


## Article

# Digital Twin-Driven Adaptive Scheduling for Flexible Job Shops

Lilan Liu <sup>1,2</sup>, Kai Guo <sup>1,2</sup>, Zenggui Gao <sup>1,2,\*</sup> , Jiaying Li <sup>1,2</sup> and Jiachen Sun <sup>1,2</sup>

<sup>1</sup> School of Mechatronic Engineering and Automation, Shanghai University, Shanghai 200444, China; lancy@shu.edu.cn (L.L.); betta@shu.edu.com (K.G.); lji1201@shu.edu.com (J.L.); sunjiachen@shu.edu.com (J.S.)

<sup>2</sup> Shanghai Key Laboratory of Intelligent Manufacturing and Robotics, Shanghai University, Shanghai 200444, China

\* Correspondence: gaozg@shu.edu.cn

**Abstract:** The traditional shop floor scheduling problem mainly focuses on the static environment, which is unrealistic in actual production. To solve this problem, this paper proposes a digital twin-driven shop floor adaptive scheduling method. Firstly, a digital twin model of the actual production line is established to monitor the operation of the actual production line in real time and provide a real-time data source for subsequent scheduling; secondly, to address the problem that the solution quality and efficiency of the traditional genetic algorithm cannot meet the actual production demand, the key parameters in the genetic algorithm are dynamically adjusted using a reinforcement learning enhanced genetic algorithm to improve the solution efficiency and quality. Finally, the digital twin system captures dynamic events and issues warnings when dynamic events occur in the actual production process, and adaptively optimizes the initial scheduling scheme. The effectiveness of the proposed method is verified through the construction of the digital twin system, extensive dynamic scheduling experiments, and validation in a laboratory environment. It achieves real-time monitoring of the scheduling environment, accurately captures abnormal events in the production process, and combines with the scheduling algorithm to effectively solve a key problem in smart manufacturing.

**Keywords:** digital twin; flexible job-shop scheduling problem (FJSSP); reinforcement learning enhanced genetic algorithm (RLEGA); dynamic job-shop scheduling



**Citation:** Liu, L.; Guo, K.; Gao, Z.; Li, J.; Sun, J. Digital Twin-Driven Adaptive Scheduling for Flexible Job Shops. *Sustainability* **2022**, *14*, 5340. <https://doi.org/10.3390/su14095340>

Academic Editors: Yoshiki Shimomura and Shigeru Hosono

Received: 25 March 2022

Accepted: 27 April 2022

Published: 28 April 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

From the perspective of each country's strategy, whether it is the "Industrial Internet" strategy of the United States [1], the mechanization, electrification and informatization of Germany to the latest "Industry 4.0" strategy [2] or China's "informatization drives industrialization, industrialization promotes informatization" to the deep integration of the two, to the latest "Made in China 2025" manufacturing power development strategy [3], we can see that intelligent manufacturing is the common development trend of all countries in the world. How to achieve the integration of the physical world and the information world is the key to realize intelligent manufacturing.

Digital twin (DT), as a bridge to realize the physical world and the information world [4], is one of the key technologies to realize smart manufacturing. DT technology can effectively perform real-time mapping and bidirectional interaction between the physical world and the information world, thus deeply integrating operational data, environmental changes, dynamic disturbances, and other information in the actual physical production line with data in the virtual space, such as statistical analysis and optimal scheduling. In recent years, scholars in various countries have conducted a lot of research on DT technology, and both academia and industry have focused on digital twin technology as the key to achieve intelligent manufacturing. Digital twin workshop is a new operation mode of future workshop proposed by Professor Tao Fei [5] in 2017. Tao Fei put forward the

concept of digital twin workshop and discussed the key technologies and implementation methods of digital twin workshop. Milton et al. [6] applied digital twin technology to the field of fault diagnosis and gave examples to verify the effectiveness of the method. Shanghua Mi et al. [7] applied digital twin technology to predictive maintenance to improve the accuracy of fault diagnosis and prediction. Song Yue et al. [8], in order to solve the multifield coupling of optoelectronic detection system caused by performance degradation problem, established a digital twin model, and the simulation verified that the digital twin technique could better solve the performance prediction problem. Gabriel Fedorko et al. [9] applied the finite element model to the digital twin technique to explore the possibility of DT in the measurement field. Chunxia Lu et al. [10] constructed a digital twin model of a gear measurement center twin model as a platform for the evaluation of various measurement software, and the results show that the proposed method can effectively identify errors in measurement software.

The flexible job-shop scheduling problem (FJSSP) is an extension of the traditional classical job-shop scheduling problem (JSSP), which has proven to be the well-known NP-hard problem (nondeterministic polynomial time hard). The traditional job-shop scheduling is to arrange the processing sequence of workpieces on the machine under certain optimization objectives. The processing path of workpieces is determined, and the processing machine of each process is unique. The flexible job-shop scheduling of a processing machine can often process multiple workpieces; the processing path of workpieces is uncertain, and the processing machine of each process is not unique, which makes the enterprise have higher flexibility and wider adaptability and increases the difficulty of solving it. The flexible job-shop scheduling problem was first proposed by Brucker and Schile in 1990 [11]. Subsequently, more and more researchers started to study this problem. Shopfloor scheduling plays an important role in the field of intelligent manufacturing to allocate tasks effectively and improve productivity. Tian Songling et al. [12] proposed a shopfloor adaptive scheduling method for the dynamic events of machine failures that may occur in FJSSP. M.B.S. Sreekara Reddy et al. [13] studied the FJSSP multiobjective model of FJSSP and discussed the system performance when machine failure occurs. Yu Tianbiao et al. [14] designed a new genetic algorithm coding method for solving the dynamic scheduling problem of FJSSP. Hao Chinchang et al. [15] proposed a hybrid genetic algorithm for distributed FJSSP, and experiments showed that the proposed method was successful. Dolgui Alexandre et al. [16] investigated the application of optimal control to the scheduling problem in an Industry 4.0 environment. Wang jin [17] proposed a multi-intelligent IoT FJSSP real-time scheduling architecture and verified that the method outperforms the traditional dynamic scheduling strategy through simulation. Yilin Fang et al. [18] made a preliminary exploration of the role of the digital twin in JSSP, proposed a new working principle of job shop, considered the dynamic events that may occur in the actual production process, and finally verified the feasibility of this new model through a prototype. Zhifeng Liu et al. [19], combined with digital twin and super network, developed a new intelligent scheduling method for multivariety and small-batch production workshops and simulated and optimized the initial scheduling scheme by using virtual workshop. Shu Luo [20] used reinforcement learning to simulate and solve for the dynamic events of new order insertion that may occur during the actual production process. Meng Zhang et al. [21] proposed a dynamic scheduling method enhanced by DT to predict the machine and trigger rescheduling in advance and verified the effectiveness of the method with an example.

In general, digital twin technology can deeply integrate the physical world and information world of the actual production workshop. By collecting the data in the actual production process and mapping the data to the virtual workshop, the virtual workshop can be optimized through the data, and finally fed back to the actual workshop to form a closed loop. However, there are few works of literature on the combination of digital twinning and job-shop scheduling. Several research attempted to combine digital twinning technology with optimization function to realize data-driven job-shop adaptive scheduling.

The overall framework of this paper is shown in Figure 1. According to the five-dimensional model proposed by Tao et al. [5], we establish the framework of digital twin workshop, including physical entity layer, simulation application layer, service layer (workshop scheduling system), data, and the connection between various layers. This paper proposes a digital twin-driven workshop adaptive scheduling method. Firstly, the digital twin workshop is established to realize the real-time mapping of the physical workshop. Then, FJSSP is modeled, and the results are obtained by reinforcement learning enhanced genetic algorithm (RLEGA). The possible dynamic events in the workshop are simulated. Finally, based on the digital twin system, the dynamic events of new order arrival and machine failure in the actual production process of the workshop are captured to trigger rescheduling. The feasibility of digital twin technology in job-shop scheduling is verified. Figure 1 includes physical entity layer, simulation application layer, and service layer (job-shop scheduling system). The physical entity layer mainly includes robot arm, materials, suction cup, table, manufacturing environment, automated guided vehicle, 3D printing fixed table, 3D printing car model, and automated unloading device. The simulation application layer mainly includes using a 3D modeling tool and weighting tool to model the actual production line and visualize the final results. In the intelligent production line, we can use sensor networks and MES (manufacturing execution system) to obtain real-time dynamic data (such as equipment data, logistics data, production progress data, order data, etc.) so as to establish a stable and real-time digital twin system. The service layer senses the dynamic events in the production process in real time through the obtained data. The dotted line in Figure 1 indicates that it can be skipped if there are no dynamic events. Every time an abnormal event occurs, it triggers rescheduling, and re-executes the algorithm to obtain the latest results.

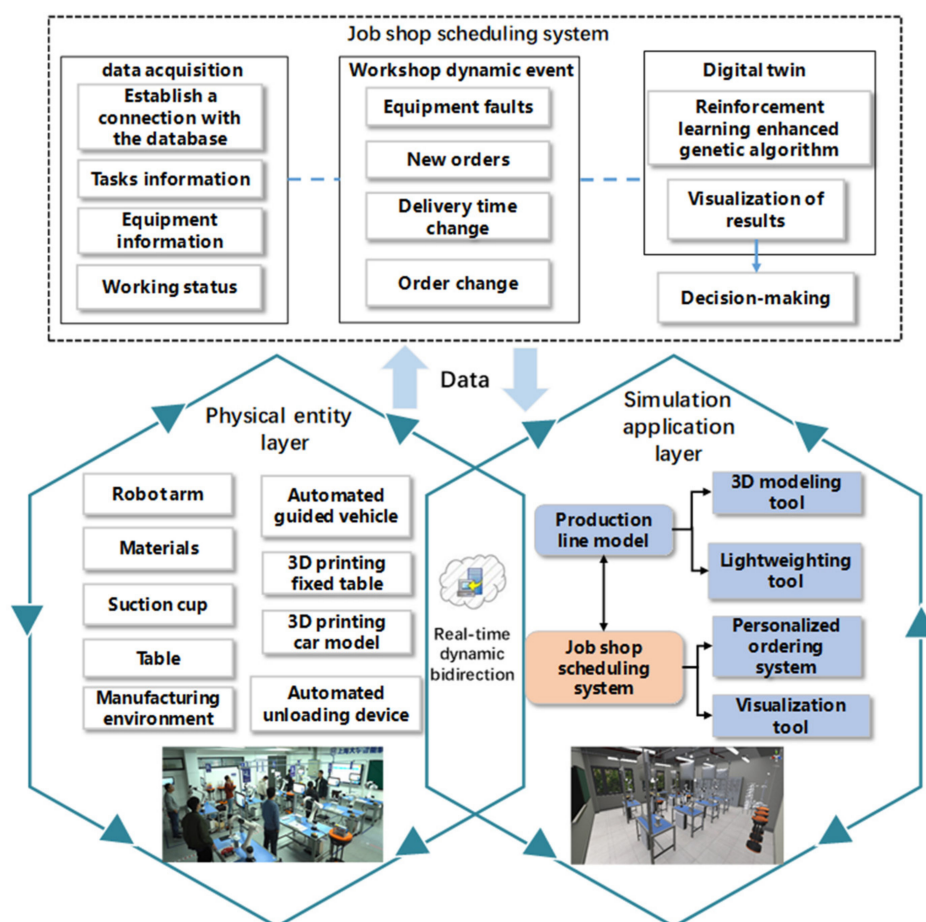


Figure 1. The framework of the proposed method.

The rest of the paper is organized as follows. In Section 2, the digital twin workshop is modeled. Section 3 describes and models the FJSSP problem. Section 4 presents a dynamic scheduling method based on RLEGA. Experimental verification is carried out in Section 5. Finally, Section 6 gives the conclusion and future work.

## 2. Digital Twin Workshop Modelling

### 2.1. Data Acquisition and Processing

The real-time acquisition and processing of relevant data in a physical workshop is an important prerequisite for establishing high-quality digital twin models. The digital twin model is established with reference to the digital twin model construction theory and the digital twin model evaluation index system proposed by Tao Fei [22,23]. The digital twin model is shown in the simulation application layer of Figure 1. Through the collection and processing of all multisource heterogeneous data involved in the real-time scheduling system, it provides data support for the digital twin-driven intelligent workshop scheduling. Based on RFID (radio frequency identification), sensors, manipulators, and the secondary development interface and communication module provided by AGV (automated guided vehicle), the data of the physical workshop is collected. Due to the complexity of the workshop production environment, the collected data often inevitably have null values, error values, or abnormal values, which are collectively referred to as noise data. Noise data seriously affects the synchronous simulation effect of a digital twin system, so it is necessary to filter the collected data to make the digital twin system achieve the ideal effect. In this paper, fuzzy c-means clustering algorithm (FCM) is used for data filtering, and the noise-filtering method based on Euclidean distance is used.

### 2.2. Data Modelling for Job-Shop Scheduling

The digital twin shop uses data collected in real time on the physical shop floor to drive 3D models that map the physical shop floor in real time. Data modelling refers to the classification, association, and aggregation of various multisource heterogeneous data collected from the shop floor to clarify the relationships between the data in a graphical way [24]. Data modelling provides the data basis for real-time monitoring of the physical shop floor and adaptive scheduling of the shop floor. In this paper, as shown in Figure 2, we build a data model for digital twin shop floor adaptive scheduling that drives the 3D model to run and perform solving. The solid diamond arrow in the figure represents the aggregation relationship, which is used to describe the relationship between “part and subject”, and the arrow points from the part to the subject. 1 and \* in the figure represent the association relationship, indicating that the properties of one class save a reference to one or more instances of another class. 1 and \* represent the number of objects of another corresponding class, 1 represents one, and \* represents multiple. In this paper, we divide the data into three parts according to “modelling-monitoring-task scheduling”: static model of virtual workshop, dynamic monitoring information of virtual workshop, and task scheduling solution system. In Figure 2, there are three classes pointing to the digital twin workshop model. Among them, the static model of the virtual workshop mainly includes the geometric and logical information of the workshop, which is not changed during the actual operation of the production line, such as “equipment-product-environment” information. The dynamic monitoring information of the virtual workshop mainly includes the data collected in real time during the operation of the physical production line, which is the core of the data model. All kinds of dynamic information collected in real time can be referred to the class of dynamic monitoring information of the virtual shop floor with arrows in Figure 2. The workshop scheduling system optimizes the scheduling of tasks based on the real-time data collected by the data twin workshop and capture various abnormal events occurring in the physical production line in real time, mainly including new order insertion and equipment failure.



is assigned to the machine that meets the processing conditions. At the same time, on the premise of meeting the process constraints and equipment processing conditions, sort the operation sets assigned to each processing equipment.

This paper considers the evaluation index commonly used in actual production: maximum completion time  $C_{max}$  (makespan) is the smallest. Completion time is the time for each job to complete all its processes, and the longest time is the maximum completion time. The maximum completion time is the most fundamental index to measure the scheduling scheme, which mainly reflects the production efficiency of the workshop. The objective function is shown in Equation (1). Where  $C_j$  is the completion time of job  $j$ . Equations (2) and (3) represent the process sequence constraints of each job; Equation (4) represents the constraint of the completion time of the job, that is, the completion time of each job cannot exceed the total completion time; Equations (5) and (6) means that only one process can be processed by the same machine at the same time; Equation (7) represents machine constraints, that is, the same process can only be processed by one machine at the same time; Equations (8) and (9) indicate that there can be cyclic operation on each machine; and Equation (10) indicates that each parameter variable must be a positive number. The parameters are described in Table 1.

**Table 1.** Meaning of parameters.

Parameters	Descriptions
$n$	The total number of jobs
$m$	The total number of machines
$i, e$	The number of machines
$j, k$	The number of jobs
$h_j$	The total number of operations of job $j$
$h, l$	The number of operations
$m_{jh}$	The number of optional processing machines for the $h$ operation of the job $j$
$O_{jh}$	The $h$ th operation of the $j$ th job
$p_{ijh}$	The time of operation $h$ of job $j$ on machine $i$
$s_{jh}$	The start time of the $h$ operation of the job $j$
$c_{jh}$	The completion time of the $h$ operation of the job $j$
$L$	A positive number large enough
$x_{ijh}$	When machine $i$ is selected for operation $O_{jh}$ , the value is 1, otherwise 0
$y_{ijhkl}$	When operation $O_{ij}$ is preceded by operation $O_{hk}$ , the value is 1, otherwise 0

$$f = \min \left( \max_{1 \leq j \leq n} (C_j) \right) \quad j = 1, \dots, n \quad (1)$$

$$s_{jh} + x_{ijh} \times p_{ijh} \leq c_{jh} \quad i = 1, \dots, m; j = 1, \dots, n; h = 1, \dots, h_j \quad (2)$$

$$c_{jh} \leq s_{j(h+1)} \quad j = 1, \dots, n; h = 1, \dots, h_j - 1 \quad (3)$$

$$c_{jh_j} \leq C_{max} \quad j = 1, \dots, n \quad (4)$$

$$s_{jh} + p_{ijh} \leq s_{kl} + L(1 - y_{ijhkl}) \quad j = 0, \dots, n; k = 1, \dots, n; h = 1, \dots, h_j; l = 1, \dots, h_k; i = 1, \dots, m \quad (5)$$

$$c_{jh} \leq s_{j(h+1)} + L(1 - y_{ikj(h+1)}) \quad j = 1, \dots, n; k = 0, \dots, n; h = 1, \dots, h_j - 1; l = 1, \dots, h_k; i = 1, \dots, m \quad (6)$$



$$\sum_{i=1}^{m_{jh}} x_{ijh} = 1 \quad h = 1, \dots, h_j; j = 1, \dots, n \quad (7)$$

$$\sum_{j=1}^n \sum_{h=1}^{h_j} y_{ijhkl} = x_{ikl} \quad i = 1, \dots, m; k = 1, \dots, n; l = 1, \dots, h_k \quad (8)$$

$$\sum_{k=1}^n \sum_{l=1}^{h_k} y_{ijhkl} = x_{ijh} \quad i = 1, \dots, m; j = 1, \dots, n; h = 1, \dots, h_k \quad (9)$$

$$s_{jh} \geq 0, c_{jh} \geq 0 \quad j = 0, 1, \dots, n; h = 1, \dots, h_j \quad (10)$$

#### 4. Reinforcement Learning Enhanced Genetic Algorithm

##### 4.1. Genetic Algorithm

A genetic algorithm is designed and proposed according to the evolution law of organisms in nature. It is one of the most classical algorithms in the field of job-shop scheduling. The main parameters of genetic algorithms are as follows:

1. Population size  $NP$ . Population size affects the final result of genetic optimization and the execution efficiency of genetic algorithm. When the population size  $NP$  is too small, the performance of genetic optimization is generally not very good. Larger population size can reduce the chance of genetic algorithm falling into local optimal solution, but larger population size means higher computational complexity.
2. Crossover probability  $P_c$ . The crossover probability  $P_c$  controls the frequency with which the crossover operation is used. Larger crossover probability can enhance the ability of genetic algorithm to open up new search areas, but the possibility of the high-performance mode being destroyed increases. If the crossover probability is too low, the genetic algorithm search may fall into a slow state.
3. Mutation probability  $P_m$ . Mutation is an auxiliary search operation in genetic algorithms. Its main purpose is to maintain the diversity of the population. Generally, low-frequency mutation can prevent the possible loss of important genes in the population. High-frequency mutation makes the genetic algorithm tend to pure random search.
4. Termination evolution algebra  $G$  of genetic operation. The terminating evolution algebra  $G$  is a parameter representing the end condition of the genetic algorithm.

The flow chart of solving the flexible job-shop production scheduling problem is shown in Figure 3.

When using genetic algorithms to solve flexible job-shop scheduling problems, it is necessary to encode the chromosome. The purpose of coding is to realize crossover, mutation, and other operations. The legitimacy and feasibility of chromosomes must be considered. FJSP includes two parts: machine selection and process selection. Machine selection is to solve which machine each process is processed on, while process selection is to solve all processes and determine the sequencing and start time after processing machines. To solve the above problems, the chromosome coding is in the form of segmented coding, which is composed of machines selection (MS) and operations selection (OS), as shown in Figure 4. Taking Table 2 as an example, in the upper-left corner of the figure are all processes of job 1 and job 2. The optional machine of process  $O_{11}$  is 5, and the number 4 corresponds to the optional fourth machine  $M_4$ . The optional machine of process  $O_{12}$  is 2, and the number 1 corresponds to the optional first machine  $M_2$ . The order in which the part number of the operation code appears indicates the processing order between the parts. As shown in Figure 4, assuming that the chromosome of the process is [2,2,1,1,2], the

first 2 represents the first process of job 2, and the second 2 represents the second process of job 2. By analogy, the final processing sequence is  $O_{21} - O_{22} - O_{11} - O_{12} - O_{23}$ .

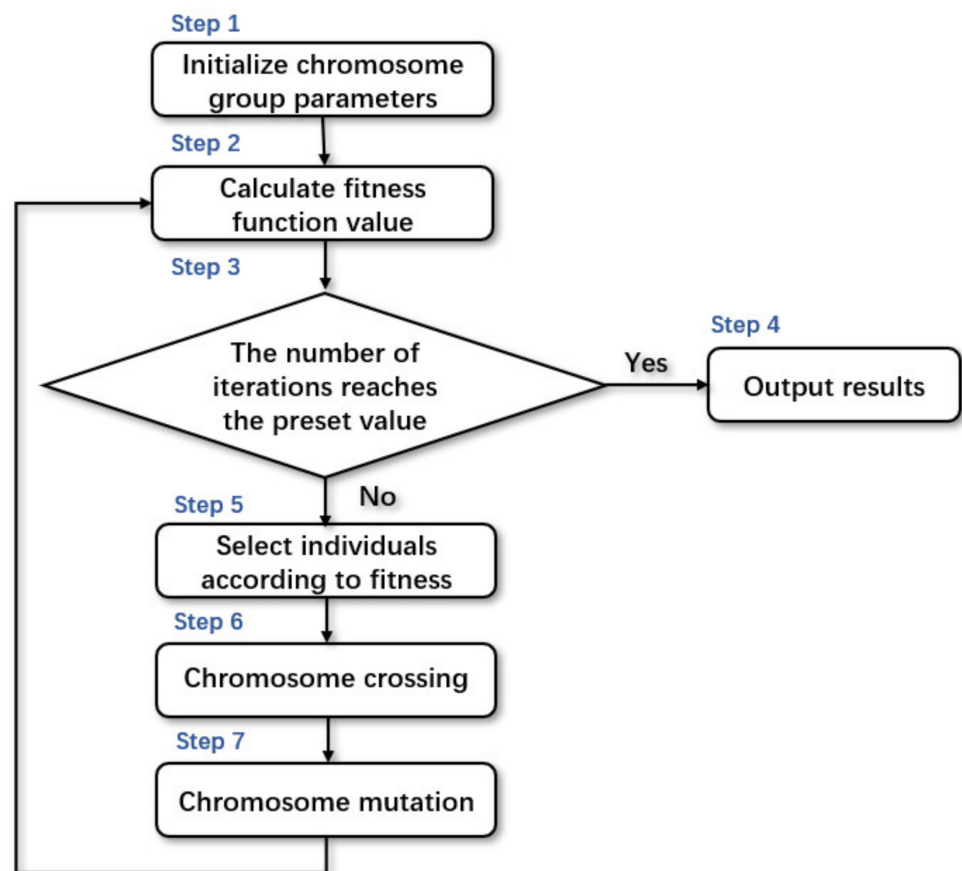


Figure 3. Flow chart of genetic algorithm.

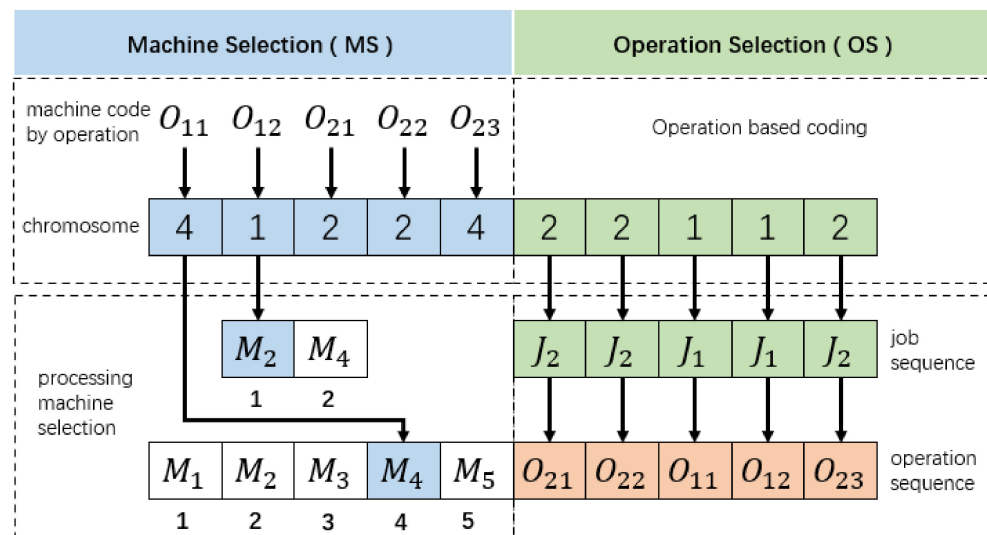


Figure 4. Chromosome coding.



**Table 2.**  $2 \times 5$  flexible job shop example.

Jobs	Operation	Optional Processing Machine				
		$M_1$	$M_2$	$M_3$	$M_4$	$M_5$
$J_1$	$O_{11}$	2	6	5	3	4
	$O_{12}$	—	8	—	4	—
$J_2$	$O_{21}$	3	—	6	—	5
	$O_{22}$	4	6	5	—	—
	$O_{23}$	—	7	11	5	8

#### 4.2. State, Action, and Reward

Crossover probability  $P_c$  and mutation probability  $P_m$  are the most critical parts of the algorithm, which directly affect the performance of the algorithm. The greater the crossover probability, the easier it is to produce new individuals. However, if it is too large, it destroys the structure of individuals. If the crossover probability is too small, the search efficiency of the algorithm is greatly reduced. For the mutation probability, if the value is too small, it is not easy to produce new individuals. If the mutation probability is too large, the genetic algorithm becomes a random search algorithm, and the efficiency of the algorithm is greatly reduced. Therefore, it is necessary to use some methods to make us choose the crossover rate and mutation rate better. In this paper, reinforcement learning is used to adaptively adjust the key parameters in the genetic algorithm.

Markov property means that the state of the current time is only related to the state and action of the previous time and has nothing to do with the actions and states of other times. Set  $h_t = \{s_1, s_2, s_3, \dots, s_t\}$  ( $h_t$  includes all states before time  $t$ ), then the Markov property can be expressed by mathematical formula as shown in Equation (11). If the state transition of the multistage decision problem satisfies the Markov property, the decision problem is a Markov decision problem (MDPs). Markov decision process can be described as 5 tuples, as shown in Equation (12). Where  $S$  represents the state space and is a description of the environment.  $\mathcal{A}(s)$  represents the action space that can be selected in state  $s$ . The  $R$  agent represents the reward given by the environment after making the action.  $\pi(a | s)$  represents the probability distribution of output action  $a$  at state  $s$ .

$$p(s_{t+1} | s_t, a_t) = p(s_{t+1} | h_t, a_t) \quad (11)$$

$$\mathcal{E} = \{S, \mathcal{A}(s), R, P, \pi(a | s)\} \quad (12)$$

**State.** In order to use reinforcement learning to realize the adaptive adjustment of key parameters of genetic algorithm, the setting of state should accurately represent the current state of genetic algorithm. The average fitness of the population, the diversity of the population, and the fitness of the optimal individual are used to represent the current state of the genetic algorithm, as shown in Equations (13)–(15).

$$f^* = \frac{\sum_{i=1}^N f(x_i^t)}{\sum_{i=1}^N f(x_i^1)} \quad (13)$$

$$d^* = \frac{\sum_{i=1}^N \left| f(x_i^t) - \frac{\sum_{i=1}^N f(x_i^t)}{N} \right|}{\sum_{j=1}^N \left| f(x_j^1) - \frac{\sum_{j=1}^N f(x_j^1)}{N} \right|} \quad (14)$$

$$p^* = \frac{\max f(x^t)}{\max f(x^1)} \quad (15)$$

$$S^* = w_1 f^* + w_2 d^* + w_3 p^* \quad (16)$$

where  $f(x_i^t)$  represents the fitness of the  $i$ th individual in generation  $t$ ,  $N$  represents the size of the population, and  $\max f(x^t)$  represents the individual with the largest fitness value in generation  $t$ .  $w_1$ ,  $w_2$ , and  $w_3$  represents weight,  $w_1 + w_2 + w_3 = 1$ , initially 0.3, 0.3, and 0.4, respectively.

**Action.** The selection of action is  $P_c$  and  $P_m$ . The common value of  $P_c$  is between 0.4 and 0.9. Set the interval between each action to 0.05. For example, when the action is selected as  $a_1$ ,  $P_c \in (0.4, 0.45)$ , Randomly select a number in the interval as the value of  $P_c$ . Similarly, the common value of  $P_m$  is between 0.01 and 0.31. Set the interval between each action to 0.03. For example, when the action is selected as  $a_1$ ,  $P_m \in [0.01, 0.04)$ , randomly select a number in the interval as the value of  $P_m$ . There are 10 actions in total.

**Reward.** The reward function is set to be related to the selection of  $P_c$  and  $P_m$ . As shown in Equations (17) and (18). Equation (17) indicates that if the optimal individual of the new generation has higher fitness than the optimal individual of the previous generation, they are rewarded. Equation (18) shows that the average individual fitness of the new generation is rewarded if it is higher than that of the previous generation.

$$r_c = \frac{\max f(x_i^t) - \max f(x_i^{t-1})}{\max f(x_i^{t-1})} \quad (17)$$

$$r_m = \frac{\sum_{i=1}^N f(x_i^t) - \sum_{i=1}^N f(x_i^{t-1})}{\sum_{i=1}^N f(x_i^{t-1})} \quad (18)$$

#### 4.3. Training Process

Deep Q network (DQN) is a method combining deep learning and reinforcement learning proposed by the DeepMind team [25]. A neural network is used to replace the table of the traditional Q learning algorithm to fit the Q value of state action value function. See Equation (19) for details.  $r_{t+1}$  is the reward obtained at  $t + 1$ ,  $\gamma$  is the discount factor,  $\theta^-$  is the parameter of the target network, and  $\arg\max_{a'} Q(s_{t+1}, a'; \theta^-)$  indicates the action of selecting the maximum Q value.

$$Q_t^{DQN} = r_{t+1} + \gamma Q\left(s_{t+1}, \arg\max_{a'} Q(s_{t+1}, a'; \theta^-); \theta^-\right) \quad (19)$$

In the process of using DQN, using a neural network to select the maximum Q value leads to overestimation, and the estimated value is greater than the real value, which affects the effect of the algorithm. To solve the problem of overestimation, the DeepMind team proposed a double deep Q network (DDQN) [26]. The specific calculation of Q value is shown in Equation (20).  $\theta$  is the main network parameter. The main network is used to select the action, and the target network estimates the Q value, which effectively avoids the problem of overestimation [27].

$$Q_t^{DDQN} = r_{t+1} + \gamma Q\left(s_{t+1}, \arg\max_{a'} Q(s_{t+1}, a'; \theta); \theta^-\right) \quad (20)$$

Based on DQN and DDQN, a dueling double deep Q network (D3QN) introduces a dueling network, improves the structure of a neural network, and adds an advantage function, as shown in Equation (21) [28].  $\theta$  is the shared network parameter,  $\theta^A$  and  $\theta^V$  represent separate network parameters,  $V(s; \theta^V)$  is the state value function, and  $A(s, a; \theta^A)$  is the advantage function. Through the advantage function, we can judge the good or bad degree of the action, which effectively increases the effect of the algorithm.

$$Q(s, a; \theta) = V(s; \theta^V) + \left( A(s, a; \theta^A) - \frac{1}{|A|} \sum_a A(s, a'; \theta^A) \right) \quad (21)$$

Based on the above theory, the pseudocode of the algorithm training process is shown in Algorithm 1.

---

**Algorithm 1 Pseudocode of the RLEGA Algorithm**

---

```

1: Initialize the GA: population size  $N$ , maximum iterations  $T$ 
2: Initialize the RL: replay memory  $D$ , initialize action-value function  $Q$  with random weights  $\theta$ ,
   and initialize target action-value function  $\hat{Q}$  with weights  $\theta^- = \theta$ 
3: for  $t = 1, 2, \dots, T$  do
4:   Calculate status value  $s_t = [f^*, d^*, p^*, S^*]$ 
5:   Randomly select action  $a_t$  with probability  $\varepsilon$ 
6:   Otherwise, choose  $a_t = \underset{a}{\operatorname{argmax}} Q(s_t, a; \theta)$  with  $1 - \varepsilon$  probability
7:   Observe state  $s_{t+1}$  and reward  $r_t$ 
8:   store  $(s_t, a_t, r_t, s_{t+1})$  in  $D$ 
9:   Randomly sample a batch of data from  $D$ 
10:  Set  $y_i = \begin{cases} r_j & \text{if episode terminates at step } j + 1 \\ r_j + \gamma \hat{Q}(s_{t+1}, \underset{a'}{\operatorname{argmax}} Q(s_{t+1}, a'; \theta^-); \theta^-) & \text{otherwise} \end{cases}$ 
11:  Compute  $(y_i - Q(s_j, a_j; \theta))^2$ 
12:  Update weights  $\theta$  using gradient descent
13:  every  $C$  steps reset  $\hat{Q} = Q$ 
13: end

```

---

## 5. Experiment and Analysis

### 5.1. Simulation Results

In order to verify the feasibility and effectiveness of the above algorithm in solving the dynamic flexible job shop, this section designs experiments to solve the dynamic scheduling problem of flexible job shop with random arrival of jobs. Because the dynamic scheduling problem has no benchmark similar to the traditional scheduling problem, this experiment constructs test cases suitable for this algorithm according to the characteristics of the problem.

In order to fully verify the rationality and effectiveness of the algorithm, this experiment generates test examples of different sizes. The number of processing jobs is divided into 20 and 100 jobs, and the corresponding number of processing machines is 10 and 20, respectively. For the number of jobs of different sizes, the number of processes also obey the discrete uniform distribution of  $[1, 10]$ . The processing time of each job follows the uniform distribution of  $[1, 100]$ . Since the job arrives randomly, assuming that the arrival of the jobs is a Poisson process, the arrival time interval of the job follows the exponential distribution, and the average arrival time interval is shown in Equation (22):

$$\lambda = \frac{\sum_{p=1}^n \sum_{q=1}^{o_p} P_{pq}}{mn\eta} \quad (22)$$

In the above formula,  $\lambda$  Represents the average interval between the arrival of jobs,  $P_{pq}$  represents the processing time of the  $q$ th operation of job  $p$ ,  $o_p$  is the number of operations of job  $p$ ,  $m$  represents the number of machines,  $n$  represents the number of jobs, and  $\eta$  represents machine utilization. In this experiment, the machine utilization rate is 75% and 90%. There are three kinds of flexibility: 20%, 50%, and 100%

In order to better compare the effect of RLEGA, ordinary GA, tabu search (TS), a commonly used heuristic algorithm, and some scheduling rules are selected for comparison in flexible job shop, machine design selection, and job selection rules. In this experiment, we select the machine selection rule as LWT (least waiting time), which means we select the machine with the smallest total processing time waiting for the processing process. In terms of job selection, we have selected SPT (shortest processing time)—the shortest process processing time; LPT (longest processing time)—the longest process time; SSO

(shortest processing time of subsequence operation)—the processing time of the remaining processes is the shortest; LSO (longest processing time of subsequence operation)—four different scheduling rules with the longest processing time of the remaining operation are combined. Through comparison, the effectiveness of RLEGA is fully verified.

The parameters set by each algorithm are shown in Table 3. On the generated test cases, the operation results of each algorithm are shown in Table 4. Taking S1\_20\_10\_75%\_20% as an example, the number of test case\_jobs\_machines\_ machine utilization\_ flexibility. Each algorithm runs 10 times, and the average value, maximum value, and minimum value are shown in the table. According to the operation results, the box diagram can be obtained, as shown in Figure 5. It can be seen from the results that the results of RLEGA on each test case are better than ordinary GA, TS algorithm and various scheduling rules. The minimum, maximum and average values of only one test case are slightly larger than ordinary genetic algorithm. Therefore, we can conclude that using the reinforcement learning enhanced genetic algorithm has a good effect in solving the dynamic scheduling problem of flexible job shop.

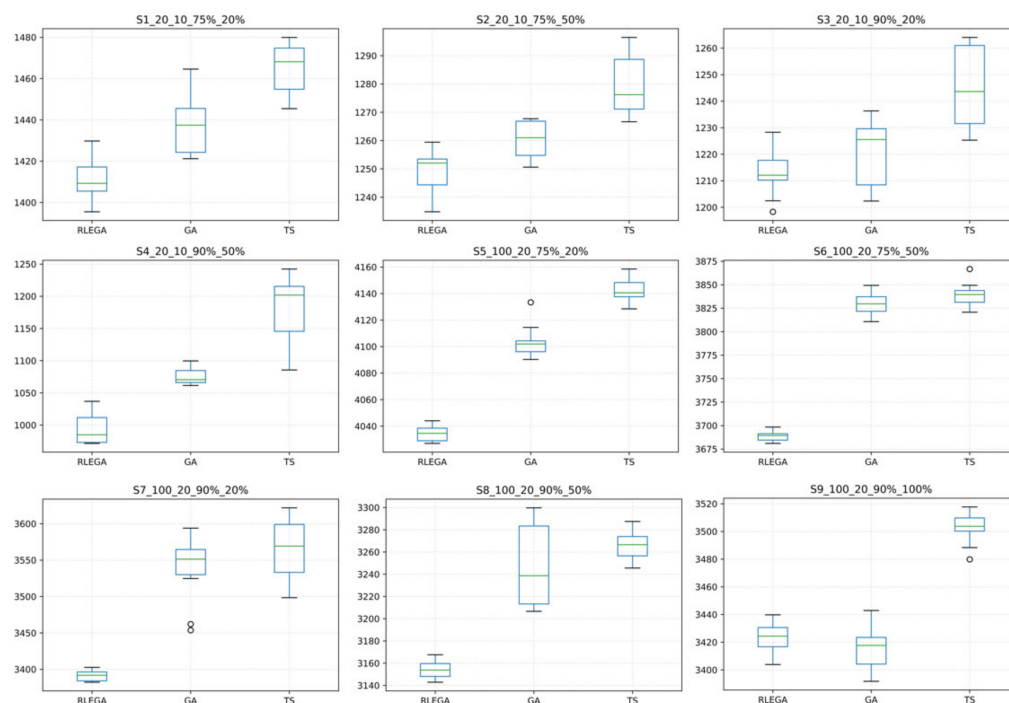


Figure 5. Box diagram.

Table 3. The parameters of algorithm.

Parameter	Value
RLEGA	
Number of iterations	1000
Learning rate	$10^{-3}$
Discount rate	0.95
Batch size	128
Buffer size	100,000
Greedy rate	0.9
TS	
Number of iterations	500
Preset probability	0.5
Taboo table length	10

**Table 4.** Operation results.

Test Case	RLEGA			GA			TS			LWT + SPT	LWT + LPT	LWT + SSO	LWT + LSO
	Avg	Max	Min	Avg	Max	Min	Avg	Max	Min	—	—	—	—
S1_20_10_75%_20%	1411.50	1429.79	1395.49	1437.32	1464.59	1421.19	1464.63	1479.95	1445.47	1575.62	1602.94	1557.72	1620.02
S2_20_10_75%_50%	1249.41	1259.43	1234.84	1261.00	1267.74	1250.62	1280.08	1296.46	1266.64	1520.51	1480.45	1540.22	1534.42
S3_20_10_90%_20%	1213.56	1228.29	1198.31	1220.30	1236.31	1202.35	1245.30	1264.03	1225.31	1402.51	1450.21	1395.66	1380.61
S4_20_10_90%_50%	993.15	1036.82	971.48	1075.19	1099.60	1061.45	1178.12	1242.36	1085.55	1320.94	1360.73	1376.72	1342.61
S5_100_20_75%_20%	4036.58	4044.01	4026.97	4103.53	4133.43	4090.29	4142.77	4158.52	4128.5	4755.28	4757.21	4927.13	4838.95
S6_100_20_75%_50%	3688.92	3698.43	3681.02	3829.52	3849.34	3810.76	3839.14	3866.76	3820.79	4668.86	4681.28	4601.55	4797.52
S7_100_20_90%_20%	3391.53	3402.65	3382.3	3538.60	3593.95	3453.70	3565.44	3621.85	3498.41	4396.96	4273.13	4029.85	4341.45
S8_100_20_90%_50%	3154.40	3167.56	3142.88	3247.31	3299.77	3206.68	3266.01	3287.50	3245.60	4164.51	4033.08	4166.68	4013.11
S9_100_20_90%_100%	3423.31	3439.81	3403.85	3415.46	3442.91	3391.76	3502.81	3517.72	3479.84	4287.91	4213.23	4333.08	4233.95

## 5.2. Case Study

Finally, taking the production content of the laboratory as an example. Our laboratory is a workshop for producing personalized customized cars, which includes eight models. Each model includes five processes: chassis, frame, left door, right door, and front cover. Each process can be processed in different machines at different times, which is in line with the flexible job-shop scheduling problem. The available machines and corresponding time for the assembly of various models are shown in Tables 5 and 6.

**Table 5.** Machines available for each model.

Job	O <sub>1</sub>	O <sub>2</sub>	O <sub>3</sub>	O <sub>4</sub>	O <sub>5</sub>
J <sub>1</sub>	[3,8]	[1,7]	[1,4]	[2,8]	[6,7]
J <sub>2</sub>	[2,8]	[4,7]	[3,5]	[1,3]	[2,3]
J <sub>3</sub>	[1,4,6]	[4,5]	[2,5]	[3,8]	[7]
J <sub>4</sub>	[4,8]	[1,2]	[6,8]	[6]	[6,7]
J <sub>5</sub>	[1,6]	[2,5]	[1,4]	[2,7]	[3,8]
J <sub>6</sub>	[1,2,4,7]	[1,6]	[4,8]	[1,3]	[6]
J <sub>7</sub>	[1,6]	[2,5]	[1,4]	[7,8]	[3,7]
J <sub>8</sub>	[2,5,7]	[2,4]	[5,8]	[1,3]	[3,8]

**Table 6.** Time required for assembly of each model.

Job	O <sub>1</sub>	O <sub>2</sub>	O <sub>3</sub>	O <sub>4</sub>	O <sub>5</sub>
J <sub>1</sub>	[38,49]	[72,54]	[49,65]	[76,59]	[73,43]
J <sub>2</sub>	[50,49]	[53,41]	[62,66]	[51,42]	[49,70]
J <sub>3</sub>	[48,60,68]	[53,59]	[61,66]	[42,59]	[43]
J <sub>4</sub>	[60,49]	[72,85]	[59,66]	[30]	[73,69]
J <sub>5</sub>	[35,68]	[42,69]	[67,49]	[42,30]	[70,88]
J <sub>6</sub>	[43,35,32,57]	[68,67]	[56,93]	[68,105]	[79]
J <sub>7</sub>	[48,32]	[85,66]	[49,43]	[51,73]	[102,52]
J <sub>8</sub>	[50,43,57]	[85,53]	[66,60]	[94,100]	[90,71]

The results of running with different algorithms are shown in Table 7. Each algorithm runs 5 times to obtain the maximum, minimum, and average values. As shown in Table 7, the minimum and average values of the results obtained by RLEGA algorithm are better than GA and TS. The parameter settings of each algorithm are shown in Section 5.1. Therefore, RLEGA algorithm is used in this experiment.

**Table 7.** Algorithm running results.

	RLEGA	GA	TS
Maximum	404	433	466
Average	400	420	453.6
Minimum	397	411	435

The initial Gantt chart is obtained according to the RLEGA algorithm, as shown in Figure 6. The abscissa in the figure is time (s) and the ordinate is the robot arm. Take 7-4 as an example to show the fourth process of the seventh job. The curve in the solution process of reinforcement learning enhanced genetic algorithm is shown in Figure 7.

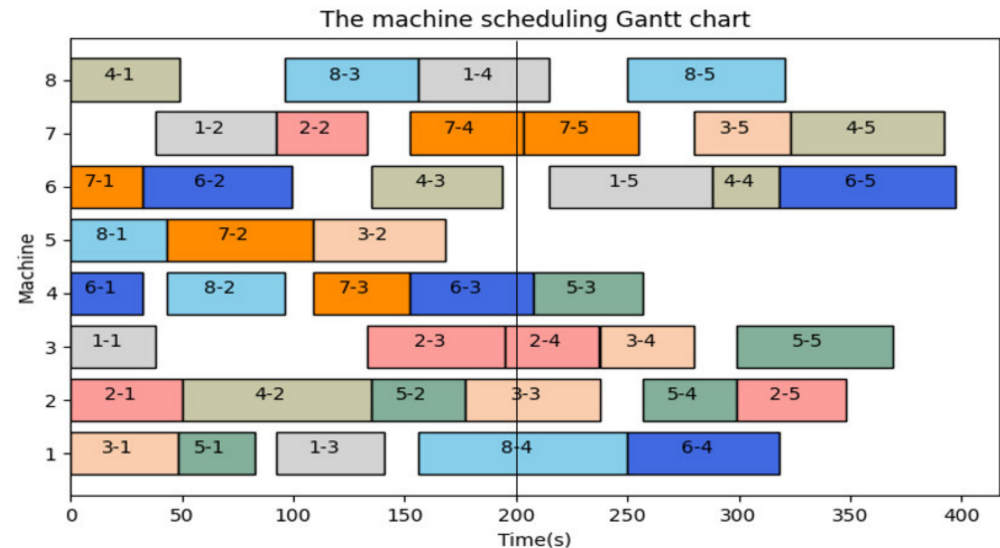


Figure 6. Initial Gantt chart.

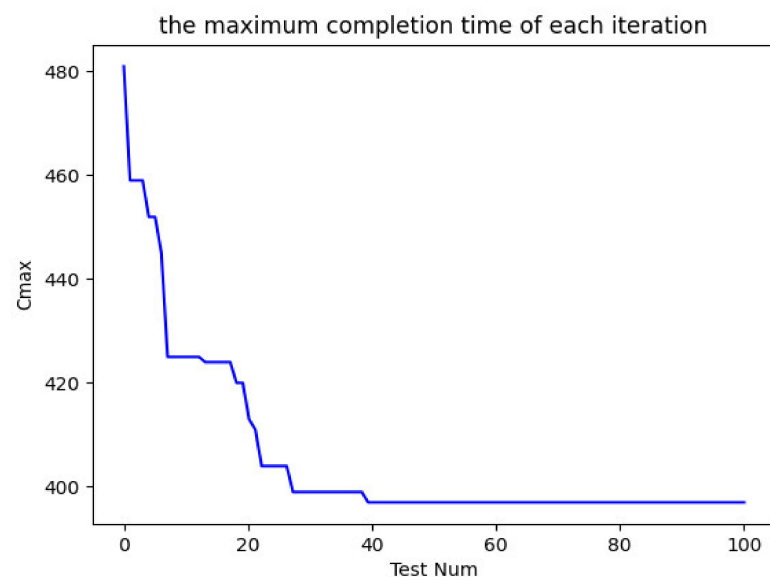


Figure 7. Iterative curve.

As shown in Figure 6, assuming that a new order is issued through the ordering system when  $t = 200$  s, the dynamic event of inserting a new order occurs in the ordering system as shown in Figure 8a. At this time, the order information is sent to the database. The digital twin system senses the insertion of the new order, triggers the rescheduling mechanism, and processes the new order together with the previously unfinished order, and the model being assembled cannot be interrupted. The resulting rescheduling Gantt chart is shown in Figure 9. The red box in the figure shows the task of a newly issued eighth model. The initial solution is 397 s, and the obtained rescheduling solution is 521 s.



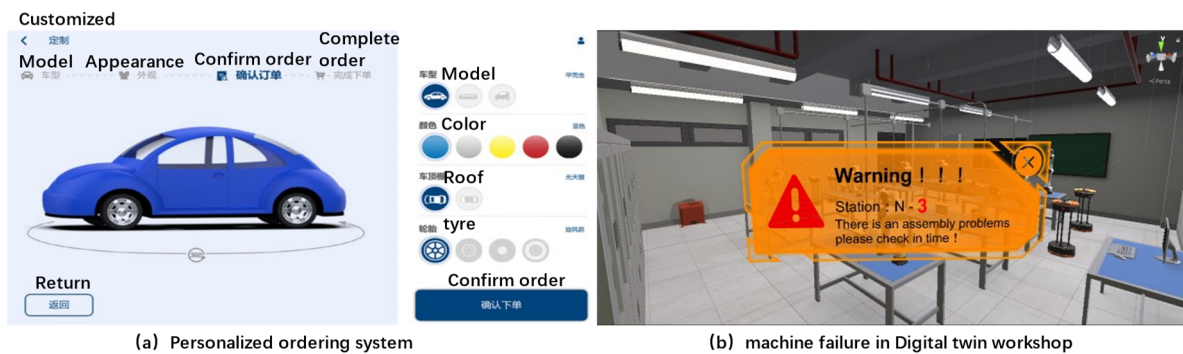


Figure 8. Initial Gantt chart. (a) Personalized ordering system. (b) Machine failure in Digital twin workshop.

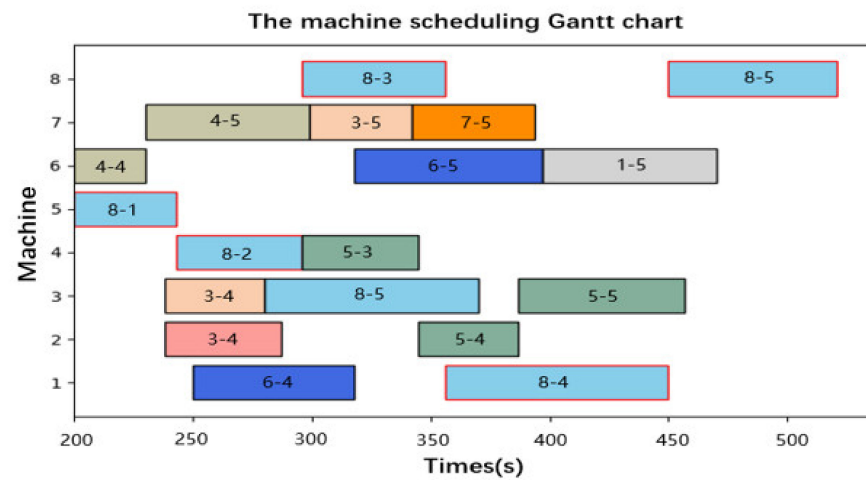


Figure 9. Rescheduling Gantt chart (insert new order).

As shown in Figure 8b, or when  $t = 200$  s, the machine 3 breaks down, obtains the abnormal information through the digital twin system, as shown in Figure 10, captures the dynamic event, and triggers the rescheduling mechanism, and the resulting rescheduling Gantt chart is shown in Figure 10. After the failure of machine 3, the job originally processed at machine 3 is rearranged to other machines.

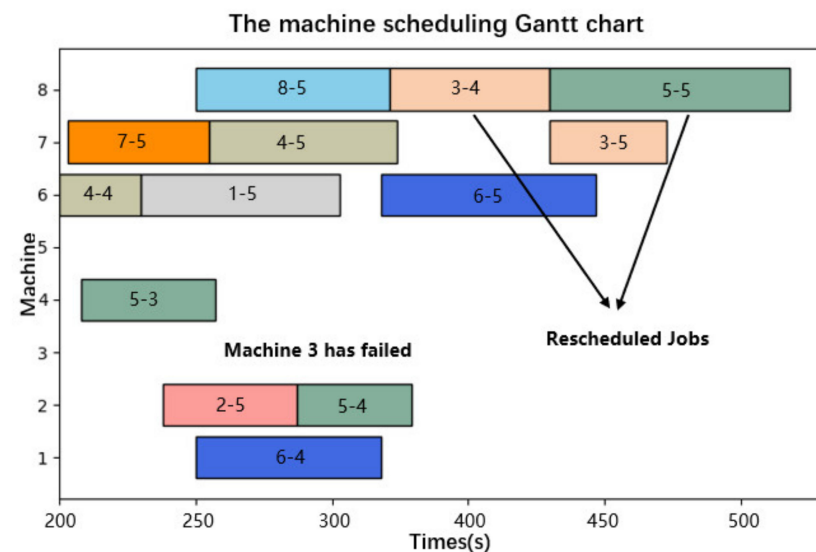


Figure 10. Rescheduling Gantt chart (machine fault).

## 6. Conclusions

Digital twin provides a new solution for effectively realizing the interaction and integration of the real world and the information world. This paper presents a digital twin-driven adaptive scheduling method for flexible job shop. This method can realize the effective combination of real-time monitoring of production process, scheduling optimization of production tasks, and real-time perception of abnormal events in production process, effectively improving the production efficiency.

The RLEGA scheduling model established in this paper has high solution quality in the production process. When dynamic events occur in the workshop, adaptive dynamic scheduling is carried out based on RLEGA. Compared with genetic algorithm, tabu search and various scheduling rules, RLEGA has higher solution quality and can well solve the dynamic events in the actual production process.

At present, reinforcement learning is only applied to optimize the key parameters of genetic algorithms, and reinforcement learning is not used to solve the production scheduling problem. Establishing a digital twin workshop with all elements and only using reinforcement learning method to solve the scheduling problem is the focus of future work. In the future work, we only use reinforcement learning to solve the flexible job-shop scheduling problem, make use of the advantages of fast decision making of the model trained by reinforcement learning, and combine the digital twin workshop to achieve a faster real-time response.

**Author Contributions:** Conceptualization, L.L. and Z.G.; Data curation, J.L.; Formal analysis, K.G.; Methodology, L.L., Z.G. and J.S.; Project administration, L.L. and K.G.; Resources, J.L.; Software, K.G. and J.L.; Supervision, Z.G.; Visualization, J.S.; Writing—original draft, L.L. and K.G.; Writing—review & editing, Z.G. All authors have read and agreed to the published version of the manuscript.

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

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** All data underlying the results are available as part of the article and no additional source data are required.

**Acknowledgments:** The authors thanks Shanghai Key Laboratory of intelligent manufacturing and robotics of Shanghai University for its support for this study.

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

## References

1. Jazdi, N. Cyber physical systems in the context of Industry 4.0. In Proceedings of the 2014 IEEE International Conference on Automation, Quality and Testing, Robotics, Cluj-Napoca, Romania, 22–24 May 2014; pp. 14–16. [\[CrossRef\]](#)
2. Mosterman, P.J.; Zander, J. Industry 4.0 as a Cyber-Physical System study. *Softw. Syst. Model.* **2016**, *15*, 17–29. [\[CrossRef\]](#)
3. Liu, S.X. Innovation Design: Made in China 2025. *Des. Manag. Rev.* **2016**, *27*, 53–58.
4. Saracco, R. Digital Twins: Bridging Physical Space and Cyberspace. *Computer* **2019**, *52*, 58–64. [\[CrossRef\]](#)
5. Tao, F.; Zhang, M. Digital Twin Shop-Floor: A New Shop-Floor Paradigm towards Smart Manufacturing. *IEEE Access* **2017**, *5*, 20418–20427. [\[CrossRef\]](#)
6. Milton, M.; De La, O.C.; Ginn, H.L.; Benigni, A. Controller-Embeddable Probabilistic Real-Time Digital Twins for Power Electronic Converter Diagnostics. *IEEE Trans. Power Electron.* **2020**, *35*, 9850–9864. [\[CrossRef\]](#)
7. Mi, S.; Feng, Y.; Zheng, H.; Wang, Y.; Gao, Y.; Tan, J. Prediction maintenance integrated decision-making approach supported by digital twin-driven cooperative awareness and interconnection framework. *J. Manuf. Syst.* **2021**, *58*, 329–345. [\[CrossRef\]](#)
8. Song, Y.; Shi, Y.Y.; Yu, J.S.; Tang, D.Y.; Tao, F. Application of digital twin model in performance prediction of electro-optical detection system. *Comput. Integr. Manuf. Syst.* **2019**, *25*, 1559–1567.
9. Fedorko, G.; Molnár, V.; Vasil', M.; Salai, R. Proposal of digital twin for testing and measuring of transport belts for pipe conveyors within the concept Industry 4.0. *Measurement* **2021**, *174*, 108978. [\[CrossRef\]](#)

10. Lu, C.; Wang, J.; Yin, P.; Wang, L. Error identification of measurement software based on digital twin of gear measuring center. *Measurement* **2021**, *173*, 108666. [[CrossRef](#)]
11. Brucker, P.; Schlie, R. Job-shop scheduling with multi-purpose machines. *Computing* **1990**, *45*, 369–375. [[CrossRef](#)]
12. Tian, S.; Wang, T.; Zhang, L.; Wu, X. Real-time shop floor scheduling method based on virtual queue adaptive control: Algorithm and experimental results. *Measurement* **2019**, *147*, 106689. [[CrossRef](#)]
13. Reddy, M.B.S.S.; Ratnam, C.; Rajyalakshmi, G.; Manupati, V.K. An effective hybrid multi objective evolutionary algorithm for solving real time event in flexible job shop scheduling problem. *Measurement* **2018**, *114*, 78–90. [[CrossRef](#)]
14. Yu, T.; Zhou, J.; Fang, J.; Gong, Y.; Wang, W. Dynamic scheduling of flexible job shop based on genetic algorithm. In Proceedings of the 2008 IEEE International Conference on Automation and Logistics, Qingdao, China, 1–3 September 2008; pp. 2014–2019. [[CrossRef](#)]
15. Chang, H.-C.; Liu, T.-K. Optimisation of distributed manufacturing flexible job shop scheduling by using hybrid genetic algorithms. *J. Intell. Manuf.* **2017**, *28*, 1973–1986. [[CrossRef](#)]
16. Dolgui, A.; Ivanov, D.; Sethi, S.P.; Sokolov, B. Scheduling in production, supply chain and Industry 4.0 systems by optimal control: Fundamentals, state-of-the-art and applications. *Int. J. Prod. Res.* **2019**, *57*, 411–432. [[CrossRef](#)]
17. Wang, J.; Zhang, Y.; Liu, Y.; Wu, N. Multiagent and Bargaining-Game-Based Real-Time Scheduling for Internet of Things-Enabled Flexible Job Shop. *IEEE Internet Things J.* **2019**, *6*, 2518–2531. [[CrossRef](#)]
18. Fang, Y.; Peng, C.; Lou, P.; Zhou, Z.; Hu, J.; Yan, J. Digital-Twin-Based Job Shop Scheduling Toward Smart Manufacturing. *IEEE Trans. Ind. Inform.* **2019**, *15*, 6425–6435. [[CrossRef](#)]
19. Liu, Z.; Chen, W.; Zhang, C.; Yang, C.; Cheng, Q. Intelligent scheduling of a feature- process-machine tool supernetwork based on digital twin workshop. *J. Manuf. Syst.* **2021**, *58*, 157–167. [[CrossRef](#)]
20. Luo, S. Dynamic scheduling for flexible job shop with new job insertions by deep reinforcement learning. *Appl. Soft Comput.* **2020**, *91*, 106208. [[CrossRef](#)]
21. Zhang, M.; Tao, F.; Nee, A.Y.C. Digital Twin Enhanced Dynamic Job-Shop Scheduling. *J. Manuf. Syst.* **2021**, *58*, 146–156. [[CrossRef](#)]
22. Tao, F.; Zhang, H.; Qi, Q.; Xu, J.; Sun, Z.; Hu, T.; Liu, X.; Liu, T.; Guan, J.; Chen, C.; et al. Construction theory and application of digital twin model. *Comput. Integr. Manuf. Syst.* **2021**, *27*, 1–15. [[CrossRef](#)]
23. Zhang, C.; Tao, F. Evaluation index system of digital twin model. *Comput. Integr. Manuf. Syst.* **2021**, *27*, 2171–2186. [[CrossRef](#)]
24. Wu, Y.; Yao, L.Y.; Xiong, H.; Zhuang, C.B.; Zhao, H.R.; Liu, J.H. Quality control method of complex product assembly process based on digital twin technology. *Comput. Integr. Manuf. Syst.* **2019**, *25*, 1568–1575.
25. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [[CrossRef](#)] [[PubMed](#)]
26. Van Hasselt, H. Double Q-learning. In *Advances in Neural Information Processing Systems*; MIT Press: Vancouver, BC, Canada, 2010; pp. 2613–2621.
27. Van Hasselt, H.; Guez, A.; Silver, D. Deep reinforcement learning with double Q-learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Menlo Park, CA, USA, 12–17 February 2016; pp. 2094–2100.
28. Wang, Z.; Schaul, T.; Hessel, M.; Hasselt, H.; Lanctot, M.; Freitas, N. Dueling network architectures for deep reinforcement learning. In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; Volume 48, pp. 1995–2003.