Deep Reinforcement Learning-Based Dynamic Reconfiguration Planning for Digital Twin-Driven Smart Manufacturing Systems With Reconfigurable Machine Tools

Jintang Huang, Sihan Huang, *Member, IEEE*, Shokraneh K. Moghaddam, Yuqian Lu, *Member, IEEE*, Guoxin Wang, Yan Yan, and Xuejiang Shi

***Abstract*—Smart manufacturing systems are a new paradigm in Industry 4.0 driven by the emerging information and communication technology and artificial intelligence that converge to digital twin, which are able to perceive, recognize, and handle the changes in demand and production. Reconfigurable machine tools (RMTs) can promote the flexibility of smart manufacturing systems. The fundamental problem lies in dynamically reconfiguring the RMTs in smart manufacturing systems efficiently and accurately by considering the flexibility of production precedence and operation sequences simultaneously. Therefore, in this article, a deep reinforcement learning-based reconfiguration planning method of digital twin-driven smart manufacturing systems with RMT is proposed to seek optimal reconfiguration policy online. The reconfiguration processes of smart manufacturing systems are modeled by considering reconfiguration cost, moving cost, and processing cost. Deep *Q*-network is adopted to explore the state space and action space to find the optimal reconfiguration scheme with the highest return. An industry case study is presented to demonstrate the effectiveness and efficiency of the proposed method, where the reconfiguration processes of a smart manufacturing system consisting of five RMTs for producing four parts are discussed.**

***Index Terms*—Deep reinforcement learning, digital twin, Industry 4.0, reconfigurable machine tools (RMTs), reconfiguration planning, smart manufacturing systems.**

Manuscript received 26 February 2024; revised 25 May 2024; accepted 8 July 2024. This work was supported in part by the Beijing Institute of Technology Research Fund Program for Young Scholars, Beijing Natural Science Foundation under Grant L243009, in part by the National Key Research and Development Program of China under Grant 2021YFB1716201, and in part by the National Natural Science Foundation of China under Grant 51975056. Paper no. TII-24-0857.*(Corresponding author: Sihan Huang.)*

Jintang Huang, Sihan Huang, Guoxin Wang, and Yan Yan are with the School of Mechanical Engineering, Beijing Institute of Technology, Beijing 100081, China, and also with the Key Laboratory of Industry Knowledge and Data Fusion Technology and Application, Ministry of Industry and Information Technology, Beijing Institute of Technology, Beijing 100081, China (e-mail: huangjt0119@163.com; hsh@bit.edu.cn; wangguoxin@bit.edu.cn; yanyan331@bit.edu.cn).

Shokraneh K. Moghaddam is with the School of Physics, Engineering, and Computer Science, University of Hertfordshire, Hatfield AL10 9AB, U.K. (e-mail: s.khashkhashimoghadam@herts.ac.uk).

Yuqian Lu is with the Department of Mechanical and Mechatronics Engineering, The University of Auckland, Auckland 1142, New Zealand (e-mail: yuqian.lu@auckland.ac.nz).

Xuejiang Shi is with Hongyun Honghe Tobacco Group, Kunming 650231, China (e-mail: 81321763@qq.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2024.3431095.

Digital Object Identifier 10.1109/TII.2024.3431095

I. Introduction

INDUSTRY 4.0, driven by emerging information and communication technology (ICT) and artificial intelligence (AI), requires a new paradigm of manufacturing systems that can respond in real time to meet the changing demands and conditions in factories and supply networks and satisfy varying customer needs, that is, smart manufacturing systems [1], [2], [3]. The responsiveness of smart manufacturing systems in case of demand fluctuation is fundamentally determined by their changeability and reconfigurability, which can be enabled by reconfigurable machine tools (RMTs) [4], [5], [6]. Digital twin as the synthesis of the emerging ICT and AI is one of the core enabling technologies of smart manufacturing systems with RMT to promote production efficiency and accuracy through seamless interaction between virtual space and physical space [7], [8], [9], which digital twin model should be capable of reconfigurability as well [10].

The concept of a reconfigurable manufacturing system (RMS) to explore reconfiguration in manufacturing systems was proposed by Koren et al. [11], where RMT is the fundamental equipment of RMS to promote its responsiveness [12]. RMS is a promising manufacturing paradigm aiming at providing exactly the functionality and capacity needed and exactly when needed [13]. Smart manufacturing systems with reconfigurability are the evolved version of RMS to some extent in Industry 4.0 era [14], [15], [16], moving toward smart RMSs. Lee and Ryu [17] proposed smart, self-reconfigurable manufacturing system and discussed its architecture and key features. Reconfiguration planning, i.e., how to reconfigure a manufacturing system, is the key problem to be solved when demand changes. Reconfiguration planning is the key phase to successfully implementing RMS or smart manufacturing systems with reconfigurability. Many researchers investigated reconfiguration planning.

Typically, this problem has been studied from two main perspectives, including scalability and convertibility [18], [19]. Koren et al. [20] studied the scalability planning problem of RMS to maximize system throughput after reconfiguration. Also, Koren et al. [21] discussed the principles, design, and development trends of RMS. Ma et al. [22] proposed a bilevel coordinated optimization model solved by a nesting genetic algorithm to address the reconfigurable process planning problem. Azab and ElMaraghy [23] put forward mathematical modeling for reconfigurable process planning in which a process plan reconfiguration index that captures the extent of changes in the plan and their implications has been introduced in this study. Recently, the production mode is shifting from mass customization to mass personalization [24]. Convertibility of manufacturing systems should, hence, be of more importance in order to meet the individual customization demand. Huang et al. [13] concerned convertibility enhancement of RMS through delayed reconfiguration. Machine-level changeovers [25] provided a great option to realize convertibility rapidly and cost effectively. Gadalla and Xue [26] proposed an approach to identify optimal reconfiguration processes by considering the configuration variation of RMT. Leng et al. [27] introduced a digital twin-driven rapid reconfiguration method of smart manufacturing systems via RMT based on open architecture. Touzout and Benyoucef [28] proposed a multiobjective model for optimizing the reconfiguration process plan by considering the configuration changes of RMT, where an iterative multiobjective integer linear programming approach is developed to solve this model. Similarly, Khezri et al. [29] attempted to generate a reconfiguration process plan by minimizing sustainability-metric value, total production time, and total production cost using heuristic algorithms. Liu et al. [30] proposed a distributed reconfiguration planning algorithm by considering modular robots that share similar characteristics of RMT. Due to the narrower production time window and higher demand uncertainty, it is necessary and important to effectively handle dynamic arrival production tasks in a dynamic environment [31]. However, most of the existing studies focus on the offline method to figure out the reconfiguration planning problem of manufacturing systems resulting in static solutions generally. They fail to perceive the real-time status of smart manufacturing systems with RMT and provide optimal solutions for reconfiguration planning dynamically. It is significant to update reconfiguration planning method and promote the production accuracy and efficiency of smart manufacturing systems with RMT at a high level. In addition, process planning [32], scheduling [33], capacity planning [34], and other points should be considered when optimizing the performance of smart manufacturing systems. The single-point optimization that largely exists in the literature is gradually unable to meet the new demands for more and more complex smart manufacturing systems.

To cope with the urgent challenges, a dynamic optimization model for reconfiguration planning of smart manufacturing systems with RMT using deep reinforcement learning is proposed in this article, where the corresponding digital twin scenario is constructed based on the *Unity 3-D* platform to manifest the power of the proposed method. The proposed method attempts to fill the above-mentioned gap by optimizing the reconfiguration planning of smart manufacturing systems with RMT by considering production flexibility and operation flexibility simultaneously. Traditionally, discrete optimization algorithms (i.e., genetic algorithm [35], simulated annealing [36], particle swarm optimization [37], etc.) can solve production planning and reconfiguration problem well, which is an offline mode and cannot meet the requirement of smart manufacturing systems. Due to its ability to dynamically interact with the environment during optimization, deep reinforcement learning shows the potential to shift reconfiguration planning from offline mode to online mode. Also, this study actively explores the effective reconfiguration planning method to catch up with the rapidly developing speed of new technologies and accommodate new application scenarios in Industry 4.0 era. So, deep reinforcement learning plays a crucial role in this article. As we know, reinforcement learning [38], [39], [40] is the problem faced by an agent that learns behavior through trial-and-error interactions with environment, which is learning what to do—how to map situations to actions—so as to maximize reward [41]. Recently, deep learning has been prevailing in reinforcement learning in recent years to scale to decision-making problems with high-dimensional state and action spaces [42], [43], [44]. The application of reinforcement learning is becoming popular in the manufacturing domain. Wang et al. [40] adopted reinforcement learning to optimize the energy efficiency of Industrial Internet of Things that can promote management efficiency alongside with Industrial Internet platforms [45]. Yang and Xu [33] used deep reinforcement learning to solve the scheduling and reconfiguration model in smart manufacturing. Epureanu et al. [46] proposed a self-repair method based on deep reinforcement learning for smart manufacturing systems to find optimal strategy by considering system status and performance. Bakopoulos et al. [47] studied the production scheduling based on deep reinforcement learning under the framework of digital twin.

In summary, the main contributions of this article are as follows.

1) A digital twin of smart manufacturing systems with RMTs is constructed to explore the online optimization of reconfiguration planning by recognizing demand changes in time to support the realization of dynamic reconfiguration.
2) The reconfiguration processes of smart manufacturing systems with RMT are directly modeled based on Markov decision processes (MDP) to construct an online optimization environment, where the flexibility of production precedence and operation sequences is included to promote the accuracy of the reconfiguration scheme.
3) A deep reinforcement learning algorithm named deep Q-network (DQN) is adopted to dynamically search the optimization reconfiguration scheme of smart manufacturing systems with RMT, in which the capability of DQN to efficiently solve high-dimensional problems can equip the proposed optimization model with high adaptability for handling dynamic demand fluctuation.

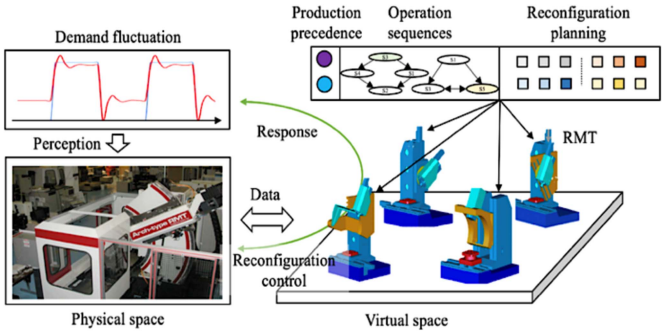


Fig. 1. Digital twin-driven smart manufacturing system based on RMT.

The rest of this article is organized as follows. Section II analyzes the problem to be solved of digital twin-driven smart manufacturing systems with RMT. The reconfiguration process modeling based on MDP and deep reinforcement learning-based optimization is elaborated in Section III. In Section IV, we provide a case study to demonstrate the effectiveness of the proposed method. Finally, Section V concludes this article.

II. Problem Statement

Smart manufacturing systems are expected to be aware of internal and external changes and respond rapidly and accurately. This requires the application of emerging ICT based on the flexibility of their physical structure (flexible layout, changeable machine tools, etc.). RMT is a typical machine tool that can change its functionality or capacity through configuration adaption to meet the demand fluctuation. A smart manufacturing system consisting of a group of RMTs can ensure its flexibility in a relatively mild way without requiring system layout adjustment, which could be a good option to mitigate production interruptions. A typical smart manufacturing system with the consideration of arch-type RMT [6] based on digital twin is shown in Fig. 1, where a couple of RMTs constitute a manufacturing system and digital twin technology is used to connect physical space and virtual space for enhancing the smart awareness of demand fluctuation and dynamically optimizing the performance of the manufacturing system. Due to the capability of digital twin, the smart manufacturing system can recognize the demand changes (e.g., red line in the left top of Fig. 1) that can be the changes of part type, part quantity, or both in both physical space and virtual space, where reconfiguration planning is dynamically executed by considering the flexibility of production precedence and operation sequences simultaneously. Also, the reconfiguration activities will synchronously happen in physical and virtual space to update the performance of the manufacturing system.

Generally, there is more than one part to be produced in a smart manufacturing system. The production precedence of these parts could be changed due to some specific reasons (production precedence flexibility), which may require different manufacturing resources. Namely, the reconfiguration planning could be different when the production precedence of parts is changed. For example, as shown in Table I, there are two parts (part a and part b) to be produced in a smart manufacturing

TABLE I
Influence of Production Precedence Flexibility

Production precedence	Smart manufacturing system	Reconfiguration planning
Part a → Part b	(c_{12}, c_{21})	$\rightarrow (c_{11}, c_{22})$
Part b → Part a		$\rightarrow (c_{11}, c_{22}) \rightarrow (c_{12}, c_{21})$

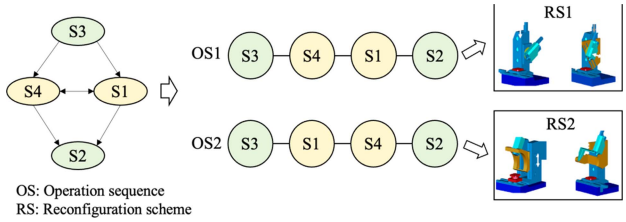


Fig. 2. Influence of operation flexibility.

system consisting of two RMTs (RMT 1 and RMT2). Part a should be produced by RMT 1 with configuration 2 (named c_{12}) and RMT 2 with configuration 1 (named c_{21}). Part b should be produced by c_{11} and c_{22}. Assuming that the current configuration of these two RMTs is c_{12} and c_{21}, if part a is produced first, these two RMTs only need to be reconfigured when producing part b, $c_{12} \rightarrow c_{11}$ and $c_{21} \rightarrow c_{22}$. However, if part b is produced at first, reconfiguration activities will occur when processing part a and part b, $c_{12} \rightarrow c_{11} \rightarrow c_{12}$ and $c_{21} \rightarrow c_{22} \rightarrow c_{21}$. Therefore, it is significant to consider the flexibility of production precedence when executing reconfiguration planning.

Similarly, the operation sequence of a part is generally not rigid. In other words, the operation sequence of a part could be flexible. Different operation sequences may require different manufacturing resources resulting in different reconfiguration schemes as well. Consider the part with four features to be machined and the corresponding precedence order, as shown in Fig. 2. Due to whether S4 or S1 can be selected when S3 is completed, the example part can be machined using flexible operation sequences (S3-S4-S1-S2 or S3-S1-S4-S2) that can lead to different reconfiguration schemes. Moreover, there is more than one option when selecting RMT for producing a specific feature of a part. That is to say, the same feature of a part can be produced in different RMTs producing different reconfiguration planning results. Take the part in Fig. 2 as an example again, although there is no precedence limitation between S4 and S1, the reconfiguration options (different RMTs or configurations of the same RMT) could affect operation sequences as well. If the configuration for S3 can be used for machining S1, operation sequence 2 could be the better one. Thus, it is also important to exploit the operation flexibility of parts when conducting reconfiguration planning for smart manufacturing systems.

Above all, the reconfiguration planning problem of smart manufacturing systems with RMT by considering the flexibility of production precedence and operation sequences simultaneously can be described as follows. A smart manufacturing system with RMT is constructed to possess the ability of handling dynamic demands, where $\mathrm{SMS} = \{\mathrm{RMT}_m | m = 1, 2, 3, \ldots\}$ denotes the structure of the smart manufacturing system and $\mathrm{PT} = \{(P_i, N_i) | i = 1, 2, \ldots\}$ means the dynamic

production tasks assigned to the smart manufacturing system by providing the information of part type P_i and part number N_i. Part i consists of several features to be produced, that is, f_j^i $(j = 1, 2, \ldots)$, where priority relationships between these features are given and will be considered during optimization. When production task PT changes, the smart manufacturing system SMS should simultaneously find the optimal reconfiguration scheme, the production precedence, and operation sequences, which the reconfiguration effort of RMTs themselves, part moving effort among RMTs, and processing cost are used to estimate the optimization process. So, the fundamental problem of this article lies in dynamically reconfiguring the RMTs in smart manufacturing systems efficiently and accurately while simultaneously optimizing production precedence and operation sequences. To increase the accuracy and efficiency of optimization, AI algorithms (i.e., deep reinforcement learning) could be adopted. It is the requirement of making the manufacturing system smarter as well.

III. Reconfiguration Planning Method Based on Reinforcement Learning

In this section, a reconfiguration planning method based on reinforcement learning will be presented, including the reconfiguration process modeling of smart manufacturing systems with RMT and the optimization solution of reconfiguration planning using DQN.

A. Assumptions

To focus on the core problem to be solved in this article, the necessary assumptions should be given at first, as shown in the following.

1) One machine only can process one part at a specific time.
2) Only one feature of a part can be arranged for processing at the same time.
3) One RMT can be reconfigured to more than one configuration.
4) A feature can be completed by a specific configuration of RMT.
5) Once feature processing is started in a specific configuration of RMT, it will not be interrupted until complete.
6) Part consists of features. Once all features are finished, the processing work of this part is completed.
7) Only one part is produced at the same time. Namely, one RMT only can process one part at the same time.
8) To control the complexity of the proposed problem, mixed model production is not considered in this article.

B. Reconfiguration Process Modeling

Generally, reinforcement learning uses MDP to define the interaction between a learning agent and its environment in terms of states, actions, and rewards. The reconfiguration of smart manufacturing systems with RMT is executed based on the current state without necessarily considering the influence of the previous states. So, the reconfiguration process is typical MDP. The reconfiguration process modeling will sort out the details of the MDP tuple $\langle S, A, P, R, \gamma \rangle$, where S denotes the state space, A represents the action space, P denotes the state transition probability, R represents the reward function, and $\gamma \in (0, 1]$ is the reward discount factor. In the proposed work, the next state is always fixed when action is taken in the current state. So, the state transition probability is deterministic. Namely, the state transition probability $P(s'|s, a) = 1$.

1) State Space: In smart manufacturing systems with RMTs, parts with one or more features will be assigned to one or more RMTs and different configurations could be needed. At time slot t, the system state consists of the following:

1) part processing progress;
2) feature processing progress;
3) part position;
4) configuration of RMT.

So, the current state $s_t \in S$ can be expressed by (1). Additionally, the agent will take appropriate action to update the current state during production

$$s_t = (\boldsymbol{U_t}, \boldsymbol{F_t}, \boldsymbol{Pos_t}, \boldsymbol{C_t}) \tag{1}$$

where $\boldsymbol{U_t} = \{u_i | i = 1, 2, \ldots\}$ denotes the part processing progress at time t, u_i records the processing progress of part i ($u_i \in \{0, 1, 2\}$ represents NotStarted, InComplete, and Completed states, respectively, of part i). For example, $\boldsymbol{U_t} = \{0, 1, 2, 2\}$ means that the production of part 1 has not started, part 2 is incomplete, while part 3 and part 4 have been completed at time t. $\boldsymbol{F_t} = \{f_j^i | i = 1, 2, \ldots; j = 1, 2, 3, \ldots\}$ represents the processing progress of feature j in part i at time t ($f_j^i = 1$ means feature j of part i is completed; otherwise, $f_j^i = 0$). For example, part 2 is incomplete at time t, the corresponding $\boldsymbol{F_t} = \{1, 0, 0, 1\}$ means that feature 1 and feature 4 have been completed, while feature 2 and feature 3 are incomplete at time t. $\boldsymbol{Pos_t} = \{p_m | m = 1, 2, \ldots\}$is used to show the position of part i at time t, where p_m means the fixed position of RMT m in the smart manufacturing system. Due to only the position changes of parts being concerned, the real position value will not affect the results. Namely, p_1 means the position of the RMT1, in which the real position value RMT1 will not be involved. As mentioned before, part 2 is incomplete, if $\boldsymbol{Pos_t} = \{p_2\}$means part 2 is located in RMT2. $\boldsymbol{C_t} = \{c_{mk} | m = 1, 2, \ldots; k = 1, 2, \ldots\}$ is a configuration set of the RMTs in the current manufacturing system at time t, where c_{mk} refers to the configuration k of RMT m. For example, $\boldsymbol{C_t} = \{c_{11}, c_{21}, c_{32}\}$ means that there are three RMTs in the smart manufacturing system and the configuration states of RMT1, RMT2, and RMT3 are configuration 1 (c_{11}), configuration 1 (c_{21}), and configuration 2 (c_{32}), respectively.

2) Action Space: In the reconfiguration process, the action can be described from two aspects: Selecting a new part, namely wp^i; or selecting a feature of the current part (InComplete) to process using a specific configuration of the selected RMT, namely $c_{mk}^{f_j^i}$, where $c_{mk}^{f_j^i}$ means that the agent selects RMT m with configuration k for processing feature j of part i. Reconfiguration activities could happen if the current configuration of the selected RMT is not the selected one. Based on the description of the state and action, the decision point at which the agent selects an action to update the state should be after a feature

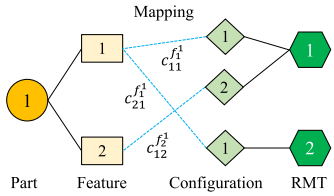


Fig. 3. Action space example.

of the current part is completed or after a new part is selected. So, the action $a_t \in A$ can be defined according to the following equation:

$$a_t = \begin{cases} wp^i, \ Condition\ 1 \\ c_{mk}^{f_j^i}, \ Condition\ 2 \end{cases} \tag{2}$$

where Condition 1 means when the current part is completed or the system is in the initial state, the agent will select a new part for production from the current production task. Condition 2 means a part has been selected in state InComplete, action $c_{mk}^{f_j^i}$ will be taken to process a new feature of the selected part (f_j^i), which can be completed by using c_{mk}. For example, a part contains two features and a feature can be processed in different RMTs; the mapping among features and configurations is shown in Fig. 3, where the corresponding action space is $\{wp^1, c_{11}^{f_1^1}, c_{21}^{f_1^1}, c_{12}^{f_2^1}\}$.

3) Policy: The process of reconfiguration from the current state to the next state by conducting the selected action is regarded as a policy $\pi : S \to A$, which is the mapping between the state space and the action space. The policy provides a definition of action selection of agent in current state. When finishing performing action a_t in state s_t, the next state $s_{t+1} \Leftarrow s_t + \ a_t$ can be obtained. Adopting the example in Fig. 3 again, supposing the current state is $s_1 = \{\ u_1 = 0\}, \{f_1^1 = 1, f_2^1 = 0\}, p_1, \{c_{11}, c_{21}\}$, the agent will choose the action a_1 from action space according to the policy π. In this situation, $a_1 = c_{22}^{f_2^1}$ will be the best action.

4) Reward: Generally, a reward will be obtained when the agent executes an action under a state, that is, $r_t \sim R(s_t, a_t)$. The reward determines the optimization direction of deep reinforcement learning, which is the crucial indicator for evaluating the optimization effectiveness. Therefore, the reward function for reconfiguration planning will be set based on the minimum production cost of multiple parts, as shown in the following equation:

$$R\ (s_t, a_t) = \begin{cases} \mathrm{GP} - \begin{pmatrix} \delta \cdot d\ (p_m,\ p_{m'}) \\ +g\ (c_{m'k},\ c_{m'k'}) \\ +N_i \cdot pd\left(c_{m'k}^{f_j^i}\right) \end{pmatrix}, (s_t, a_t) \in \text{Condition 3} \\ \text{positive value}, \quad (s_t, a_t) \in \text{Condition 4} \\ \text{negative value}, \quad (s_t, a_t) \in \text{Condition 5} \end{cases} \tag{3}$$

where Condition 3 denotes selecting a feature of the current part for producing, which the production cost will be calculated. GP denotes the gross profit of producing a specific production task. $d(p_m,\ p_{m'})$ means the cost of moving part from RMT_m to $\mathrm{RMT}_{m'}$[if the part is processed in the same RMT (i.e., $m = m'$), $d\ (p_m,\ p_{m'}) = 0$]. $\delta = \lfloor N_i/B \rfloor$ is the moving factor considering parts that can be moved in batches for saving cost, where N_i is the total number of part i, B denotes the batch size, and $\lfloor\rfloor$ means round up to an integer. $g(c_{m'k},\ c_{m'k'})$ means the reconfiguration cost of $\mathrm{RMT}_{m'}$ from configuration k to configuration k' [similarly, if the configuration of $\mathrm{RMT}_{m'}$ is exactly needed (reconfiguration is not needed), $g\ (c_{m'k},\ c_{m'k'}) = 0$]. $pd(c_{m'k}^{f_j^i})$ means the processing cost of producing feature i of part j in configuration k of $\mathrm{RMT}_{m'}$. Condition 4 is selecting an unprocessed part from the production task when initializing production or the current part is completed, in which a positive value (the value should be greater than zero) will be assigned to reward this action. Condition 5 means other situations, including selecting the same part, reprocessing the same feature, violating operation sequence, etc., in which a negative value (the value should be less than zero) will be assigned to penalize this action. Above all, the reward is obtained based on the evaluation of actions, which can be explained through two phases: First, an action will meet one of the three different conditions. Second, the action will be evaluated under a specific condition. In Condition 3, the benefit of completing a production task will be calculated by considering the gross profit, processing cost, reconfiguration cost, and moving cost. In Condition 4, if the right part is selected, a positive value will be assigned. Condition 5 means a wrong action will be punished.

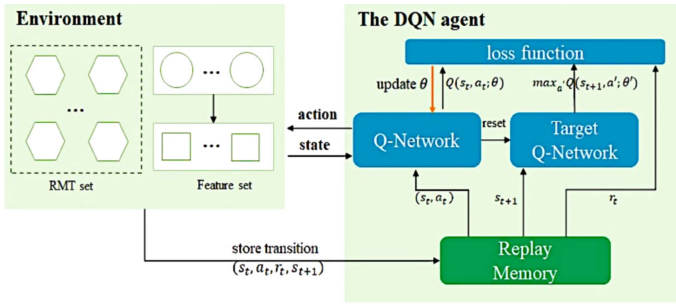


Fig. 4. Reconfiguration planning based on DQN.

C. Reconfiguration Planning Using DQN

Q-learning [38] is one of the classical reinforcement learning algorithms, but it suffers from the curse of dimensionality when solving high-dimensional problems. DQN [39], [40] is proposed to overcome the limitation of Q-learning by introducing a deep neural network to approximate the Q-function, which can describe high-dimensional state spaces better. Also, DQN has stronger scalability than Q-learning benefiting from the deep neural network. According to the previous problem analysis, the reconfiguration planning of smart manufacturing systems with RMT is a typical high-dimensional problem. So, DQN is adopted to solve the proposed reconfiguration process model in this article. The optimization details of DQN for reconfiguration planning are shown in Fig. 4.

DQN is also a value-based algorithm. The goal of DQN is to learn an optimal strategy by approximating the Q-value function so that the agent can obtain the maximum return in its interaction with the environment. The Q-value function represents the long-term expected return of an agent performing an action in a given state, as shown in the following equation:

$$Q^{\pi}(s,a) = E_{\pi}[G_t | s_t = s, a_t = a]$$

$$= E_{\pi}\left[r_{t+1} + \gamma\left(r_{t+2} + \ldots + \gamma^{k-1} r_{t+k}\right) | s_t, a_t\right]$$

$$= E_{\pi}\left[r_{t+1} + \gamma Q^{\pi}(s_{t+1}, a_{t+1}) | s_t, a_t\right] \quad (4)$$

where return G_t is the total discounted reward of the agent from time t to the end. γ is the discount factor, which is used to tradeoff the influence of future reward on return. It is also capable of ensuring the convergence of G_t.

According to DQN, the agent attempts to seek maximum return by constantly updating the Q-value. To guarantee the efficiency of training, the temporal-difference method based on the Bellman expectation equation is adopted to execute a single-step update, as shown in the following equation:

$$Q^{\text{new}}(s_t, a_t) \leftarrow (1-\alpha) Q^{\text{old}}(s_t, a_t) + \alpha\left[r_{t+1} + \gamma \max_{a' \epsilon A} Q^{\pi}(s_{t+1}, a')\right] \quad (5)$$

where s_{t+1} and a' are the next state and action, respectively. $\alpha \in (0, 1)$ is the learning rate, which determines the influence of new information on current Q-value. A lower learning rate may require more learning rounds due to smaller influence; however, a higher learning rate could lead to suboptimal and even cannot converge.

As the state space and action space of MDP are too large in the proposed problem, a deep neural network (Q-network) is used to approximate Q-value function, that is $Q(s, a; \boldsymbol{\theta})$. By feeding the current state into the Q-network, a Q-value prediction will be executed for each possible action. Then, the DQN agent will select an action based on these predictions and gather experience by interacting with the environment. These experiences are used to update the parameters $\boldsymbol{\theta}$ of Q-network to make it more accurate as shown in the following equation:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\theta} L_{\theta} \quad (6)$$

where η is the learning rate that controls the weight updating speed of the neural network; ∇ denotes the gradient function, and L is the loss function, as shown in the following equation:

$$L_{\theta} = \frac{1}{2}[\text{target} - \text{prediction}]^2$$

$$= \frac{1}{2}\left[r_{t+1} + \gamma \max_{a' \epsilon A} Q_{\theta}(s_{t+1}, a') - Q_{\theta}(s_t, a_t)\right]^2. \quad (7)$$

Besides, the techniques of experience replay and target network are used to help DQN agent stabilize the training process and solve the problem of sample correlation and instability in reinforcement learning.

The pseudocode of DQN for reconfiguration planning is given in Table II.

TABLE II
DQN PSEUDOCODE

Inputs: Reconfiguration planning problem
Output: Parameter $\boldsymbol{\theta}$ for Q-Network
Initialize replay memory D to capacity N
Initialize Q-Network with weight $\boldsymbol{\theta}$
Initialize target Q-Network with weights $\boldsymbol{\theta}'$
For episode = 1, $\boldsymbol{M}$ **do**:
  Initialize state $s = s_0$;
  For t=1, $\boldsymbol{T}$ do
    With probability $\boldsymbol{\varepsilon}$ select a random action a_t;
    Otherwise select $a_t = argmax_{a \epsilon A} Q^*(s_t, a)$;
    Execute action a_t in emulator
    Observe reward r_t and new state s_{t+1};
    Save transition (s_t, a_t, r_t, s_{t+1}) in D
    Sample random batch of transitions (s_j, a_j, r_j, s_{j+1}) from D
    Set $y_j = \begin{cases} r_j, & for\ terminal\ s_{j+1} \\ r_j + \gamma max_{a'} Q(s_{j+1}, a'; \boldsymbol{\theta}), & for\ non-terminal\ s_{j+1} \end{cases}$
    Perform a gradient decent regarding weights $\boldsymbol{\theta}$
    Every C steps reset $\theta' = \theta$
  End for
End for

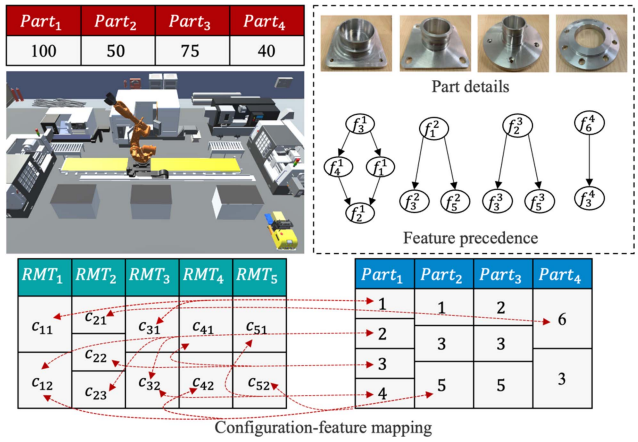


Fig. 5. Case study scenario and production task 1.

IV. CASE STUDY

To demonstrate the effectiveness of the proposed reconfiguration planning method, a case study is presented in this section.

A. Basic Experiment

A smart manufacturing system consisting of five RMTs is adopted to inspect the reconfiguration process of a production task with four parts (Production task 1) in which the corresponding digital twin is constructed using the *Unity 3-D* platform to provide high-fidelity virtual scenario and monitor real-time production activities, as shown in Fig. 5. Also, the feature precedence of these four parts and the configuration-feature mapping is given in Fig. 5.

The optimization goal of this case study is to maximize the return of completing the production task in the selected smart manufacturing system with five RMTs. GP value and the batch size of production can be set according to the feature of production task.

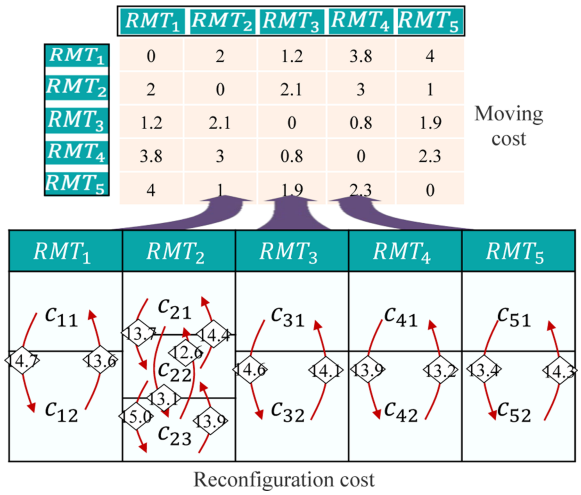


Fig. 6. Moving cost and reconfiguration cost.

TABLE III
PROCESSING COST

Parameter	Value	Parameter	Value	Parameter	Value
$pd(c_{11}^{f_1^{1(2)}})$	0.68(0.59)	$pd(c_{22}^{f_3^{1(2)(3)(4)}})$	0.54(0.51) (0.47)(0.44)	$pd(c_{12}^{f_5^{2(3)}})$	0.74(0.47)
$pd(c_{31}^{f_1^{1(2)}})$	0.63(0.68)	$pd(c_{41}^{f_3^{1(2)(3)(4)}})$	0.60(0.33) (0.65)(0.40)	$pd(c_{42}^{f_5^{2(3)}})$	0.42(0.59)
				$pd(c_{52}^{f_5^{2(3)}})$	0.30(0.55)
$pd(c_{12}^{f_2^{1(3)}})$	0.33(0.43)	$pd(c_{32}^{f_4^1})$	0.32		
$pd(c_{23}^{f_2^{1(3)}})$	0.75(0.37)	$pd(c_{51}^{f_4^1})$	0.59	$pd(c_{21}^{f_4^4})$	0.50
$pd(c_{32}^{f_2^{1(3)}})$	0.37(0.55)				

TABLE IV
RANDOMLY INITIAL CONFIGURATIONS OF FIVE RMTS

RMT	Situation 1	Situation 2	Situation 3
RMT_1	c_{11}	c_{12}	c_{11}
RMT_2	c_{22}	c_{21}	c_{21}
RMT_3	c_{32}	c_{31}	c_{31}
RMT_4	c_{41}	c_{42}	c_{41}
RMT_5	c_{52}	c_{52}	c_{51}

The necessary parameters should be preset before optimization, that is, positive value $=$ 1 and negative value $=$ -50. In addition, the moving cost and the reconfiguration cost are given in Fig. 6. The red arrow means reconfiguration direction. The number on the red arrow means the value of reconfiguration cost. The processing cost is given in Table III.

To inspect the effectiveness and adaptability of the proposed method, we explore the optimal solution based on three randomly initial situations of the smart manufacturing system consisting of five RMTs. The corresponding configurations of these five RMTs are given in Table IV.

Also, the parameters of DQN should be appropriately set for the three situations based on the experience of the existing literature and the situation details, as shown in Table V. For example, in order to explore more states and actions at the beginning of training, the initial epsilon value is set as $\varepsilon = 0.9$, which gradually decrease linearly to 0.1 for increasing the utilization of the learned strategies.

TABLE V
DQN PARAMETER SETTINGS OF THREE SITUATIONS

Parameter	Value
Learning rate η	0.01
Discount factor γ	0.95
Epsilon ε	0.9
EPISODES M	400
Replay Buffer Size N	3000
Batch Size B	128
Update Frequency C	100

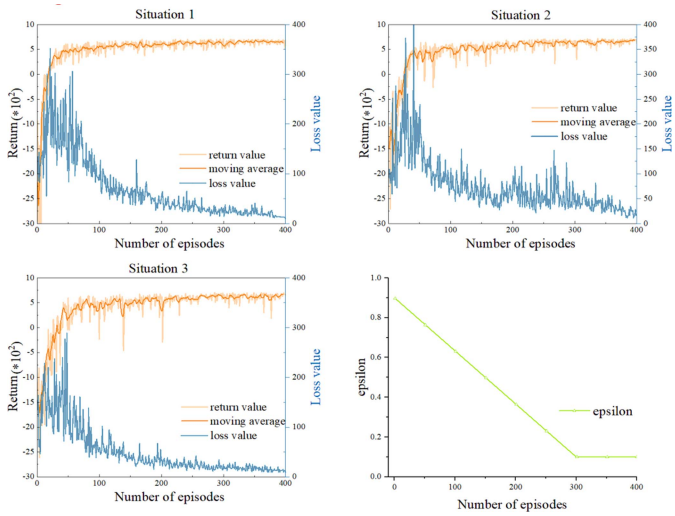


Fig. 7. Training process of three situations.

The experiments of these three situations are executed on PyCharm with Python in a 3.20 GHz AMD Ryzen 7-5800H laptop. The training processes of these experiments are shown in Fig. 7.

In DQN, the loss value of the neural network is used to measure the difference between the predicted value and the target value, which reflects the prediction accuracy of Q-value in the current state of the neural network. Also, if the change in the return function is small, the value function is considered to have converged. Therefore, the training effectiveness and convergence of DQN can be determined by evaluating the changes in the return value curve and loss value curve. The ε value (light green curve) in all experiments decreases gradually to sufficiently explore the potential solutions until ε value equals 0.1. The return value (light orange curve) and loss value (blue curve) in all experiments can converge to a stable level within 400 rounds, indicating that the agent's strategy or value function has converged.

The experimental results are shown in Table VI, including returns and optimal policies. The returns of these three situations are 713.4, 718.4, and 706.2, respectively. The policy of these experiments includes the description of the production sequence of parts, the operation sequence of each part, and the reconfiguration sequence of RMTs. As shown in Fig. 8, in situation 1, the production sequence of parts is part2 $\rightarrow$ part 3 $\rightarrow$ part 1 $\rightarrow$ part 4; the operation sequence of part 2 is to process feature 1 $\rightarrow$ feature 3 $\rightarrow$ feature 5 using RMTs

TABLE VI
EXPERIMENTAL RESULTS OF THREE SITUATIONS

Situation	Return	Optimal policy π^*
Situation 1	717.4	part 2 $\left[c_{11}^{f_1^2} \rightarrow c_{22}^{f_2^2} \rightarrow c_{52}^{f_5^2}\right]$ → part 3 $\left[c_{32}^{f_3^2} \rightarrow c_{22}^{f_2^3} \rightarrow c_{52}^{f_5^3}\right]$ → part 1 $\left[c_{22}^{f_2^1} \rightarrow c_{32}^{f_3^1} \rightarrow c_{11}^{f_1^1} \rightarrow c_{32}^{f_3^1}\right]$ → part 4 $\left[c_{21}^{f_2^4} \rightarrow c_{41}^{f_4^3}\right]$
Situation 2	718.4	part 4 $\left[c_{21}^{f_2^4} \rightarrow c_{22}^{f_2^4}\right]$ → part 2 $\left[c_{31}^{f_3^2} \rightarrow c_{52}^{f_5^2} \rightarrow c_{41}^{f_4^2}\right]$ → part 3 $\left[c_{12}^{f_2^3} \rightarrow c_{12}^{f_2^3} \rightarrow c_{22}^{f_2^3}\right]$ → part 1 $\left[c_{22}^{f_2^1} \rightarrow c_{31}^{f_3^1} \rightarrow c_{32}^{f_3^1} \rightarrow c_{12}^{f_2^1}\right]$
Situation 3	706.2	part 4 $\left[c_{21}^{f_2^4} \rightarrow c_{41}^{f_4^3}\right]$ → part 2 $\left[c_{11}^{f_1^2} \rightarrow c_{41}^{f_4^2} \rightarrow c_{52}^{f_5^2}\right]$ → part 3 $\left[c_{23}^{f_2^3} \rightarrow c_{22}^{f_2^3} \rightarrow c_{52}^{f_5^3}\right]$ → part 1 $\left[c_{22}^{f_2^1} \rightarrow c_{31}^{f_3^1} \rightarrow c_{32}^{f_3^1} \rightarrow c_{32}^{f_3^1}\right]$

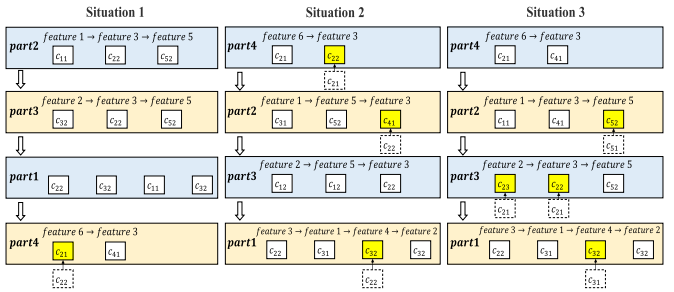


Fig. 8. Optimal policy of these three situations.

Production task 2

$Part_1$	$Part_2$	$Part_3$	$Part_4$
60	55	30	75

Production task 3

$Part_1$	$Part_2$	$Part_3$	$Part_4$
100	30	66	77

Production task 5

$Part_1$	$Part_2$	$Part_3$	$Part_4$
30	0	50	100

Production task 4

$Part_1$	$Part_2$	$Part_3$	$Part_4$
0	40	70	55

Fig. 9. Details of production tasks 2–5.

c_{11}, c_{22}, and c_{52} in turn, and part 3, part 4, and part 1 follow the same pattern; the reconfiguration activity is $c_{22} \rightarrow c_{21}$, means that RMT2 is reconfigured from c_{22} to c_{11} to process feature 6 of part 4. Similarly, the optimal policy of situation 2 and situation 3 can be obtained from Fig. 8 as well. It can be observed that these reconfiguration schemes never violate feature precedence in Fig. 5 and only require reconfiguration activities when necessary. Besides, the reconfiguration planning results demonstrate that the proposed method can fully facilitate the initial configurations to reduce reconfiguration activities to avoid frequent production interruption, which empowers the smart manufacturing system with RMT to deal with demand fluctuation easily.

B. Dynamic Production Tasks

To further present the application process of the proposed method and verify its effectiveness, a series of experiments with dynamic production tasks are conducted. The details of the dynamic production tasks are shown in Fig. 9. The part quantity changes in the new production tasks, in which the zero quantity of a part means this part is not needed in the production task. These four new production tasks are assumed to arrive in order, which will start from the last states of situation 1 of production task 1 in Fig. 5. The robustness of the trained DQN model can be tested as well.

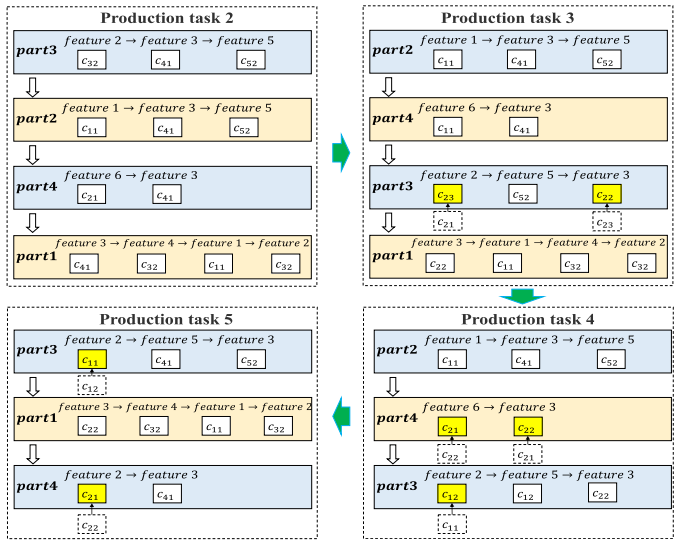


Fig. 10. Optimal policy of new production tasks.

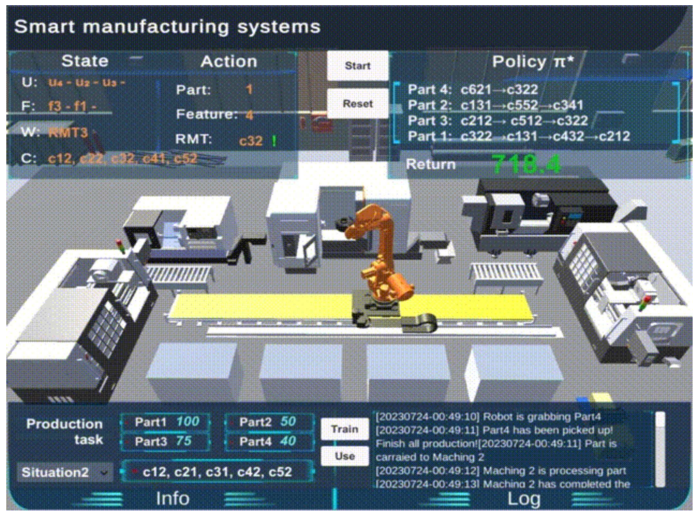


Fig. 11. Automatic reconfiguration optimization (.GIF) (*Adobe Acrobat reader* can show the GIF).

The optimal policy of these new production tasks is shown in Fig. 10. Again, these results based on the trained DQN model for different production tasks do not violate the assumptions and restrictions in this article. That means the trained DQN model can deal with different production tasks well.

In order to make the proposed method easier to use, the corresponding optimization model is integrated with the digital twin of the smart manufacturing system with five RMTs to automatically execute reconfiguration optimization when changes happen, including different initial states and dynamic production tasks, as shown in Fig. 11 (GIF). A digital Kanban is used to dynamically show the reconfiguration policy of the current production task in the upper left. The initial states of the smart manufacturing systems can be selected using the button "Situation" in the bottom left. The current state and action are shown in the upper left, where "?" is used to label reconfiguration

TABLE VII
RANDOM EXPERIMENTAL RESULTS

Situation	Return	Random π
Situation 1	560.65	part 1 $\left[c_{22}^{f_3^1} \rightarrow c_{32}^{f_4^1} \rightarrow c_{11}^{f_1^1} \rightarrow c_{23}^{f_2^1}\right]$ → part 3 $\left[c_{12}^{f_2^3} \rightarrow c_{22}^{f_3^3} \rightarrow c_{42}^{f_5^3}\right]$ → part 4 $\left[c_{21}^{f_6^4} \rightarrow c_{41}^{f_3^4}\right]$ → part 2 $\left[c_{31}^{f_1^2} \rightarrow c_{22}^{f_3^2} \rightarrow c_{52}^{f_5^2}\right]$
Situation 2	574.15	part 2 $\left[c_{31}^{f_1^2} \rightarrow c_{12}^{f_2^2} \rightarrow c_{41}^{f_3^2}\right]$ → part 1 $\left[c_{41}^{f_3^1} \rightarrow c_{51}^{f_4^1} \rightarrow c_{11}^{f_1^1} \rightarrow c_{12}^{f_2^1}\right]$ → part 3 $\left[c_{23}^{f_2^3} \rightarrow c_{12}^{f_2^3} \rightarrow c_{22}^{f_3^3}\right]$ → part 4 $\left[c_{21}^{f_6^4} \rightarrow c_{41}^{f_3^4}\right]$
Situation 3	552.35	part 4 $\left[c_{21}^{f_6^4} \rightarrow c_{22}^{f_3^4}\right]$ → part 2 $\left[c_{31}^{f_1^2} \rightarrow c_{41}^{f_3^2} \rightarrow c_{12}^{f_2^2}\right]$ → part 1 $\left[c_{22}^{f_3^1} \rightarrow c_{11}^{f_1^1} \rightarrow c_{51}^{f_4^1} \rightarrow c_{12}^{f_2^1}\right]$ → part 3 $\left[c_{32}^{f_2^3} \rightarrow c_{22}^{f_3^3} \rightarrow c_{42}^{f_5^3}\right]$

TABLE VIII
GREEDY ALGORITHM RESULTS

Situation	Return	Optimal policy π^*
Situation 1	698.10	part 2 $\left[c_{11}^{f_1^2} \rightarrow c_{52}^{f_5^2} \rightarrow c_{41}^{f_3^2}\right]$ → part 3 $\left[c_{23}^{f_2^3} \rightarrow c_{52}^{f_5^3} \rightarrow c_{22}^{f_3^3}\right]$ → part 1 $\left[c_{22}^{f_3^1} \rightarrow c_{32}^{f_4^1} \rightarrow c_{11}^{f_1^1} \rightarrow c_{32}^{f_2^1}\right]$ → part 4 $\left[c_{21}^{f_6^4} \rightarrow c_{41}^{f_3^4}\right]$
Situation 2	686.20	part 1 $\left[c_{22}^{f_3^1} \rightarrow c_{32}^{f_4^1} \rightarrow c_{31}^{f_1^1} \rightarrow c_{12}^{f_2^1}\right]$ → part 4 $\left[c_{21}^{f_6^4} \rightarrow c_{22}^{f_3^4}\right]$ → part 2 $\left[c_{31}^{f_1^2} \rightarrow c_{52}^{f_5^2} \rightarrow c_{22}^{f_3^2}\right]$ → part 3 $\left[c_{12}^{f_2^3} \rightarrow c_{12}^{f_5^3} \rightarrow c_{22}^{f_3^3}\right]$
Situation 3	674.10	part 4 $\left[c_{21}^{f_6^4} \rightarrow c_{41}^{f_3^4}\right]$ → part 1 $\left[c_{41}^{f_3^1} \rightarrow c_{32}^{f_4^1} \rightarrow c_{11}^{f_1^1} \rightarrow c_{32}^{f_2^1}\right]$ → part 2 $\left[c_{11}^{f_1^2} \rightarrow c_{41}^{f_3^2} \rightarrow c_{42}^{f_5^2}\right]$ → part 3 $\left[c_{23}^{f_2^3} \rightarrow c_{22}^{f_3^3} \rightarrow c_{42}^{f_5^3}\right]$

demand and "!" means that reconfiguration has happened in specific RMT.

C. Comparison Experiments

The random experiment, greedy algorithm experiment, and Q-learning experiment are presented to compare with DQN, where the effectiveness of the proposed optimization model is also assessed through cross validation. Similarly, the random experiment, greedy algorithm experiment, and Q-learning experiment are executed according to production task 1 (see Fig. 5) for comparison.

In the random experiment, all feasible actions that match the current state are first selected from the action space. And then, a random function is used to select an action randomly, in which the current state will be updated to a random new state. Repeat the above steps until a random feasible solution is obtained. The results of the random experiment are shown in Table VII. The returns of the three situations based on the random experiment are 560.65, 574.15, and 552.35, respectively. Obviously, the results of DQN are far better than random experiments, which proves the validity of DQN.

The greedy algorithm is one of the typical heuristic algorithms. The core idea of greedy algorithm is to make a locally optimal choice at each step based on the current state, without considering the global optimum. In this experiment, a greedy choice rule is used to select the action with the maximum reward at each step (if multiple actions have the same reward, one is randomly chosen). Repeat the above step until a feasible solution is obtained. The results of the greedy algorithm experiment are shown in Table VIII. The returns of the three situations are 698.10, 686.20, and 674.10, respectively. Similarly, the results of greedy algorithm are better than random experiment but worse than DQN, which further verify the effectiveness of DQN as well.

TABLE IX
Q-LEARNING HYPERPARAMETERS' SETTINGS

Parameter	Value
Learning rate α	0.001
Discount factor γ	0.95
Epsilon ε	0.9
Min epsilon ε_{min}	0.1
Decay ratio ρ	0.9998
EPISODES M	15 000

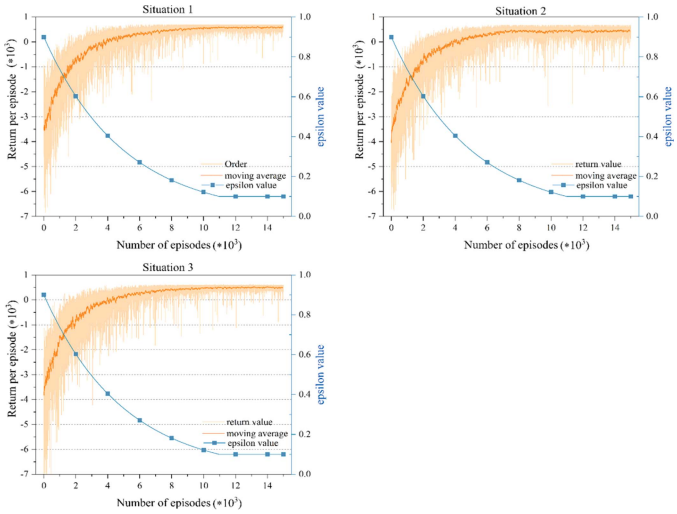


Fig. 12. Convergence process of three situations based on *Q*-learning.

TABLE X
EXPERIMENTAL RESULTS OF Q-LEARNING

Situation	Return	Optimal policy π^*
Situation 1	702.15	part 3 $\left[c_{32}^{f_2^3} \rightarrow c_{52}^{f_5^3} \rightarrow c_{22}^{f_3^3}\right]$ → part 4 $\left[c_{21}^{f_6^4} \rightarrow c_{41}^{f_3^4}\right]$ → part 1 $\left[c_{41}^{f_3^1} \rightarrow c_{32}^{f_4^1} \rightarrow c_{11}^{f_1^1} \rightarrow c_{32}^{f_2^1}\right]$ → part 2 $\left[c_{11}^{f_1^2} \rightarrow c_{52}^{f_5^2} \rightarrow c_{41}^{f_3^2}\right]$
Situation 2	712.75	part 2 $\left[c_{31}^{f_1^2} \rightarrow c_{41}^{f_3^2} \rightarrow c_{52}^{f_5^2}\right]$ → part 1 $\left[c_{41}^{f_3^1} \rightarrow c_{31}^{f_1^1} \rightarrow c_{32}^{f_4^1} \rightarrow c_{12}^{f_2^1}\right]$ → part 4 $\left[c_{21}^{f_6^4} \rightarrow c_{41}^{f_3^4}\right]$ → part 3 $\left[c_{12}^{f_2^3} \rightarrow c_{12}^{f_5^3} \rightarrow c_{22}^{f_3^3}\right]$
Situation 3	693.45	part 2 $\left[c_{31}^{f_1^2} \rightarrow c_{52}^{f_5^2} \rightarrow c_{41}^{f_3^2}\right]$ → part 4 $\left[c_{21}^{f_6^4} \rightarrow c_{22}^{f_3^4}\right]$ → part 3 $\left[c_{12}^{f_2^3} \rightarrow c_{22}^{f_3^3} \rightarrow c_{52}^{f_5^3}\right]$ → part 1 $\left[c_{22}^{f_3^1} \rightarrow c_{31}^{f_1^1} \rightarrow c_{32}^{f_4^1} \rightarrow c_{12}^{f_2^1}\right]$

In the Q-learning experiment, the hyperparameters' settings are shown in Table IX. The convergence processes of these experiments are shown in Fig. 12. All experiments can reach convergence within 12 000 iterations, which is less efficient than DQN. The experimental results are shown in Table X, and the returns of the three situations are 702.15, 712.75, and 693.45, respectively.

The optimal policies obtained from Q-learning also can satisfy the assumptions and restrictions, which can demonstrate the correctness of the proposed optimization model based on MDP. However, the training efficiency and results of Q-learning are inferior to DQN. DQN shows its advantage in solving high-dimensional problems.

V. CONCLUSION

Technology-driven Industry 4.0 is setting the trend of integrating the emerging ICT and AI with traditional industrial scenarios

to promote production efficiency and accuracy in the mass personalization era. Smart manufacturing systems with RMT equipped with high flexibility are a new paradigm to deal with demand fluctuation growing out of individual customization. Digital twin, as the synthesis of the emerging ICT and AI, is the core enabler of smart manufacturing systems with RMT to increase responsiveness. Digital twin-driven smart manufacturing systems with RMT are gradually unfolding the potential of accelerating the development of current industrial evolution. It is significant and necessary to investigate digital twin-driven smart manufacturing systems with RMT.

How to take advantage of the new technologies to promote the efficiency and accuracy of reconfiguration activities is the key problem to be solved when implementing smart manufacturing systems with RMT. Therefore, in this article, a deep reconfiguration planning method of digital twin-driven smart manufacturing systems with RMTs was proposed to dynamically adapt and optimize production activities when demand changes. MDP was used to model the reconfiguration processes of RMTs within a specific smart manufacturing system. The DQN was adopted to search for the optimal reconfiguration scheme, which provides the flexibility to integrate with digital twin smoothly. A case study with three randomly initial situations of given smart manufacturing systems was presented. Besides, the experiment based on different production tasks was used to verify the effect of the proposed method. The experimental results showed that the proposed method can efficiently find optimal reconfiguration schemes online, which also demonstrates the effectiveness and adaptability of the proposed method as well. The comparison between DQN and random experiment, greedy algorithm experiment, and Q-learning experiment provides a cross validation of the proposed optimization model showing its correctness and effectiveness. However, there are some limitations in the proposed work due to the space restriction.

1) The mixed model production is not considered.
2) The production *makespan* is not involved.
3) The optimal algorithm (e.g., SARSA, Dyna-Q, etc.) for reconfiguration planning is not studied as well.

We will investigate the influence of these factors in future work. Moreover, we will further study reconfiguration and scheduling problems simultaneously and figure out the relationships between reconfiguration and scheduling based on digital twin.

References

[1] A. Kusiak, "Smart manufacturing," *Int. J. Prod. Res.*, vol. 56, no. 1/2, pp. 508–517, 2018.

[2] P. Zheng et al., "Smart manufacturing systems for Industry 4.0: Conceptual framework, scenarios, and future perspectives," *Front. Mech. Eng.*, vol. 13, no. 2, pp. 137–150, 2018.

[3] I. Ahmed, G. Jeon, and F. Piccialli, "From artificial intelligence to explainable artificial intelligence in Industry 4.0: A survey on what, how, and where," *IEEE Trans. Ind. Inform.*, vol. 18, no. 8, pp. 5031–5042, Aug. 2022.

[4] Y. H. Yin, J. Y. Xie, L. D. Xu, and H. Chen, "Imaginal thinking-based human-machine design methodology for the configuration of reconfigurable machine tools," *IEEE Trans. Ind. Inform.*, vol. 8, no. 3, pp. 659–668, Aug. 2012.

[5] S. Huang, G. Wang, and Y. Yan, "Building blocks for digital twin of reconfigurable machine tools from design perspective," *Int. J. Prod. Res.*, vol. 60, no. 3, pp. 942–956, 2022.

[6] J. Dhupia, B. Powalka, R. Katz, and A. G. Ulsoy, "Dynamics of the arch-type reconfigurable machine tool," *Int. J. Mach. Tools Manuf.*, vol. 47, no. 2, pp. 326–334, 2007.

[7] J. Leng, D. Wang, W. Shen, X. Li, Q. Li, and X. Chen, "Digital twins-based smart manufacturing system design in Industry 4.0: A review," *J. Manuf. Syst.*, vol. 60, pp. 119–137, 2021.

[8] X. Zhou et al., "Intelligent small object detection for digital twin in smart manufacturing with industrial cyber-physical systems," *IEEE Trans. Ind. Inform.*, vol. 18, no. 2, pp. 1377–1386, Feb. 2022.

[9] F. Tao, H. Zhang, A. Liu, and A. Y. C. Nee, "Digital twin in industry: State-of-the-art," *IEEE Trans. Ind. Inform.*, vol. 15, no. 4, pp. 2405–2415, Apr. 2019.

[10] C. Zhang, W. Xu, J. Liu, Z. Liu, Z. Zhou, and D. T. Pham, "Digital twin-enabled reconfigurable modeling for smart manufacturing systems," *Int. J. Comput. Integr. Manuf.*, vol. 34, no. 7/8, pp. 709–733, 2021.

[11] Y. Koren et al., "Reconfigurable manufacturing systems," *CIRP Ann.*, vol. 48, no. 2, pp. 527–540, 1999.

[12] Y. Koren, "The rapid responsiveness of RMS," *Int. J. Prod. Res.*, vol. 51, no. 23/24, pp. 6817–6827, 2013.

[13] S. Huang, G. Wang, and Y. Yan, "Delayed reconfigurable manufacturing system," *Int. J. Prod. Res.*, vol. 57, no. 8, pp. 2372–2391, 2019.

[14] J. Morgan, M. Halton, Y. Qiao, and J. G. Breslin, "Industry 4.0 smart reconfigurable manufacturing machines," *J. Manuf. Syst.*, vol. 59, pp. 481–506, 2021.

[15] H. Tang, D. Li, J. Wan, M. Imran, and M. Shoaib, "A reconfigurable method for intelligent manufacturing based on industrial cloud and edge intelligence," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4248–4259, May 2020.

[16] J. Wan et al., "Reconfigurable smart factory for drug packing in healthcare industry 4.0," *IEEE Trans. Ind. Inform.*, vol. 15, no. 1, pp. 507–516, Jan. 2019.

[17] S. Lee and K. Ryu, "Development of the architecture and reconfiguration methods for the smart, self-reconfigurable manufacturing system," *Appl. Sci.*, vol. 12, no. 10, 2022, Art. no. 5172.

[18] M. Bortolini, F. G. Galizia, and C. Mora, "Reconfigurable manufacturing systems: Literature review and research trend," *J. Manuf. Syst.*, vol. 49, pp. 93–106, 2018.

[19] A. R. Yelles-Chaouche, E. Gurevsky, N. Brahimi, and A. Dolgui, "Reconfigurable manufacturing systems from an optimisation perspective: A focused review of literature," *Int. J. Prod. Res.*, vol. 59, no. 21, pp. 6400–6418, 2021.

[20] Y. Koren, W. Wang, and X. Gu, "Value creation through design for scalability of reconfigurable manufacturing systems," *Int. J. Prod. Res.*, vol. 55, no. 5, pp. 1227–1242, 2017.

[21] Y. Koren, X. Gu, and W. Guo, "Reconfigurable manufacturing systems: Principles, design, and future trends," *Front. Mech. Eng.*, vol. 13, pp. 121–136, 2018.

[22] Y. Ma, G. Du, and R. J. Jiao, "Optimal crowdsourcing contracting for reconfigurable process planning in open manufacturing: A bilevel coordinated optimization approach," *Int. J. Prod. Econ.*, vol. 228, 2020, Art. no. 107884.

[23] A. Azab and H. A. ElMaraghy, "Mathematical modeling for reconfigurable process planning," *CIRP Ann.*, vol. 56, no. 1, pp. 467–472, 2007.

[24] Z. Qin and Y. Lu, "Self-organizing manufacturing network: A paradigm towards smart manufacturing in mass personalization," *J. Manuf. Syst.*, vol. 60, pp. 35–47, 2021.

[25] S. Huang and K. Saitou, "Configuration design for make-to-order production considering individual order arrivals and machine level changeovers," in *Proc. IEEE 16th Int. Conf. Autom. Sci. Eng.*, 2020, pp. 630–635.

[26] M. Gadalla and D. Xue, "An approach to identify the optimal configurations and reconfiguration processes for design of reconfigurable machine tools," *Int. J. Prod. Res.*, vol. 56, no. 11, pp. 3880–3900, 2018.

[27] J. Leng et al., "Digital twin-driven rapid reconfiguration of the automated manufacturing system via an open architecture model," *Robot. Comput.-Integr. Manuf.*, vol. 63, 2020, Art. no. 101895.

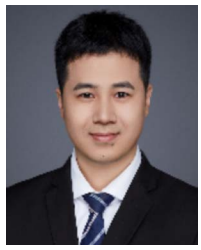
[28] F. A. Touzout and L. Benyoucef, "Multi-objective sustainable process plan generation in a reconfigurable manufacturing environment: Exact and adapted evolutionary approaches," *Int. J. Prod. Res.*, vol. 57, no. 8, pp. 2531–2547, 2019.

[29] A. Khezri, H. H. Benderbal, and L. Benyoucef, "Towards a sustainable reconfigurable manufacturing system (SRMS): Multi-objective based approaches for process plan generation problem," *Int. J. Prod. Res.*, vol. 59, no. 15, pp. 4533–4558, 2021.

[30] C. Liu, M. Whitzer, and M. Yim, "A distributed reconfiguration planning algorithm for modular robots," *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4231–4238, Oct. 2019.
[31] D. Johnson, G. Chen, and Y. Lu, "Multi-agent reinforcement learning for real-time dynamic production scheduling in a robot assembly cell," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 7684–7691, Jul. 2022.
[32] A. S. Khan, L. Homri, J. Y. Dantan, and A. Siadat, "An analysis of the theoretical and implementation aspects of process planning in a reconfigurable manufacturing system," *Int. J. Adv. Manuf. Technol.*, vol. 119, no. 9/10, pp. 5615–5646, 2022.
[33] S. Yang and Z. Xu, "Intelligent scheduling and reconfiguration via deep reinforcement learning in smart manufacturing," *Int. J. Prod. Res.*, vol. 60, no. 16, pp. 4936–4953, 2022.
[34] K. Alexopoulos, N. Papakostas, D. Mourtzis, and G. Chryssolouris, "A method for comparing flexibility performance for the lifecycle of manufacturing systems under capacity planning constraints," *Int. J. Prod. Res.*, vol. 49, no. 11, pp. 3307–3317, 2011.
[35] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: Past, present, and future," *Multimedia Tools Appl.*, vol. 80, pp. 8091–8126, 2021.
[36] S. Kirkpatrick, C. D. Gelatt Jr, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.
[37] E. H. Houssein, A. G. Gad, K. Hussain, and P. N. Suganthan, "Major advances in particle swarm optimization: Theory, analysis, and application," *Swarm Evol. Comput.*, vol. 63, 2021, Art. no. 100868.
[38] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *J. Artif. Intell. Res.*, vol. 4, pp. 237–285, 1996.
[39] S. Lee and D.-H. Choi, "Federated reinforcement learning for energy management of multiple smart homes with distributed energy resources," *IEEE Trans. Ind. Inform.*, vol. 18, no. 1, pp. 488–497, Jan. 2022.
[40] J. Wang, C. Jiang, K. Zhang, X. Hou, Y. Ren, and Y. Qian, "Distributed Q-learning aided heterogeneous network association for energy-efficient IIoT," *IEEE Trans. Ind. Inform.*, vol. 16, no. 4, pp. 2756–2764, Apr. 2020.
[41] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[42] Y. Li, "Deep reinforcement learning: An overview," 2017, *arXiv:1701.07274*.
[43] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Mach. Learn.*, vol. 8, pp. 279–292, 1992.
[44] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.
[45] R. Liu and X. Xie, "Improve the industrial digital transformation through Industrial Internet platforms," *Front. Eng. Manage.*, vol. 11, pp. 167–174, 2024.
[46] B. I. Epureanu, X. Li, A. Nassehi, and Y. Koren, "Self-repair of smart manufacturing systems by deep reinforcement learning," *CIRP Ann.*, vol. 69, no. 1, pp. 421–424, 2020.
[47] E. Bakopoulos, V. Siatras, P. Mavrothalassitis, N. Nikolakis, and K. Alexopoulos, "Digital-twin-enabled framework for training and deploying AI agents for production scheduling," in *Artificial Intelligence in Manufacturing: Enabling Intelligent, Flexible and Cost-Effective Production Through AI*. Berlin, Germany: Springer, 2024, pp. 147–179.

Jintang Huang received the B.Eng. degree in mechanical design, manufacturing, and automation from South China Agricultural University, Guangzhou, China, in 2021, and the M.Eng. degree in mechanical engineering from the Beijing Institute of Technology, Beijing, China, in 2024.

He is currently an Assistant Engineer with Beijing Fanuc, Beijing, China. His current research interests include deep reinforcement learning, dynamic reconfiguration planning, digital twin, smart manufacturing systems, and the development and application of reconfigurable machine tools.

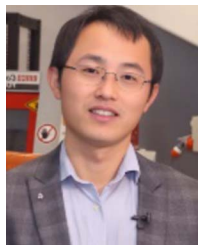


Sihan Huang(Member,IEEE) received the B.Eng. degree in industrial engineering and the Ph.D. degree in mechanical engineering from the Beijing Institute of Technology, Beijing, China, in 2014 and 2020, respectively.

He visited the University of Michigan-Ann Arbor, USA, from 2017 to 2019. He is currently an Associate Professor with the School of Mechanical Engineering, Beijing Institute of Technology. His research interests include reconfigurable manufacturing systems, human-centric smart manufacturing, and digital twin. He has authored or coauthored more than 30 papers (one highly cited paper included in the ESI database). He also served as an Associate Editor for the *Journal of Intelligent Manufacturing*, an Associate Editor and Special Corresponding Expert of *Frontiers of Engineering Management*, and a Youth Editorial Board Member of the *Journal of Mechanical Engineering*, *Industrial Engineering Journal*, and *Journal of Advanced Manufacturing Science and Technology*.

Shokraneh K. Moghaddam received the B.Sc. degree from Islamic Azad University, North Tehran Branch, Tehran, Iran, in 2011, the M.Sc. degree from Tarbiat Modares University, Tehran, Iran, in 2013, and the Ph.D. degree from the Sharif University of Technology, Tehran, Iran, in 2019, all in industrial engineering.

She is currently a Lecturer with the School of Physics, Engineering, and Computer Science, University of Hertfordshire, Hatfield, U.K. She was a short-term Visiting Scholar with the University of Michigan, Ann Arbor, USA, from 2017 to 2018. Her fields of interest include advanced manufacturing systems, applied operations research, optimization, and simulation. Her current research is majorly focused on reconfigurable manufacturing systems configuration design as well as reconfigurable supply chain network design. She has authored or coauthored papers in prestigious journals, such as international journal of production research (IJPR), journal of manufacturing systems (JMS), and computers & industrial engineering (CAIE), and has served on editorial boards of a number of them as a reviewer.



Yuqian Lu (Member, IEEE) received the B.Eng. degree in mechatronics engineering from Dalian University of Technology, in 2012, the Ph.D. degree in mechatronics engineering from The University of Auckland, in 2016. He is a Senior Lecturer with the Department of Mechanical and Mechatronics Engineering, The University of Auckland (UoA), Auckland, New Zealand, where he leads Industrial Artificial Intelligence Research Group. His research mainly focuses on manufacturing systems, industrial artificial intelligence, and human–robot interaction. He is on the board of multiple international scientific committees, journals, and conferences.

Guoxin Wang received the B.Eng. degree in mechanical manufacture and automation and M.Eng. degree in vehicle engineering from Lanzhou Jiaotong University, Lanzhou, China, and the Ph.D. degree in mechanical manufacture and automation from the Beijing Institute of Technology, Beijing, China, in 2001, 2004, and 2007, respectively.

He was a Visiting Scholar with the University of Oklahoma, USA, from 2014 to 2015. He is currently a Professor with the Beijing Institute of Technology. His current research interests include reconfigurable manufacturing systems, intelligent design, systems engineering, and knowledge engineering.

Yan Yan received the B.Eng. and Ph.D. degrees in mechanical engineering from the Beijing Institute of Technology, Beijing, China, in 1989 and 2001, respectively.

She is currently a Professor with the Beijing Institute of Technology. Her current research interests include reconfigurable manufacturing systems, and intelligent design and knowledge engineering.

Xuejiang Shi received the B.Eng. degree in mechanical engineering and automation and the M.Eng. degree in mechanical and electronic engineering from the Beijing Institute of Technology, Beijing, China, in 2005 and 2007, respectively.

He visited FOCKE Company, Germany, from July to September 2009. He is currently an Engineer with Hongyun Honghe Group, Kunming, China. His research interests include intelligent manufacturing, intelligent logistics, logistics supply chain, etc.