machines

MDPI

Article

Research on Scheduling Algorithm of Knitting Production Workshop Based on Deep Reinforcement Learning

Lei Sun [1], Weimin Shi [1,*], Chang Xuan [1] and Yongchao Zhang [1,2]

[1] Key Laboratory of Modern Textile Machinery & Technology of Zhejiang Province, Zhejiang Sci-Tech University, Hangzhou 310018, China
[2] School of Automation, Zhejiang Institute of Mechanical & Electrical Engineering, Hangzhou 310053, China
* Correspondence: swm@zstu.edu.cn

Abstract: Intelligent scheduling of knitting workshops is the key to realizing knitting intelligent manufacturing. In view of the uncertainty of the workshop environment, it is difficult for existing scheduling algorithms to flexibly adjust scheduling strategies. This paper proposes a scheduling algorithm architecture based on deep reinforcement learning (DRL). First, the scheduling problem of knitting intelligent workshops is represented by a disjunctive graph, and a mathematical model is established. Then, a multi-proximal strategy (multi-PPO) optimization training algorithm is designed to obtain the optimal strategy, and the job selection strategy and machine selection strategy are trained at the same time. Finally, a knitting intelligent workshop scheduling experimental platform is built, and the algorithm proposed in this paper is compared with common heuristic rules and metaheuristic algorithms for experimental testing. The results show that the algorithm proposed in this paper is superior to heuristic rules in solving the knitting workshop scheduling problem, and can achieve the accuracy of the metaheuristic algorithm. In addition, the response speed of the algorithm in this paper is excellent, which meets the production scheduling needs of knitting intelligent workshops and has a good guiding significance for promoting knitting intelligent manufacturing.

Keywords: knitting workshop scheduling; deep reinforcement learning; multi-proximal policy optimization; flexible job-shop scheduling problem; intelligent manufacturing

Citation: Sun, L.; Shi, W.; Xuan, C.; Zhang, Y. Research on Scheduling Algorithm of Knitting Production Workshop Based on Deep Reinforcement Learning. *Machines* **2024**, *12*, 579. https://doi.org/10.3390/machines12080579

Academic Editor: Angelos P. Markopoulos

Received: 25 July 2024
Revised: 19 August 2024
Accepted: 21 August 2024
Published: 22 August 2024

1. Introduction

Digitalization, networking, and intelligence have become the main development lines of knitting [1]. With the development of emerging technologies such as the Internet of Things, artificial intelligence, and cyber-physical systems, the knitting industry has also begun to transform towards intelligence, and the production model of enterprises has also evolved from the original assembly line production to multi-variety, small batch, and customized production [2]. Traditional centralized and static scheduling algorithms can no longer meet the high flexibility and reconfigurability requirements of production.

In a knitting intelligent workshop, the processes of order production, fabric cutting, fabric dropping, fabric transportation, and fabric inspection are completed by the workshop management system according to the process flow. Therefore, the knitting workshop scheduling problem belongs to the typical flexible job-shop scheduling problem (FJSP), in which at least one process has multiple parallel machines that can be processed simultaneously, and it has been proved to be an NP-hard combinatorial optimization problem [3]. In FJSP, a job consists of a series of specific continuous operations, in which each operation is assigned to a machine in a group of parallel machines to optimize one or more objectives, such as operation time, average completion time, maximum flow time, and total delay time [4]. Compared with the traditional job shop scheduling problem, the flexible job shop scheduling problem is more difficult to solve and has higher engineering application value.

Currently, existing methods for solving NP-hard combinatorial optimization problems can be classified into two categories: exact methods and approximate methods. Exact

optimization algorithms, such as mathematical solutions and integer linear programming, search for the optimal solution in the entire solution space. These methods are not suitable for solving large scheduling problems within a reasonable time [5]. Therefore, more and more approximate optimization algorithms, including heuristic methods, meta-heuristic methods, and machine learning, have been developed to solve practical problems. In particular, swarm intelligence (SI) and evolutionary algorithms (EAs), such as genetic algorithms (GAs) [6,7], particle swarm optimization (PSO) [8,9], and the firefly algorithm (FA) [10], have shown outstanding advantages in solving FJSP instances.

Although SI and EAs can solve FJSP in a reasonable time compared with the exact mathematical optimization methods, they are not applicable in the actual knitting production scheduling because the algorithm iterations require a lot of iterative computing time to obtain a satisfactory solution. In general, the scheduling rules used to solve FJSP can be divided into two categories: job selection rules and machine selection rules. Efficient scheduling rules require a lot of domain expertise and trial and error, and cannot guarantee local optimality. Deep reinforcement learning (DRL) algorithms provide a scalable method for solving scheduling problems with common characteristics. This motivates us to use reinforcement learning-based methods for FJSP.

For the production scheduling problem of knitting intelligent workshops, this paper considers this problem as a combinatorial optimization problem and optimally arranges production orders by allocating machines and equipment to satisfy the production requirements. Usually in DRL, the agent interacts with the environment according to the following behavior: the agent first accepts the state s_t, and selects the action a_t according to the state of each step, then obtains the reward r_t, and moves to the next state s_{t+1}. In DRL, the action a_t is selected from the action space. However, the action space in the knitting intelligent workshop scheduling problem is constructed by the job operation action space and the machine action space. In order to solve this problem, the DRL architecture agent in this paper will first select an operation action from the job operation space, and then select a machine action from the machine space for the selected job operation action. The main contributions of this paper are as follows:

A deep reinforcement learning model applied to knitting intelligent workshops is designed, and multi-action space is introduced into the Markov decision process representation, the disjunctive graph representation of the knitting intelligent workshop scheduling problem is introduced, and the graph neural network is embedded in the local state.

The multi-proximal policy optimization (multi-PPO) algorithm is used to train the job operation action strategy and machine action strategy of knitting smart workshop scheduling. The effectiveness of the algorithm framework is verified by orders in actual production.

The paper is organized as follows: Section 2 describes the related work of knitting intelligent workshop scheduling solution. Section 3 explains the mathematical model of knitting intelligent workshop production scheduling. Section 4 proposes a DRL algorithm framework through a multi-strategy algorithm. Section 5 builds a knitting intelligent production experimental platform to compare and validate the algorithm. Section 6 introduces the conclusion.

2. Literature Review

When solving the flexible job-shop scheduling problem, exact methods based on mathematical solutions have been applied to find small FJSP instances. Initially, Shapiro [11] proposed a mathematical programming model and solution method. Özgüven [12] developed a mixed integer linear programming model (MILP-1) and improved MILP-1, which improved the solution time of the model. Meng [13] established a mixed integer linear programming (MILP) model to solve the FJSP-SDST-T energy minimization problem. Although the medium-scale solution efficiency is high, the running memory requirements are relatively high and the solution time is relatively long. Although various exact methods have been successful, they are not suitable for solving large FJSP instances due to the NP-hardness of FJSP.

In recent years, approximate solutions such as SI and EAs are usually used to solve scheduling problems. Although the computation time of SI and EA methods is less than that of exact solutions, they are not competent in real-time scheduling environments. In order to solve scheduling problems in practice, a series of scheduling rules have been designed. Yang [14] proposed an improved dragonfly algorithm (DA), which adopts a dynamic reverse learning strategy to improve the search ability of DAs. Sun [15] improved the genetic algorithm by using a variable neighborhood search algorithm, which improves the local search ability of the algorithm and accelerates the convergence speed of the algorithm. However, the algorithm only considered the influence of crossover probability and mutation probability on the performance of the algorithm. Pan [16] considered energy saving, proposed a dual-population evolutionary algorithm with feedback, and conducted a large number of experiments to verify the superiority of the evolutionary algorithm with feedback. As the scale of FJSP instances increases, the meta-heuristic algorithm needs to go through a large number of iterations and consumes more computing time. Therefore, the computational complexity of the meta-heuristic algorithm is an obvious limiting factor.

Meanwhile, some reinforcement learning algorithms have achieved remarkable results in board and video games [17], especially successful examples of DRL from simulations to the real world [18], which have become another option for solving FJSP. Shang [19] proposed a combination of deep learning network LSTM and genetic algorithms, and experiments show that the algorithm significantly improves the utilization of equipment, but the predictive processing level of the algorithm is insufficient. Wang [20] proposed a dynamic scheduling method based on DRL, which uses proximal policy optimization (PPO) to find the optimal scheduling policy, which reduces the complexity and the solution results are better than heuristic rules and meta-heuristics algorithm.

There are more and more related studies on workshop production scheduling in the fields of manufacturing, production planning, and logistics [18]. Methods based on mathematical solutions and metaheuristics cannot be applied due to the complexity of large FJSP problems. Many researchers have solved combinatorial optimization problems through DRL and achieved good results, but the scheduling of knitting intelligent workshops has received less attention. Most of the methods for solving FJSP based on DRL choose composite scheduling rules instead of directly finding scheduling solutions. The performance of the selected composite scheduling rules depends on the design of the scheduling rules. The literature shows that there are few studies on solving knitting intelligent workshop scheduling through a multi-action DRL framework without predetermined scheduling rules. To this end, this paper designs a DRL framework based on the multi-PPO algorithm to solve the knitting intelligent workshop scheduling problem. It can be directly trained to find the optimal strategy, and the effectiveness of the algorithm architecture is verified.

3. Mathematical Modeling of Production Scheduling Problem in Knitting Workshop

3.1. System Model

In this section, the knitting intelligent workshop production scheduling problem is defined as a flexible job shop scheduling problem, which can be described as follows: given a set $J = \{J_1, J_2, \cdots, J_n\}$ consisting of n jobs and a set $M = \{M_1, M_2, \cdots, M_m\}$ consisting of m machines, each job contains one or more processes, and the order of the processes is predetermined. Each process can be processed on multiple parallel machines, and the processing time of the process varies with the processing machines. In addition, in order to satisfy the actual production requirements, this problem needs to meet the following constraints:

1. The same machine can only process a maximum of one workpiece at a given time;
2. The same job can only be processed by one machine at the same time in the same process;
3. Each process of each job cannot be interrupted once it starts (that is, each process is considered to be non-preemptive);
4. Different artifacts have the same priority;

5. There are no priority constraints between the processes for different jobs, but there are sequential constraints between processes for the same job;
6. All jobs and machines are available within the dispatch scope until the dispatch is completed, regardless of equipment failures.

In the knitting intelligent workshop model scheduling problem, the optimization objective is to assign job operations to machines for processing and to determine the operation sequence on the machines to minimize the completion time makespan C_{max}, which is expressed by Equation (1), where C_{in_i} represents the completion time of job i.

$$C_{max} = \max\{C_{in_i}\},\ i \in \{1, 2, \cdots, n\} \tag{1}$$

3.2. Problem Formulation

In the production scheduling problem of knitting intelligent workshops, the goal of scheduling is to select the most suitable machine for each process, and determine the optimal processing sequence and start time of each process on each machine, so that certain performance indicators of the entire system can be optimized. Therefore, this knitting intelligent production scheduling problem contains two sub-problems: determining the processing machine for each job (machine selection sub-problem) and determining the processing sequence on each machine (process sequencing sub-problem). For the convenience of the following description, the variable symbols of the knitting shop scheduling model are defined as shown in Table 1:

Table 1. Mathematical symbol definition table.

Symbol	Definition
n	Total number of jobs
m	Total number of machines
i	Machine number, $i = 1,2,3,\cdots,m$
j	Job number, $j = 1,2,3,\cdots,n$
h_j	Number of processes for job j
h	Process number, $h = 1,2,3,\cdots,h_j$
M_{jh}	The set of optional processing machines for process h of job j
m_{jh}	The number of optional processing machines for process h of job j
O_{jh}	The process h for job j
P_{ijh}	The process h of job j is processed on machine i
T_{ijh}	The processing time on machine i for process h of job j
ST_{jh}	The processing start time of process h of job j
ET_{jh}	The processing completion time of process h of job j
Et_j	The processing completion time of job j
C_{max}	The maximum completion time
N_O	The total number of processes for all jobs, $N_O = \sum_{j=1}^{n} h_j$
x_{ijh}	$x_{ijh} = \begin{cases} 1 & \text{if machine i is selected for operation } O_{jh} \\ 0 & \text{otherwise} \end{cases}$

The processing time, i.e., the total time required to complete all production jobs, is used as a measure of the quality of the scheduling algorithm. Mathematically, we define the knitting intelligent workshop production scheduling problem to satisfy the following constraints:

$$ST_{jh} + x_{ijh} \times T_{ijh} \leq ET_{jh} \tag{2}$$

$$ET_{jh} \leq ST_{j(h+1)} \tag{3}$$

$$Et_j \leq C_{max} \tag{4}$$

$$ST_{jh} \leq ST_{(j+1)h} \tag{5}$$

$$ET_{jh} \leq ST_{j(h+1)} \tag{6}$$

$$\sum_{i=1}^{m_{jh}} x_{ijh} = 1 \tag{7}$$

$$ST_{jh} \geq 0, ET_{jh} \geq 0 \tag{8}$$

Among them, constraints (2) and (3) represent the sequence constraints of the processes of each job; constraint (4) represents the completion time constraint of the job (that is, the completion time of each job does not exceed the total completion time); constraints (5) and (6) represent that only one process can be processed by the same machine at the same moment; constraint (7) represents the machine constraint (that is, the same process can only be processed by one machine at the same moment); and constraint (8) represents the processing condition constraint (that is, the processing parameter variables of each job must be positive numbers).

The knitting intelligent workshop scheduling problem can be represented by a disjunctive graph $G = (O, C, D)$, where $O = \{O_{jh}\}$; C represents the processing priority constraints between different processes of the same job, which is a set of connecting arcs; and D is a set of disjunctive arcs, in which the processes connected by each arc are processed by the same machine equipment. Taking the 3×3 FJSP as an example as shown in Figure 1, the upper part of the figure is a disjunctive graph, and the lower part is a set of feasible solutions. The black line represents the connecting arc, the colored line represents the non-connecting arc, and the colored line with the arrow represents the processing order of the assigned machine.

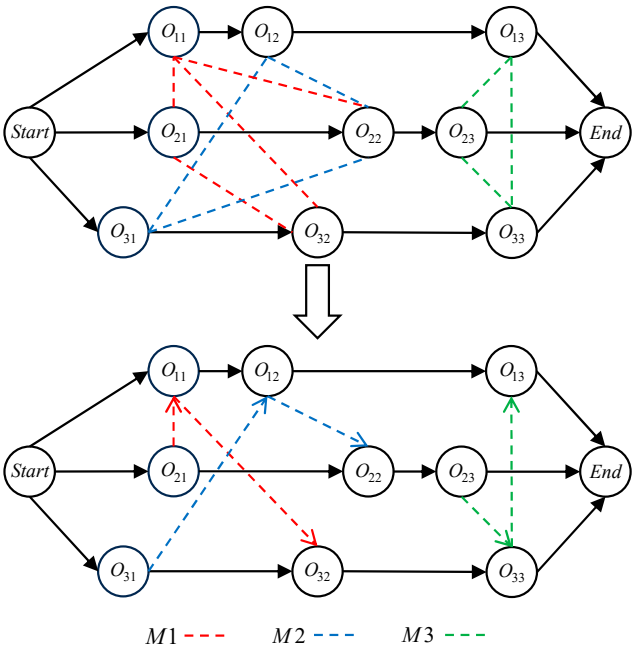

Figure 1. FJSP instance disjunction graph.

4. DRL Architecture for the Knitting Workshop Production Scheduling Problem

Reinforcement learning is one of the most powerful tools for solving sequential decision problems. In this section, the knitting intelligent workshop scheduling process is conceptualized as a multi-action reinforcement learning task, which is defined as a multi-Markov decision process (MMDP). Therefore, we propose a multi-pointer graph network (MPGN) algorithm [21] based on Graph Neural Networks (GNNs), and train the MPGN through a multi-proximal strategy optimization (multi-PPO) algorithm with an actor-critic architecture to solve multi-task combinatorial optimization problems such as knitting intelligent workshop scheduling.

Standard reinforcement learning based on Markov decision process (MDP) is described by a tuple $\langle S, A, P, R, \gamma \rangle$, where S is a finite set of states; A is a finite set of actions; P is the state transition probability; R is the reward function; and γ is a discount factor, which is used to calculate the cumulative return. The agent interacts with the environment at each discrete time step. At time step t, the agent accepts state s_t from the state space S and takes action a_t from the action space A. Then, the agent receives a reward r_t from the environment and enters the next state s_{t+1} according to the transition probability distribution P and s_t and a_t. The policy function π:S×A→[0,1] is defined as a probability density function, where $\pi(a|s) = P(A = a|S = s)$ is the probability of choosing action a given state s. The goal of reinforcement learning is to learn a strategy that maximizes the expected return $R(\pi) = E_\pi[\sum_t \gamma^t r_t]$, where $\gamma \in [0,1]$ is the discount factor.

4.1. Problem Setting

The core of the knitting intelligent workshop scheduling problem is to guide the production order sequencing and processing machine selection in the knitting intelligent workshop through a reasonable job action policy and machine action policy. The multi-task scheduling diagram is shown in Figure 2. This section regards the scheduling process as a sequential decision task. The agent controls multiple actions at the same time to realize the scheduling of the knitting intelligent workshop. This multi-task reinforcement learning problem is an MMDP defined by a tuple $\langle S, A, P, R, \gamma \rangle$. In this section, the knitting intelligent workshop scheduling process is modeled by MMDP, and then the policy network is constructed.

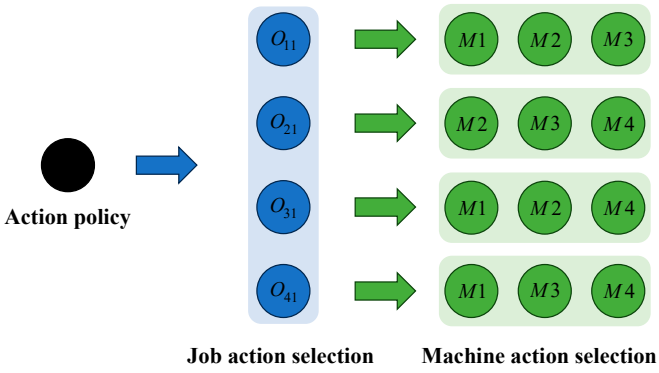

Figure 2. Multi-task scheduling diagram.

States: The scheduling state at a certain moment is approximately described by extracting important features of the scheduling process. The state of the workshop scheduling environment at time t is represented by S_t. The workshop environment includes jobs and machines, so this paper also divides the state space into two categories, s_t^j and s_t^m. After being represented by a disjunctive graph, each operation $O_{jh} \in O$ at a time includes two features $\left[LB_t\left(O_{jh}\right), I_t\left(O_{jh}\right)\right]$, where $LB_t\left(O_{jh}\right)$ is the prediction of the minimum completion time of the processing operation O_{jh}, and $I_t\left(O_{jh}\right)$ is a one-dimensional feature. If O_{jh} is scheduled to a machine, $I_t\left(O_{jh}\right)$ is 1, otherwise it is 0.

Action: Action refers to the decision made by the agent based on the current state. The action at time t also consists of job action $a_j \in A_j$ and machine action $a_m \in A_m$. At time t, the set of all job actions is recorded as A_j, and the set of all machine actions is recorded as A_m. Therefore, A_j is the set of all jobs that meet the processing conditions, and A_m is the set of machines that can be processed by action a_j.

State transition: State transition is the process of the agent transferring from the current state s_t at time t to the state s_{t+1} at the next time $t + 1$. The agent updates the direction of the non-connected arc based on the current job action a_j and machine action a_m, and generates a new disjunctive graph as the new state space, as shown in Figure 1.

Reward: Reward is the evaluation value returned to the agent by the environment after the agent performs an action. Through rewards, an optimal scheduling policy is learned to minimize the total processing time. To this end, this paper predicts the remaining processing operation time $\hat{q} = \max\left\{LB_{t+1}\left(O_{jh}\right)\right\} - \max\left\{LB_t\left(O_{jh}\right)\right\}$ at the current time t and time $t+1$, and defines the negative value of the difference $\hat{q}$ as the reward, that is, $r\left(s_t, a_j, a_m\right) = -\hat{q}$.

4.2. Multi-Proximal Policy Optimization

In order to deal with the multi-action reinforcement learning problem in the knitting intelligent workshop, the MPGN algorithm architecture is adopted, which includes two parts: a job operation action policy and a machine selection action policy. The algorithm architecture is shown in Figure 3. The decoders of the two parts based on multi-layer perceptron (MLP) have the same network structure, but do not share data. The job operation action encodes the disjunctive graph through the L layer graph isomorphism network (GIN). After MLP decoding, the action decision of the non-connected arc in the disjunctive graph is obtained. As shown in Figure 3, at time $t = 1$, the job operation may execute O_{11}, O_{21}, O_{31} processes, and the agent chooses to execute O_{11} operation through the job operation action policy. There is no graph structure in machine selection, so we use a fully connected layer to encode. Similarly, as shown in Figure 3, at time $t = 1$, the execution of O_{11} operation may be processed by M1,M2, and the agent chooses M2 processing through the machine selection decision.

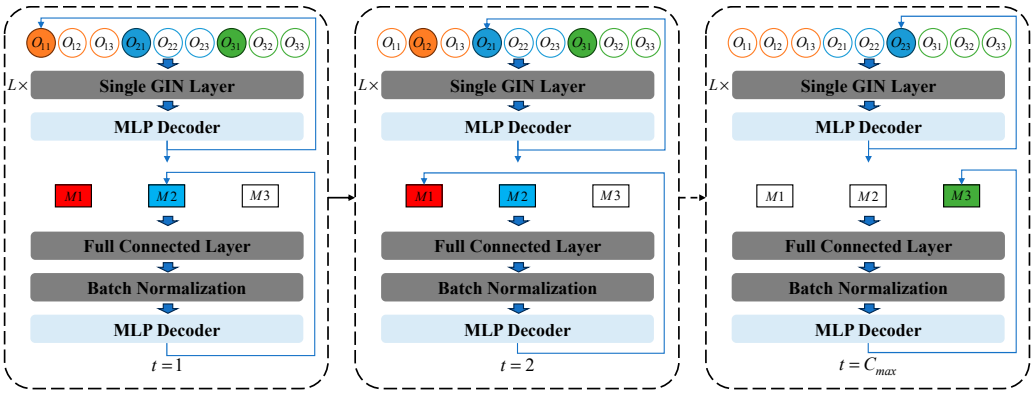

Figure 3. MPGN algorithm architecture diagram.

The proximal policy optimization (PPO) algorithm is a very popular reinforcement learning algorithm based on the actor-critic architecture and is widely used in continuous action space problems. In this paper, the multi-proximal policy optimization (multi-PPO) algorithm is used to learn two sub-policies, namely, the job operation policy $\pi_{\theta_j}\left(a_j|s\right)$ selects the job operation action a_j, and the machine selection policy $\pi_{\theta_m}\left(a_m|s\right)$ selects the machine selection action a_m. Through this algorithm, the action of the agent is represented by a tuple $\left(a_j, a_m\right)$. In multi-PPO, the state value function depends on both the current state s_t and the parameter θ of the policy network π. That is, the better the current state s_t and the better the policy network π, the greater the expected state value. The advantage function $\hat{A}_t$ is estimated by Equation (9).

$$\hat{A}_t = \sum_{t'>t} \gamma^{t'>t} r_{t'} - V_{\varnothing}(S_t) \tag{9}$$

where γ is the reward discount factor and $V_{\varnothing}(S_t)$ is the value function of the current state.

In the knitting intelligent workshop scheduling, the job operation action policy π_{θ_j} and the machine selection action policy π_{θ_m} are both discrete. Therefore, the output of the multi-PPO actor network is the probability distribution on the job operation action space and the machine selection action space. Then, the job operation and machine selection actions are

selected on the generated probability distribution by random sampling or greedy decoding. During the policy network training process, the training experience samples are collected through the actor network to update the two policies π_{θ_j} and π_{θ_m}, as shown in Figure 4.

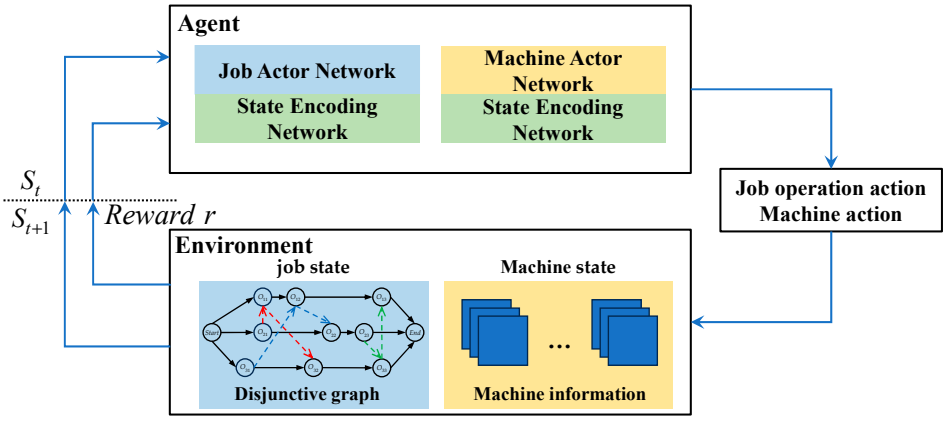

Figure 4. Multi-PPO algorithm for multi-action space scheduling.

The objective of both sub-policies is to find a policy π'_θ that is better than the original policy π_θ. This paper directly clips the objective function used for policy gradient to obtain a more conservative update. We define $\rho_t(\theta')$ to represent the ratio of the two policies $\frac{\pi'_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)}$. The objective function is shown in Equation (10), which allows the policy gradient method to have stable learning performance.

$$L(\pi'_\theta) = E_{\pi_\theta}\left[\min\left\{\rho_t(\theta')\hat{A}_t, \text{clip}\left(\rho_t(\theta'), 1-\varepsilon, 1+\varepsilon\right)\hat{A}_t\right\}\right] \tag{10}$$

where $\text{clip}(x, 1-\varepsilon, 1+\varepsilon)$ means truncating x in $[1-\varepsilon, 1+\varepsilon]$ to ensure that π_θ and π'_θ are similar. Finally, the smaller of the truncated objective function and the untruncated objective function is taken as the final objective function of the learning. The algorithm is described in Algorithm 1.

Algorithm 1 Multi-PPO Algorithm

Parameters: Truncated factorization ε, number of sub-iterations M, B
Input: Initial policy function parameters θ, initial value function parameters $\varnothing$.
Output: Optimal solution s
Begin
  for k = 0, 1, 2, $\cdots$, **do**
    Execute the policy π_θ in the environment and save the trajectory set $D_k = \{\tau_i\}$.
    Calculate the reward $\hat{r}_t$.
    Based on the current value function $V_\varnothing$, calculate the advantage function $\hat{A}_t$
    for m $\in \{1, \cdots, M\}$ **do**
      $\rho_t(\theta') = \frac{\pi'_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)}$
      The Adam stochastic gradient ascent algorithm is used to maximize the objective function of PPO-Clip to update the policy:
      $$\theta_{k+1} = \arg\max_\theta \frac{1}{|D_k|T} \sum_{\tau \in D_k} \sum_{t=0}^{T} \min\left\{\rho_t(\theta\prime)\hat{A}_t, \text{clip}(\rho_t(\theta\prime), 1-\varepsilon, 1+\varepsilon)\hat{A}_t\right\}$$
    end for
    for b $\in \{1, \cdots, B\}$ **do**
      The value function is learned by minimizing the mean square error using the gradient descent algorithm:
      $$\varnothing_{k+1} = \arg\max_\theta \frac{1}{|D_k|T} \sum_{\tau \in D_k} \sum_{t=0}^{T} (V_\varnothing(S_t) - \hat{r}_t)^2$$
    end for
  end for
End

5. Simulation Result and Analysis

In this section, we evaluate the performance of the knitting intelligent workshop production scheduling algorithm based on reinforcement learning through experimental simulation. First, we train it through publicly available small and medium-sized FJSP benchmark Hurink instances. Secondly, a real knitting intelligent production platform is built, and a batch of order information actually produced in the workshop is used as a test instance. The effectiveness of the proposed algorithm is demonstrated by comparing it with commonly used heuristic rules and meta-heuristic algorithms.

5.1. Parameters and Training

In this paper, the reinforcement learning model is built with PyTorch, and the code is implemented in Python 3.10. The hardware system configuration is Intel(R) Core(TM) i7 processor and NVIDIA GeForce RTX 3060 Laptop GPU. For the job action policy and machine action policy, this paper uses the MPGN algorithm architecture with GIN layers. In each GIN layer, the MLP includes two hidden layers with a dimension of 128. In the multi-PPO algorithm, the clipping coefficient is 0.2, the loss coefficient of the Critic is 1, the loss coefficient of the policy is 2, and the entropy loss coefficient is 0.01.

First, we use the examples in Hurink for training and verification. The experimental verification shows that the algorithm has good convergence during the training process. The training process and makespan change curve are shown in Figure 5.

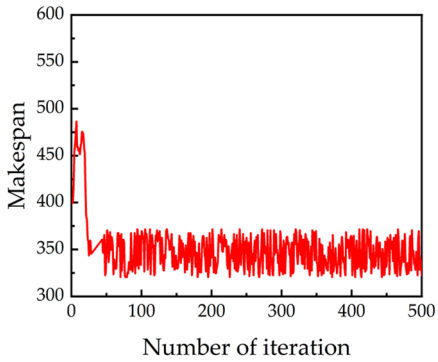

Figure 5. Makespan curve during algorithm training.

5.2. System Validation Parameters

The experimental verification data intercepts a batch of actual production order information from the workshop management system. Each production order is divided into four processing procedures: knitting production, fabric roller handling, fabric inspection, and fabric handling. Knitting production can be processed by any circular knitting machine, and the production parameters of the circular knitting machine equipment that can be processed are known, meaning that the processing and production time of each production order on each circular knitting machine equipment has been determined. The fabric roller handling and fabric handling processes are completed by an automated guided vehicle(AGV). In actual production, the configurations of the two AGVs are the same, and the process duration of the AGV has also been determined. The fabric inspection process can be completed automatically by any fabric inspection machine, and the process duration has been determined. For the convenience of description, the orders, processing procedures, and equipment are numbered. The 20 orders are numbered from Job 1 to Job 20, the four processing procedures are numbered from Process 1 to Process 4, the eight circular knitting machines are numbered from M1 to M8, and the two AGVs are numbered M9 and M10. The five fabric inspection machines are numbered from M11 to M15. The optional mechanical equipment for each production order is shown in Table 2, and the production and processing time of each process of the production order is shown in Table 3.

Table 2. List of optional machines and equipment for processing operations.

Workpiece	Process 1	Process 2	Process 3	Process 4
Job 1	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 2	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 3	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 4	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 5	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 6	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 7	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 8	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 9	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 10	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 11	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 12	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 13	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 14	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 15	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 16	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 17	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 18	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 19	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)
Job 20	(M1,M2,M3,M4,M5,M6,M7,M8)	(M9,M10)	(M11,M12,M13,M14,M15)	(M9,M10)

Table 3. List of processing times of corresponding machines and equipment.

Workpiece	Process 1 (min)	Process 2 (min)	Process 3 (min)	Process 4 (min)
Job 1	(194,153,174,173,179,163,153,189)	(13,13)	(10,11,16,16,13)	(12,12)
Job 2	(171,143,196,183,153,195,195,170)	(12,12)	(22,21,14,19,21)	(13,12)
Job 3	(183,197,141,166,158,138,157,165)	(12,13)	(19,15,13,20,12)	(15,14)
Job 4	(182,201,173,188,145,173,184,188)	(12,12)	(20,20,20,22,25)	(14,14)
Job 5	(211,208,130,174,214,135,210,151)	(11,14)	(19,19,14,16,14)	(14,15)
Job 6	(201,190,166,182,166,149,205,197)	(12,12)	(18,23,18,25,14)	(12,12)
Job 7	(174,197,150,180,133,154,183,200)	(10,12)	(16,14,14,20,17)	(13,14)
Job 8	(183,163,193,154,156,207,216,179)	(14,14)	(21,14,20,16,11)	(12,10)
Job 9	(104,159,114,191,192,179,117,192)	(12,11)	(13,15,18,14,20)	(11,13)
Job 10	(149,168,152,203,141,193,207,206)	(11,12)	(13,20,17,18,19)	(12,12)
Job 11	(199,209,109,150,179,187,144,146)	(13,15)	(20,23,20,15,16)	(11,11)
Job 12	(123,125,141,199,179,132,192,120)	(14,11)	(19,19,18,19,20)	(10,13)
Job 13	(122,118,122,197,187,127,169,180)	(12,13)	(14,14,17,16,11)	(12,11)
Job 14	(201,200,151,150,169,176,153,201)	(13,14)	(18,12,19,18,13)	(11,11)
Job 15	(164,157,173,194,196,199,150,181)	(11,12)	(23,20,18,14,12)	(11,11)
Job 16	(195,198,148,193,164,143,160,145)	(14,15)	(13,20,14,15,19)	(12,12)
Job 17	(146,193,168,137,189,200,139,139)	(12,12)	(13,11,11,16,13)	(12,13)
Job 18	(208,203,208,152,203,197,137,181)	(13,13)	(17,15,15,20,16)	(11,11)
Job 19	(122,152,143,159,114,189,152,159)	(11,12)	(22,21,21,20,19)	(12,15)
Job 20	(191,188,181,185,181,212,212,161)	(12,11)	(20,13,19,19,18)	(12,12)

5.3. Knitting Intelligent Production Experiment Platform

In order to verify the feasibility and effectiveness of the proposed algorithm through experiments, a knitting intelligent production platform was built, as shown in Figure 6. The platform includes eight circular knitting machines, two AGVs, and five fabric inspection machines. After the circular knitting machines complete the production of product orders, the fabric transport AGV supplies the full-shaft fabric rolls in the circular knitting machines to the fabric inspection machines, which then complete the fabric inspection process for the knitted fabrics, thus realizing unmanned continuous production in the knitting intelligent workshop.

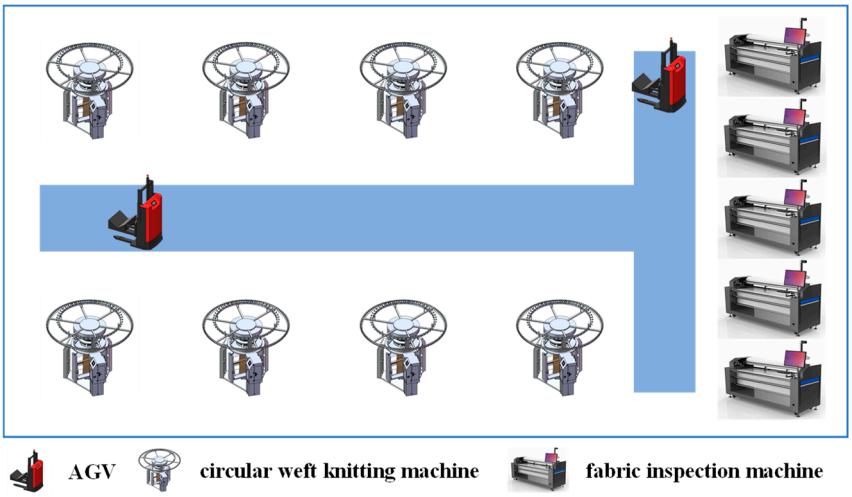

Figure 6. Knitting intelligent production experimental platform.

5.4. Multi-Proximal Policy Optimization

The actual production data in the knitting workshop (see Tables 2 and 3) are used as experimental instances to verify the algorithm in this paper, and the resulting Gantt chart is shown in Figure 7. Taking Job 8 as an example, the first processing step $O_{8,1}$ of job 8 is completed by M4, followed by the second processing step $O_{8,2}$ by M10, and then the third processing step $O_{8,3}$ is processed on M15, and the last processing step $O_{8,4}$ is completed by M10. At the same time, taking the processing of equipment M2 as an example, equipment M2 completes the first process $O_{2,1}$ of Job 2, the first process $O_{12,1}$ of Job 12, and the first process $O_{13,1}$ of Job 13 in sequence.

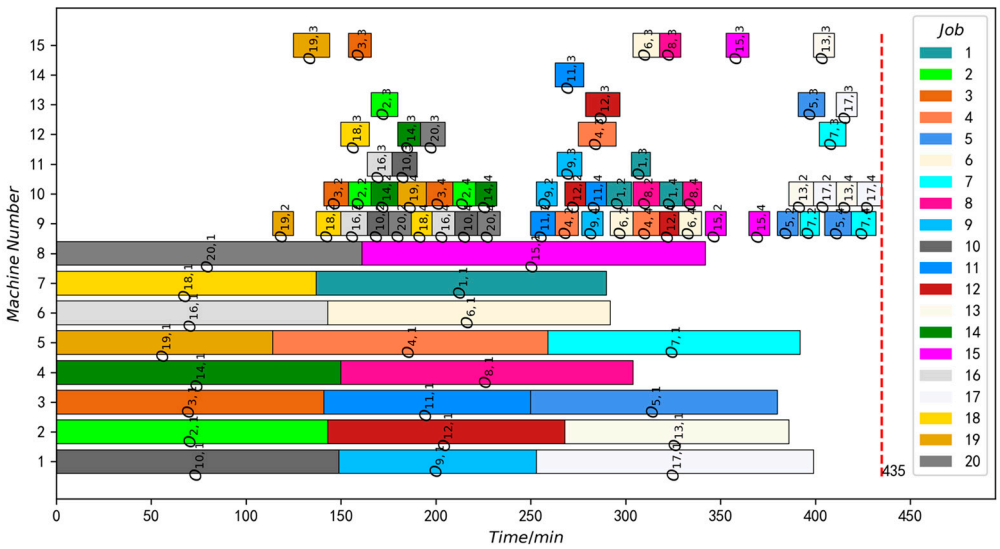

Figure 7. Gantt chart of algorithm verification results.

It can be clearly seen from the Gantt chart that the production order has completed the knitting production on the M1~M8 circular knitting machines. After completing the first process, the AGV transports the fabric roll to the visual fabric inspection machine for inspection, and then transports the fabric on the visual fabric inspection machine to the intermediate warehouse. This algorithm is applied to the knitting intelligent experimental platform. The task scheduling time is compact and the machine can be reasonably scheduled to meet the production scheduling requirements.

In order to further evaluate the performance of the proposed algorithm, we designed two groups of comparative experiments, one using Longest Processing Time (LPT), Shortest

Processing Time (SPT), and First In First Out (FIFO) heuristic rules [22], and the other using the meta-heuristic algorithm of the genetic algorithm. The two groups of experimental algorithms were used to solve the knitting intelligent experimental platform data. The solution results are shown in Table 4.

Table 4. Comparison of experimental solution results.

Experimental Algorithms	Algorithm Solution Results	Relative Error (%)	Running Time (s)
LPT	462	5.84	0.57
SPT	527	17.46	0.59
FIFO	654	33.49	0.41
Genetic algorithms	437	0.46	6.83
The algorithms in this paper	435	0.00	1.08

As can be seen from Table 4, the solution results of the algorithm in this paper are smaller than those of the heuristic rule method, and are comparable to the metaheuristic algorithm. The algorithm in this paper is better than the metaheuristic algorithm as a whole. In addition, the relative error of the algorithm in this paper is better than that of the heuristic rule method. When the GA is similar, the proposed algorithm is more efficient. In summary, the DRL algorithm proposed in this paper can effectively meet the production scheduling needs of the knitting intelligent workshop and effectively improve the production and processing efficiency of the workshop.

6. Conclusions

This paper studies the production scheduling problem in knitting intelligent workshops and proposes a DRL algorithm framework to train scheduling strategies with the goal of minimizing completion time. In order to solve the complexity of knitting intelligent workshop scheduling, a disjunctive graph is introduced to represent local states, and a graph neural network is used to embed local states. At the same time, two neural networks are designed to train workpiece operation action strategies and machine selection action strategies, and tested in real workshop examples. Experiments show that the proposed algorithm is significantly better than the existing heuristic rule algorithm. The main contributions of this paper are as follows:

(1) A production scheduling model for the knitting intelligent workshop is built to reasonably schedule knitting intelligent workshop equipment such as circular knitting machines, AGVs, and visual fabric inspection machines to achieve sustainable production in the knitting workshop.

(2) A multi-strategy optimization algorithm is designed to train the DRL neural network, and the DRL algorithm is trained through public instances. A knitting intelligent workshop experimental platform is built for simulation experiments. The experiments show that the algorithm can effectively, efficiently, and stably solve the production scheduling problem of the knitting intelligent workshop.

Many constraints are added during the execution of the knitting intelligent production platform, such as the processing process cannot be interrupted and equipment failures are not considered. At the same time, there are still more equipment uncertainties and multiple goals in solving problems in the actual work of the knitting intelligent workshop, which are all valuable research directions in the future.

Author Contributions: Conceptualization, L.S. and W.S.; methodology, L.S. and C.X.; software, L.S.; validation, L.S., W.S., and C.X.; formal analysis, Y.Z.; investigation, Y.Z.; resources, W.S.; data curation, C.X.; writing—original draft preparation, L.S.; writing—review and editing, L.S.; visualization, L.S.; supervision, W.S.; project administration, W.S.; funding acquisition, W.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Key Research and Development Program of China, Grant/Award Number: 2017YFB1304000.

Data Availability Statement: Data are contained within the article.

Acknowledgments: The authors sincerely thank the anonymous reviewers for their critical comments and suggestions for improving the manuscript.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Sun, L.; Shi, W.M.; Wang, J.R.; Mao, H.M.; Tu, J.J.; Wang, L.J. Research on Production Scheduling Technology in Knitting Workshop Based on Improved Genetic Algorithm. *Appl. Sci.* **2023**, *13*, 5701. [CrossRef]
2. Chen, X.J.; Cui, W.Z.; Wei, Y.L. Application of information technology in textile industry. *Qing Fang Gong Ye YuJishu* **2021**, *50*, 72–73.
3. Wang, C.Y.; Li, Y.; Li, X.Y. Solving flexible job shop scheduling problem by a multi-swarm collaborative genetic algorithm. *J. Syst. Eng. Electron.* **2021**, *32*, 261–271. [CrossRef]
4. Gen, M.; Lin, L.; Ohwada, H. Advances in Hybrid Evolutionary Algorithms for Fuzzy Flexible Job-shop Scheduling: State-of-the-Art Survey. In Proceedings of the 13th International Conference on Agents and Artificial Intelligence (ICAART), Electr Network, Online, 4–6 February 2021; pp. 562–573.
5. Gao, K.Z.; Cao, Z.G.; Zhang, L.; Chen, Z.H.; Han, Y.Y.; Pan, Q.K. A Review on Swarm Intelligence and Evolutionary Algorithms for Solving Flexible Job Shop Scheduling Problems. *IEEE-CAA J. Autom. Sin.* **2019**, *6*, 904–916. [CrossRef]
6. Wang, J.F.; Du, B.Q.; Ding, H.M. A Genetic Algorithm for the Flexible Job-Shop Scheduling Problem. In Proceedings of the International Conference on Advanced Research on Computer Science and Information Engineering, Zhengzhou, China, 21–22 May 2011; pp. 332–339.
7. Sun, W.; Pan, Y.; Lu, X.H.; Ma, Q.Y. Research on flexible job-shop scheduling problem based on a modified genetic algorithm. *J. Mech. Sci. Technol.* **2010**, *24*, 2119–2125. [CrossRef]
8. Zhang, G.H.; Shao, X.Y.; Li, P.G.; Gao, L. An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem. *Comput. Ind. Eng.* **2009**, *56*, 1309–1318. [CrossRef]
9. Nouiri, M.; Jemai, A.; Ammari, A.C.; Bekrar, A.; Niar, S. An effective particle swarm optimization algorithm for flexible job-shop scheduling problem. In Proceedings of the 5th International Conference on Industrial Engineering and Systems Management (IEEE IESM), Mohammadia Sch Engn, Rabat, Morocco, 28–30 October 2013; pp. 29–34.
10. Devi, K.G.; Mishra, R.S.; Madan, A.K. A Dynamic Adaptive Firefly Algorithm for Flexible Job Shop Scheduling. *Intell. Autom. Soft Comput.* **2022**, *31*, 429–448. [CrossRef]
11. Shapiro, J.F. Mathematical programming models and methods for production planning and scheduling. In *Handbooks in Operations Research and Management Science*; Elsevier: Amsterdam, The Netherlands, 1993; Volume 4, pp. 371–443.
12. Özgüven, C.; Özbakir, L.; Yavuz, Y. Mathematical models for job-shop scheduling problems with routing and process plan flexibility. *Appl. Math. Model.* **2010**, *34*, 1539–1548. [CrossRef]
13. Meng, L.L.; Zhang, B.; Gao, K.Z.; Duan, P. An MILP Model for Energy-Conscious Flexible Job Shop Problem with Transportation and Sequence-Dependent Setup Times. *Sustainability* **2023**, *15*, 776. [CrossRef]
14. Yang, D.S.; Wu, M.L.; Li, D.; Xu, Y.L.; Zhou, X.Y.; Yang, Z.L. Dynamic opposite learning enhanced dragonfly algorithm for solving large-scale flexible job shop scheduling problem. *Knowl.-Based Syst.* **2022**, *238*, 16. [CrossRef]
15. Sun, K.X.; Zheng, D.B.; Song, H.H.; Cheng, Z.W.; Lang, X.D.; Yuan, W.D.; Wang, J.Q. Hybrid genetic algorithm with variable neighborhood search for flexible job shop scheduling problem in a machining system. *Expert Syst. Appl.* **2023**, *215*, 18. [CrossRef]
16. Pan, Z.X.; Lei, D.M.; Wang, L. A Bi-Population Evolutionary Algorithm with Feedback for Energy-Efficient Fuzzy Flexible Job Shop Scheduling. *IEEE Trans. Syst. Man Cybern.-Syst.* **2022**, *52*, 5295–5307. [CrossRef]
17. Badia, A.P.; Piot, B.; Kapturowski, S.; Sprechmann, P.; Vitvitskyi, A.; Guo, D.; Blundell, C. Agent57: Outperforming the Atari Human Benchmark. In Proceedings of the International Conference on Machine Learning (ICML), Electr Network, Online, 13–18 July 2020.
18. Bellemare, M.G.; Naddaf, Y.; Veness, J.; Bowling, M. The Arcade Learning Environment: An Evaluation Platform for General Agents. *J. Artif. Intell. Res.* **2013**, *47*, 253–279. [CrossRef]
19. Shang, X.F. A Study of Deep Learning Neural Network Algorithms and Genetic Algorithms for FJSP. *J. Appl. Math.* **2023**, *2023*, 13. [CrossRef]
20. Wang, L.; Hu, X.; Wang, Y.; Xu, S.; Ma, S.; Yang, K.; Liu, Z.; Wang, W. Dynamic job-shop scheduling in smart manufacturing using deep reinforcement learning. *Comput. Netw.* **2021**, *190*, 107969. [CrossRef]

21. Lei, K.; Guo, P.; Zhao, W.C.; Wang, Y.; Qian, L.M.; Meng, X.Y.; Tang, L.S. A multi-action deep reinforcement learning framework for flexible Job-shop scheduling problem. *Expert Syst. Appl.* **2022**, *205*, 18. [CrossRef]
22. Liang, X.J.; Song, W.; Wei, P.F. Dynamic Job Shop Scheduling via Deep Reinforcement Learning. In Proceedings of the 35th IEEE International Conference on Tools with Artificial Intelligence (ICTAI), Atlanta, GA, USA, 6–8 November 2023; pp. 369–376.