International Journal of Mechanical and Electrical Engineering
ISSN: 3005-9615 (Print), ISSN: 3005-7132 (Online) | Volume 2, Number 3, Year 2024
DOI: https://doi.org/10.62051/ijmee.v2n3.07
Journal homepage: https://wepub.org/index.php/IJMEE/index

# Improved Q-learning Algorithm to Solve the Permutation Flow Shop Scheduling Problem

Zifeng Xuan, Yunfei Liu, Xinxin Peng

School of Mechanical Engineering and Automation, Wuhan Textile University, China

## ABSTRACT

A modified Q-learning algorithm is proposed for the permutation flow shop scheduling problem. This algorithm initializes the environment with the job sequence and considers each processable job as an executable action. A reward function is defined as the reciprocal of the completion time. Moreover, the completion time is calculated using the principle of diagonalization of a two-dimensional matrix, significantly enhancing computational efficiency. The Boltzmann action exploration strategy is designed, where the probability of selecting an action decreases as the temperature coefficient T decreases, and the probability of randomly selecting an action decreases, favoring the selection of actions corresponding to larger Q values. Finally, the performance of the proposed algorithm is validated using instances of permutation flow shop scheduling problems of different scales. By comparing the results with standard instances and other algorithms, the accuracy of the algorithm is demonstrated.

## KEYWORDS

Q-learning Algorithm; Boltzmann Action Exploration Strategy; Permutation Flow Shop.

## 1. INTRODUCTION

In an era marked by the rapid advancement of technology in the 21st century, the scale of production and operations in the manufacturing industry continues to expand, accompanied by increasing complexity[1]. This shift has made scheduling problems increasingly crucial in the industrial and manufacturing sectors. Among these, the permutation flow shop scheduling problem (PFSP) stands out as a significant research topic in manufacturing. This problem finds broad applications in industries such as industrial manufacturing, semiconductor manufacturing, pharmaceuticals, and food production. Therefore, effectively addressing the permutation flow shop scheduling problem has become a focal point for enterprises and research institutions[2]. However, the complexity of the PFSP cannot be ignored, as it has been proven to be an NP-Hard problem.

Research on PFSP by domestic and international scholars is extensive, particularly in the realm of intelligent optimization algorithms. Yan Hongchao et al[3]conducted in-depth research on the NEH-based heuristic algorithm and proposed an innovative hybrid Crow Search Algorithm. They introduced novel methods to enhance the quality and diversity of the initial population. Li Yang et al[4] proposed an innovative improved Simulated Annealing algorithm. This algorithm optimizes the initial annealing temperature deeply, introduces corresponding computational functions, and adopts a probability-based multi-strategy cooperative search to generate new solutions, significantly improving the quality of solutions.

## 2. ALGORITHM INTRODUCTION

The Q-learning algorithm is an important reinforcement learning algorithm, and the foundation of many reinforcement learning algorithms is the Markov Decision Process (MDP), on which the Q-learning algorithm is based. Specifically, the machine operates within an environment where each state represents the machine's perception of the current environment. The machine can only influence the environment through actions, and when it takes an action, the environment transitions to another state according to some probability. Additionally, the environment provides the machine with a reward based on a potential reward function. In summary, reinforcement learning mainly consists of four elements: states, actions, transition probabilities, and reward functions, as illustrated in Fig.1.

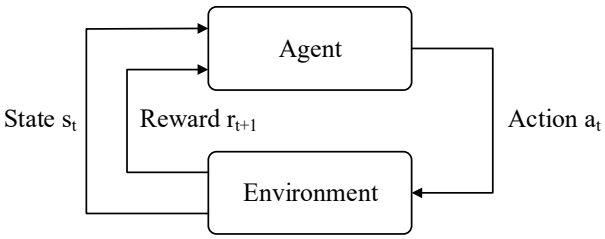


**Figure 1.** Markov state transfer diagram

## 3. ALGORITHM DESIGN

### 3.1. State Space Design

The processing status of jobs is abstracted into an index sequence, where each index corresponds to a unique identifier for a job awaiting processing. In the initial state, this sequence contains the indices of all jobs. Given that the algorithm is implemented in Python, the index of the first job is defined as zero. Therefore, the initial sequence is $S_1 = [0,1,2,\dots,n-1]$, where n represents the total number of jobs, indicating that all jobs are yet to be processed. After selecting a job for processing in each step, the index of the processed job is removed from the sequence, transitioning to the next state $S'$. For instance, if the job with index 1 is selected for processing, the next state becomes $S_2 = [0,2,3,\dots,n-1]$, indicating that job 1 has been processed. This process continues until all jobs are processed, resulting in an empty sequence, $S = []$, signifying the end of one iteration. Based on the designed number of iterations, the state space will be trained to obtain the optimal state representation.

### 3.2. Action Set Design

The choice of action depends on the current state. Given that there are $n$ jobs waiting to be processed, each job may be chosen as the current action. That is, $A_i = \{a_1, a_2, \dots, a_n\}$. According to the selection policy, one of the available jobs $a_i$ is optimally selected for processing. Based on this selection policy, the state is updated accordingly. This process continues until the maximum number of iterations is reached, at which point the actions are selected optimally based on the trained $Q(s,a)$ values.

### 3.3. Reward Function Design

In this section, the performance objective is to minimize the maximum completion time. The reward function $R(s,a)$ is also chosen to minimize the maximum completion time as a feedback function. For the current state $S_i$ and the selected action $a_i$, the reward calculation considers the completion time associated with selecting that action. If the chosen action leads to a smaller maximum completion time, the reward value will increase accordingly; otherwise, it will decrease. Thus, the reward function is inversely proportional to the objective. The reward function is expressed as:

$$R(s,a) = \frac{1}{makespan} \tag{1}$$

### 3.4. $C_{max}$Solution Design

The specific computation steps are as follows:

Step 1: According to the above description, the two-dimensional matrix has a total of $n+m-1$ diagonals.

Step 2: There are $n+m-1$ diagonals in total, and adjacent diagonals have different traversal directions. If the current traversal direction is from top-left to bottom-right, then the next adjacent diagonal's traversal direction is from top-right to bottom-left.

Step 3: Let the index of the diagonal from top to bottom be $i \in [0, m+n-2]$. When $i$ is even, the traversal direction of the $i$ diagonal is from bottom to top; when $i$ is odd, the traversal direction of the $i$ diagonal is from top to bottom.

Step 4: When $i < m$, the starting position of the diagonal traversal is $(i, 0)$; when $i \geq m$, the starting position of the diagonal traversal is $(m-1, i-m+1)$.

Step 5: When the $i$ diagonal is traversed from top to bottom, increment the row index and decrement the column index until reaching the edge of the matrix. When $i < n$, the starting position of the diagonal traversal is $(i, 0)$; when $i \geq n$, the starting position of the diagonal traversal is $(i-n+1, n-1)$.

### 3.5. Algorithm Process

The process of using the Q-learning algorithm to solve the replacement flow shop scheduling problem is shown in Fig.2:

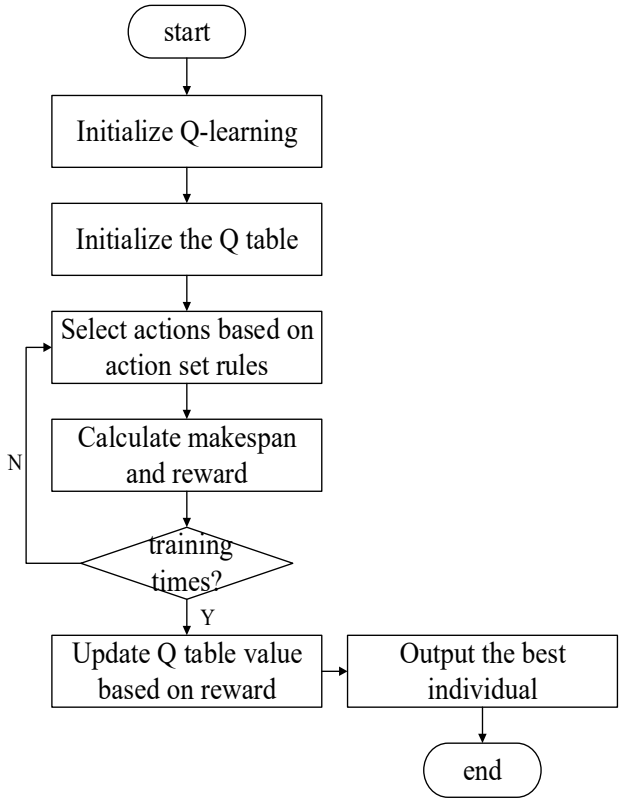


**Figure 2.** Flowchart of Q-learning algorithm for solving PFSP

# 4. EXPERIMENTAL RESULTS AND ANALYSIS

## 4.1. Parameter Settings

The Q-learning algorithm uses python3.11 as the programming language and runs in the Pycharm2023 development environment. After preliminary comparison of experimental results, the impact of different parameters on the results was obtained, and it was finally determined that the number of iterations was 1000, the learning rate was 0.4, and the discount factor was 0.8.

## 4.2. Result Analysis

In order to verify the performance of the proposed algorithm, comparisons are made from the best relative error (BRE), average relative error (ARE). The best relative error (BRE), average relative error (ARE)of all compared algorithms are calculated as follows:

$$ARE = \frac{C_{av} - C_{max}}{C_{max}} \times 100\% \tag{2}$$

$$BRE = \frac{C_i - C_{max}}{C_{max}} \times 100\% \tag{3}$$

Firstly, the Q-learning algorithm for solving the PFSP problem is compared with the Discrete Bat Algorithm (DBA)[5], the Differential Evolution and Estimation of Distribution Algorithm (DE-EDA). A set of 5 Rec test instances is used to test the ARE, BRE, and WRE values of these algorithms. Smaller values of ARE, BRE indicate better optimization performance of the algorithm. The specific test results are shown in Table 1, where the Q-learning algorithm is denoted as QL.

**Table 1.** Comparison of Algorithmic Metrics

| | DBA | | DE-EDA | | HSOS | | QL | |
|---|---|---|---|---|---|---|---|---|
| | BRE | ARE | BRE | ARE | BRE | ARE | BRE | ARE |
| Rec01 | 0.000 | 0.080 | **0.000** | 0.020 | **0.000** | **0.000** | **0.000** | **0.000** |
| Rec03 | **0.000** | 0.081 | **0.000** | **0.000** | **0.000** | **0.000** | **0.000** | **0.000** |
| Rec05 | 0.242 | 0.242 | **0.000** | 0.230 | **0.000** | **0.000** | **0.000** | 0.222 |
| Rec07 | **0.000** | 0.575 | **0.000** | **0.000** | **0.000** | **0.000** | **0.000** | **0.000** |
| Rec09 | **0.000** | 0.638 | **0.000** | 0.010 | **0.000** | **0.000** | **0.000** | **0.000** |
| Rec11 | 1.049 | 2.580 | 1.620 | 2.320 | 0.831 | 2.488 | **0.000** | **0.889** |

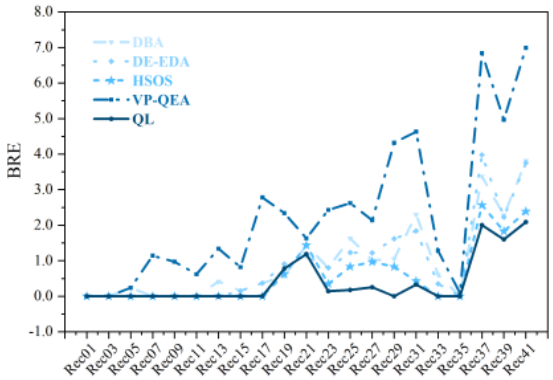


**Figure 3.** Comparison of best relative error results

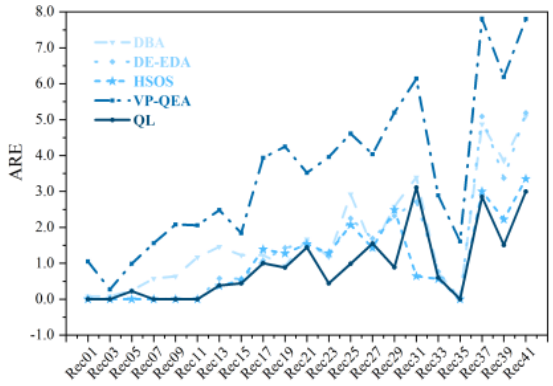


**Figure 4.** Comparison of average relative error results

The BRE, ARE curves of the comparison algorithms are plotted using the Rec test set in Fig.3 Fig.4.The comparative indicator values are all smaller than those of other optimization algorithms. Since smaller values indicate better optimization performance, it can be concluded that the Q-learning algorithm achieves values closer to the optimal solution. This further demonstrates that the optimization performance achieved by the Q-learning algorithm is superior.

To validate the improvement in solving speed of the enhanced Q-learning algorithm, the computation time for each algorithm was calculated for the Rec21 instance. Figure 5 shows the comparison of computation time for each algorithm. It is evident from the graph that the proposed algorithm has the shortest computation time at 43.6 seconds, which is 2.1 seconds faster than the fastest intelligent optimization algorithm, DBA. This demonstrates that using the calculation method with a two-dimensional matrix to compute completion time can indeed enhance computation speed.

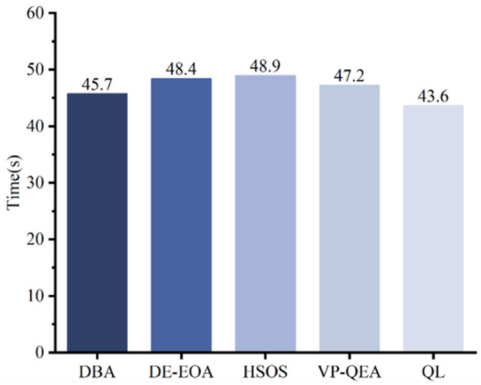


**Figure 5.** Computation time of different algorithms

## 5. CONCLUSION

This paper proposes an improved Q-learning algorithm for solving PFSP, which involves enhancements to the state set, action set, and reward function. Additionally, by leveraging the structure of a two-dimensional array, the algorithm improves efficiency in computing completion time and significantly enhances the convergence speed of the problem.In summary, by improving the Q-learning algorithm for PFSP, faster and more accurate optimization results can be achieved. Future research can further explore the potential application of this algorithm in other domains and combine it with other optimization algorithms for improvement in performance and applicability.

# REFERENCES


[1] WANG L, PAN Z and Wang J. " A Review of Reinforcement Learning Based Intelligent Optimization for Manufacturing Scheduling," in Complex System Modeling and Simulation, vol. 1, no. 4, pp. 257-270, December 2021.

[2] Li XinYu, Huang JiangPin, Li JiangHang. Research and Development Trend Analysis of Dynamic Scheduling in Intelligent Workshops[J]. Science in China: Technical Sciences, 2023,53(07):1016-1030.

[3] Lu Y, Jiang T. Bi-population based discrete bat algorithm for the low-carbon job shop scheduling problem [J]. IEEE Access, 2019, 7: 14513-14522.

[4] Dai M, Zhang Z, Giret A, et al. An enhanced estimation of distribution algorithm for energy-efficient job-shop scheduling problems with transportation constraints[J]. Sustainability, 2019, 11(11): 3085.

[5] Qin Xuan, Fang ZiHan, Zhang ZhaoXin. Solving Permutation Flow Shop Scheduling Problem using Hybrid Symbiotic Organisms Search Algorithm [J]. Journal of Zhejiang University (Engineering Science Edition),2020,54(04):712-721.