Journal of Computing and Electronic Information Management

ISSN: 2413-1660

Research on twin-ASC priority rules in an automated container terminal based on deep reinforcement learning

Yifeng Xu, Jin Zhu*
Institute of Logistics Science and Engineering, Shanghai Maritime University, Shanghai 201306, China.

*Corresponding author Email: jinzhu@shmtu.edu.cn

Abstract: In this paper, the priority assignment problem of twin automated stacking cranes (ASCs) operations in a single block of an automated container terminal based on deep reinforcement learning is considered. First, the basic theoretical knowledge of deep learning and reinforcement learning is introduced, and then a Markov decision model for the twin-ASC priority assignment problem in a single block is established for the problem studied in this paper, and the three elements of Markov decision are defined. And select the six priority rules mentioned in the literature as the candidate action set for decision-making. A deep Q network algorithm is designed for this problem. Finally, the makespan obtained by six priority rules and the DQN algorithm are compared and analyzed through several groups of examples under different production conditions, and the effectiveness of the DQN algorithm is verified.

Keywords: Container terminal, Automated stacking crane priority, deep Q network.

1. Introduction

Automated container terminal uses unmanned equipment to operate, which has the advantages of low labor cost and high safety, but puts forward higher requirements for scheduling and control. It is of great significance to study the equipment control and scheduling strategy of automated terminal to improve the operation efficiency and meet the rising throughput of the port. This paper aims at the vertical coastline type terminal layout described by Zey[1] in the literature.A single block is equipped with two identical ASCs, and the research is carried out when the container needs to be transported from the sea side to the land side by ship to port. ASC will generate job interference during the job process, and the DQN algorithm is used to select appropriate priority rules to minimize the task delay of the makespan.

2. Model Construct

2.1 Model assumptions

In order to simplify the process and fit the reality, this chapter makes the following assumptions in the analysis:(1) The I/O point capacity on both sides is sufficient, ASC does not need to wait for loading and unloading at I/O point. (2) Two ASCs travel at a constant speed, ignore acceleration and deceleration. (3) The picking and dropping of ASC take the same time.

2.2 Define State Features

To transform the twin-ASC priority assignment problem in a single block of an automated container terminal into a Markov decision process, the first thing to do is to define the state characteristics of each moment in the process of assigning priorities. In order to represent the characteristics and changes of the decision-making environment in the process of ASC priority allocation, and to achieve a reasonable prediction of the unknown production process through all current information, the state characteristics obtained information from the decision-making environment need to accurately reflect the twin-ASC in a single block such as location, job status, and task processing volume at different times, including the target position of the task sequence, the task sequence to be processed, the time required for the remaining tasks to be processed, the completion time of the task sequence without interference, and the delay state completion time of tasks.We describe the position and task processing information of the ASC in the yard block at a certain time by defining six different state features. The state features are expressed in the form of the following equations:

$$S = \{f_1, f_2, f_3, f_4, f_5, f_6\}$$

The specific expressions of the above six state characteristics are as follows:

(1)The shortest delay time f_1 for the ASC to process the current task: at the decision moment,the shortest task delay caused by the ASC completing the currently executed task container.

(2) The moving distance f_2 of the task sequence to be processed by ASC: at the decision time, find the job position of the next task container of the current ASC in the task sequence table, and calculate the distance from the current task container group to the next task container group .

(3) Remaining processing time f_3 of the current task request of ASC: at the decision moment, the remaining processing time of the task container currently being processed by ASC.

(4) The length f_4 of the ASC task sequence to be processed: at the initial moment, it represents all the task container assigned by the current ASC that need to be processed. With the continuous change of the decision-making process, the currently

executing tasks and completed tasks are removed, and the obtained the remaining task sequence is the length of the task sequence at the decision moment.

(5) ASC pending task sequence time f_5: At the decision time, the current ASC task sequence list removes currently executing tasks and completed tasks, and the remaining task sequence obtained does not consider the processing required for subsequent decisions time.

(6) Average delay time of tasks completed by ASC: at the decision time, calculate the sum of the delay time generated by the current task sequence completed by ASC.

2.3 Select priority rules to determine action set

In the block, the ASC scheduling priority rule is used as a common strategy to determine the priority when the ASC interferes in the yard. According to the existing rules, the priority can be determined for the ASC at each decision moment, and it can respond in real time when dynamic events occur. Give feedback to increase the efficiency of equipment operation. Different priority rules will inevitably show different advantages and disadvantages under different working conditions. A single priority rule is often not as random or linear as a variety of simple priority rules in the process of dealing with interference. Selecting an appropriate ASC scheduling priority rule to deal with the situation of ASC in a single block can reduce the task delay of ASC due to job interference, and improve the learning efficiency of DQN.

The DQN algorithm is essentially an algorithm framework similar to making decisions through agents. Therefore, in this paper, the twin-ASC scheduling system in a single block, that is, the scheduling environment, is used as the feedback state feature of the agents, so that the ASC priority rules are adopted in this paper. Make a decision to solve the dual ASC priority assignment problem in a single block. Therefore, at each decision moment, the ASC priority rule needs to select a certain job interference and assign the priority to a certain ASC. Since different ASC priority rules are applicable to different production situations, this chapter combines the performance tests of different priority rules in the previous chapter, and selects some five ASCs, such as the seaside priority rule, the remaining completion time smaller rule, and the nearest rule. The priority rule constitutes the action set of the twin-ASC priority assignment problem, and selects the appropriate ASC to assign the priority for the decision moment.The priority rules used are mentioned in the literature[2-5] and the specific description of the species priority rule is as follows:

(1)At the decision time, calculate the respective waiting time of the two ASCs to process the current job interference, and the ASC with less delay time will get priority;

(2)At the decision time, after completing the current task request, the ASC that is closer to the starting point of the next task gets priority;

(3) At the decision time, calculate the time required for two ASCs to complete the

current task request, and the ASC with the shorter time is given priority;
(4) At the decision time, calculate the number of remaining tasks currently assigned by the two ASCs, and the ASC with more remaining tasks will get priority;
(5) At the decision moment, determine the time required for each ASC to complete the remaining tasks without considering the other ASC (without interference), and the ASC that needs the most time to complete the remaining tasks gets priority;
(6) At the decision time, calculate the delay caused by the currently completed tasks of the two ASCs to the original task sequence list, and the ASC with longer delay time will get priority.

2.4 Reward function

In this section, the task sequence table without delay is used as the benchmark, the minimum total delay time for completing the given task is the objective function, and the DQN algorithm is used to find the priority allocation scheme with the largest cumulative reward through iteration.Therefore, after executing action $\mathrm{a_t}$, the state of the ASC scheduling system goes from $\mathrm{s_t}$ to $\mathrm{s_{t+1}}$, and get rewarded $\mathrm{r_t}$, and set it as the opposite of the average delay time of the task sequence before and after the decision, and its formula is as follows:

$$r_t = -[\frac{1}{n_2}\sum_{g\in G'} max(D_g^{'} - W_g, 0) - \frac{1}{n_1}\sum_{g\in G} max(D_g - W_g, 0)]$$

where D_g represents the actual completion time of the total tasks assigned by ASC before the current decision time, the calculation method is the addition of two periods of time, the former period is the completion time calculated from the delayed task table obtained by the previous decision, and the latter period is the completion time calculated without considering the operation interference; $D_g^{'}$ represents the actual completion time for ASC to complete the currently assigned total tasks after executing the current decision, and the calculation method is similar to the above; W_g represents the completion time calculated by the task table without task delay, which is a fixed constant in the case of a given task sequence table; G represents the set of interference that has been processed before the current decision time; G' represents the set of interference that have been processed after the current decision action; n_1 indicates the number of interference that have been processed before the current decision time; n_2 indicates the number of interference that have been processed after completing the current decision action.

3. Numerical experiment

In order to evaluate the effect of the DQN algorithm on the priority assignment problem when the double ASC operation interference occurs in the container area of

the automated container terminal, since there is no ready-made standard example, the examples used in the experiment are all randomly generated. All calculation examples only consider the ASC priority allocation in one block of the automated container terminal yard. Each container area contains 41 bays, the 0th bay is the seaside I/O point, and the 41st bay is the landside I/O point, one block is equipped with two identical ASCs. The task sequence list generated at the initial moment contains 300 tasks. The start or end point of the task is the landside or seaside I/O point, and the corresponding end point or start point obeys the uniform distribution of [1,40] bays. Based on the job completion time of each task obtained by the task sequence table without considering the job interference, calculate the delay of the makespan. It takes one unit time to set the ASC to move one bay, the time to load and unload one container is 30 time units, and the two ASCs need to maintain a safe distance of one bay.

The proposed DQN algorithm is implemented based on Tensorflow 2.0 in a Python 3.8 environment under 64-bit Windows 10, and runs on an Intel Core i5-10400, 2.9 GHz computer with 16 GB of memory. The main parameters of the algorithm are set as shown in the following : training round is set to 500, experience playback volume is set to 3000, sampling batch is set to 30, Network update step size is set to 350,greedy parameter sets the minimum value to 0.5 and the maximum value to 0.9, increment step size is 0.05, discount rate is set to 0.9,learning rate is set to 0.005.

In order to verify the effectiveness of the DQN model obtained after training for the twin-ASC priority assignment problem in a single block, the trained DQN algorithm model was tested for delays under different numbers of tasks, and the results obtained with a single priority rule comparing. Each result is obtained by running ten times independently and taking the average value. Table 1 shows the amount of makepan delay obtained under different methods, in which each example represents the total number of tasks as 50, 100, 200, 300 , 400 and 500.

Table 1 Comparison of completion time delays under different task scales

Case number	total makepan delay						
	priority rule 1	priority rule 2	priority rule 3	priority rule 4	priority rule 5	priority rule 6	DQN algorithm
1	534	546	524	489	475	499	449
2	952	937	942	891	871	908	821
3	1724	1756	1735	1671	1598	1700	1548
4	2678	2645	2686	2514	2486	2597	2388
5	3394	3375	3387	3277	3247	3324	3074
6	4598	4612	4588	4405	4358	4486	4175

As can be seen from Table 1 above, in the case of different generated examples, the priority rules and the DQN algorithm show different performances. As the amount of tasks increases, different rules and algorithms require the delay is also increasing, but the delay obtained by the DQN algorithm is always smaller than the makespan delay obtained by a single rule, so the effect of a single priority rule is obviously not as good as that of the trained DQN model. Choose the appropriate rules for different situations. And the performance of different rules under different task instances is also different. This is due to the randomness of the task, and the task situations that different rules adapt to are obviously different, so it is necessary to select appropriate rules according to different task situations.

Figure 1 shows the action selection frequency of the DQN algorithm when the task volume is 100 and 200. Priority rule 4, priority rule 5, and priority rule 6 are selected more frequently, and their performance under a single rule is relatively good, while the other three priority rules are selected less frequently, so the setting and selection of priority rules is also a factor that affects the performance of the algorithm.

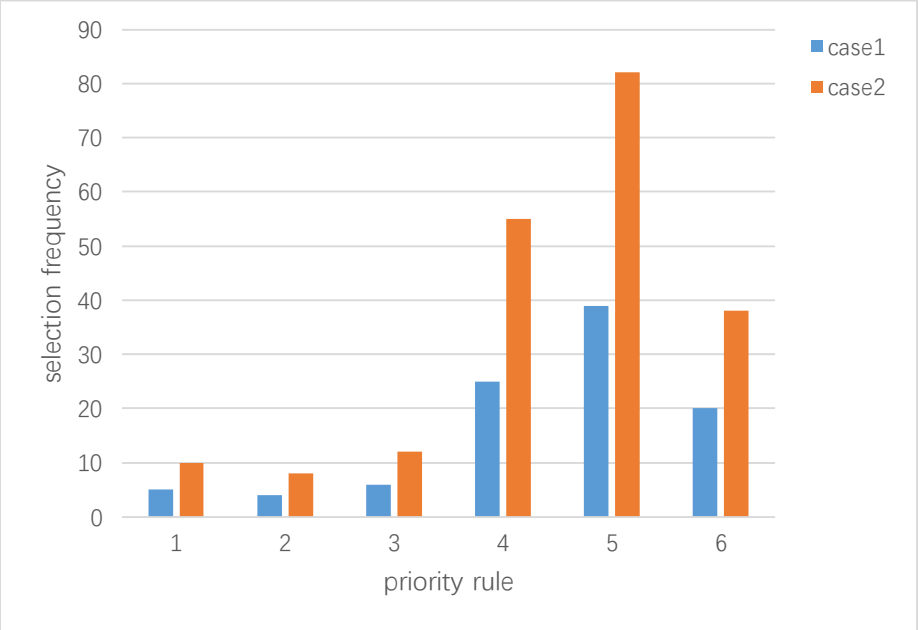

Figure 1 Decision Frequency of DQN in Different Cases

Due to the increase in the number of tasks, it will inevitably lead to an increase in the number of job interference, resulting in an increase in the delay of completion time. Therefore, the following experiments were carried out under the same number of tasks and with different frequencies of interference in the initial task sequence list. The number of tasks is set to 300, and the frequency of interference is controlled by controlling the difference in the number of tasks passing through the middle bay.

Table 2 Comparison of makepan delays under different interfering task ratios

Interference ratio	total makepan delay						
	priority rule 1	priority rule 2	priority rule 3	priority rule 4	priority rule 5	priority rule 6	DQN algorithm
30%	910	905	895	889	875	891	857
40%	1386	1413	1405	1324	1232	1238	1204
50%	1732	1698	1711	1678	1598	1620	1521
60%	1832	1858	1849	1795	1777	1804	1752
70%	2208	2211	2218	2145	2085	2171	2057

As can be seen from Table 2 above, as the ratio of interference tasks increases, the delay time increases, and in the case of using a single priority rule, the priority rule 5 shows better effect, and the effect of priority rule 4, priority rule 5, and priority rule 6 is better than that of priority rule 1, priority rule 2, and priority rule 3. Among them, priority rule 5 has the best effect, but the effect is still slightly worse than that obtained by the DQN algorithm. As a result, the effects of the first three priority rules are also random in different calculation cases.

4. Conclusion

This paper studies the use of the DQN algorithm to solve the priority assignment problem when two ASCs conflict in a single block. The effectiveness of the algorithm is verified by different examples, but the research is carried out given the task sequence. Therefore, in the next work, we can consider combining the heuristic algorithm with the DQN algorithm, use the heuristic algorithm to adjust the job order to reduce conflicts, and use the DQN algorithm to resolve conflicts.

References

[1] Zey L, Briskorn D, Boysen N. Twin-crane scheduling during seaside workload peaks with a dedicated handshake area[J]. Journal of Scheduling, 2021: 1-32.

[2] Carlo H J, Vis I F A. Sequencing dynamic storage systems with multiple lifts and shuttles[J]. International Journal of Production Economics, 2012, 140(2): 844-853.

[3] Park T, Choe R, Ok S M, et al. Real-time scheduling for twin RMGs in an automated container yard[J]. or Spectrum, 2010, 32(3): 593-615.

[4] Dell R F, Royset J O, Zyngiridis I. Optimizing container movements using one and two automated stacking cranes[J]. Journal of Industrial & Management Optimization, 2009, 5(2): 285.

[5] Carlo H J, Martínez-Acevedo F L. Priority rules for twin automated stacking cranes that collaborate[J]. Computers & Industrial Engineering, 2015, 89: 23-33.