of researchers, and many of them tried to apply RL to production scheduling. Zhang et al.[35] addressed a dynamic unrelated parallel machine scheduling problem with a mean weighted tardiness objective, but unexpected events, order details, and rewards in addition to tardiness are not involved. Shiue et al.[36] proposed a real-time scheduling method based on RL using multiple dispatching rules by incorporating two main mechanisms, that is, an offline learning module and a Q-learning-based module. Shahrabi et al.[37] used RL to improve the scheduling performance for dynamic job-shop scheduling problems, considering random job arrivals and machine failures. Kardos et al.[38] designed a scheduling algorithm based on Q-learning to solve the dynamic job-shop scheduling problem for reducing the average lead-time of production orders. However, it is difficult for basic Q-learning algorithms to adapt to problems with a large action space. Wang et al.[39] used correlated equilibrium to propose a multi-agent RL algorithm for makespan and cost optimization to guide the scheduling of multi-workflows over clouds, but many real events and factors existing in real manufacturing conditions are neglected. Such research brought new approaches to manufacturing scheduling, dealing with dynamic task assignments, and solving uncertainties well in job-shop scheduling problems. Nevertheless, the researches mentioned above are mainly concerned with optimizing makespan and costs from the operational research aspect, regardless of an overall study of various disturbances, reward-scheduling mechanisms, and industrial implementation. Meanwhile, most approaches to job-shop scheduling own fake real-time characteristics without online features that are largely influenced by job numbers and computing efficiency.

This current paper concentrates on establishing an online scheduling system to deal with the tendency of globalized and individualized manufacturing. A smart factory testbed was built to realize communication between machines and the smart scheduler. RL is used to establish a smart scheduler for online scheduling orders and handling unexpected events, which equips the self-organized manufacturing system with self-adaptive abilities. The smart scheduler improves its comprehensive capabilities through operating processes guided by innovative composite rewards. An instant reward is given to the smart scheduler after a scheduling action is taken rather than after all orders are completed, which increases the training efficiency of the smart scheduler.

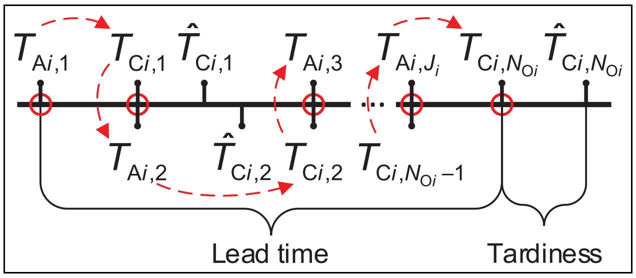


Figure 1. The timeline of job J_i and its component operations: scheduling moments are marked by red circles.

Research methodology

Model description

An order contains only one job denoted by J_i, and a job has one or more operations denoted by $O_{i,j}$, where J_i denotes the i th job and $O_{i,j}$ denotes the j th operation of the i th job. The total number of jobs is N_{J}, and the total number of operations of job J_i is $N_{\mathrm{O}i}$. The attributes of an operation $O_{i,j}$ include initialization time $T_{\mathrm{A}i,j}$, operation type $C_{i,j}$, workload time $T_{i,j}$. The first operation of a job is initialized when the job is generated at time $T_{\mathrm{A}i}$ (i.e. $T_{\mathrm{A}i} = T_{\mathrm{A}i,1}$). The completion time and estimated completion time of a job are respectively denoted as $T_{\mathrm{C}i}$ and $\hat{T}_{\mathrm{C}i}$, and they are $T_{\mathrm{C}i,j}$ and $\hat{T}_{\mathrm{C}i,j}$ for operation $O_{i,j}$. The initialization time, completion time, and estimated completion time of the operations of job J_i are shown in Figure 1, where scheduling moments are marked by red circles. Job J_i is initialized at time $T_{\mathrm{A}i,1}$ and completed at time $T_{\mathrm{C}i,N_{\mathrm{O}_i}}$.

The total number of machines is denoted by N_{M}. The attributes of a machine include machine type C_m, machining speed factor K_{MT}, energy consumption factor K_{ME}. Each machine is equipped with a buffer, whose remaining length is $B_{\mathrm{L}m}$, for holding the unprocessed jobs.

Some constraints are set up to keep the shop floor operating robustly. First, the chosen machine is capable of undertaking the corresponding operations, that is C_m matches $C_{i,j}$. Second, the sequence of operations should be in accordance with designed requirements, that is $T_{Ai,j} \leqslant T_{Ci,j}$. Third, the number of tasks assigned to a machine should be no more than B_{Lm}. Fourth, the makespan of each order is supposed to satisfy the target completion time $\hat{T}_{Ci}$.

Jobs are generated in real time and are automatically added to the job list, ensuring that the actual machining conditions are met. As shown in Figure 2, each block represents an operation of a job marked by a time label with initialization time and completion time, and the boundaries of blocks in the job list represent scheduling moments. As the former operation of a job is completed, the next operation of this job is initialized and added to the operation list. The scheduler handles operations along the dash line in the operation list of Figure 2. For example, job J_1 is decomposed into two operations (i.e. $O_{1,1}$ and $O_{1,2}$) after process planning; operation $O_{1,1}$ is initialized at time $T_{A1,1}$ and completed at time $T_{C1,1}$; operation $O_{1,2}$ is initialized at time $T_{A1,2}$ and completed at time $T_{C1,2}$. The job list is sequenced by the generating time of jobs, and the operation list is sequenced by the initialization time of operations. Scheduling moments are at the initialization or completion time of each operation. When an operation is completed, the next operation of the same job will be initialized and appended to the operation list.

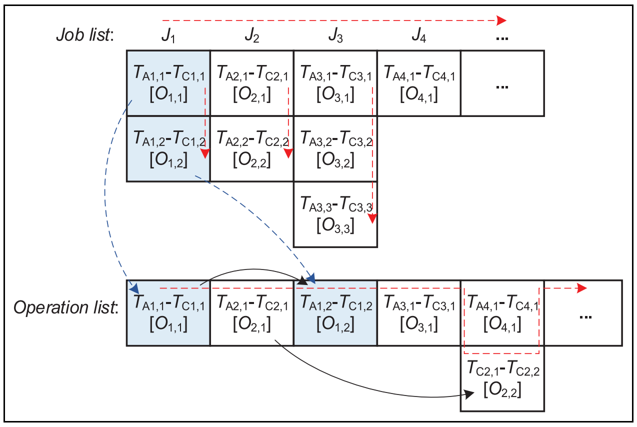

Job list:
J_1
J_2
J_3
J_4
...
$T_{A1,1}$-$T_{C1,1}$ [$O_{1,1}$]
$T_{A2,1}$-$T_{C2,1}$ [$O_{2,1}$]
$T_{A3,1}$-$T_{C3,1}$ [$O_{3,1}$]
$T_{A4,1}$-$T_{C4,1}$ [$O_{4,1}$]
$T_{A1,2}$-$T_{C1,2}$ [$O_{1,2}$]
$T_{A2,2}$-$T_{C2,2}$ [$O_{2,2}$]
$T_{A3,2}$-$T_{C3,2}$ [$O_{3,2}$]
$T_{A3,3}$-$T_{C3,3}$ [$O_{3,3}$]
Operation list:
$T_{A1,1}$-$T_{C1,1}$ [$O_{1,1}$]
$T_{A2,1}$-$T_{C2,1}$ [$O_{2,1}$]
$T_{A1,2}$-$T_{C1,2}$ [$O_{1,2}$]
$T_{A3,1}$-$T_{C3,1}$ [$O_{3,1}$]
$T_{A4,1}$-$T_{C4,1}$ [$O_{4,1}$]
$T_{C2,1}$-$T_{C2,2}$ [$O_{2,2}$]

Figure 2. Online scheduling with RL according to the operation list: $T_{A4,1} = T_{A2,2} = T_{C2,1}$.

RL for online scheduling

Reinforcement learning. RL is a series of algorithms to learn how to map situations to actions so as to maximize a numerical reward signal.[40] RL can be formalized as Markov decision process (MDP) which is a straightforward framework of learning from interactions with a dynamic environment to maximize the total reward. A typical RL model comprises two parts of agent and environment. The agent learns to choose an action a from the action space A according to the environment state s from the state space S. At a decision time step t, the agent takes an action a_t based on the observation s_t derived from the environment and receives a reward r_t. The environment state is updated to s_{t+1} after taking the action a_t. The agent learns a policy π that maps state s to action a, that is $a = \pi(s)$, from the interactions with the environment. The objective of an agent is to master an optimal policy π^* for achieving more discounted future rewards as follows:

$$G_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}, 0 \leqslant \gamma \leqslant 1 \tag{1}$$

where γ is the discount rate that denotes the contribution of future rewards to the current action value; T is the terminal time step.

Many algorithms are designed to obtain optimal policies of MDPs such as temporal difference (TD), dynamic programming (DP), and Monte Carlo (MC) methods. Like DP, TD methods update estimates individually or partly rather than waiting for an eventual result, which increases the efficiency of calculation. Like MC methods, TD methods interact directly with environments to acquire their statuses, which avoids depending on the transition possibility of models. Industrial environments need high stabilization, swift reaction, and low latency, so TD methods are appropriate for the operation of smart factories. Q-learning[41] is an off-policy TD algorithm that incorporates the advantages of DP and MC methods. The iterative equation of Q-learning is shown as follows:

$$q(s_t, a_t) \leftarrow q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a_{t+1}} q(s_{t+1}, a_{t+1}) - q(s_t, a_t)], 0 \leqslant \alpha \leqslant 1 \tag{2}$$

where α is the learning rate that denotes the iterative speed.

In a Q-learning algorithm, the maps from the state space to the action space are recorded in a Q-table. If the state space is so large, a neural network can be used to take place of the Q-table, that is Deep Q-Network (DQN) is invented.[42] The DQN has two neural networks: a target network and an evaluation network. The loss function of DQN, shown in equation (3), is used to update the parameters of the evaluation network. Parameters of the target network are copied from the evaluation network at every few steps.

$$L(\theta) = E[(\hat{y} - q)^2] \tag{3}$$

$$\hat{y} = r_{t+1} + \gamma \max_{a_{t+1}} \hat{q}(s_{t+1}, a_{t+1}) \tag{4}$$

where $\hat{q}$ is an action-value given by the target network, and q is an action-value given by the evaluation

Table 1. The relationships between RL and JSSP models.

RL	JSSP
Agent	Scheduler
Environment	Shop floor with machines and jobs
State	Attributes of machines and operations
Action	Scheduling or not
Reward	Optimization objectives

network. The iterative equation of DQN can be rewritten as follows:

$$q(s_t, a_t) \leftarrow q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a_{t+1}} \hat{q}(s_{t+1}, a_{t+1}) - q(s_t, a_t)] \tag{5}$$

From RL to job-shop scheduling problem. The precondition of solving scheduling problems with RL methods is turning scheduling problems into multi-step decision problems. In a job-shop scheduling problem, there are three main characters, including jobs, machines, and schedulers. Jobs are assigned to appropriate machines by schedulers. To implement online scheduling, schedulers should make decisions as soon as jobs are generated, which is a continuous single-step decision manner. Online scheduling is dependent on the current state of shop floors, and this scheduling approach turns a conventional JSSP into an RL model. The relationships between RL and JSSP models are listed in Table 1. The environment of a shop floor contains machines, material handlers, warehouses, etc. All jobs come from warehouses, and they will be brought back to them when the last operation is finished. Material handlers transport jobs between machines or between machines and warehouses. Machines play an important role in the processing of jobs.

State space of the shop floor. In a job-shop scheduling system, all machines are interconnected to build an environment for machining jobs and give back rewards for the scheduling actions. Each action can receive a reward, and all the rewards of a workpiece constitute a value for estimating the trace. A smart scheduler helps workpieces choose the appropriate machines according to the current state of the shop floor. The state space is made up of the attributes of schedulable operations and machines. The detailed state features of the shop-floor environment are listed in Table 2. At time step t, the state of operation $O_{i,j}$ is $(s_1, s_2, s_3, s_4)_{i,j}$, and the state of machine m is $(s_1, s_2, s_3, s_4, s_5)_m$. The integrated state is denoted as:

$$\mathbf{s}_t = \left[(s_1, ..., s_4)_{i,j}; (s_1, ..., s_5)_1, ..., (s_1, ..., s_5)_{N_\mathrm{M}}\right] \tag{6}$$

Table 2. State features of the shop-floor environment.

State types	$\mathbf{s}_t$	State features
Operation attributes	$(s_1)_{i,j}$	Operation type $C_{i,j}$
	$(s_2)_{i,j}$	Initialization time $T_{\mathrm{A}i,j}$
	$(s_3)_{i,j}$	Workload time $T_{i,j}$
	$(s_4)_{i,j}$	Target completion time $\hat{T}_{\mathrm{C}i,j}$
Machine attributes	$(s_1)_m$	Machine type C_m
	$(s_2)_m$	Machining speed factor $K_{\mathrm{MT}m}$
	$(s_3)_m$	Energy consumption factor $K_{\mathrm{ME}m}$
	$(s_4)_m$	Remaining workload time $T_{\mathrm{W}m}$
	$(s_5)_m$	Remaining buffer length $B_{\mathrm{L}m}$

Rewards for scheduling optimization

Evaluation criteria of manufacturing scheduling methods come from three aspects including productivity (e.g. time and flow efficiency), cost (e.g. resource consumption), and customer satisfaction (e.g. tardiness and quality). Most researches aim at only one objective that minimizes the makespan which belongs to the first aspect. Actually, resource consumption should be considered along with shortening the machining period, which not only meets customers' satisfaction but also saves manufacturing costs. Increasing utilization rate of machines or choosing high-efficiency machines can minimize the makespan of jobs. Generally, it is necessary to balance increasing efficiency and saving costs to get higher profits.

Shortening makespan. One of the optimization objectives is to minimize the mean tardiness rate of all jobs as follows:

$$\text{minimize } \overline{T} = \frac{1}{N_\mathrm{JC}} \sum_{i=1}^{N_\mathrm{JC}} \frac{T_{\mathrm{C}i} - \hat{T}_{\mathrm{C}i}}{\hat{T}_{\mathrm{C}i} - T_{\mathrm{A}i}} \tag{7}$$

where N_JC denotes the total number of completed jobs. The equation for calculating the target completion time of job J_i is:

$$\hat{T}_{\mathrm{C}i} = T_{\mathrm{A}i} + K_{\mathrm{D}i} \cdot \sum_{j=1}^{N_{\mathrm{O}i}} T_{i,j}, 1 \leq i \leq N_\mathrm{J} \text{ and } K_{\mathrm{D}i} > 0 \tag{8}$$

where $K_{\mathrm{D}i}$ is the urgency factor of job J_i.

Saving production costs. The other optimization objective is to minimize the mean consumption of resources. The amount of resource consumption rises along with the elevation of machining efficiency. The resource consumption $e_{i,j}$ of operation $O_{i,j}$ depends on the energy consumption factor $K_{\mathrm{ME}m}$ and actual machining time $K_{\mathrm{MT}m}\ T_{i,j}$, and its calculation equation is:

$$e_{i,j} = K_{\mathrm{ME}m} \cdot K_{\mathrm{MT}m} T_{i,j}, K_{\mathrm{ME}m} > 0 \text{ and } K_{\mathrm{MT}m} > 0 \tag{9}$$

Profits increase with the reduction of costs. Machines with higher speed can complete operations swiftly, but consume more resources. The prices of orders depend on the workload time $T_{i,j}$ and urgency factor $K_{\mathrm{D}i}$. We define the price $P_{\mathrm{R}i,j}$ of operation $O_{i,j}$ as:

$$P_{\mathrm{R}i,j} = K_{\mathrm{PR}i} T_{i,j} \tag{10}$$

where $K_{\mathrm{PR}i}$ is a price factor that increases with the decrease of urgency factor $K_{\mathrm{D}i}$.

The profit $P_{\mathrm{F}i,j}$ of operation $O_{i,j}$ can be calculated by equation (11).

$$\begin{aligned} P_{\mathrm{F}i,j} &= P_{\mathrm{R}i,j} - e_i \\ &= (K_{\mathrm{PR}i} - K_{\mathrm{ME}m} K_{\mathrm{MT}m}) T_{i,j} \end{aligned} \tag{11}$$

Increasing machine utilization rates. Different machines have different properties, and schedulers are willing to choose efficient machines that complete tasks quickly. If a large number of tasks are assigned to specific capable machines, it may cause system congestion and workload imbalance. Some bottleneck resources can also slow down the operating efficiency of manufacturing systems. It is necessary to balance the workloads among the targeted machines. The standard deviation of the utilization rates of machines of type c, shown in equation (12), is an important criterion of workload distribution.

$$U_c = \sqrt{\frac{1}{N_{\mathrm{M}c}} \sum_m (u_m - \bar{u}_c)^2}, \ \ C_m = c \tag{12}$$

where $N_{\mathrm{M}c}$ is the total number of machines of type c; u_m denotes the utilization rates of machine m (m = 1 and 2, …, N_{M}; N_{M} is the total number of machines); $\bar{u}_c$ is the mean value of the utilization rates of machines of type c.

The utilization rate of machine m can be calculated by equation (13).

$$u_m = \frac{1}{T} \sum_{n=1}^{N_m} T_n \tag{13}$$

where T denotes the production duration; N_m is the total number of operations completed by machine m; T_n is the machining time of the n th operation conducted on machine m.

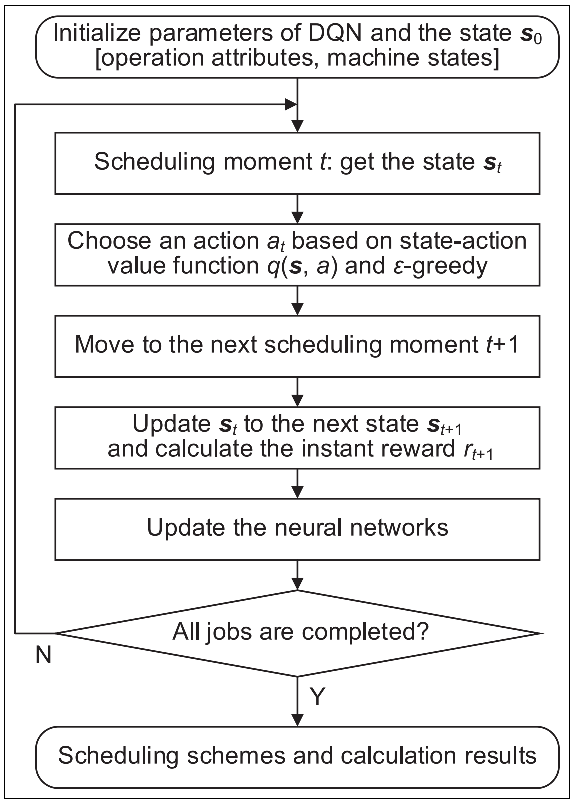
Composite reward functions. There are many evaluation criteria and optimization objectives on job-shop scheduling for different working conditions, including time, cost, machine utilization, and system capacity. The machining time of each workpiece is recorded. One of the important objectives is to minimizing the makespan to guarantee the delivery time of orders. Many methods aim to reduce the flow time of workpieces, such as improving machining efficiency and shortening waiting time. High-speed machines can complete tasks rapidly, but require higher costs. Low-speed machines consume lower equipment costs, but need more time for machining workpieces. A workload balance is supposed to be maintained among machines with different working abilities. The operating time and idle time of each machine are recorded to calculate the utilization rate of each machine, and a comparison is made among machines. It is necessary to balance workloads among machines and improve their utilization rates. The warehouse downloads orders from the order system and stores them in its buffers. If there are some spare targeted machines, the warehouse will assign tasks to them. If the shop floor is always busy and the task buffers are full of tasks all the time, the scheduling system can provide improvement suggestions to entrepreneurs.

According to the above criteria, the optimization objectives of job-shop scheduling problems mainly contain three types, including lead time, resource consumption, and machine utilization rates. Most researches on job-shop scheduling optimization are concerned with time parameters such as makespan, tardiness, switching time, deadline, etc. Therefore, under actual conditions, resource consumption should be taken into consideration as well. Thus, a multi-objective optimization measure is invited into the RL-based scheduling method, where makespan and resource consumption are considered together. Such approaches are not restricted to minimizing the makespan of orders to guarantee the target completion time, but they conform to actual machining conditions as well.

When a scheduler takes an action at time step t, it will get an immediate reward r_{t+1} which constitutes the value function. Optimization objectives are used to regulate the performances of schedulers and help them learn useful knowledge for enriching their experience. The reward r_{t+1} is a weighted value that is formed as equation (14).

$$r_{t+1} = \sum_{i=1}^{n} k_i \cdot r_{i,t+1}, \ \sum_{i=1}^{n} k_i = 1 \text{ and } k_i \geqslant 0 \tag{14}$$

where k_1, k_2, …, k_n are weights of rewards $r_{1,t+1}$, $r_{2,t+1}$, …, $r_{n,t+1}$ corresponding to different optimization objectives. The composite reward function includes rewards for time saving rate R_{D} in equation (15), energy saving rate R_{P} in equation (16), machine utilization rate R_{U} in equation (17), and workload distribution deviation R_{V} in equation (18). The composite reward R is



Initialize parameters of DQN and the state s_0 [operation attributes, machine states]
Scheduling moment t: get the state s_t
Choose an action a_t based on state-action value function $q(s, a)$ and ε-greedy
Move to the next scheduling moment t+1
Update s_t to the next state s_{t+1} and calculate the instant reward r_{t+1}
Update the neural networks
All jobs are completed?
N
Y
Scheduling schemes and calculation results

Figure 3. Online scheduling processes of the smart scheduler based on RL.

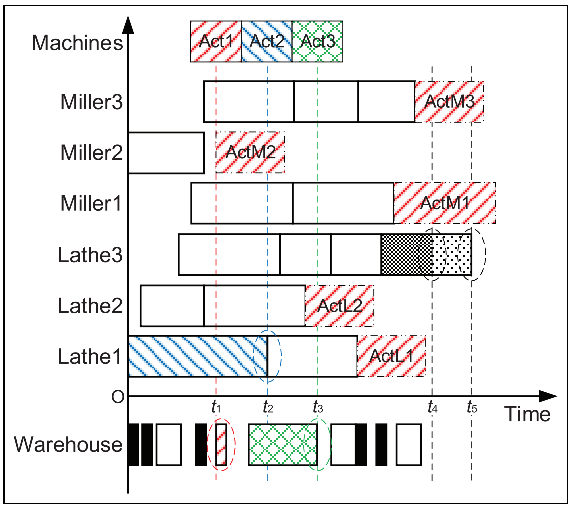


Figure 4. Action types of the smart scheduler.

defined as equation (19). To get denser rewards at scheduling moments, each operation can give back rewards according to order details.

$$R_{\mathrm{D}} = \left(\frac{\hat{T}_{Ci,j} - T_{Ci,j}}{\hat{T}_{Ci,j} - T_{Ai,j}}\right)^{-K_{\mathrm{D}i}}, \hat{T}_{Ci,j} = T_{Ai,j} + K_{\mathrm{D}i} T_{i,j} \quad (15)$$

$$R_{\mathrm{P}} = K_{\mathrm{PR}i} - K_{\mathrm{ME}m} K_{\mathrm{MT}m} \quad (16)$$

$$R_{\mathrm{U}} = \frac{1}{N_{\mathrm{M}c}} \sum_m u_m, \ \ C_m = c \quad (17)$$

$$R_{\mathrm{V}} = -U_c, \ \ C_m = c \quad (18)$$

$$R = K_1 \cdot R_{\mathrm{D}} + K_2 \cdot R_{\mathrm{P}} + K_3 \cdot R_{\mathrm{U}} + K_4 \cdot R_{\mathrm{V}} \quad (19)$$

The weights K_1, K_2, K_3, K_4 own the same value normally, but they differ in specific conditions. For example, if some urgent orders are generated or the deadline approaches, K_1 and K_3 increase as K_2 decreases. Thus, the reward for time saving rate R_{D} contributes more to the total weighted rewards and scheduling decisions are adjusted autonomously for adapting to the new manufacturing condition. If the remaining completion time of orders is so long, the weight K_2 can be raised slightly for saving energy.

Scheduling actions

Scheduling methods aim to achieve a balance between jobs and machines to obtain optimized combinations and sequences of these two resources. In general, the actions of a scheduler include two types of scheduling and machining, which can change the state of the shop floor. Detailed steps are shown in Figure 3 to illustrate the online scheduling processes based on DQN algorithms. The scheduler should take actions at scheduling moments, including the operation initialization moments and completion moments. At these moments, the scheduler collects all the states of related machines and makes decisions based on operation attributes and machine states. As a scheduling action is chosen, the job will be transported to the target machine. The states of machines are updated and a reward for the action is given. Between two scheduling moments, the scheduler and machines do nothing but process operations. If there is no operation to do or all the operations are finished, machines will be in a standby mode and take no action. If all machine buffers are occupied, scheduled jobs have to wait until a job is finished and a buffer is released. In these operation actions, the scheduler moves the timeline and updates all machine states to the next scheduling moment.

Regarding scheduling actions, when a new order is generated or the current operation is finished, the scheduler will choose a machine for its next operation. Apart from scheduling actions, the environment will not move until the operations on a machine are finished. The RL method simplifies scheduling models, and all conditions will not be ignored, as shown in Figure 4. Figure 4 is a real-time Gantt chart that is used for the explanation of action types. It is assumed that there are three millers, three lathes, and a

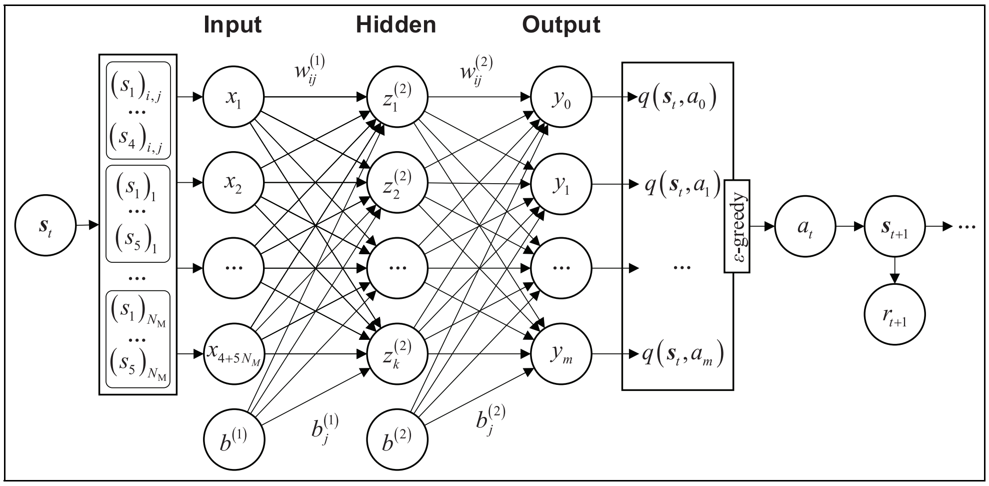


Figure 5. The neural network of DQN for getting a_t based on $\boldsymbol{s}_t$.

warehouse on the shop floor, and each machine has a buffer whose length is 4. The horizontal axis is a timeline and the vertical axis represents machines. Above the timeline, operations that have been scheduled are marked by rectangles in bold lines, and operations that will be scheduled are in double dot-dash lines. The operations waiting for being scheduled are below the timeline and the solid bricks represent that they are processed without delay. At time t_1, an operation is initialized at the shop floor. If it is a milling operation, it can be scheduled on Miller 2 and processed immediately, that is, ActM2. It can also choose to queue up behind the operation lists of Miller 1 or Miller 3. Miller 3 needs more waiting time than Miller 1, but it needs less machining time on the same operation. If it is a lathing operation, optional machines are Lathe 1 and Lathe 2. It cannot choose Lathe3 because the remaining buffer length of Lathe 3 is 0. At time t_1, a new operation in the warehouse is scheduled and arrives at the shop floor. At time t_2, the current operation on Lathe 2 is completed and scheduled for the next operation. At time t_3, an operation in the warehouse ends its waiting status and is scheduled to be machined. At time t_4, an operation on Lathe 3 is finished, but it has to remain still until t_5 when there is a vacant machine for its next operation. Between the adjacent scheduling moments, all the machines just process operations and the scheduler moves the timeline.

Neural networks are used to establish a mapping relationship between operation-machine states $\boldsymbol{s}_t$ and scheduling action a_t. As shown in Figure 5, the neural network is composed of an input layer, a hidden layer, an output layer, and each layer comprises several neurons. At a scheduling moment, the scheduler chooses an action from (a_0, a_1, a_2, …, a_m), where a_0 denotes waiting and a_1 to a_m respectively denote target machines. After being processed by the interconnected neurons, a tuple [$q(\boldsymbol{s}_t, a_0)$, $q(\boldsymbol{s}_t, a_1)$, $q(\boldsymbol{s}_t, a_2)$, …, $q(\boldsymbol{s}_t, a_m)$] is obtained, and the optimal action a_t is given by the maximum $q(\boldsymbol{s}_t, \ )$ in this tuple. Actions are chosen based on the ε-greedy policy, $\varepsilon \in [0, 1]$, that is, the optimal action is selected by the probability of ε, and a stochastic action is selected by the probability of 1 - ε. As the optimal machine is chosen for the current operation, the state space of the manufacturing system will be updated to $\boldsymbol{s}_{t+1}$, and a corresponding reward r_{t+1} is derived from the environment by a comprehensive evaluation system. The states of all machines are updated to provide real-time observations for subsequent tasks. The evaluation network is used only for getting the optimal scheduling policy instantly. A target network with previous parameters of the evaluation network is used to provide the next observation $\boldsymbol{s}_{t+1}$ for training the evaluation network.

Experimental design

Manufacturing resource specifications

Smart factory testbed: As shown in Figure 6, a testbed of the smart manufacturing system is established to verify the performances of the proposed methodology and other scheduling methods. Apart from the cloud, the shop floor includes a warehouse (m = 0), three millers (m = 1, 2, 3), three lathes (m = 4, 5, 6), an AGV (m = 7), and two robot arms (m = 8, 9). The buffer length of each machine is 4, and the total length of machine buffers is 24. The automated warehouse is used to place workpieces, and it has an exit numbered 0 and an entrance numbered 1. The AGV transports workpieces among the warehouse and machine buffers. When jobs enter the buffer of a machine, a robot arm

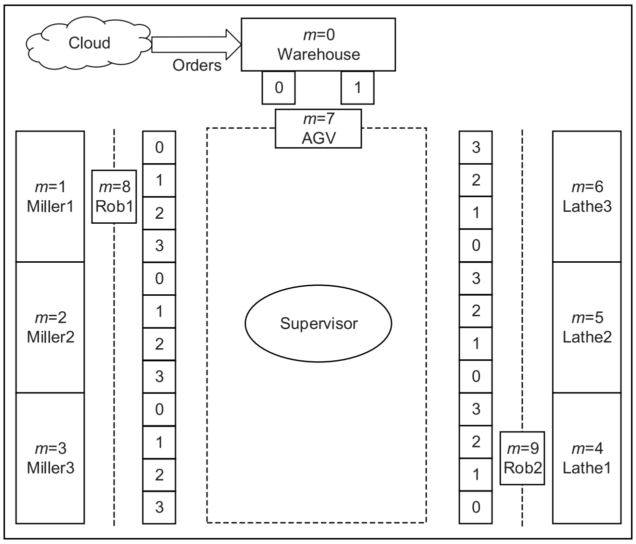


Figure 6. The layout of the smart manufacturing factory.

can transit it between the machine and its affiliated buffers. The supervisor coordinates all machines and gathers the operating data of them.

Order attributes: The RL-based scheduling system improves its decision-making abilities at any time by interacting with the environment, and a trained scheduler can schedule resources instantly. On the shop floor, two typical workpieces of shafts and panels are chosen as the products. A shaft has two operations of lathing and milling, and a panel has only one milling operation. As shown in Table 3, it is assumed that $K_{\mathrm{D}i} = 3$ (i.e. $P_{\mathrm{R}i} = 3$) represents a normal order, and $K_{\mathrm{D}i} = 2$ (i.e. $P_{\mathrm{R}i} = 4$) represents an urgent order.

Machine attributes: As shown in Table 4, there are six machines (i.e. $N_{\mathrm{M}} = 6$) in the smart factory including three lathes ($m = 1, 2, 3$) and three millers ($m = 4, 5, 6$). The former three machines do the lathing operations, while the latter three machines do the milling operations. Each machine has a buffer zone with a capacity of four jobs ($B_{\mathrm{L}m} = 4$) to hold the unprocessed or schedulable jobs. The machining speed of machines is depicted by the machining speed factor $K_{\mathrm{MT}m}$, and the energy consumption rate is depicted by the energy consumption factor $K_{\mathrm{ME}m}$.

Case studies for the scheduling model

Parameter specifications: Several case studies are conducted to test the overall performances of a smart scheduler. The parameters of DQN are set as: learning rate $\alpha = 0.01$, discount rate of rewards $\gamma = 0.9$, greedy rate $\varepsilon = 0.9$. According to equation (6), the state dimension for the dynamic attributes of operations and machines is 34. At a scheduling moment, the smart scheduler chooses one from six machines for processing the next operation, otherwise the operation waits for the next scheduling moment. The neural network has an input layer with 34 neurons, a hidden layer with 200 neurons, and an output layer with seven neurons.

Learning abilities: Orders contain different customer requirements, including product quality and delivery time. To validate the self-learning abilities of schedulers, several study cases are designed to test their performance. Four reward functions are designed to help schedulers operate effectively, including time saving rate R_{D}, energy saving rate R_{P}, machine utilization rate

Table 3. The details of orders for scheduling.

i	j	$T_{\mathrm{A}i,j}$ (s)	$C_{i,j}$	$T_{i,j}$ (s)	$K_{\mathrm{D}i}$	$\hat{T}_{\mathrm{C}i}$ (s)	$P_{\mathrm{R}i}$
1	1	10	Lathe	35	3	115	3
	2	-	Mill	12	3	151	3
2	1	45	Lathe	32	3	141	3
	2	-	Mill	10	3	171	3
3	1	70	Mill	43	2	156	4
4	1	95	Mill	38	3	209	3
5	1	125	Mill	47	3	266	3
6	1	150	Lathe	30	2	210	4
	2	–	Mill	14	2	238	4
7	1	175	Lathe	33	3	274	3
	2	–	Mill	15	3	319	3
8	1	190	Mill	50	2	290	4
9	1	200	Lathe	39	3	317	3
	2	–	Mill	17	3	368	3
10	1	210	Mill	29	3	297	3
11	1	235	Lathe	33	3	334	3
	2	–	Mill	13	3	373	3
12	1	260	Mill	44	2	348	4
13	1	270	Mill	38	3	384	3
14	1	295	Lathe	29	3	382	3
	2	–	Mill	10	3	412	3
15	1	320	Lathe	26	3	398	3
	2	–	Mill	9	3	425	3
16	1	350	Lathe	28	2	406	4
	2	–	Mill	10	2	426	4
17	1	370	Mill	35	3	475	3
18	1	385	Mill	37	3	496	3
19	1	430	Lathe	31	3	523	3
	2	–	Mill	16	3	571	3
20	1	495	Lathe	34	3	597	3
	2	–	Mill	17	3	648	3

Table 4. The inherent attributes of machines.

m	C_m	$B_{\mathrm{L}m}$	$K_{\mathrm{MT}m}$	$K_{\mathrm{ME}m}$
1	Lathe	4	0.5	2.0
2	Lathe	4	1.0	1.0
3	Lathe	4	2.0	0.5
4	Mill	4	0.5	2.0
5	Mill	4	1.0	1.0
6	Mill	4	2.0	0.5

R_U, workload distribution deviation R_V. These four types of rewards are adopted separately in the testing algorithms to validate the contribution of each component. A composite reward $R = (K_1, K_2, K_3, K_4)\cdot(R_D, R_P, R_U, R_V)$ is designed for building a more adaptive scheduler. Machining speed factor K_{MTm} and energy consumption factor K_{MEm} of each machine are set as the same value 1.0.

Scheduling abilities: As a scheduler has been trained, it can make advisable decisions on its own instantly. Therefore, orders are generated at stochastic time with different attributes. Thus, 30 orders of the same urgency factor $K_{Di} = 3$ are generated stochastically to validate the smart scheduler's scheduling performance for new tasks. The testing orders have the same format as the training orders shown in Table 3. The composite reward R is put into practice, and the factors K_{MTm} and K_{MEm} of each machine still keep the same value 1.0. Under real manufacturing conditions, more than one order may be generated at the same time. The smart scheduler can also give rational schedules for the simultaneously initialized operations. A study case is conducted by setting the initialization time of all orders with the same value.

Adaptive abilities: Unexpected tasks or events (e.g. urgent orders and machine failures) potentially exist in the operating and scheduling processes. Urgent orders, with urgency rate $K_{Di} = 2$, require shorter lead time. The scheduler should choose a machine with a shorter waiting time and higher efficiency. Sometimes, urgent operations will be moved to the head of the waiting list if it is necessary. The factors K_{MTm} and K_{MEm} of each machine are set with the values shown in Table 4. On traditional shop floors, a failure machine can stop the operation of the entire system, while the RL-based scheduling algorithm can handle such unexpected events without downtime. To validate the adaptive ability for machine failures, the fourth machine in Table 4 is shut down.

Experimental results and discussion

Learning performance of the smart scheduler

Composite reward functions are designed to help the smart scheduler improve its decision-making ability. Different reward functions bring different scheduling results, and combinations of reward functions may obtain better improvements than a single component. The composite reward R is constituted by rewards (R_D, R_P, R_U, R_V), and the components are respectively applied in the smart scheduler. The learning curves show the changing rates of the costs between $\hat{q}$ and q, which can be derived from equation (3). Each RL-based scheduling model has been trained for 300 episodes, and the learning curves of them are shown in Figure 7.

At the very beginning, the scheduler may have many attempts many steps to schedule the orders in an episode. With the accumulation of experience, its decision-making abilities improved a lot during these training episodes, and it can make scheduling decisions instantly with its rich experience. Figure 7(a) to (d) are learning curves of the model based on R_D, R_P, R_U, R_V. These four curves all converge within 300 episodes, and the convergence points of them are respectively near the episodes of 170, 240, 175, 120. Although machining speed and energy consumption are not considered in this study case, the learning curve of the R_P-based scheduler also converges with the help of the discount rate γ. The learning curves of the R_D, R_U, R_V-based schedulers show rapid convergence, and their detailed performances are elaborated in the following paragraphs. The learning curve of composite reward R is shown in Figure 7(e) and converges near episode 70. The heights of the curves are influenced by the weights (K_1, K_2, K_3, K_4) of the components.

After an order is completed, reward R_D can be calculated using equation (15). In most cases, it is advisable to shorten the lead time of orders to meet the completion time set by customers. Figure 8 shows the learning processes of an R_D-based scheduler. All the time saving rates of each episode are summed up, and all the total rates of the former 300 episodes are put together in Figure 8(a). According to the fitting curve, the total time saving rates increase from 11.51 to 12.97 by 12.69%. The Gantt chart of the scheduling results in the 300th episode is shown in Figure 8(b). The trained scheduler with reward R_D for time saving rates attempts to schedule operations on free machines for minimizing the lead time so that the target completion time is well guaranteed. Each operation is processed as soon as it is initialized, but the workload distribution of machines is unbalanced.

The reward R_D depends on the difference between the completion time and target completion time that is influenced by the efficiency and current state of machines. Another criterion showing the utilization rates of machines is the waiting time of jobs. The reward R_U helps shorten the waiting time of jobs by increasing the utilization rate of the target machine between two adjacent time steps. All the waiting time of jobs in each episode is summed up, and all the total values of the former 300 episodes are put together in Figure 9(a). The Gantt chart of the scheduling results in the 300th episode is shown in Figure 9(b). According to the fitting curve, the total waiting time decreases from 221.56 to 41.90 by 69.72%. The actual waiting time of the 300th episode is 0, which means that all the operations are conducted without delay.

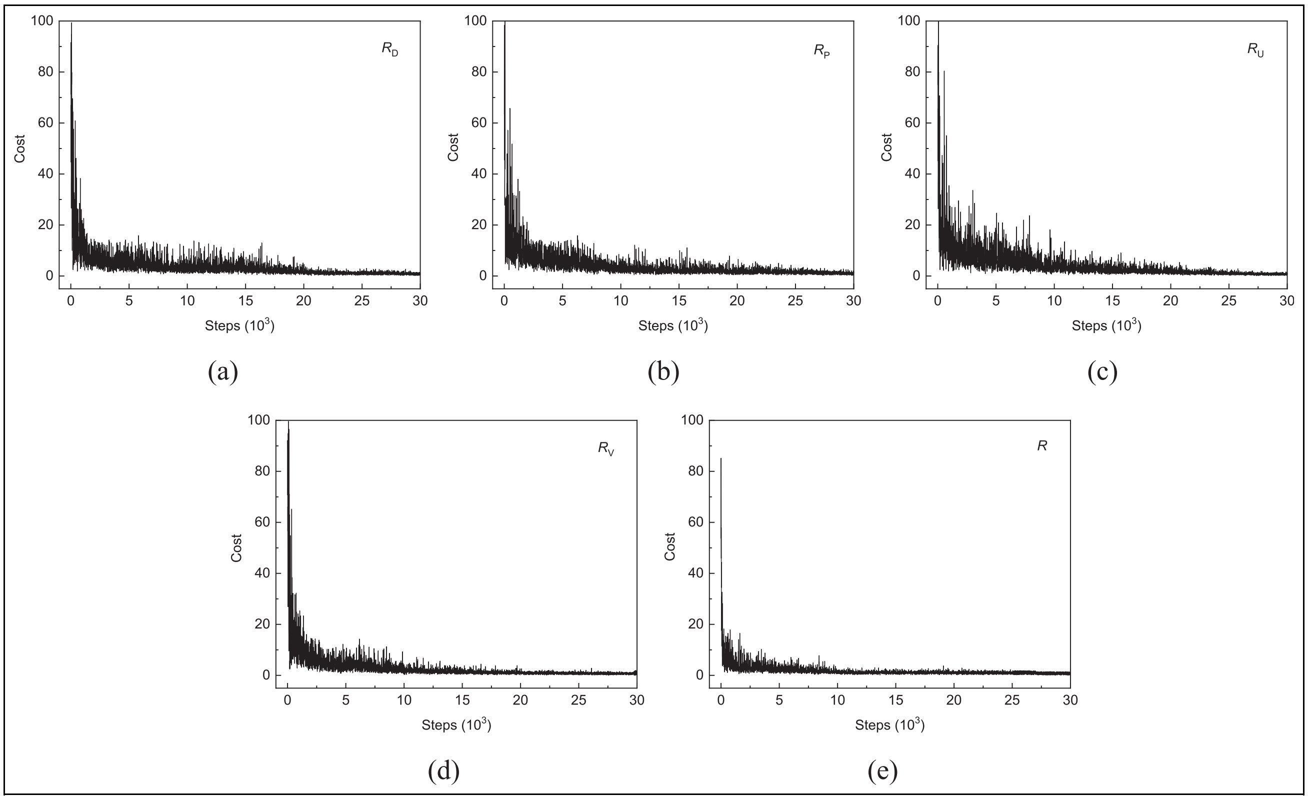


Figure 7. Learning curves of all the components of composite rewards: (a) reward for time saving rate, (b) reward for energy saving rate, (c) reward for machine utilization rate, (d) reward for workload distribution deviation, and (e) composite rewards.

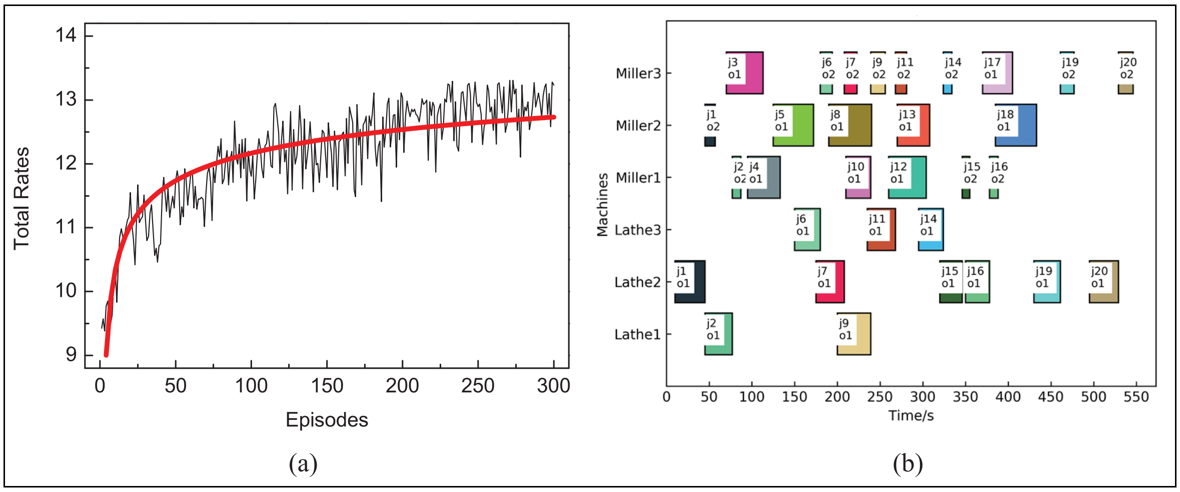


Figure 8. Learning processes of the scheduler based on time saving rates: (a) time saving rates and (b) scheduling processes of the scheduler.

The workload balance of machines is a significant criterion for evaluating the performance of shop floors. The standard deviations of the machine utilization rates, given by equation (12), is a useful parameter for illustrating the workload balance of machines. Standard deviations of machine operating time are respectively calculated on lathes and millers, and all the results of the former 300 episodes are shown in Figure 10(a). Both the fitting curves of lathes and millers show a decreasing trend. The standard deviations of lathe operating time decrease from 70.48 to 23.92 by 66.06%, and these of millers decrease from 92.82 to

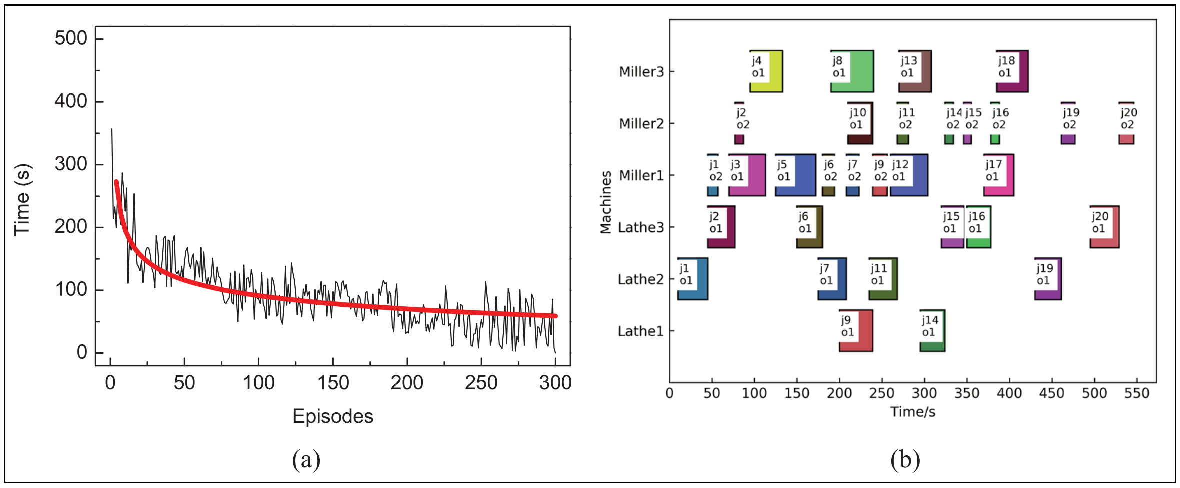


Figure 9. Learning processes of the scheduler based on machine utilization rates: (a) waiting time of jobs and (b) scheduling processes of the scheduler.

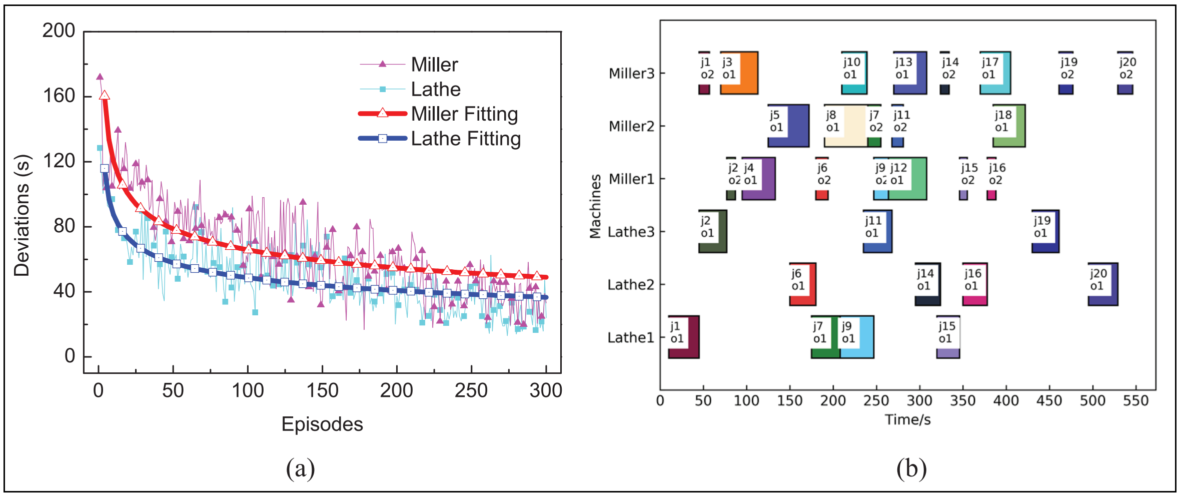


Figure 10. Learning processes of the scheduler based on workload distribution deviations: (a) the standard deviations of workloads and (b) scheduling processes of the scheduler.

30.34 by 67.31%. As shown in Figure 10(b), the scheduler trained with reward R_V gradually acquires the ability to balance the workloads among machines, but some operations still waste time waiting to be processed, because reward R_V pays more attention to global workload distribution rather than the performance of a single operation.

The composite reward R considers shortening the lead time and balancing workloads at the same time. It combines many optimization objectives to achieve global optimization of the shop floor. The scheduling results with the help of composite rewards are shown in Figure 11. Figure 11(a) is the Gantt chart of the untrained scheduler, and Figure 11(b) is the Gantt chart of the trained scheduler. The scheduler shows poor performances at the very beginning of training. After being trained by 300 episodes, its scheduling abilities have improved significantly. The standard deviations of operating time of lathes and millers respectively decrease from 62.47, 125.86 to 26.45, 62.76, by 57.66%–50.14%. The total waiting time of jobs decreased from 292 s to 0 by 100%. The total time saving rates of orders increase from 11.09 to 13.33 by 20.20%. Integrating the typical features of all components, the scheduler trained with composite rewards shows an overall good performance. The excellent performances are shown by the smart scheduler autonomously in real time, and the smart scheduler can perceive the states of the environment to make optimal schedules.

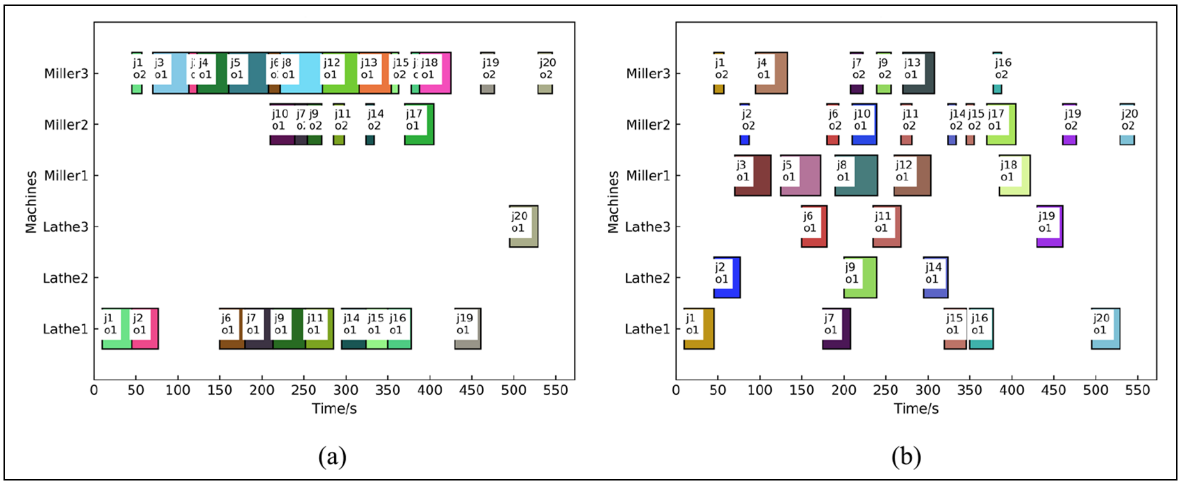


Figure 11. Scheduling results of the scheduler based on composite rewards: (a) results of the untrained scheduler and (b) results of the trained scheduler.

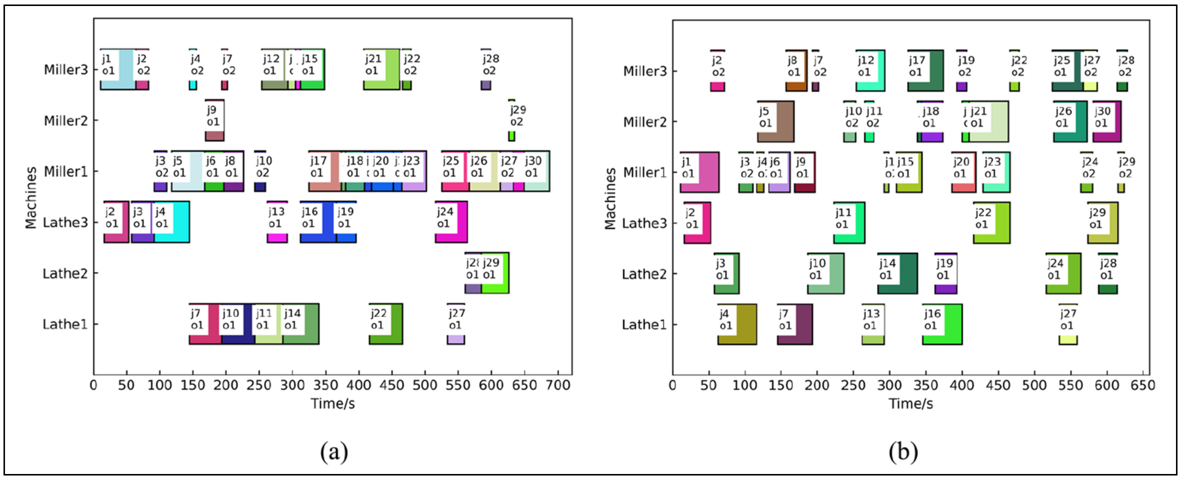


Figure 12. Scheduling performances for new orders of the scheduler based on composite rewards: (a) results of the untrained scheduler and (b) results of the trained scheduler.

Scheduling performance for resource variations

According to the above experimental results, the trained scheduler performs well in scheduling processes. Thirty new orders are created stochastically to evaluate the adaptive abilities of a trained scheduler. The smart scheduler is trained with composite rewards R. Gantt charts of Figure 12 show the comparison of the scheduling results between the untrained and trained schedulers. As the untrained scheduler (i.e. Figure 12(a)) turns into a well-trained scheduler (i.e. Figure 12(b)), the standard deviations of lathe operating time decrease from 100.15 to 28.64 by 71.40%, and these of millers decrease from 182.10 to 12.25 by 93.27%, the total waiting time decreases from 653 to 77 by 88.21%, and the total remaining deadline rates increase from 14.87 to 19.46 by 30.87%. Their detailed results are recorded in the first two rows of Table 5. Although the features and sequences are not the same as before, they are scheduled with good performance. It is obvious that a trained scheduler can schedule new tasks well with the knowledge accumulated during its training period in the past.

Machine failures are common in production processes, and such problems are inevitable and may disturb shop floors. The scheduler faces great challenges in dealing with these unexpected technical failures. According to the Gantt chart in Figure 13, the smart scheduler perceived the failure of Miller 1 and scheduled the following milling operations to the other normal millers. In comparison with Figure 12(b), utilization rates of the normal millers increased a lot while the actions of three lathes almost keep the same.

Table 5. Self-adaptive performances for new orders, failure machines, and simultaneous jobs.

Conditions	Deviation of operating time (s)		Waiting time (s)	Time saving rates
	Lathes	Millers		
Untrained scheduler	100.15	182.10	653	14.87
Trained scheduler	28.64 (−71.40%)	12.25 (−93.27%)	77 (−88.21%)	19.46 (+ 30.87%)
Machine failure	27.40 (−4.33%)	22.00 (+ 79.59%)	250 (+ 224.68%)	18.26 (−6.17%)
Simultaneous jobs	11.44	9.93	3065	16.87

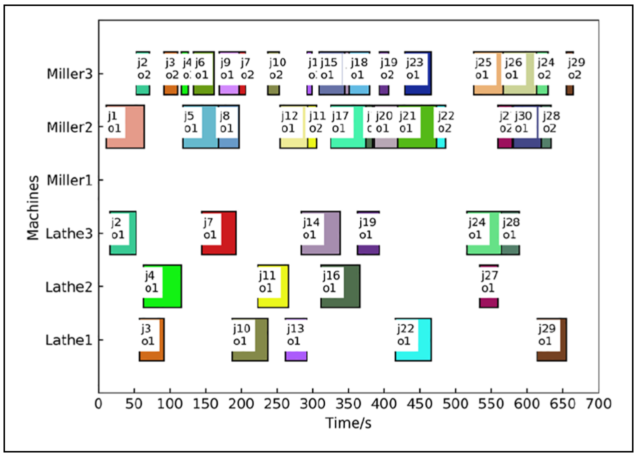


Figure 13. Scheduling performance for machine failures.

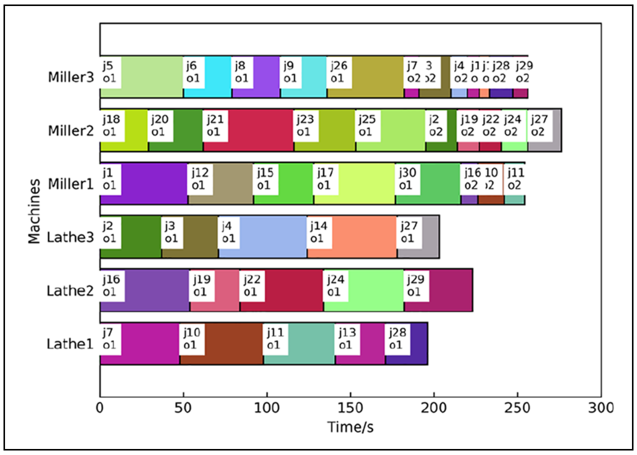


Figure 14. Scheduling for simultaneous orders.

Standard deviations of lathe utilization rates decrease from 26.86 to 27.40 by 4.33%, and these of millers increase from 12.25 to 22.00 by 79.59%. The total waiting time of jobs increases from 77 to 250 by 224.68%. The total time saving rates decrease from 19.46 to 18.26 by 6.17%. Detailed results are recorded in the third row of Table 5. Obviously, the smart scheduler can eliminate the effects of machine failures and keep the stability of the manufacturing system.

Large amounts of orders may be generated simultaneously. As shown in Figure 14, 30 orders are assumed to be generated at the same time. Jobs queue to be processed without delay, and the utilization rates of all machines are very high. The detailed scheduling results of simultaneous orders are recorded in the last row of Table 5. Although the total waiting time is so long, the target completion time can be guaranteed as well. Although the scheduler has not met such events before, it still keeps a preferable and robust performance in handling machine failures. Hence, the smart scheduler has good adaptability for scheduling different kinds of orders online according to the real-time statuses of machines on a shop floor.

Adaptive performance for different objectives

On most occasions, there are various machines with different machining abilities and efficiency on the shop floor. The efficiency of machines is denoted by the machining speed factor $K_{\mathrm{MT}m}$ in Table 4, and machines with smaller $K_{\mathrm{MT}m}$ can finish jobs more efficiently. All the testing orders in this section are obtained from Table 3. Only the reward R_{D} for time saving rates is adopted in the training process. Scheduling results considering order urgency and machine efficiency are shown in Figure 15. Figure 15(a) shows the operating time of each machine, and Figure 15(b) shows the scheduling actions of a trained scheduler. As the training continues, more orders are scheduled on Lathe 1 and Miller 1 than on the low-efficiency machines. The total operating time of lathes and millers decreases with the help of high-efficiency machines. In Figure 15(b), the

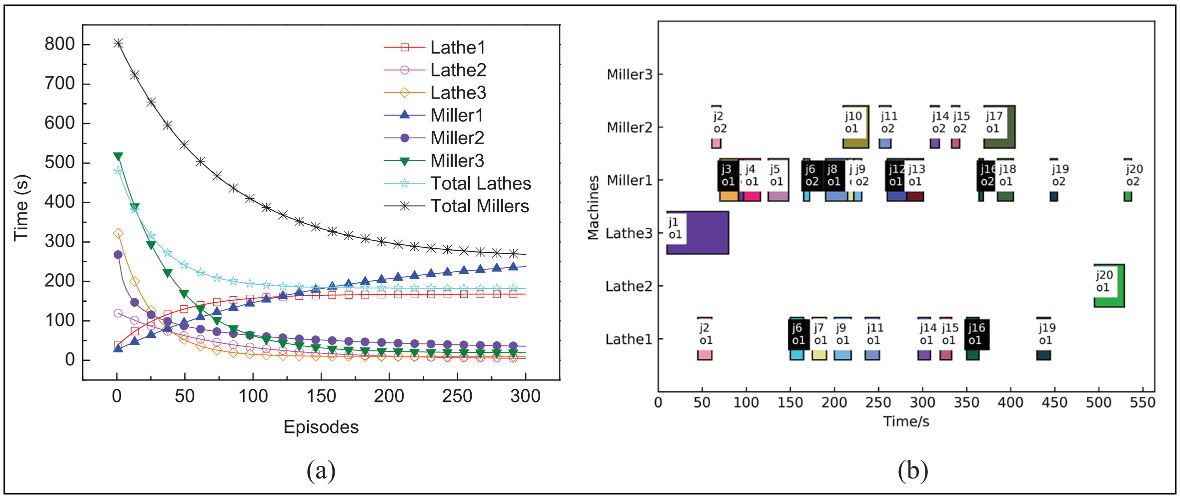


Figure 15. Scheduling results considering order urgency and machine efficiency: (a) operating time of machines and (b) dealing with urgent orders.

Table 6. Operating time of machines considering order urgency and machine efficiency.

Machines	Operating time (s)		Change rate
	Untrained scheduler	Trained scheduler	
Lathe 1	39.50	167.64	+ 324.41%
Lathe 2	119.70	5.06	−95.77%
Lathe 3	321.87	9.84	−96.94%
All lathes	481.07	182.54	−62.06%
Miller 1	27.93	237.63	+ 750.81%
Miller 2	256.84	11.60	−95.48%
Miller 3	519.21	19.47	−96.25%
All millers	803.98	268.70	−66.58%

Table 7. Machine profits considering resource consumption rates.

Machines	Profits (i.e. energy savings)		Change Rate
	Untrained scheduler	Trained scheduler	
Lathe 1	76.87	7.29	−90.52%
Lathe 2	310.59	260.50	−16.16%
Lathe 3	504.25	678.65	+ 34.59
Miller 1	83.18	65.38	−21.40%
Miller 2	700.72	666.78	−4.84%
Miller 3	515.15	713.59	+ 38.52
All Machines	2190.76	2392.20	+ 9.19

urgent operations are marked by white characters with black tags. The trained scheduler is capable of scheduling tasks on high-efficiency machines, and all urgent orders are scheduled on the most efficient machines autonomously.

As shown in Table 6, the total operating time of machines changes during the training period of the smart scheduler. The operating time of Lathe 1 and Miller 1 increases dramatically by 324.41%–750.81%. Meanwhile, the operating time of other machines decreases to some extent. The total utilization time of lathes and millers decreases by 62.06%–66.58%. The scheduling performance of the smart scheduler improved a lot during the training period considering order urgency and machine efficiency.

The production objectives of companies include not only improving operating efficiency but also saving costs to increase profits. The energy consumption factor $K_{\mathrm{ME}m}$, shown in Table 4, is considered separately at this time. Machines with smaller $K_{\mathrm{ME}m}$ consume fewer resources while processing the same operations. Only the reward R_{P} for energy saving rates is adopted in the training period. Scheduling results considering profits (i.e. energy savings) and energy consumption factors are shown in Figure 16. Figure 16(a) shows the profit of every machine, and Figure 16(b) shows the scheduling actions of a trained scheduler. As the training period continues, more orders are scheduled on Lathe 3 and Miller 3 than on machines with high resource consumption rates. The total profits contributed by lathes and millers increase continuously. In Figure 16(b), the urgent operations marked by white characters with black tags are neglected by the scheduler, because order urgency and machine efficiency are not considered in this case.

As shown in Table 7, the total profits of machines increase from 2190.76 to 2392.20 by 9.19% during the training period of the scheduler. Profit factors contributed by Lathe 3 and Miller 3 respectively increase from

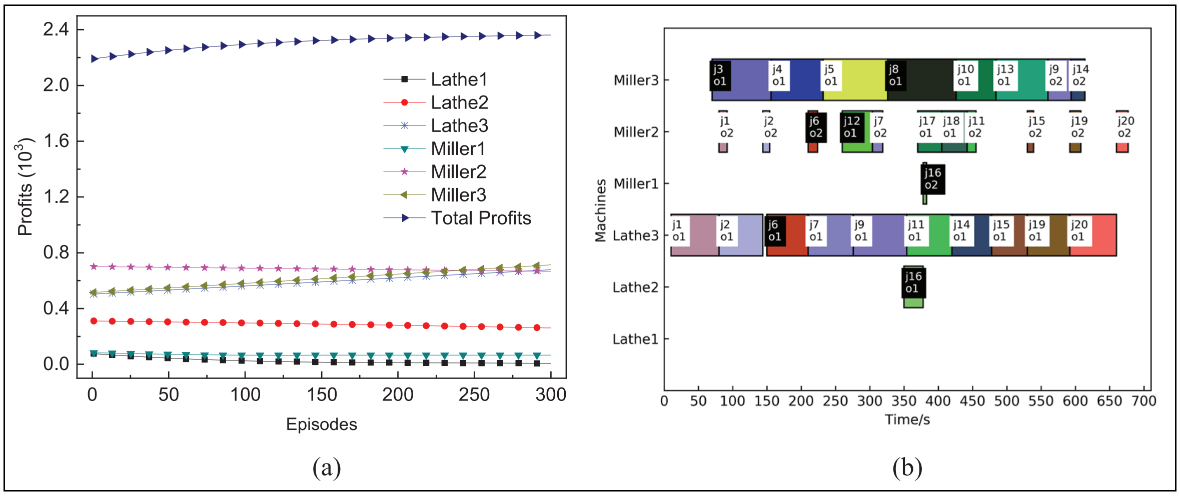


Figure 16. Scheduling results considering profits and energy consumption: (a) profits contributed by machines and (b) scheduling actions for saving energy.

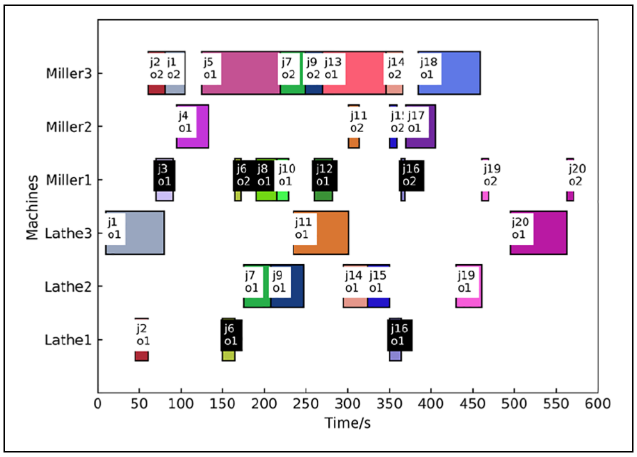


Figure 17. Scheduling results considering order urgency and resource consumption.

504.25, 515.15 to 678.65, 713.59 by 34.59%–38.52%. Meanwhile, profits from other machines show a certain reduction to some extent. Apparently, the trained scheduler considering energy saving rates is capable of scheduling operating on machines with lower energy consumption rates autonomously.

Scheduling performances of the scheduler considering machining efficiency and resource consumption has been separately verified in the previous parts of this section. In real manufacturing processes, efficiency and profits should be taken into consideration at the same time. In this part, these two criteria are considered simultaneously by considering all the components of composite reward R and initializing machines with parameters in Table 4. The order attributes are the same as before in this section. After 300 training episodes, the scheduler can schedule operations among machines in consideration of the attributes of orders and machines. The scheduling results considering order urgency and resource consumption are shown in Figure 17. In Figure 17, the urgent operations are marked by white characters with black tags. All the urgent orders are assigned on the most efficient machines, while most normal orders are autonomously placed on the machines with lower energy consumption rates. The composite rewards show excellent performance in optimizing the comprehensive scheduling actions.

Comparison of different scheduling methods

Most production scheduling methods aim at achieving time-related objectives that are concerned with the order sequence and completion time. RL-based scheduling with composite rewards (RL-C) is compared with some common online or offline scheduling methods. About 100 orders with 152 operations are generated stochastically to evaluate the scheduling performances of Genetic Algorithm (GA), Shortest Processing Time First (SPTF), First Come First Serve (FCFS), and RL-based methods. FCFS is a common rule-based method for dynamic manufacturing scheduling. GA is an offline heuristic method that can be used to obtain optimal schedules offline. Another offline scheduling method, SPTF, is adopted to obtain a schedule with the shortest waiting time in total. FCFS is an online method and schedules jobs according to their initialization time. A basic RL-based scheduling method (RL-B) is also adopted to make a comparison, while aiming to minimize the idle time of machines. The experimental algorithms belong to three typical categories of scheduling methods, including iterative optimization (i.e. GA), rule-based simulation (i.e.

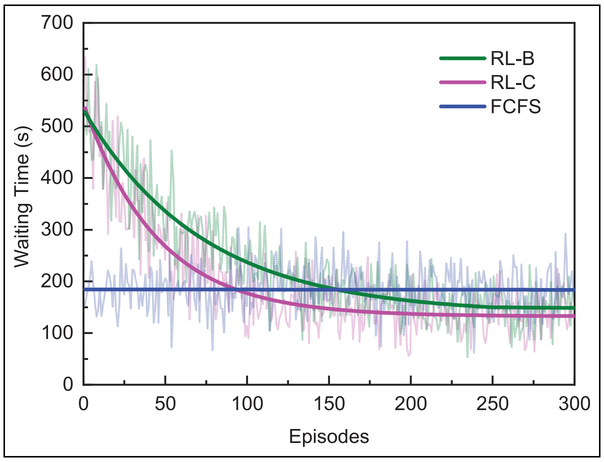


Figure 18. The learning performance of three online scheduling methods.

SPTF and FCFS), and AI-based self-learning (i.e. RL-B and RL-C).

The learning performances of three online scheduling methods (i.e. FCFS, RL-B, and RL-C) are shown in Figure 18, where the performance metric is the total waiting time of 100 orders in each episode. The waiting time of an order is from the scheduling completion moment to the machining start moment. FCFS schedules orders based on specific rules and is not capable of improving decision-making abilities during scheduling processes. At the very beginning, RL-based scheduling methods (i.e. RL-B and RL-C) learn to minimize the waiting time of orders in real time. The learning curves of RL-B and RL-C converge near episode 270 and episode 170. RL-C converges faster than RL-B and needs 37.0% fewer training episodes than RL-B. Therefore, the proposed RL-C scheduling method shows better learning performances than the traditional rule-based method and basic RL scheduling method.

GA and SPTF are offline scheduling method that reschedule the jobs when unexpected events (e.g. urgent orders and machine failures) happens in the manufacturing environment. FCFS, RL-B, and RL-C are online scheduling methods that schedule jobs and handle unexpected events in real time. The calculation time and makespan of each scheduling method are recorded in Table 8. GA can give the best schedules, but it takes a long time to get the result by a complex iteration period, and some unexpected events may cause an incredible increase in computing time. As 10 urgent orders are generated stochastically, online methods can handle the events well with a negligible increase in computing time and makespan. As a machine breaks down, all the scheduling methods can reschedule the operations on normal machines with an increase of makespan. As shown in Figure 19, the curves of RL-C nearly coincide with the results of GA and are more robust than other scheduling methods. Composite rewards improve the online decision-making abilities of the smart scheduler.

Conclusions and future work

This paper presents a smart scheduler for online scheduling low-volume-high-mix orders in a smart manufacturing factory. RL algorithms equip the smart scheduler with the abilities of self-organization, self-learning, and self-adaptation. A new composite reward function enables the smart scheduler for online optimization of multiple scheduling objectives. A series of experiments are performed to evaluate how the reward functions (i.e. R_D, R_P, R_U, R_V) influence the learning performances and optimize the scheduling performances. Experimental results show that the proposed scheduling model not only effectively optimizes scheduling policies but also handles unexpected events such as simultaneously generated orders, urgent orders, and machine failures.

Low-volume-high-mix orders and unexpected events bring a large number of uncertainties to real-world production environment. RL and composite reward functions help the smart scheduler learn autonomously to schedule manufacturing resources with dynamic features in real-time. The smart scheduler can adapt to different real case situations by optimizing the components of the composite reward function. Therefore, engineers do not need to spend much time rescheduling resources when the attributes of orders or machines vary in the dynamic manufacturing environment, which reduces the human costs, computing resources, and lead time.

Real production processes are more complex than experimental environments, and various factors (e.g. transition time and transportation optimization) existing in real working conditions should be considered in

Table 8. Comparison of different scheduling methods (values: computing time/makespan; units: s).

	GA	SPTF	FCFS	RL-B	RL-C
Normal	431/983	15/1034	12/1095	9/1001	7/990
Urgent Order	496/1002	21/1130	16/1208	11/1046	7/1014
Machine Failure	532/1310	23/1630	17/1703	15/1452	8/1347

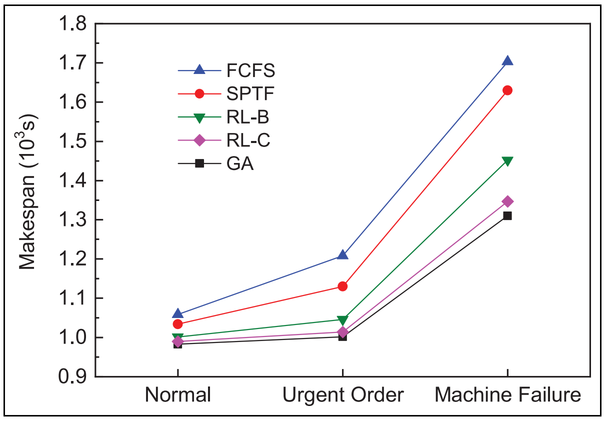


Figure 19. Comparison of the makespan of different scheduling methods.

future researches. To share the computing tasks with the centralized scheduler, the researches on improving the decision-making abilities of each machine are being conducted by the authors of this paper. In addition, we will delve into how to optimize the framework and parameters of DQN or other RL-based scheduling algorithms.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: The National Key Research and Development Program of China (No. 2018YFE0117000), the National Natural Science Foundation of China (No. 52075257), the Fundamental Research Funds for the Central Universities (No. NS2021070), and the Key Research and Development Program of Jiangsu Province (No. BE2021091).

ORCID iDs

Tong Zhou https://orcid.org/0000-0002-8311-0058
Changchun Liu https://orcid.org/0000-0001-8854-6136
Wei Shi https://orcid.org/0000-0002-9780-2629

References

1. Mehrabi MG, Ulsoy AG, Koren Y, et al. Trends and perspectives in flexible and reconfigurable manufacturing systems. *J Intell Manuf* 2002; 13: 135–146.
2. Zaeh MF and Ostgathe M. A multi-agent-supported, product-based production control. In: *2009 IEEE international conference on control and automation*, Christchurch, New Zealand, 2009, pp.2376–2383. IEEE.
3. Bannat A, Bautze T, Beetz M, et al. Artificial cognition in production systems. *IEEE Trans Autom Sci Eng* 2011; 8: 148–174.
4. Leitão P and Restivo F. A holonic approach to dynamic manufacturing scheduling. *Robot Comput Integr Manuf* 2008; 24: 625–634.
5. Yuan M, Li Y, Zhang L, et al. Research on intelligent workshop resource scheduling method based on improved NSGA-II algorithm. *Robot Comput Integr Manuf* 2021; 71: 102141.
6. He Y and Stecke KE. Simultaneous part input sequencing and robot scheduling for mass customisation. *Int J Prod Res* 2021; 59: 1–16.
7. Luo S, Zhang L and Fan Y. Dynamic multi-objective scheduling for flexible job shop by deep reinforcement learning. *Comput Ind Eng* 2021; 159: 107489.
8. Shen W. Distributed manufacturing scheduling using intelligent agents. *IEEE Intell Syst* 2002; 17: 88–94.
9. Gershwin SB. Hierarchical flow control: a framework for scheduling and planning discrete events in manufacturing systems. *Proc IEEE* 1989; 77: 195–209.
10. Webster S and Azizoglu M. Dynamic programming algorithms for scheduling parallel machines with family setup times. *Comput Oper Res* 2001; 28: 127–137.
11. Kutanoglu E and Sabuncuoglu I. Routing-based reactive scheduling policies for machine failures in dynamic job shops. *Int J Prod Res* 2001; 39: 3141–3158.
12. Sabuncuoglu I. A study of scheduling rules of flexible manufacturing systems: a simulation approach. *Int J Prod Res* 1998; 36: 527–546.
13. Park J, Mei Y, Nguyen S, et al. An investigation of ensemble combination schemes for genetic programming based hyper-heuristic approaches to dynamic job shop scheduling. *Appl Soft Comput* 2018; 63: 72–86.
14. Ruiz D, Cantón J, Maŕia Nougués J, et al. On-line fault diagnosis system support for reactive scheduling in multipurpose batch chemical plants. *Comput Chem Eng* 2001; 25: 829–837.
15. Metaxiotis KS, Askounis D and Psarras J. Expert systems in production planning and scheduling: a state-of-the-art survey. *J Intell Manuf* 2002; 13: 253–260.
16. Zhou H, Feng Y and Han L. The hybrid heuristic genetic algorithm for job shop scheduling. *Comput Ind Eng* 2001; 40: 191–200.
17. Lin TL, Horng SJ, Kao TW, et al. An efficient job-shop scheduling algorithm based on particle swarm optimization. *Expert Syst Appl* 2010; 37: 2629–2636.
18. Reisen K, Teschemacher U, Niehues M, et al. Biomimetics in production organization: a literature study and framework. *J Bionic Eng* 2016; 13: 200–212.
19. Hou Y, Fu Y, Gao K, et al. Modelling and optimization of integrated distributed flow shop scheduling and distribution problems with time windows. *Expert Syst Appl* 2022; 187: 115827.
20. Fu Y, Zhou M, Guo X, et al. Scheduling dual-objective stochastic hybrid flow shop with deteriorating jobs via Bi-population evolutionary algorithm. *IEEE Trans Syst Man Cybern Syst* 2020; 50: 5037–5048.
21. Wang C and Jiang P. Manifold learning based rescheduling decision mechanism for recessive disturbances in

RFID-driven job shops. *J Intell Manuf* 2018; 29: 1485–1500.

22. Park HS and Tran NH. An autonomous manufacturing system based on swarm of cognitive agents. *J Manuf Syst* 2012; 31: 337–348.
23. Jiang Z, Yuan S, Ma J, et al. The evolution of production scheduling from Industry 3.0 through industry 4.0. *Int J Prod Res* 2021; 59: 1–21.
24. Shen W and Norrie DH. Agent-based systems for intelligent manufacturing: a state-of-the-art survey. *Knowl Inf Syst* 1999; 1: 129–156.
25. Wang S, Wan J, Zhang D, et al. Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination. *Comput Netw* 2016; 101: 158–168.
26. Wong TN, Leung CW, Mak KL, et al. Dynamic shopfloor scheduling in multi-agent manufacturing systems. *Expert Syst Appl* 2006; 31: 486–494.
27. Li K, Zhou T, Liu BH, et al. A multi-agent system for sharing distributed manufacturing resources. *Expert Syst Appl* 2018; 99: 32–43.
28. Yeung WL. Agent-based manufacturing control based on distributed bid selection and publish-subscribe messaging: a simulation case study. *Int J Prod Res* 2012; 50: 6339–6356.
29. Zhang Y, Wang J and Liu Y. Game theory based real-time multi-objective flexible job shop scheduling considering environmental impact. *J Clean Prod* 2017; 167: 665–679.
30. Wang LC, Cheng CY and Lin SK. Distributed feedback control algorithm in an auction-based manufacturing planning and control system. *Int J Prod Res* 2013; 51: 2667–2679.
31. Hsu CY, Kao BR, Ho VL, et al. Agent-based fuzzy constraint-directed negotiation mechanism for distributed job shop scheduling. *Eng Appl Artif Intell* 2016; 53: 140–154.
32. Fu Y, Hou Y, Wang Z, et al. Distributed scheduling problems in intelligent manufacturing systems. *Tsinghua Sci Technol* 2021; 26: 625–645.
33. Silver D, Huang A, Maddison CJ, et al. Mastering the game of go with deep neural networks and tree search. *Nature* 2016; 529: 484–489.
34. Silver D, Schrittwieser J, Simonyan K, et al. Mastering the game of Go without human knowledge. *Nature* 2017; 550: 354–359.
35. Zhang Z, Zheng L, Li N, et al. Minimizing mean weighted tardiness in unrelated parallel machine scheduling with reinforcement learning. *Comput Oper Res* 2012; 39: 1315–1324.
36. Shiue YR, Lee KC and Su CT. Real-time scheduling for a smart factory using a reinforcement learning approach. *Comput Ind Eng* 2018; 125: 604–614.
37. Shahrabi J, Adibi MA and Mahootchi M. A reinforcement learning approach to parameter estimation in dynamic job shop scheduling. *Comput Ind Eng* 2017; 110: 75–82.
38. Kardos C, Laflamme C, Gallina V, et al. Dynamic scheduling in a job-shop production system with reinforcement learning. *Procedia CIRP* 2021; 97: 104–109.
39. Wang Y, Liu H, Zheng W, et al. Multi-objective workflow scheduling with deep-Q-network-based multi-agent reinforcement learning. *IEEE Access* 2019; 7: 39974–39982.
40. Sutton RS and Barto AG. *Reinforcement learning: an introduction*. Cambridge, MA: MIT Press, 2018.
41. Watkins CJCH. *Learning from delayed rewards*. Ph.D. Thesis, King's College, London, UK, 1989.
42. Mnih V, Kavukcuoglu K, Silver D, et al. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:13125602 2013*.