mathematics

MDPI

Article

Fusion Q-Learning Algorithm for Open Shop Scheduling Problem with AGVs

Xiaoyu Wen [1], Haobo Zhang [1], Hao Li [1], Haoqi Wang [1], Wuyi Ming [1,2,*], Yuyan Zhang [1] and Like Zhang [1]

[1] Henan Provincial Key Laboratory of Intelligent Manufacturing of Mechanical Equipment, Zhengzhou University of Light Industry, Zhengzhou 450002, China
[2] Guangdong Intelligent Engineering Technology Research Center for Manufacturing Equipment & Guangdong HUST Industrial Technology Research Institute, Huazhong University of Science and Technology, Dongguan 523770, China
* Correspondence: mingwuyi@zzuli.edu.cn

Abstract: In accordance with the actual production circumstances of enterprises, a scheduling problem model is designed for open-shop environments, considering AGV transport time. A Q-learning-based method is proposed for the resolution of such problems. Based on the characteristics of the problem, a hybrid encoding approach combining process encoding and AGV encoding is applied. Three pairs of actions are constituted to form the action space. Decay factors and a greedy strategy are utilized to perturb the decision-making of the intelligent agent, preventing it from falling into local optima while simultaneously facilitating extensive exploration of the solution space. Finally, the proposed method proved to be effective in solving the open-shop scheduling problem considering AGV transport time through multiple comparative experiments.

Keywords: open-shop scheduling; AGV transport time; Q-learning; hybrid coding

MSC: 90C27

Citation: Wen, X.; Zhang, H.; Li, H.; Wang, H.; Ming, W.; Zhang, Y.; Zhang, L. Fusion Q-Learning Algorithm for Open Shop Scheduling Problem with AGVs. *Mathematics* **2024**, *12*, 452. https://doi.org/10.3390/math12030452

Academic Editors: Kaizhou Gao, Naiqi Wu and Yaping Fu

Received: 2 January 2024
Revised: 23 January 2024
Accepted: 24 January 2024
Published: 31 January 2024

1. Introduction

The open-shop scheduling problem (OSP) constitutes a paradigmatic NP-hard class problem [1]. In contrast to the job-shop scheduling problem (JSP), where job operations adhere to a fixed sequence of processing steps, the OSP employs a non-fixed sequence-processing methodology. The OSP adopts a non-fixed sequence-processing methodology to mitigate the challenges associated with a fixed sequencing of operations during job processing. This approach enhances the flexibility of manufacturing processes within enterprises, consequently promoting advancements in machine-utilization rates and overall production efficiency. It is noteworthy that the OSP has a larger solution space compared to the traditional JSP, thereby containing greater potential for optimization.

Presently, the minimization of maximum completion time stands as the most widely studied performance-evaluation metric in the field of open-shop scheduling. Wan et al. [2] proposed a method for solving the open-shop scheduling problem through link prediction with graph convolutional networks. They represented the state of the open-shop scheduling problem using a disjunctive graph and designed a scheduling model and algorithm based on graph convolutional networks. Finally, they demonstrated that their approach achieved better and more stable experimental results when solving random instances of the problem. Rahimi et al. [3] formulated a no-wait open-shop scheduling problem model considering transportation time, with the objective of minimizing the maximum completion time. They devised a hybrid simulated annealing metaheuristic algorithm to solve small-scale instances of the problem. Experimental results indicate a significant improvement in performance with the enhanced approach. Li et al. [4] incorporated discounting memory into a graph neural-network model to address the open-shop scheduling problem with the objective of

minimizing makespan. They utilized the construction of incremental graph representations to transform the scheduling problem into a sequence problem. In the final experimental results, they demonstrated that their method outperformed traditional approaches and could achieve higher-quality solution sets. Abreu et al. [5] proposed a mixed-integer linear programming model and designed a novel efficient cross-domain search method for solving the makespan objective in open-shop scheduling. The final computations demonstrated the effectiveness of their metaheuristic approach. Yan et al. [6] designed a solving framework based on deep reinforcement learning for addressing workshop scheduling problems. The proposed method's high performance was validated through numerical results. Kurdi [7], with the objective of minimizing makespan, introduced a novel metaheuristic algorithm to enhance the exploration capability of the optimized ant colony. This method was compared with traditional approaches, and the results indicated that, within a finite timeframe, it could provide superior solution outcomes. Wang et al. [8] improved a metaheuristic approach for solving open-shop scheduling problems with the task objective of minimizing makespan. The feasibility of the method was ultimately validated through experiments. Abreu et al. [9] proposed an approach that combines constraint programming with an adaptive large neighborhood search to address the open-shop scheduling problem with an integer linear programming model. The method was validated with the objective of minimizing makespan, demonstrating its reliability. Wang et al. [10] employed a disjunctive graph model to extract production times for the open-shop scheduling in the wind turbine-production process. An enhanced Pythagorean hesitant fuzzy approach was utilized to address multi-objective concerns related to the maximum completion time and machine workload. Experimental results demonstrated its capability to enhance production efficiency by over 15% in both small- and large-scale problem instances. Fu et al. [11] proposed a heuristic, knowledge-based, enhanced artificial bee colony algorithm and designed rules for initializing the population. The effectiveness of the method was ultimately validated through comparison and statistical analysis. Gao et al. [12] combined heuristic methods with Q-learning to solve a mathematical model problem considering battery capacity and uncertain drawing times with the objective of minimizing the maximum completion time. The performance of the proposed method was ultimately demonstrated through experiments on various scales. Abreu et al. [13] designed two novel constructive heuristic algorithms and combined them to solve the open-shop scheduling problem with the objectives of minimizing makespan and sequence-dependent setup times. The effectiveness of the method was ultimately validated through experimental testing. Gong et al. [14] considered the objectives of minimizing makespan and total delay time. They transformed the problem into the state features and actions of an intelligent agent using reinforcement learning, they designed heuristic rules for problem-solving, and they experimentally demonstrated the effectiveness of the proposed method. Xu et al. [15] employed Q-learning to solve a two-stage hybrid workshop scheduling problem with the objectives of minimizing makespan and total energy consumption. Experimental results demonstrated an average performance improvement of over 5%, validating the effectiveness of their approach. Mejia et al. [16] designed an improved variable neighborhood search method to solve the open-shop scheduling problem and validated the effectiveness of the method across multiple instances.

With the continuous evolution of global trade and logistics technologies, countries worldwide are increasingly establishing "smart factories" [17,18]. In recent years, in-depth research by scholars on the modeling and optimization of scheduling problems has led to the gradual proliferation of scheduling methods in practical applications within enterprises [19]. Automated Guided Vehicles (AGV) are increasingly favored by enterprises as flexible and efficient transportation tools in automated material-handling systems, manufacturing workshop systems, and container-handling applications [20]. Wu et al. [21] designed an improved narrow-path search method to optimize AGV paths. This method incorporates strategies such as parent-node reselection and path pruning, significantly reducing the number of turning points. Through comparison with other algorithms, the

efficiency of the method was ultimately demonstrated. Durst et al. [22] designed a novel deep reinforcement learning method optimized with a proximity strategy for solving AGV scheduling problems. The effectiveness of the method was validated through comparisons with cases and traditional heuristic algorithms. Li et al. [23] consider the interdependence between workshop CNC machines and AGVs, proposing an algorithm that combines AGV coordination and harmony search. Experimental validation indicates the effectiveness of this method in reducing the waiting time for CNC machines and enhancing overall production efficiency. Liu et al. [24] applied an improved reinforcement learning method to AGV path planning. They designed a new dynamic reward function and action selection method, ultimately demonstrating through experimental cases that the method can effectively assist AGVs in obtaining better paths. Zhou et al. [25], aiming at the completion time of job processing, machine workload, and carbon emissions, investigated AGV allocation strategies using the NSGA-II method, incorporating heuristic strategies. The effectiveness of the AGV allocation strategy is validated, and the optimal number of AGVs is determined under specific conditions. Liang et al. [26] designed a three-stage integrated scheduling algorithm to solve the AGV route-planning problem in a road-grid model. Through comparisons with genetic algorithms with the task objective of minimizing makespan, they ultimately validated the efficiency of the proposed method. Wu et al. [27] introduced the Laplace distribution into the update process of the ant colony algorithm, thereby accelerating the convergence speed of the algorithm and optimizing the smoothness of paths during AGV scheduling. To address unforeseen events such as machine failures and emergency dispatching during AGV scheduling, Yang et al. [28] designed a hierarchical planning-based method for coordinating and scheduling AGV fleets in workshops. Through simulation experiments, they demonstrated that this method could more efficiently organize AGV transportation within the workshop, showcasing the effectiveness of the approach. Zhang et al. [29] established a hybrid linear model and applied it to an enhanced NSGA-II method to tackle scheduling problems with objectives, including completion time, machine workload, and energy consumption. Experimental results demonstrated the reliability and robustness of the proposed method. Xue et al. [30] employed reinforcement learning methods to solve the AGV scheduling problem with the objectives of minimizing makespan and minimizing job delays. Experimental results indicated that this method could provide better solutions, demonstrating its good performance. Peng et al. [31] designed a multi-agent reinforcement learning approach to solve the flexible job-shop scheduling problem, considering flexibility and variations in transportation time. In case experiments, they ultimately demonstrated characteristics of their method, such as high stability and strong generalization capability. Li et al. [32] proposed a discrete invasive weed-optimization algorithm to study the AGV scheduling problem with time and capacity constraints. They designed a heuristic method to reduce the computational complexity of local searches. Ultimately, the effectiveness of the method was verified through instances.

In the context of traditional open-shop scheduling problems, the transportation time between machines is often overlooked. Typically, it is assumed that transportation times are negligible and are not taken into consideration during the scheduling process. However, this assumption deviates significantly from reality, and the transportation time of AGVs has an undeniable impact on the actual scheduling solution [33]. The open-shop scheduling problem, considering AGV transportation time, represents an extension of the traditional open-shop scheduling problem. This question comprehensively incorporates the time required for the transportation of jobs between machines, making it more aligned with real production processes and posing increased computational challenges.

With the development of artificial intelligence, reinforcement learning has garnered widespread attention from researchers since its inception. Reinforcement learning possesses the capability to engage in continuous trial-and-error interactions with the environment, learn through rewards, and select optimal or near-optimal actions to achieve long-term objectives. This characteristic, as outlined in reference [34], has gradually found widespread applications in various domains such as robot control, gaming, and automated driving.

However, currently, reinforcement learning is primarily applied to job-shop scheduling problems, with a limited application to open-shop scheduling problems that consider AGV transportation times. Yang et al. [35] integrated deep reinforcement learning and graph neural networks to construct an agent model that translates the state of job-shop scheduling problems into scheduling rules using a disjunctive graph representation. They applied this approach to solve job-shop problems with the objective of minimizing makespan, demonstrating the feasibility of the method. Fu et al. [36] proposed a stochastic simulation-based multi-universe optimization method using a Markov model to solve scheduling problems with the objectives of minimizing energy consumption and maximizing disassembly profit. Through experiments, they demonstrated that the performance of their solution surpassed some current methods. Song et al. [37] applied deep reinforcement learning to job-shop scheduling, transforming the scheduling problem into a Markov decision process, and solving it. Through experimental examples, they demonstrated that their method outperformed traditional approaches. Abbod et al. [38] proposed a new reinforcement learning model based on the Q-learning model to solve the flexible job-shop scheduling problem with the objective of minimizing makespan. In the end, they demonstrated its superior performance. Liu et al. [39] proposed a framework based on graph neural networks and deep reinforcement learning, transforming graph states into node embeddings for learning. They validated the method for the workshop-scheduling problem, demonstrating excellent efficiency and scalability. Zhu et al. [40] designed a dual deep Q-network method to solve problems considering intra-job and machine scheduling. They demonstrated the effectiveness of their method through comparisons with benchmark cases. Park et al. [41] combined reinforcement learning and graph neural networks to solve job-shop scheduling problems. They trained the algorithm using a proximity-based strategy optimization method and experimentally demonstrated that the proposed method exhibits high generalization capability and fast-solving efficiency. Ma et al. [42], in solving scheduling problems with the objective of minimizing makespan, employed reinforcement learning to obtain high-quality solution sets through local search. This enhancement indicates improved overall solution efficiency and performance. Finally, through experiments on various scales, the effectiveness of the approach was validated. Zhang et al. [43] combined deep reinforcement learning with graph neural networks to solve job-shop scheduling problems. This approach effectively captures information about different types of nodes and topologies within the graph, and its effectiveness was demonstrated on benchmark cases. Liu et al. [44] proposed a deep reinforcement learning-based approach to solve job-shop scheduling problems. They used the dual deep Q-network algorithm to describe the relationship between scheduling objectives and production. Simulation experiments ultimately demonstrated that their proposed framework significantly improves solution quality. Chang et al. [45] employed a combination of double-layered Q-learning and deep Q-learning to address multi-objective flexible problems, demonstrating superior solution quality and generalization capability in the obtained results. Lin et al. [46] devised two Q-learning-based strategies to enhance the solving capability of local search, substantiating the effectiveness of their approach through a case study.

Therefore, this study aims to extend the application of Q-learning, a reinforcement learning method, to address open-shop scheduling problems considering AGV transportation time. The potential advantages of the proposed method lie in its multifaceted contributions. Firstly, the application of Q-learning, a reinforcement learning method, to address open-shop scheduling problems considering AGV transportation time extends the method's applicability to a broader domain. Secondly, the method introduces a hybrid encoding scheme, enabling the intelligent agent to continuously explore the environment and obtain a set of optimal target solutions. This encoding scheme enhances the algorithm's adaptability to complex problems. Thirdly, the research objectives encompass a comprehensive consideration of energy consumption metrics for AGV transport time and load under different scenarios, as well as job completion time. This approach aligns the method more closely with practical production needs. Fourthly, the evaluation of Q-learning un-

der diverse weightings examines the algorithm's performance under various conditions, enhancing the method's flexibility and applicability. Finally, thorough validation through numerous comparative experiments provides substantial support for the effectiveness of the proposed method in practical applications.

The remaining sections are organized as follows: Section 2 presents an overview of the problem description and the assumed conditions. Section 3 introduces an optimization method and establishes the corresponding models. Section 4 provides an experimental evaluation of the algorithms. In Section 5, conclusions and future research directions are provided.

2. Problem Description and Basic Assumptions

2.1. Problem Description of Scheduling in Open Shop Considering AGV Transportation Time

The open-shop scheduling problem considering AGV transportation time is characterized by the processing of i jobs on M machines, where each job consists of j operations. Within a specified timeframe, each machine can process only one job, and each job can be handled by only one machine. The processing and transportation of jobs between different machines in the open shop scheduling problem, requiring the assistance of V AGV, present specific operational scenarios. At the initiation of the processing cycle, both jobs and AGV are initially positioned in an inspection area with the assumption that they have already undergone incoming inspection and are ready for processing. Upon completion of all job processing, AGVs are presumed to be automatically stationed in proximity to the machine where the last operation was conducted. This problem entails the consideration of various temporal factors, including the time required for transporting jobs between the inspection area and machines, the duration for AGV to transfer jobs between different machines, and the pre-emptive movement of AGV while anticipating the completion of operations for subsequent transportation. Consequently, the scheduling task necessitates the arrangement of the processing sequence of jobs on different machines and the assignment of tasks for the transportation of each job by different AGVs. Finally, the performance evaluation is based on the completion time and the energy consumption of the AGV.

By incorporating a weighted treatment into the energy consumption during AGV transportation processes for both empty and loaded states, as well as the completion time, the final objective function is expressed as Formula (1). In the formula, α represents the weighting factor, C_{max} pertains to the minimization of the maximum completion time, EC denotes AGV energy consumption, P is assumed to be a constant default value of 1, and L defaults to 1.3 [47]. The objective functions are provided by Equation (1), and some constraints are provided in Equations (2)–(4).

$$F = min(\alpha \times C_{max} + (1 - \alpha) \times EC) \tag{1}$$

$$EC = EC_f + EC_k \tag{2}$$

$$EC_f = P * L * (\sum t_f(AGV_1)(m_1, m_2) + \sum t_f(AGV_2)(m_3, m_4)) \tag{3}$$

$$EC_k = P * L * (\sum t_k(AGV_1)(m_1, m_2) + \sum t_k(AGV_2)(m_3, m_4)) \tag{4}$$

In the open-shop scheduling problem considering AGV transportation time, certain symbols are represented as shown in Table 1. Symbols referenced in subsequent chapters will adhere to the consistency maintained with those described in the table.

Table 1. Related mathematical notations and their definitions.

Symbol	Description
o_{ij}	The *j*-th processing operation of job *i*.
M	The number of Machines.
V	The number of AGV.
m_b	The machine at which AGV docks upon the completion of the preceding operation.
m_b	The machine at which AGV docks upon the completion of the preceding.
n	Total Number of Processing Operations.
to_{ij}	The processing time for the *j*-th operation of job *i*.
A_t	Action Space.
mo_{ij}	The machine where the *j*-th operation of job *i* is processed.
$m_b o_{ij}$	The machine where the preceding operation of the *j*-th operation of job *i* is processed.
$T_b m$	The completion time of the machine after finishing the preceding operation.
To_{ij}	The completion time of the *j*-th operation of job *i*.
$T_b AGV_r$	The completion time of the AGV after finishing the preceding transportation task.
C_{max}	Minimizing the Maximum Completion Time.
$C_M{}^j$	The completion time on a specific machine.
$EF_M{}^j$	The effective processing time on a specific machine.
P	The AGV power consumption.
L	The AGV load factor.
α	The weighting factor.
AGV_r	The AGV Identification Number.
$t_k(m_1,m_2)$	The empty transit time of AGV from machine m_1 to machine m_2.
$t_f(m_1,m_2)$	The loaded transit time of AGV from machine m_1 to machine m_2.
EC	The total energy loss generated by AGV during the transportation process.
EC_f	The energy loss incurred by AGV during the loaded transportation process.
EC_k	The energy loss incurred by AGV during the empty transportation process.

2.2. Basic Assumptions

To simplify the problem, the assumptions (1)–(7) are made for the open-shop scheduling process, considering AGV transportation time.

1. Prior to the commencement of processing, all jobs are assumed to have undergone incoming inspection, are parked in the inspection area, and are a unit distance away from the first processing machine.
2. After the initiation of processing, the distance covered by AGV between machines is considered the unit length.
3. Each AGV can transport only one job at a time, and the transportation speed is constant.
4. Upon completing the current transportation task, an AGV can dock near the processing machine if there is no subsequent transportation task.
5. At any time, each job can be processed on only one machine and transported by one AGV.
6. The processing times for each operation of the jobs are provided by the test dataset.
7. Time losses, occurrences of faults, and collisions are temporarily disregarded during the loading and unloading of jobs by AGV.

The constraint conditions are represented by Equations (5)–(9).

8. When n equals 1, Equation (5) represents the constraint condition for the first operation of processing the initial job:

$$C(To_{ij}) \geq t_f(0, mo_{ij}) + to_{ij} \tag{5}$$

9. When n is not equal to 1, Equations (6)–(9) represent the scheduling constraints involving AGVs. These include constraints related to the transportation of jobs by AGVs and constraints concerning the availability of machines for processing jobs before the start of operations.

$$T_{\mathrm{b}}AGV_{\mathrm{r}} + t_k(m_b, m_b o_{ij}) \leq m_b o_{ij} \tag{6}$$

$$T_{\mathrm{b}}AGV_{\mathrm{r}} \geq m_b o_{ij} + t_f(m_b o_{ij}, mo_{ij}) \tag{7}$$

$$T_{\mathrm{b}}AGV_{\mathrm{r}} + t_k(m_b, m_b o_{ij}) > m_b o_{ij} \tag{8}$$

$$T_{\mathrm{b}}m \geq T_{\mathrm{b}}AGV_{\mathrm{r}} + t_k(m_b, m_b o_{ij}) + t_f(m_b o_{ij}, mo_{ij}) \tag{9}$$

3. Algorithm and Model Design

The central concern in the discussion of reinforcement learning is the maximization of rewards by an agent operating in a complex and uncertain environment. In reinforcement learning, the agent and the environment engage in continuous interaction. Upon receiving the current state of the environment, the agent takes corresponding actions and feeds back these actions to the environment. Subsequently, the environment, upon receiving the agent's action, transitions to the next state and conveys this new state to the agent. The specific interactive process is illustrated in Figure 1.

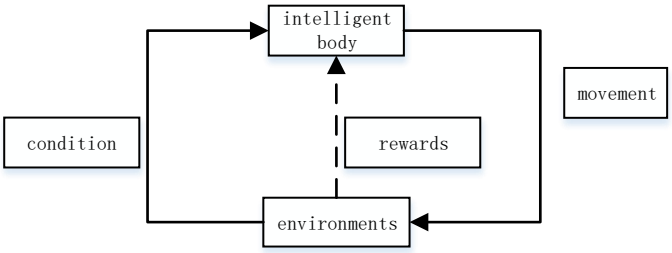

Figure 1. Intensive learning schematic.

Q-learning is a reinforcement learning method based on value functions. It can learn from data generated by scheduling systems according to different strategies and apply this knowledge to train its model during the solving process, thereby improving the training results of the scheduling system. The Q-value update computation is depicted by Equation (10). Table 2 shows the interpretation of the symbols in the Equation (10).

$$Q(s, a)' \leftarrow Q(s, a) + \alpha[r + \gamma \max_a Q(s', a) - Q(s, a)] \tag{10}$$

Table 2. The interpretation of symbols in the formula.

Symbol	Interpretations
α	The learning rate determines the speed of model convergence during the solving process. If it is too small, it may affect training efficiency, while if it is too large, it can lead to the model's inability to converge.
γ	The discount rate represents the impact of future outcomes on current behavior.
r	The environment provides rewards to the agent based on its actions.
Q (s, a)	In state s, the action a corresponding to a certain Q-value of the agent.
Q (s', a')	In the new state s', the action a' corresponding to a certain Q-value of the agent.

In the process of solving the open-shop scheduling problem considering AGV transportation time using Q-learning, the focus of Q-learning is to construct a Q-table during the training process to store the computed Q-values. The Q-table plays a crucial role in Q-learning as it is used to store Q-values for different actions corresponding to each state. In solving the open-shop scheduling problem with the objective of minimizing completion time, the experimental cases involve T7, T10, T15, and T20. The expansion

from the small-scale T7 to the large-scale T20 is illustrated. In the small-scale T7 case, the Q-table is 20,000 × 6, where 20,000 represents the number of states obtained when the agent interacts with the environment during exploration, and 6 indicates the number of actions the agent can take for each exploration (six actions in total). The values in the Q-table for states are derived from the decoding mapping of changes in the two-dimensional encoding composed of process codes and AGV codes when the agent completes a random action based on the greedy policy in the current state. Due to the relatively small scale of T7, the values of states change relatively little, and the agent can effectively explore the entire state space with epochs ranging from 100 to 300, supplementing and completing the Q-table. In the large-scale T20 case, the Q-table is designed at 300,000 × 6. Due to the significant increase in the problem scale, the uncertainty in state values obtained by the agent in space undergoes considerable variation after random actions based on the greedy policy. Therefore, the design of the Q-table is much larger than that of the small-scale case, leading to a rapid increase in the content of the Q-table, occupying a considerable amount of computer memory and increasing computational costs. Our approach to addressing this issue in experiments is to utilize the GPU acceleration feature provided by PyTorch, migrating the Q-table calculation process to the GPU for parallel computation to enhance computational efficiency. By leveraging the parallel processing capability of the GPU, we can accelerate the update and computation processes of the Q-table, thereby reducing memory usage and lowering computational costs. This method effectively addresses the challenge of the Q-table rapidly increasing in large-scale problems, improving the performance and scalability of the reinforcement learning algorithm.

Random noise and greedy strategies are introduced to disrupt the agent's judgments and provide ample opportunities for environmental exploration. Finally, the Q-table is completed to offer a comprehensive and authentic data source for the selection of agent actions. The calculation method for Q-values is provided in Algorithm 1.

Algorithm 1 Q-value updating process

1: **Input:**
2: To_{ij}, α, γ, n, $Q\ (s, a)$
3: **Output:**
4: $Q\ (s, a)'$
5: Procedure
6: **for** $(i = 1; i \leq n^2; i{+}{+})$ **do**
7: at ← At[noisy-greedy]
8: S_{t+1} ← at
9: $A_{t+1}, r \leftarrow max(Q), S_{t+1}$
10: $Q\ (s, a)' \leftarrow Q\ (s, a) + \alpha\ [r + \gamma maxQ\ (s', a) - Q\ (s, a)]$
11: **end for**
12: **Return** $Q\ (s, a)'$

The specific iteration process of the Q-learning strategy is as follows: Firstly, initialize the number of read workpieces and processing machines. Set the gamma to 0.75, learning rate to 0.0001, epoch to 300, and step size to 500. Then, enter the training and solving processes of the model. Subsequently, the intelligent agent randomly selects an action based on the greedy policy and enters a new state after completing the action. In the new state, after taking an action, the agent examines all actions in the new state, selects the action with the maximum Q value, and then chooses that action. Using Equation (10), the Q value of the current action is updated after completing all calculations. After the update, the intelligent agent continues to explore the new state, repeating the above process. It then moves to the maximum number of steps set for termination. The output prints the scheduling strategy and results implied by the maximum Q value in the Q-table.

For the characteristics of the open-shop scheduling problem considering AGV transportation time, a two-dimensional hybrid encoding is employed, combining process encoding and AGV sequence encoding. The first dimension represents process encoding, and the

second dimension represents AGV encoding. The process encoding consists of randomly ordered non-repeating integers from 1 to $i \times j$, where 1, 2. . ., j represent different processing operations of Job 1; $j + 1$, $j + 2$. . ., $j + j$ represent different processing operations of Job 2, and so on. The AGV sequence encoding consists of 3×3 instances of 1 or 2 (assuming V is 2).

Table 3 and Figure 2 illustrate a coding instance for the open-shop scheduling problem considering AGV transportation time. The symbol "8" represents the second operation of processing Job 3, which is transported by AGV2, indicated by the first digit "2" in the AGV encoding, to the corresponding machine for completion. In addition, "5" represents the second operation of processing Job 2, which is transported by AGV2, as indicated by the second digit "2" in the AGV encoding. Also, "3" represents the third operation of processing Job 1, which is transported by AGV1, indicated by the third digit "1" in the AGV encoding, to the respective machine for completion. The remaining digits follow a similar pattern.

Table 3. Processing time of a job on a machine.

	Machine 1	**Machine 2**	**Machine 3**
Job 1	1 (1)	3 (2)	2 (3)
Job 2	2 (4)	1 (5)	3 (6)
Job 3	3 (7)	2 (8)	1 (9)

Process Code	8	5	3	1	2	6	4	9	7
AGV Code	2	2	1	1	2	1	1	2	1

Figure 2. An encoding case study.

States are commonly employed to depict the context or features of a system or environment at a specific moment, enabling intelligent systems to make corresponding decisions. States can be comprehended as descriptions encompassing a set of action-related information. This enables intelligent systems to perceive and understand their surrounding environment, facilitating appropriate decision-making. In solving the open-shop scheduling problem considering AGV transportation time, to enhance the efficiency and diversity of interaction between the intelligent agent and the environment, the state is set as a two-dimensional entity. The first dimension represents the current process encoding of the intelligent agent, and the second dimension denotes the current AGV encoding of the intelligent agent. Through decoding, the two-layer encoding provides the state observation corresponding to the actions in the Q-table. Figure 3 illustrates a schematic representation of the state features in the Q-table.

An action refers to the output received by the environment based on the current state of the intelligent agent. When using Q-learning to solve the open-shop scheduling problem considering AGV transportation time, the encoding method involves a combination of process encoding and AGV sequence encoding, resulting in different crossover points as the action space. The size of the action space directly influences the efficiency and convergence speed of the algorithm. A smaller action space may lead to faster convergence in the learning process but could be prone to local optima. Conversely, a larger action space may require more attempts to find the optimal strategy in a complex environment, leading to increased learning time. Therefore, considering the practical situations in open-shop scheduling, the task complexity of finishing time and energy-consumption objectives under different weights, and the optimal strategy of the Q-learning algorithm's random exploration, three different actions were designed by randomly swapping process or AGV encoding positions at 1, 2, and 3 locations as the action space. For instance, if the intelligent agent acts 1, firstly, a random target is selected from the process or AGV encoding as the action target. Then, a random swap of 1 position is performed on the action target, while the unselected action target remains unchanged. The combination of the two forms of the action variation, as illustrated in Figure 4, depicts two action variation diagrams.

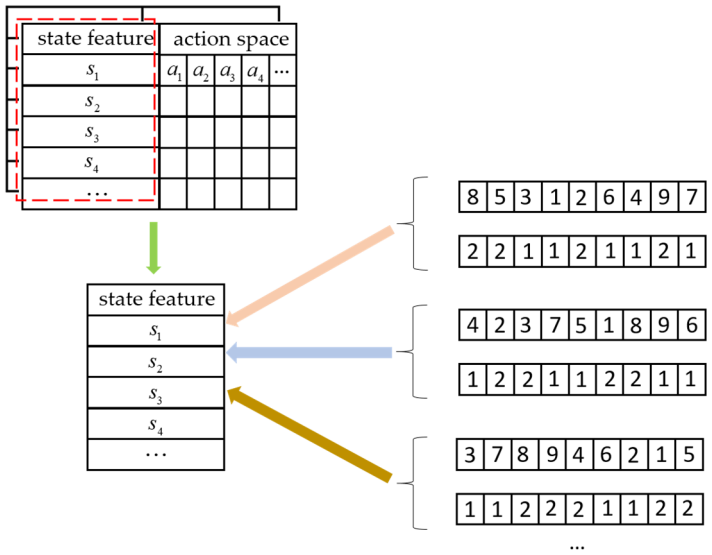

Figure 3. Schematic representation of a state feature in the *Q*-value table.

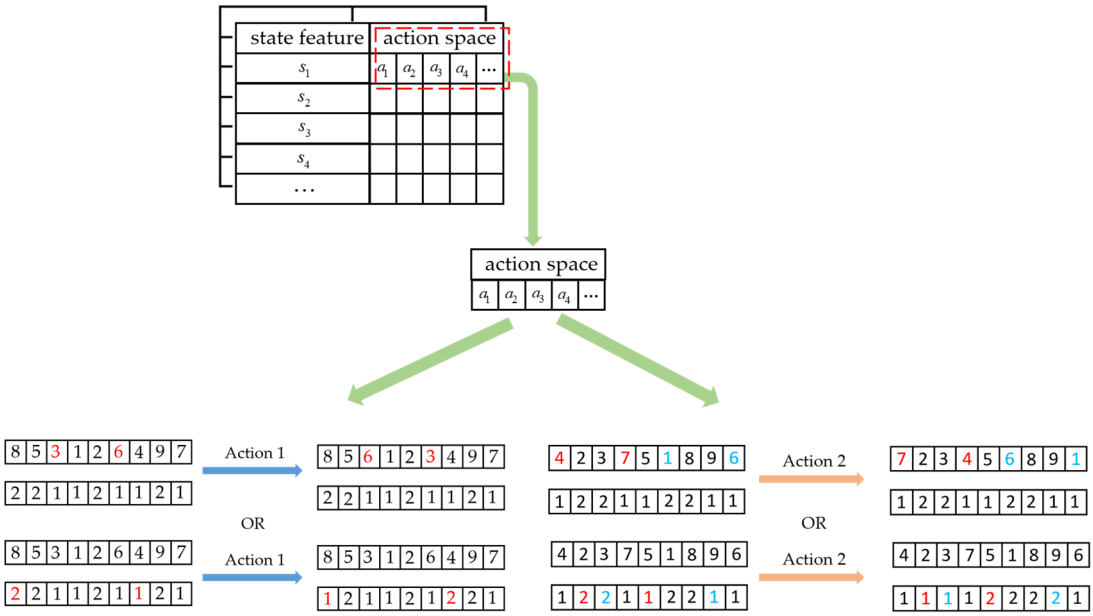

Figure 4. Schematic design of two pairs of action spaces.

Rewards play a crucial role in the interaction between the intelligent agent and the environment, serving as feedback information that the intelligent agent receives from the environment. This feedback accurately indicates the amount of reward obtained when the intelligent agent adopts a specific strategy in a particular environment. By continuously receiving and interpreting this feedback information, the intelligent agent can adjust its behavioral strategy, aiming to achieve more rewards. Therefore, to better balance the relationship between production efficiency and logistics efficiency in the learning and action of the intelligent agent, the design incorporates the machine utilization rate and AGV effective payload rate as weighted factors for the rewards obtained after the interaction between the intelligent agent and the environment.

The effective utilization rate of machine equipment in the production process refers to the ratio of the time a single machine takes to process a job to the total completion time of a single machine. A higher ratio indicates a higher efficiency and capacity utilization of the machine equipment during job processing, directly impacting production efficiency and production costs. The AGV effective payload rate is typically described as the ratio of

the time a single AGV carries a job in operation to the overall operating time of that AGV. Higher logistics efficiency can indirectly reduce the processing time of jobs. The reward calculation method is provided in Algorithm 2.

Algorithm 2 Reward updating process

Input:
$V, C_M{}^j, EF_M{}^j, t_f, m, t_k, M, a_1, a_2$
Output:
r
Procedure
for ($i = 1$; $i \leq$ epoch; i++) **do**
$c_1 \leftarrow (\sum_{m}^{M} \frac{EF_{M^j}}{C_{M^j}} / M)$
$c_2 \leftarrow \frac{\sum_{AGV_1}^{V} t_f}{\sum_{V_1}^{V} t_f + \sum_{V_2}^{V} t_k}$
if $c_1 < a_1$ || $c_2 < a_2$ **then**
$r = c_1 \times 12 + (1 - c_1) \times 8 + c_2 \times 7 + (1 - c_2) \times 3$
else then
$r = c_1 \times 37 + (1 - c_1) \times 3 + c_2 \times 25 + (1 - c_2) \times 5$
end if
end for
Return r

Using the considered open-workshop scheduling code in Table 3 as an example, with the parameter set to 0.4, the result is obtained by solving the objective function formula (1), yielding a value of 24.48. In this context, the load time for AGV_1 is [3, 2, 1, 2, 2], with corresponding idle times of [0, 0, 1, 0, 2]; AGV_2 has load times [2, 2, 1, 1] and idle times [2, 1, 0]; the completion times for the jobs are [12,16,21]; the completion times for the machines are [12,13,21]; and the completion times for AGV_1 and AGV_2 are 18 and 12, respectively. The Gantt chart is illustrated in Figure 5.

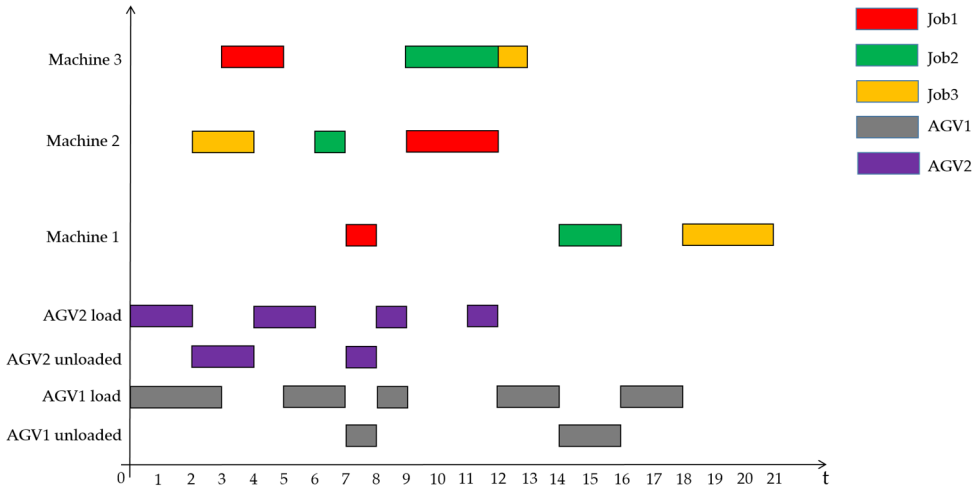

Figure 5. Gantt chart for case 3 × 3.

4. Experiments and Analyses

The experiment was conducted using Python 3.9 on a 64-bit PC with an Intel (R) CoreTM i7-12700 processor running at 2.10 GHz, equipped with 16 GB of RAM. The parameter data for the case originates from the 1993 Taillard benchmark test problems. A comparative analysis was performed against the cases presented in references [47,48].

To maintain consistency with the comparative literature, all experimental processes were conducted through training and testing on the same problem instances.

4.1. Solving the Open-Shop Scheduling Problem with Completion Time as the Objective

To validate the effectiveness of the reinforcement learning algorithm in addressing the open-shop scheduling problem with completion time as the objective, a comparative analysis was initially conducted with Q-learning and publicly disclosed open-shop cases. Experiments were deliberately chosen to match the scale of those reported in the referenced literature. For each experiment, a set of five trials was executed, and each trial was run 10 times. The resulting completion time data were recorded. The algorithm proposed in this paper is denoted as QL, while the neighborhood search mixed with the genetic algorithm from the literature [48] is labeled as NSMGA. Additionally, GA, SA, and PSO control algorithms were employed for comparative analysis. The experimental results are presented in Table 4, and the conclusions are discussed in Figures 6 and 7. D represents the constraints provided in the Taillard benchmark test problems. The RAS definition in Table 4 is provided by Equation (11).

From Table 4, it can be observed that in the Taillard benchmark test cases, Q-learning based on reinforcement learning performs well overall in the process of solving the open-shop scheduling problem. In the provided 20 sets of instances, the Q-learning method shows an improvement ranging from 77.5% to 95.44% compared to the neighborhood search mixed with genetic algorithm designed in the literature [48], with an average improvement of 88.81%.

Table 4. Objective values of resultant schedules.

CASE	D	NSMGA	QL	GA	SA	PSO	RAS
T71	435	494	440	626	580	584	91.53%
T72	443	526	453	649	601	594	87.95%
T73	468	576	483	713	634	648	86.11%
T74	416	501	428	665	612	620	85.88%
T75	451	531	469	599	559	587	77.50%
T101	637	847	664	846	986	1004	87.14%
T102	588	732	601	785	926	920	90.97%
T103	598	792	619	795	922	975	89.18%
T104	577	726	588	715	902	906	92.62%
T105	640	798	663	861	993	993	85.44%
T151	937	1159	953	1116	1621	1631	92.79%
T152	918	1283	960	1046	1700	1716	88.49%
T153	871	1217	902	1114	1584	1618	91.04%
T154	934	1276	948	1004	1645	1700	95.91%
T155	946	1309	983	1138	1697	1682	89.81%
T201	1155	1819	1239	1296	2295	2345	87.35%
T202	1241	1954	1351	1390	2461	2505	84.57%
T203	1257	2154	1309	1406	2402	2499	94.20%
T204	1248	1924	1323	1400	2420	2494	88.91%
T205	1256	1895	1328	1416	2481	2490	88.73%

$$RAS = ((NSMGA - D) - (QL - D))/((NSMGA - D)). \quad (11)$$

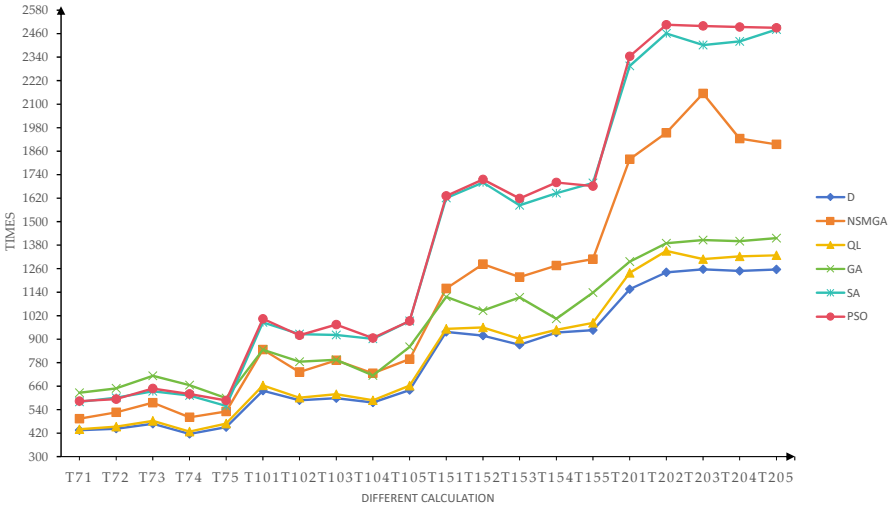

Figure 6. Line chart comparing different algorithms.

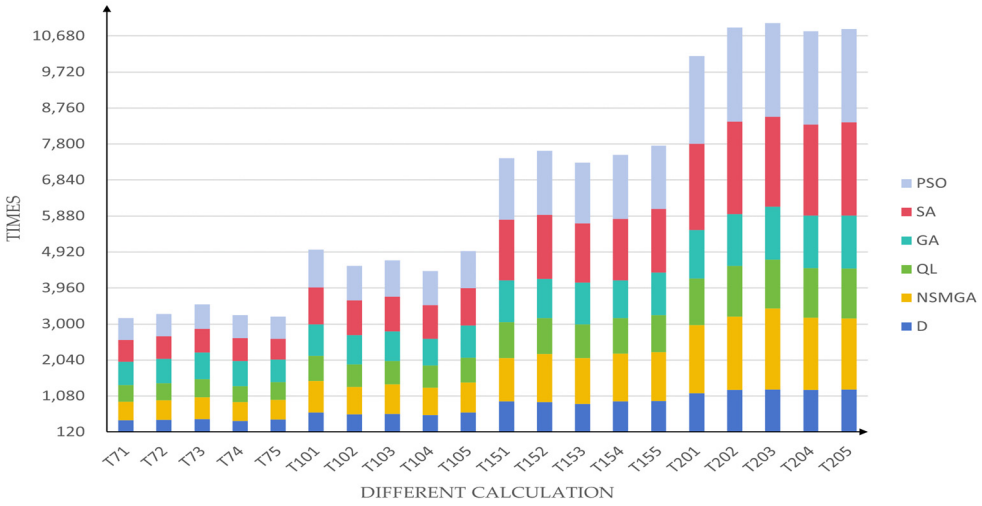

Figure 7. Bar chart comparing different algorithms.

From the changes in Figures 6 and 7, it can be observed that the NSMGA results can maintain relatively consistent approximation curves in the T7 case. However, with the increase in problem size, there is a significant deviation between the solution results and the standard cases. On the other hand, Q-learning (QL) demonstrates relatively good approximation fitting in the T7, T10, and T15 standard cases, showing good stability. In the T20 case, there is an increase in instances where the solution deviates, but overall, the solution remains good, meeting the overall expected performance. The overall performance of the PSO method is influenced by design factors such as inertia weight in the two attributes of speed and position. Moreover, PSO methods are generally suitable for solving small-scale, continuous problems. The SA method is sensitive to parameter settings and can be influenced by initial temperature, cooling temperature, and annealing strategy settings. In the comparison of results in this case, the PSO and SA methods finally yield solutions that are inferior to NSMGA and QL.

The GA algorithm possesses strong global search capabilities and the ability to parallelly evaluate multiple solutions, but it may get stuck in local optima. In the Q-learning method, random perturbations and a greedy strategy are introduced, and over time, the random value of the greedy strategy is gradually reduced. This allows the method to emphasize exploration at the beginning and utilization of learned knowledge later, resulting in Q-learning outperforming GA to varying degrees in the 20 cases. This indicates that when dealing with the open-shop scheduling problem with completion time as the sole objective,

Q-learning provides more trial-and-error opportunities for exploring the solution space and demonstrates better processing results in problem-solving. Therefore, extending the application of Q-learning to open-shop scheduling problems considering AGV transport time is warranted.

4.2. Solving the Open-Shop Scheduling Problem Considering AGV Transport Time

To validate the effectiveness of the Q-learning method in solving the open-shop scheduling problem considering AGV transport time, the objective function is formulated as Equation (1), aiming to minimize the completion time and AGV energy consumption under different weights. The number of AGVs is set to two, and AGV energy consumption is defined as the sum of idle and loaded energy consumption, as provided in Equation (2). The weight factor α is set to 0.4, 0.5, 0.6, and 1. For each group, the cases mentioned in the literature [47] are independently run 10 times, and the results are saved. The results obtained by the literature [47] using the hybrid genetic algorithm based on critical arcs are denoted as HGA, and the proposed Q-learning method for solving the open-shop scheduling problem considering AGV transport time is denoted as QL. The RAS definition in Tables 5–8 is provided by Equation (12). It is worth noting that, in solving the open-shop scheduling problem considering AGV transportation time, changing the alpha value in the objective function Equation (1) is done to compare and explore its impact on the objective function F with results obtained using different alpha values in the literature. Under the same alpha value, solving different instances does not require repetitive training in Q-learning. However, when the alpha value changes, it is necessary to retrain the Q-learning.

When α is 0.4, as shown in Table 5 and Figure 8, the Q-learning method, compared to the literature's HGA method, achieves 12 current optimal solutions. The overall average improvement was 4.43%, with the highest improvement reaching 10.93%. The algorithm demonstrated good overall performance. Due to the stochastic nature of the exploration in Q-learning, three sets of results were slightly lower than the HGA results, with improvements of −0.62%, −0.37%, and −5.52%, respectively. These minor deviations fall within the expected range, satisfying the overall design expectations.

Table 5. Experimental data when α is 0.4.

CASE	HGA	QL	GA	SA	PSO	RAS
T41	109.32	104.48	126.18	119.70	121.86	4.43%
T42	130.94	122.72	143.76	135.96	138.52	6.28%
T43	138.50	135.52	155.10	151.06	148.00	2.15%
T44	130.08	130.88	158.96	138.72	145.46	−0.62%
T45	150.12	148.04	180.48	160.84	165.02	1.39%
T51	201.36	179.36	251.64	223.66	221.82	10.93%
T52	171.14	165.74	240.60	204.72	204.78	3.16%
T53	201.88	198.62	286.90	242.24	243.52	1.61%
T54	202.10	196.68	273.34	232.82	236.50	2.68%
T55	202.72	203.48	276.28	244.12	242.42	−0.37%
T71	371.72	360.56	482.18	463.40	466.02	3.00%
T72	383.12	364.88	585.34	472.88	464.70	4.76%
T73	394.68	390.30	542.32	505.18	497.52	1.11%
T74	367.96	388.26	510.90	486.72	490.08	−5.52%
T75	385.38	349.02	489.54	470.22	472.02	9.43%

$$RAS = (HGA - QL)/HGA. \quad (12)$$

Table 6. Experimental data when α is 0.5.

CASE	HGA	QL	GA	SA	PSO	RAS
T41	126.80	119.25	141.70	138.60	136.20	5.95%
T42	158.40	142.60	170.85	157.35	155.00	9.97%
T43	163.40	159.40	180.75	168.90	168.40	2.45%
T44	156.30	152.35	184.55	172.25	168.40	2.53%
T45	180.90	174.40	211.45	189.25	186.30	3.59%
T51	220.45	202.35	303.85	239.65	256.15	8.21%
T52	198.25	188.60	228.60	228.45	229.70	4.87%
T53	224.05	224.45	333.00	278.65	272.50	−0.18%
T54	231.95	218.95	338.15	274.55	275.20	5.60%
T55	237.55	221.70	276.28	279.60	279.65	6.67%
T71	413.60	402.10	545.95	530.20	532.60	2.78%
T72	413.40	406.20	572.30	537.15	501.95	1.74%
T73	443.40	426.75	590.80	566.05	560.50	3.76%
T74	392.25	430.90	571.95	536.60	551.85	−9.85%
T75	423.15	396.30	533.80	527.40	496.05	6.35%

Table 7. Experimental data when α is 0.6.

CASE	HGA	QL	GA	SA	PSO	RAS
T41	138.88	138.40	165.08	155.16	155.60	0.35%
T42	181.28	162.48	192.08	175.64	179.44	10.37%
T43	187.36	182.72	197.88	190.32	197.40	2.48%
T44	176.04	174.28	198.92	188.04	192.40	1.00%
T45	207.92	196.76	231.64	226.08	218.92	5.37%
T51	242.64	232.40	345.64	278.24	287.56	4.22%
T52	214.04	211.56	314.08	263.20	257.04	1.16%
T53	262.36	252.68	346.88	317.52	308.44	3.69%
T54	236.80	243.88	359.12	301.72	296.20	−2.99%
T55	259.48	258.28	339.16	289.64	316.92	0.46%
T71	438.20	430.68	587.76	569.60	554.64	1.72%
T72	440.28	437.12	613.20	571.20	575.80	0.72%
T73	486.24	480.00	641.40	628.72	606.64	1.28%
T74	423.84	458.40	620.76	589.16	603.52	−8.15%
T75	482.08	421.52	581.40	569.88	568.28	12.56%

Table 8. Experimental data when α is 1.

CASE	HGA	QL	GA	SA	PSO	RAS
T41	202	198	229	230	220	1.98%
T42	272	242	278	264	262	11.03%
T43	279	276	313	289	300	1.08%
T44	257	256	316	282	271	0.39%
T45	314	301	359	321	328	4.14%
T51	345	323	401	397	388	6.38%
T52	301	282	467	344	334	6.31%
T53	375	356	445	437	461	5.07%
T54	352	336	451	421	437	4.55%
T55	385	364	457	459	466	5.45%
T71	553	539	664	770	761	2.53%
T72	580	562	716	784	774	3.10%
T73	628	604	806	826	851	3.82%
T74	561	596	851	823	786	−6.24%
T75	568	524	778	748	746	7.75%

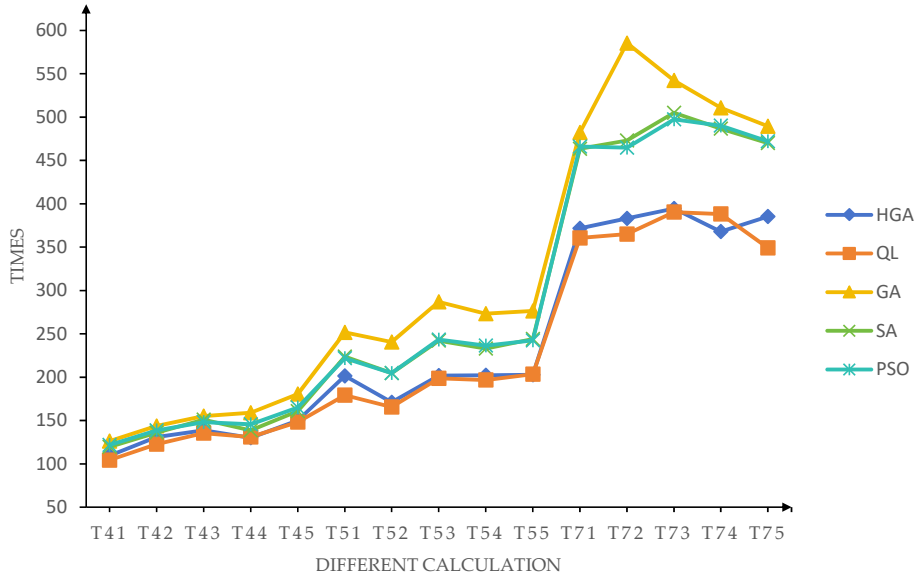

Figure 8. Line graph comparing algorithms when α is 0.4.

When α is 0.5, as depicted in Table 6 and Figure 9, the Q-learning method, compared to the literature's HGA method, achieves 13 current optimal solutions. The overall average improvement was 5.44%, with the highest improvement reaching 9.97%. The algorithm demonstrated good overall performance. In the first half of the instances, the Q-learning method consistently outperformed the HGA, GA, SA, and PSO methods. In the latter part of the instances, there were occasional abrupt increases in the numerical values, possibly due to the increased difficulty in exploring the action space with the expansion of the workshop scale or getting stuck in suboptimal solutions, impacting the exploration for global optimal solutions.

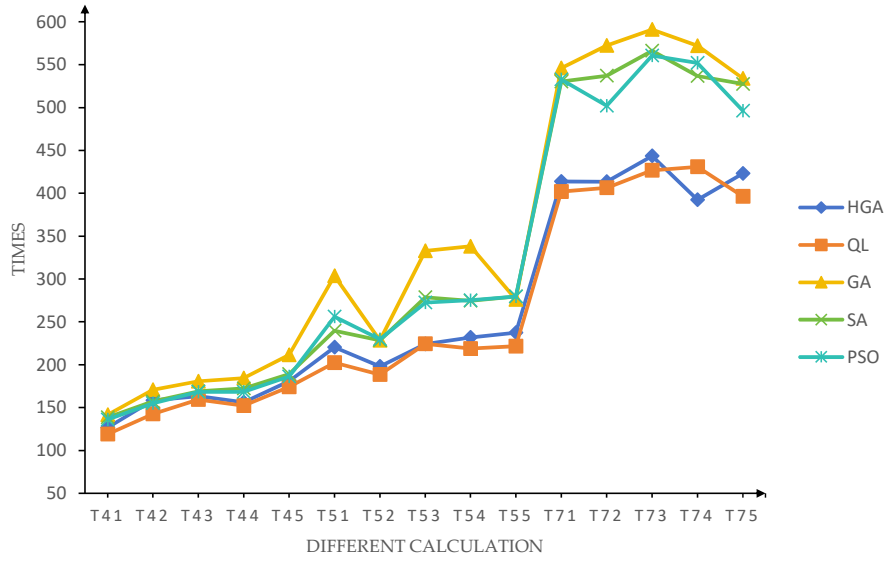

Figure 9. Line graph comparing algorithms when α is 0.5.

When α is 0.6, based on the objective function and combined with Table 7 and Figure 10, it is evident that, in solving the open-shop scheduling problem considering AGV transport time, the Q-learning method achieved 13 current optimal solutions. The completion time accounted for 60% of the experimental results, while the energy consumption accounted for 40%. The overall average performance improvement was 3.42%. When the problem scale was relatively small, the results obtained by HGA and QL were similar. As the

problem scale gradually increased, Q-learning chose to sacrifice some completion time to explore a larger solution space more extensively, resulting in better processing outcomes. Therefore, the Q-learning method, based on reinforcement learning, is suitable for both small- and large-scale problems and can effectively reduce the objective function value within a particular timeframe.

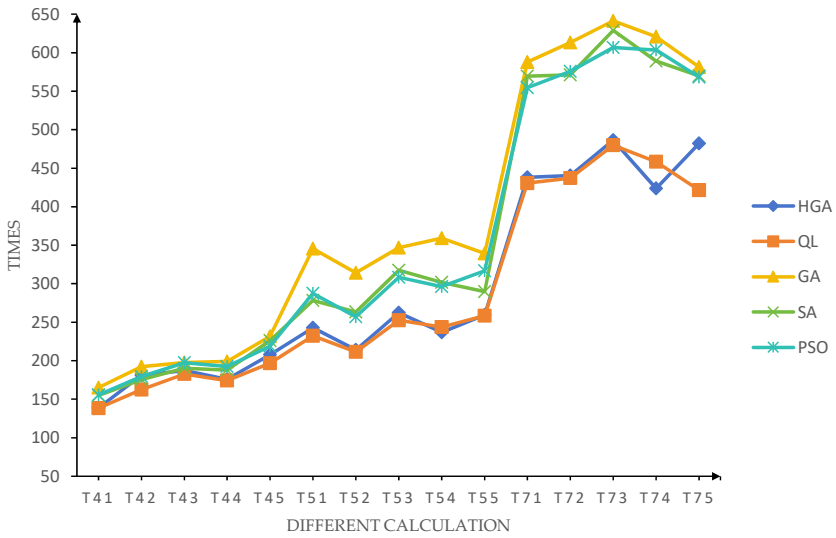

Figure 10. Line graph comparing algorithms when α is 0.6.

When α is 1, using the same decoding method to solve the objective function of the open-shop scheduling problem considering AGV transport time, with completion time accounting for 100%, the Q-learning method obtained 14 current optimal solutions. This result surpasses the solving records of the comparative algorithms, showcasing the stable superiority of the Q-learning method (Figure 11).

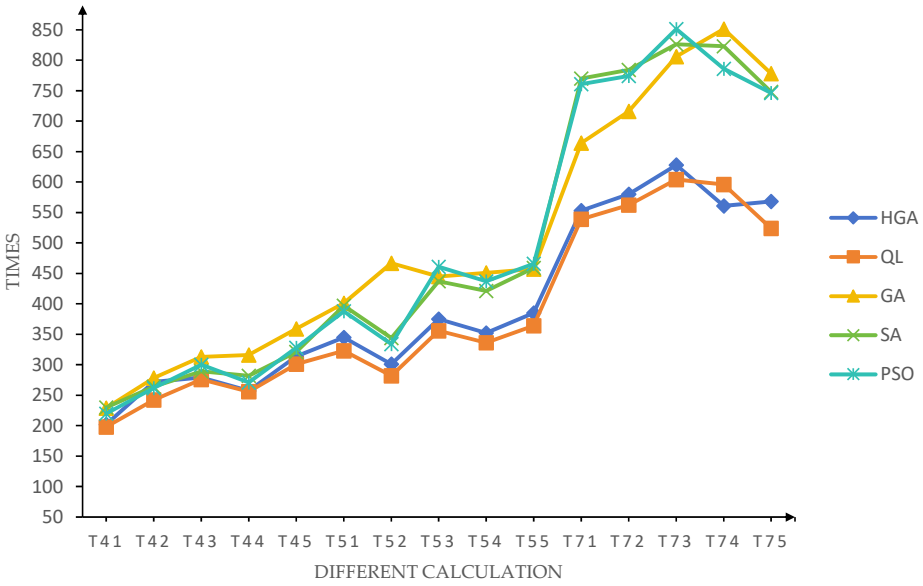

Figure 11. Line graph comparing algorithms when α is 1.

The benchmark method's paper did not provide specific runtime information, making it impossible to compare the runtime. Therefore, for each scale, the five cases were repeated five times (300 epochs), and the average calculation was performed. The summarized results are presented in Tables 9 and 10.

Table 9. The runtime results for job completion time calculation.

CASE	Time/s
T7	219.4
T10	463.5
T15	1120.3
T20	2299.7

Table 10. The runtime results for the objective functions are weighted by job completion time and energy consumption.

CASE	Time/s			
	$\alpha = 0.4$	$\alpha = 0.5$	$\alpha = 0.6$	$\alpha = 1$
T4	217.8	219.5	222.6	214.6
T5	386.2	378.1	385.9	374.3
T7	957.8	968.8	972.7	931.4

From Tables 9 and 10, it can be observed that with the gradual increase in problem size, the computation time for solving the problems significantly increases. In the actual solving process, we leveraged the parallel processing capability of the GPU to accelerate the update and computation processes of the Q-table, thereby reducing the utilization of computer memory and lowering the computational time cost.

5. Conclusions and Future Work

In dealing with the open-shop scheduling problem model considering AGV transport time, the minimization of both the maximum completion time and energy consumption is employed as influential factors within the objective function. To afford the intelligent agent greater opportunities for exploration within the solution space and the accumulation of substantial reward values, a hybrid encoding scheme is devised, comprising process encoding and AGV encoding. Experimental comparisons are conducted utilizing Taillard benchmark test cases and the literature [48], incorporating the design of GA, SA, and PSO as control groups. The results showed an average improvement of 88.81% compared to the NS-MGA algorithm. Furthermore, the Q-learning method was applied to address the objective problem of minimizing completion time and energy consumption with weighted goals, and its performance was evaluated against the literature [47]. The data results demonstrated that the Q-learning method outperforms HGA, GA, SA, and PSO, exhibiting superior performance and scalability in handling open-shop scheduling problems considering AGV transport time.

Future research can consider improvements in the following aspects:

(1) Scheduling strategies can be employed to constrain the processing of the waiting job layer, while different strategies can be applied to handle the processed job. Distinct methods and rewards can be employed for improvement in each scheduling layer, enhancing the overall efficiency of system optimization.
(2) Additional factors influencing objectives could be considered, such as the impact of external disturbances, the effects of urgent job insertions, and the influence of preventive maintenance.
(3) More effective methods can be explored to address open shop scheduling problems, considering AGV transport time.

Author Contributions: Conceptualization, X.W. and H.Z.; methodology, H.L. and Y.Z.; investigation, W.M. and H.W.; supervision, W.M.; software, L.Z. and H.Z.; writing—original draft preparation, X.W. and H.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research work is supported by the National Natural Science Foundation of China (Grant Nos. 51905494 and 52105536), the Henan Province University Science and Technology Innova-

tion Talent Support Plan (Grant No. 24HASTIT048), and Key Scientific and Technological Research Projects in Henan Province (Grant No. 232102221009).

Data Availability Statement: Data are contained within the article.

Acknowledgments: The authors would like to thank the anonymous reviewers and the editor for their positive comments.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study, in the collection, analysis, or interpretation of data, in the writing of the manuscript, or in the decision to publish the results.

References

1. Spindler, S.; Steinhardt, C.; Klein, R. Exact Solution Approaches for Order Acceptance and Scheduling Decisions in M-Machine Open Shops. *Comput. Ind. Eng.* **2023**, *176*, 108952. [CrossRef]
2. Wan, L.; Zhao, H.; Cui, X.; Li, C.; Deng, X. Solving Large-Scale Open Shop Scheduling Problem via Link Prediction Based on Graph Convolution Network. In Proceedings of the Advanced Intelligent Computing Technology and Applications, Zhengzhou, China, 10–13 August 2023; Huang, D.-S., Premaratne, P., Jin, B., Qu, B., Jo, K.-H., Hussain, A., Eds.; Springer Nature: Singapore, 2023; pp. 109–123.
3. Rahimi, A.; Hejazi, S.M.; Zandieh, M.; Mirmozaffari, M. A Novel Hybrid Simulated Annealing for No-Wait Open-Shop Surgical Case Scheduling Problems. *Appl. Syst. Innov.* **2023**, *6*, 15. [CrossRef]
4. Li, J.; Dong, X.; Zhang, K.; Han, S. Solving Open Shop Scheduling Problem via Graph Attention Neural Network. In Proceedings of the 2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI), Baltimore, MD, USA, 9–11 November 2020; pp. 277–284.
5. De Abreu, L.R.; Araújo, K.A.G.; de Athayde Prata, B.; Nagano, M.S.; Moccellin, J.V. A New Variable Neighbourhood Search with a Constraint Programming Search Strategy for the Open Shop Scheduling Problem with Operation Repetitions. *Eng. Optim.* **2022**, *54*, 1563–1582. [CrossRef]
6. Yan, Q.; Wu, W.; Wang, H. Deep Reinforcement Learning for Distributed Flow Shop Scheduling with Flexible Maintenance. *Machines* **2022**, *10*, 210. [CrossRef]
7. Kurdi, M. Ant Colony Optimization with a New Exploratory Heuristic Information Approach for Open Shop Scheduling Problem. *Knowl.-Based Syst.* **2022**, *242*, 108323. [CrossRef]
8. Wang, H.; Chen, W. Task Scheduling for Heterogeneous Agents Pickup, and Delivery Using Recurrent Open Shop Scheduling Models. *Robot. Auton. Syst.* **2024**, *172*, 104604. [CrossRef]
9. Abreu, L.R.; Nagano, M.S. A New Hybridization of Adaptive Large Neighborhood Search with Constraint Programming for Open Shop Scheduling with Sequence-Dependent Setup Times. *Comput. Ind. Eng.* **2022**, *168*, 108128. [CrossRef]
10. Wang, W.; Xiao, J.; Feng, D.; Wei, S.; Wang, Z. Multi-Objective Production and Scheduling Optimization of Offshore Wind Turbine Steel Pipe Piles Based on Improved Hesitant Fuzzy Method. *J. Mar. Sci. Eng.* **2023**, *11*, 1505. [CrossRef]
11. Fu, Y.; Ma, X.; Gao, K.; Li, Z.; Dong, H. Multi-Objective Home Health Care Routing and Scheduling with Sharing Service via a Problem-Specific Knowledge-Based Artificial Bee Colony Algorithm. *IEEE Trans. Intell. Transp. Syst.* **2023**, 1–14. [CrossRef]
12. Gao, M.; Gao, K.; Ma, Z.; Tang, W. Ensemble Meta-Heuristics and Q-Learning for Solving Unmanned Surface Vessels Scheduling Problems. *Swarm Evol. Comput.* **2023**, *82*, 101358. [CrossRef]
13. Abreu, L.R.; Cunha, J.O.; Prata, B.A.; Framinan, J.M. A Genetic Algorithm for Scheduling Open Shops with Sequence-Dependent Setup Times. *Comput. Oper. Res.* **2020**, *113*, 104793. [CrossRef]
14. Gong, H.; Xu, W.; Sun, W.; Xu, K. Multi-Objective Flexible Flow Shop Production Scheduling Problem Based on the Double Deep Q-Network Algorithm. *Processes* **2023**, *11*, 3321. [CrossRef]
15. Xu, K.; Ye, C.; Gong, H.; Sun, W. Reinforcement Learning-Based Multi-Objective of Two-Stage Blocking Hybrid Flow Shop Scheduling Problem. *Processes* **2024**, *12*, 51. [CrossRef]
16. Mejía, G.; Yuraszeck, F. A Self-Tuning Variable Neighborhood Search Algorithm and an Effective Decoding Scheme for Open Shop Scheduling Problems with Travel/Setup Times. *Eur. J. Oper. Res.* **2020**, *285*, 484–496. [CrossRef]
17. Yang, Y.; Zhong, M.; Dessouky, Y.; Postolache, O. An Integrated Scheduling Method for AGV Routing in Automated Container Terminals. *Comput. Ind. Eng.* **2018**, *126*, 482–493. [CrossRef]
18. Shiue, Y.-R.; Lee, K.-C.; Su, C.-T. Real-Time Scheduling for a Smart Factory Using a Reinforcement Learning Approach. *Comput. Ind. Eng.* **2018**, *125*, 604–614. [CrossRef]
19. Fu, Y.; Hou, Y.; Wang, Z.; Wu, X.; Gao, K.; Wang, L. Distributed Scheduling Problems in Intelligent Manufacturing Systems. *Tsinghua Sci. Technol.* **2021**, *26*, 625–645. [CrossRef]
20. Tian, M.; Sang, H.; Zou, W.; Wang, Y.; Miao, M.; Meng, L. Joint Scheduling of AGVs and Parallel Machines in an Automated Electrode Foil Production Factory. *Expert Syst. Appl.* **2024**, *238*, 122197. [CrossRef]
21. Wu, B.; Zhang, W.; Chi, X.; Jiang, D.; Yi, Y.; Lu, Y. A Novel AGV Path Planning Approach for Narrow Channels Based on the Bi-RRT Algorithm with a Failure Rate Threshold. *Sensors* **2023**, *23*, 7547. [CrossRef]

22. Durst, P.; Jia, X.; Li, L. Multi-Objective Optimization of AGV Real-Time Scheduling Based on Deep Reinforcement Learning. In Proceedings of the 2023 42nd Chinese Control Conference (CCC), Tianjin, China, 24–26 July 2023; pp. 5535–5540.
23. Li, G.; Zeng, B.; Liao, W.; Li, X.; Gao, L. A New AGV Scheduling Algorithm Based on Harmony Search for Material Transfer in a Real-World Manufacturing System. *Adv. Mech. Eng.* **2018**, *10*, 1687814018765560. [CrossRef]
24. Liu, Y.; Yan, S.; Zhao, Y.; Song, C.; Li, F. Improved Dyna-Q: A Reinforcement Learning Method Focused via Heuristic Graph for AGV Path Planning in Dynamic Environments. *Drones* **2022**, *6*, 365. [CrossRef]
25. Zhou, X.; Wang, F.; Shen, N.; Zheng, W. A Green Flexible Job-Shop Scheduling Model for Multiple AGVs Considering Carbon Footprint. *Systems* **2023**, *11*, 427. [CrossRef]
26. Liang, C.; Zhang, Y.; Dong, L. A Three Stage Optimal Scheduling Algorithm for AGV Route Planning Considering Collision Avoidance under Speed Control Strategy. *Mathematics* **2023**, *11*, 138. [CrossRef]
27. Wu, C.; Xiao, Y.; Zhu, X. Research on Optimization Algorithm of AGV Scheduling for Intelligent Manufacturing Company: Taking the Machining Shop as an Example. *Processes* **2023**, *11*, 2606. [CrossRef]
28. Yang, Q.; Lian, Y.; Xie, W. Hierarchical Planning for Multiple AGVs in Warehouse Based on Global Vision. *Simul. Model. Pract. Theory* **2020**, *104*, 102124. [CrossRef]
29. Zhang, H.; Qin, C.; Zhang, W.; Xu, Z.; Xu, G.; Gao, Z. Energy-Saving Scheduling for Flexible Job Shop Problem with AGV Transportation Considering Emergencies. *Systems* **2023**, *11*, 103. [CrossRef]
30. Xue, T.; Zeng, P.; Yu, H. A Reinforcement Learning Method for Multi-AGV Scheduling in Manufacturing. In Proceedings of the 2018 IEEE International Conference on Industrial Technology (ICIT), Lyon, France, 20–22 February 2018; pp. 1557–1561.
31. Peng, S.; Xiong, G.; Yang, J.; Shen, Z.; Tamir, T.S.; Tao, Z.; Han, Y.; Wang, F.-Y. Multi-Agent Reinforcement Learning for Extended Flexible Job Shop Scheduling. *Machines* **2024**, *12*, 8. [CrossRef]
32. Li, Z.-K.; Sang, H.-Y.; Li, J.-Q.; Han, Y.-Y.; Gao, K.-Z.; Zheng, Z.-X.; Liu, L. Invasive Weed Optimization for Multi-AGVs Dispatching Problem in a Matrix Manufacturing Workshop. *Swarm Evol. Comput.* **2023**, *77*, 101227. [CrossRef]
33. Yao, F.; Alkan, B.; Ahmad, B.; Harrison, R. Improving Just-in-Time Delivery Performance of IoT-Enabled Flexible Manufacturing Systems with AGV Based Material Transportation. *Sensors* **2020**, *20*, 6333. [CrossRef]
34. Zhang, Z.; Zheng, L.; Li, N.; Wang, W.; Zhong, S.; Hu, K. Minimizing Mean Weighted Tardiness in Unrelated Parallel Machine Scheduling with Reinforcement Learning. *Comput. Oper. Res.* **2012**, *39*, 1315–1324. [CrossRef]
35. Yang, Z.; Bi, L.; Jiao, X. Combining Reinforcement Learning Algorithms with Graph Neural Networks to Solve Dynamic Job Shop Scheduling Problems. *Processes* **2023**, *11*, 1571. [CrossRef]
36. Fu, Y.; Zhou, M.; Guo, X.; Qi, L.; Sedraoui, K. Multiverse Optimization Algorithm for Stochastic Biobjective Disassembly Sequence Planning Subject to Operation Failures. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *52*, 1041–1051. [CrossRef]
37. Song, L.; Li, Y.; Xu, J. Dynamic Job-Shop Scheduling Based on Transformer and Deep Reinforcement Learning. *Processes* **2023**, *11*, 3434. [CrossRef]
38. Momenikorbekandi, A.; Abbod, M. Intelligent Scheduling Based on Reinforcement Learning Approaches: Applying Advanced Q-Learning and State–Action–Reward–State–Action Reinforcement Learning Models for the Optimisation of Job Shop Scheduling Problems. *Electronics* **2023**, *12*, 4752. [CrossRef]
39. Liu, C.-L.; Huang, T.-H. Dynamic Job-Shop Scheduling Problems Using Graph Neural Network and Deep Reinforcement Learning. *IEEE Trans. Syst. Man Cybern. Syst.* **2023**, *53*, 6836–6848. [CrossRef]
40. Zhu, X.; Xu, J.; Ge, J.; Wang, Y.; Xie, Z. Multi-Task Multi-Agent Reinforcement Learning for Real-Time Scheduling of a Dual-Resource Flexible Job Shop with Robots. *Processes* **2023**, *11*, 267. [CrossRef]
41. Park, J.; Chun, J.; Kim, S.H.; Kim, Y.; Park, J. Learning to Schedule Job-Shop Problems: Representation and Policy Learning Using Graph Neural Network and Reinforcement Learning. *Int. J. Prod. Res.* **2021**, *59*, 3360–3377. [CrossRef]
42. Ma, Z.; Gao, K.; Yu, H.; Wu, N. Solving Heterogeneous USV Scheduling Problems by Problem-Specific Knowledge Based Meta-Heuristics with Q-Learning. *Mathematics* **2024**, *12*, 339. [CrossRef]
43. Zhang, C.; Song, W.; Cao, Z.; Zhang, J.; Tan, P.S.; Xu, C. Learning to Search for Job Shop Scheduling via Deep Reinforcement Learning. *Comput.* **2022**, *137*, 1621–1632.
44. Liu, R.; Piplani, R.; Toro, C. A Deep Multi-Agent Reinforcement Learning Approach to Solve Dynamic Job Shop Scheduling Problem. *Comput. Oper. Res.* **2023**, *159*, 106294. [CrossRef]
45. Chang, J.; Yu, D.; Zhou, Z.; He, W.; Zhang, L. Hierarchical Reinforcement Learning for Multi-Objective Real-Time Flexible Scheduling in a Smart Shop Floor. *Machines* **2022**, *10*, 1195. [CrossRef]
46. Lin, Z.; Gao, K.; Wu, N.; Suganthan, P.N. Scheduling Eight-Phase Urban Traffic Light Problems via Ensemble Meta-Heuristics and Q-Learning Based Local Search. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 14415–14426. [CrossRef]
47. Huang, Y.; Zhang, L.; Tang, Q.; Xu, Y. Analytic Fetch Map Modeling and Solving for Open Shop Floor Scheduling with AGVs. *Mech. Des. Manuf.* **2023**, *10*, 15. (In Chinese)
48. Huang, Y. Research on open shop scheduling problems with AGVs based on a hybrid evolutionary algorithm. *Wuhan Univ. Sci. Technol.* **2022**, *15*, 32. (In Chinese)