International Journal of Production Research

ISSN: 0020-7543 (Print) 1366-588X (Online) Journal homepage: https://www.tandfonline.com/loi/tprs20

Spatial arrangement using deep reinforcement learning to minimise rearrangement in ship block stockyards

Byeongseop Kim, Yongkuk Jeong & Jong Gye Shin

To cite this article: Byeongseop Kim, Yongkuk Jeong & Jong Gye Shin (2020): Spatial arrangement using deep reinforcement learning to minimise rearrangement in ship block stockyards, International Journal of Production Research

To link to this article: https://doi.org/10.1080/00207543.2020.1748247

Published online: 20 Apr 2020.

International Journal of Production Research, 2020
https://doi.org/10.1080/00207543.2020.1748247

Spatial arrangement using deep reinforcement learning to minimise rearrangement in ship block stockyards

Byeongseop Kim[a], Yongkuk Jeong[b*] and Jong Gye Shin[a,c]

[a]*Department of Naval Architecture and Ocean Engineering, Seoul National University;* [b]*Department of Sustainable Production Development, KTH Royal Institute of Technology;* [c]*Research Institute of Marine System Engineering, Seoul National University*

(*Received 3 March 2019; accepted 21 March 2020*)

As the shipbuilding industry is an engineering-to-order industry, different types of products are manufactured according to customer requests, and each product goes through different processes and workshops. During the shipbuilding process, if the product is not able to go directly to the subsequent process due to physical constraints of workshop, it temporarily waits in a stockyard. Since the waiting process involves unpredictable circumstances, plans regarding time and space cannot be established in advance. Therefore, unnecessary movement often occurs when ship blocks enter or depart from the stockyard. In this study, a reinforcement learning approach was proposed to minimise rearrangement in such circumstances. For this purpose, an environment in which blocks are arranged and rearranged was defined. Rewards based on the simplified rules were logically defined, and simulation was performed for quantitative evaluation using the proposed reinforcement learning algorithm. This algorithm was verified using an example model derived from actual data from a shipyard. The method proposed in this study can be used not only to the arrangement problem of ship block stockyards but also to the various arrangement and allocation problems or logistics problems in the manufacturing industry.

Keywords: Asynchronous advantage actor-critic (A3C) algorithm; Deep reinforcement learning; Ship block arrangement; Spatial arrangement

1. Introduction

1.1. Background

In most large shipyards, a ship is divided into several blocks to increase productivity and reduce the time for final assembly process in a dry dock, which is a process that causes a major bottleneck in shipbuilding process. This approach has a benefit of enabling the construction of many ships simultaneously. The blocks that constitute a ship generally follow the unit processes which are forming, assembly, painting, outfitting, and final assembly process. The workshops in which these unit processes are performed are widely distributed in a shipyard and each unit workshop is located considering the shipbuilding process flow and geographic characteristics.

Since multiple blocks are assembled and produced simultaneously in a shipyard, that blocks must be transported from and to the workshops according to the pre-determined schedule from the production planning phase in order to finish the shipbuilding process by the given delivery dates. However, the production schedule can be delayed due to human factors, weather, and unpredictable situation. If the ship production system is a flow type production system such as the automotive and semiconductor production industry, the entire process can delay or halt in this situation. However, only some processes may be partially delayed in the shipbuilding industry since most of the blocks will follow different processes and be transported to different workshops according to the product characteristics. A block that cannot be transported to the next process or workshop due to physical constraints is temporarily transported to the block stockyard and stored until the next process or workshop is free. The storage period in the block stockyard can be as short as a few days and as long as several weeks. A block stockyard is a place where delayed blocks are temporarily stored due to the unpredictable situation. So, it is impossible to consider the time and spatial constraint of the stockyard in the production planning phase.

Generally, a large shipbuilding company has an individual department to manage the transportation of the blocks between the workshops in their shipyard area. This department manages the transportation of the blocks in response to transportation requests from the actual unit processes. When block transportation is requested, the location of the departure and destination is requested in terms of addresses. If the destination is a block stockyard in the transport request, the

*Corresponding author. Email: yongkuk@kth.se

destination is not specified, and it is often provided roughly. Because the block stockyard is not a place to perform the shipbuilding process, it is only a temporary waiting space. In such cases, the transportation driver will transport the block to an empty random address. A block stockyard generally has several entrances and exits. So, there must be no blocks interrupting the entry or exit path of the transportation facility in order to deliver the block. However, the blocks are arranged in the stockyard without considering such circumstances, therefore there are many cases in which unnecessary movement occurs in order to transport and store the block in the stockyard.

Up to now, metaheuristic approaches have been employed in most studies on the stockyard arrangement problem to obtain optimal solutions, and it has been proven that the number of times that blocks are moved can be reduced. However, actual stockyards in the shipyards still rely on the judgment of the workers who operate the transporters rather than using algorithms or systems. In the production stage in shipyards, schedule changes may occur, and the actual circumstances may not match the situation that was previously planned for. In such situations, it is difficult to use solutions that have been pre-calculated through the metaheuristic methods. So, additional calculations must be performed again to obtain optimal results. Therefore, it is difficult to respond flexibly to variable production processes. Also, even if it is possible to estimate rearrangement costs, but the changes in block location after rearrangement cannot be reflected.

Recently, neural network-based machine learning known as deep learning has been garnering attention, and it can help to overcome the limitations of the previously mentioned methods. In deep learning, data are used to train a model so that results similar to the learned data can be produced when new data are entered. During the training stage, forward and backward propagation are performed repeatedly on a large amount of data, which takes a long time. However, during the prediction stage in which the trained model is employed, forward propagation is performed only once, which takes a relatively short time. When a worker transporting blocks in a stockyard decides upon a location to store the block, the worker chooses a specific location that is expected to cause less rearrangement tasks based on their existing knowledge and intuition. If deep learning is applied, a model can be trained to receive the current status of the stockyard as input and decide upon a location that can reduce rearrangement, similar to the decision-making process performed by workers. In this way, the judgments can be made during the arrangement stage based on the current situation rather than choosing a location that was optimised only in the previously calculated production plan.

When applying machine learning to problems, supervised learning can be applied when there is a dataset that can be considered as the correct answer, and reinforcement learning can be applied when a dataset does not exist but there is a viable environment to explore a solution. If supervised learning is applied to actual data created based on the judgment of workers, the expertise of the workers can be learned. However, because it is difficult to improve upon the current situation, a method was selected that can achieve better arrangements by using reinforcement learning.

1.2. Related works

1.2.1. Ship block arrangement problem

The ship block arrangement problem in stockyards was introduced by Park et al. (2007) and has subsequently been the subject of various studies. This problem is also defined as a planar storage location assignment problem (PSLAP), and it is known to be NP-hard problem (Park and Seo 2009; Park and Seo 2010; Tao, Jiang, and Qu 2013). Therefore, as the number of blocks increases, the calculation time increases following an exponential function (Preston and Kozan 2001). Park et al. (2007) proposed a heuristic algorithm that can reduce the number of obstacle block's movement, and Park and Seo (2009) defined a mathematical model of the problem and improved the previous heuristic algorithm. Park and Seo (2010) formalised the problem and developed a heuristic algorithm based on a genetic algorithm (GA) as well as a dynamic heuristic algorithm. Roh and Im (2011) also performed a study on minimising rearrangement based on a GA. Regarding studies related to shipyard block logistics, Roh and Cha (2011) studied the scheduling problem in which a transportation between stockyards, and Jeong et al. (2018) suggested a spatial arrangement algorithm for the ship block stockyards that can increase space utilisation.

The ship block arrangement problem is similar to the container arrangement problem in cargo terminal area. This problem is an NP-hard problem, and various studies have been conducted. Preston and Kozan (2001) defined the container location model using a mixed integer linear programming and solved it with the GA. Bazzazi, Safaei, and Javadian (2009) applied the GA to actual container schedule data, and the calculation time was increased according to a polynomial function as the size of the problem increased.

1.2.2. Reinforcement learning

Reinforcement learning is a method in which an agent learns behaviour by trial and error in a dynamic environment, and it has been used for a long time in the machine learning and artificial intelligence fields (Kaelbling, Littman, and Moore

1996). The history of reinforcement learning is long, but in 2013, deep reinforcement learning research developed rapidly, beginning with the Q-learning algorithm, which is based on deep learning technology (Mnih et al. 2013). Mnih et al. (2015) developed an algorithm called deep Q-network (DQN) and trained an agent that exhibits levels of performance similar to those of humans in the simple computer game. In 2016, the AlphaGo used reinforcement learning that it performed autonomously, supervised learning in which it learned the games of Go players, and the Monte Carlo tree search (MCTS), becoming the first program to beat the world champion in Go (Silver et al. 2016). In addition, AlphaGo Zero, which was developed in 2017, showed vastly better performance than the existing the AlphaGo by using only reinforcement learning and MTCS without supervised learning (Silver et al. 2017). Most recently, the AlphaStar was developed by applying reinforcement learning and supervised learning to the StarCraft II and which was rated above 99.8% of officially ranked human players (Vinyals et al. 2019).

However, since the AlphaGo and the AlphaGo Zero performed the MCTS simulation process in addition to neural network learning, they required considerable amounts of hardware performance. Regarding the algorithm, that can be applied to somewhat simpler and more common problems, the asynchronous advantage actor-critic (A3C) approach developed by Mnih et al. (2016) is known as better performing algorithm than the previous algorithms. In this method, multi-threading is used to train actor-critic agents in parallel. Only one multi-core CPU and a shorter amount of time are required to train agents that show better performance than when the DQN approach is employed, which uses a GPU. AlphaStar (Vinyals et al. 2019) also uses the advantage actor-critic based algorithm, and asynchronously update the network.

There have been many studies to apply reinforcement learning to the production research area. Wang, Li, and Zhu (2012) proposed a reinforcement learning algorithm fused with heuristics and applied it to lot scheduling problems. Hwang and Jang (2020) applied a Q-learning based algorithm to the routing of semiconductor logistics facilities. Recently, there is a case where reinforced learning is applied to train the transfer unit of the automation line. At this time, Q-learning was used, and a system environment was defined using a discrete event system model (Shi et al. 2020).

1.3. Method and structure

An environment for a spatial arrangement problem in the ship block stockyards was defined in this study. It logically defines rewards for arrangement and rearrangement tasks and simulates the status to calculate the rewards. The deep learning model helps to find the solution with the highest reward value. The separated deep learning model is constructed using separate stages: one for the finding a location when block arrives, and one for the rearrangement.

This method was employed to find solutions to the problem of determining the block locations while simultaneously considering time and spatial constraints in the stockyards. For this purpose, a simple example model was developed, and the proposed model and reward evaluation method were confirmed to be suitable. Actual shipyard data were used to find a solution based on the block movement requests given to a block stockyard during the shipbuilding process. The objective of the proposed method is to determine the location that minimises unnecessary transporter movement during the rearrangement stage. The proposed model was verified using actual shipyard production planning data, and it was confirmed to provide improved results compared to when the operators manually arrange blocks or when heuristic methods are used.

The rest of this paper is organised as follows. In Section 2, we discuss the ship block arrangement problem and the decision process in the ship block stockyards. In Section 3, we summarise the analysis and learning algorithms for applying reinforcement learning to given problems. Section 4 describes the use of the proposed algorithm to train the agents that make decisions and presents an evaluation of the trained results.

2. Problem definition

2.1. Ship block logistics and arrangement in shipyards

Stockyards in a large shipyard can be divided into normal stockyards, which belong to particular workshops, and common stockyards, which store blocks from multiple workshops. If the block has finished the preceding process while the next process cannot start immediately, the block will be stored in a normal stockyard near the subsequent workshop. If there is no space left in a normal stockyard, the block will be stored in a common stockyard with the blocks that will be transported to different workshops.

Normal stockyards are relatively small, and not many blocks can be stored in there. Normal stockyards are close to the workshops to which the blocks will be transported. In normal stockyards, there is little rearrangement work due to obstacles when blocks are moved in and out of the stockyard. As such, there are few problems related to this situation. However, common stockyards store large numbers of blocks, and there are many cases of obstacles occurring when blocks are moved in and out. It is also important to reduce transporter usage in this case since the distances that transporters must move are

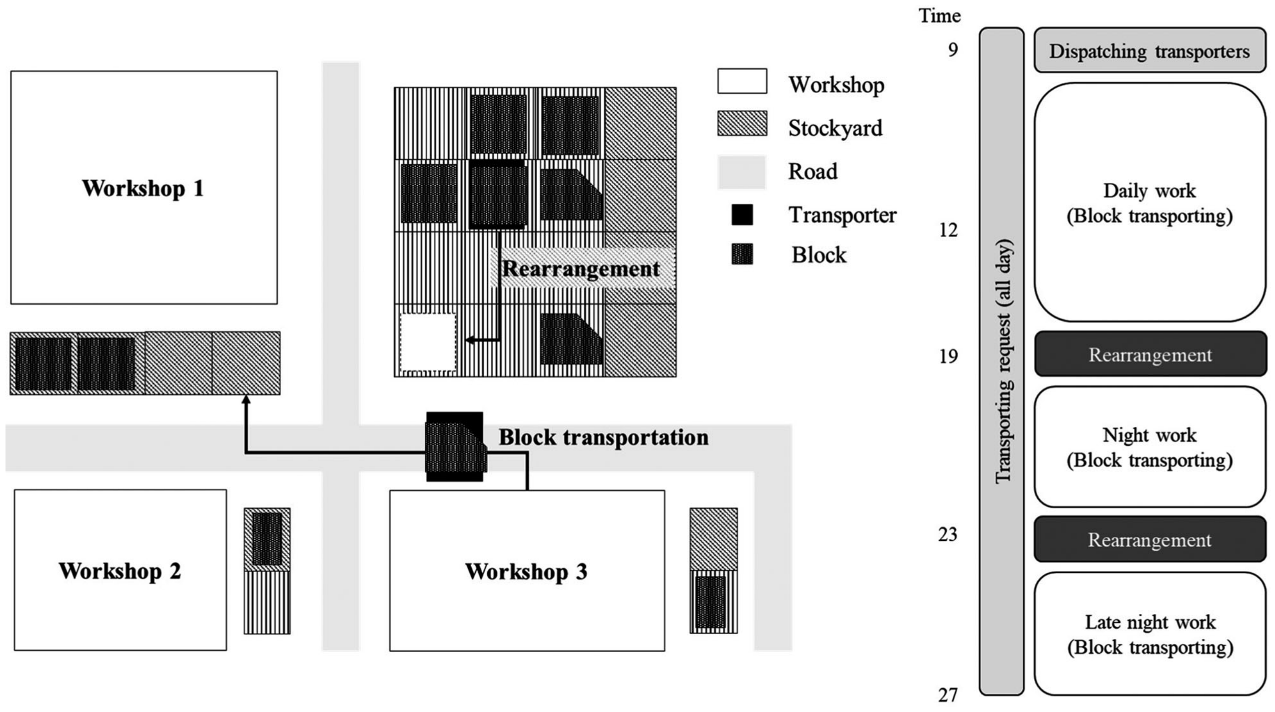

Figure 1. Block arrangement problem and operational processes in shipyards.

long and cannot be ignored. For these reasons, the management of common stockyards at the shipyard is generally an issue, and in this study the target problem was focused on the usage of transporters in common stockyards.

The task of moving the block in a stockyard to a different address is defined as a rearrangement task, and this task is generally performed to remove blocks that are acting as obstacles when moving other blocks. In terms of transporter usage, rearrangement can be seen as additional waste, and it is one of the tasks that must be minimised in ship block logistics. If transporters perform this kind of rearrangement work each time a block is transported, situations occur in which several transporters that are needed for other tasks are working simultaneously in a single stockyard. In such cases, task efficiency is reduced. Therefore, in large shipyards, rearrangement work is normally performed during off hours after the normal working hours. The block arrangement problem and the operational processes are shown in Figure 1.

Rearrangement work in the common stockyard is limited to moving the blocks that are expected to be shipped out the next day to the specific address that is close to the main roads. Using this approach removes the need to perform rearrangement that involves moving blocks to different addresses within the stockyard during the block transportation work. Also, in terms of the transporter usage, it can be separated into block entry/exit work and rearrangement work. Therefore, the decision process was also defined as consisting of two stages in this study to reflect these transporter operation processes.

2.2. *Decision processes in the ship block arrangement problem*

The ship block arrangement problem consists of two decision processes: 1) determining the arrangement of incoming blocks and 2) deciding on rearrangements. The number of block movements will be different, and it depends on the original location of the block and status of the stockyard. Similarly, the location of the next block must be chosen differently according to the results of the rearrangement. These two decision processes are performed by the block transport operators in a shipyard, and the overall decision process in block stockyards arrangement problem is shown in Figure 2.

An entity that receives information about a given environment as input and then makes a decision about the environment is defined as an agent (Russel and Norvig 2010). To create a system for reinforcement learning, the two decision processes were assigned to separate agents in this study. A system that consists of several such interacting agents and the environment is herein called a multi-agent system (MAS). In this study, the block stockyard problem was defined as a MAS, and a reinforcement learning algorithm was used to train each agent.

To perform reinforcement learning, the environment for the ship block arrangement problem must first be defined. Generally, to solve a reinforcement learning problem, the environment must be defined so that it conforms to a Markov

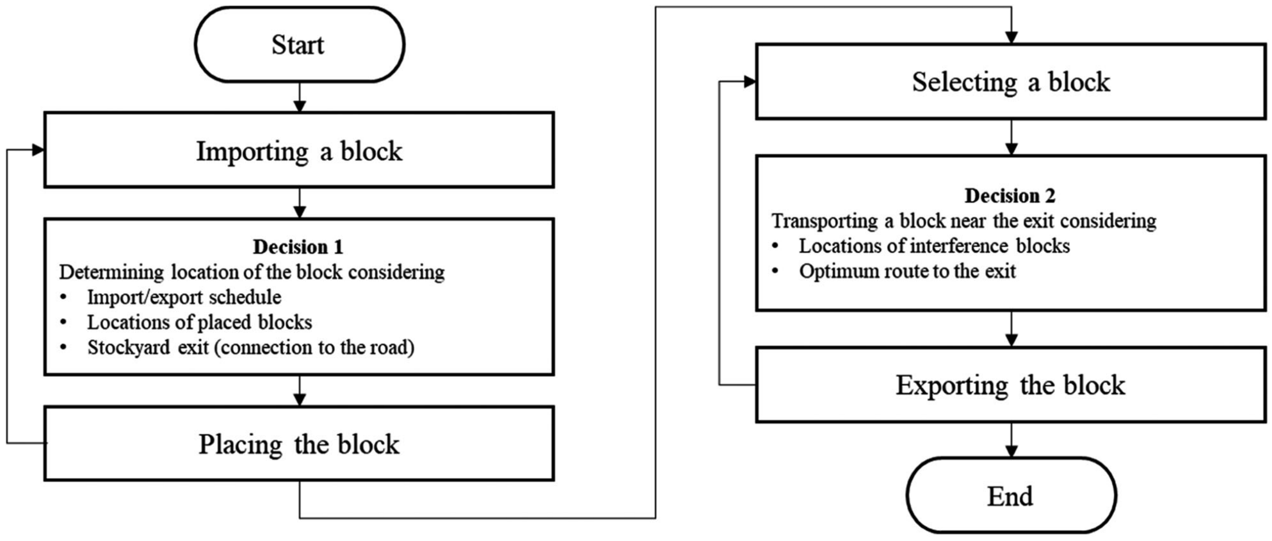
Start
Importing a block
Decision 1
Determining location of the block considering
• Import/export schedule
• Locations of placed blocks
• Stockyard exit (connection to the road)
Placing the block
Selecting a block
Decision 2
Transporting a block near the exit considering
• Locations of interference blocks
• Optimum route to the exit
Exporting the block
End

Figure 2. Decision processes in block stockyards arrangement problem.

decision process (MDP). In the MDP, if the t^{th} state (S_t) and action (a_t) of the system are given, the next state (S_{t+1}) is determined by the transition probability $p(S_{t+1}|S_t, a_t)$, and rewards are determined by the state (S_t), action (a_t), and next state (S_{t+1}) (Sutton and Barto 2012). However, in the MDP theory, there must be no other adaptive agents in the environment other than the agent who will be trained. Therefore, the MAS problem cannot be defined by the ordinary MDP. Consequently, to apply reinforcement learning to the MAS problem, the Markov game framework was proposed. This framework introduces the concept of a transition probability function that includes a set of several actions in a single state (Littman 1994). However, most recent reinforcement learning algorithms that show good performance are focussed on single-agent systems, which are defined by the MDP. Therefore, the MDP must be defined to apply this algorithm. Considering the characteristics of the ship block arrangement problem that was the focus of this study, a method was developed that assumes a limited situation, defines the decision processes of each agent as MDPs, and uses a reinforcement learning algorithm that can be applied in general problems.

3. Reinforcement learning algorithm for ship block arrangement in stockyards

3.1. Analysis of the ship block arrangement problem in stockyards

Table 1 defines the states, actions, and rewards of the transporting agent (TA) and the locating agent (LA), which are the two agents that were trained in this study. The LA decides upon the location so as to minimise the number of times rearrangement occurs during the next rearrangement based on the current status of the blocks in the stockyard. Ship block arrangement in a stockyard is performed by small address units. The stockyard was assumed to have a rectangular grid with m cells horizontally and n cells vertically, and it was assumed that only the right vertical side was used as an entrance/exit. Because the state must include not only the current status of the blocks, but also the road direction, it is defined in the form of m cells horizontally and $n+1$ cells vertically. As such, the t^{th} state is $S_t = [s_{ij}] \in \mathbb{R}^{m\times(n+1)}$. Here, the agent must be able to recognise not only the location information of the blocks that are being stocked, but also the remaining period in stockyards. Therefore, a case in which there is no block is given a value of -1, and when there is a block, a value from 0

Table 1. Definitions of reinforcement learning components pertaining to the LA and TA.

Agent	State	Action	Reward
LA	Blocks in stockyard, including location and remaining days information	Selecting the arrangement location of the next block	Number of blocks moved during the rearrangement stage
TA	Blocks and transporter in stockyard	Moving a transporter or carrying a block on a transporter	Success or failure of exporting a block

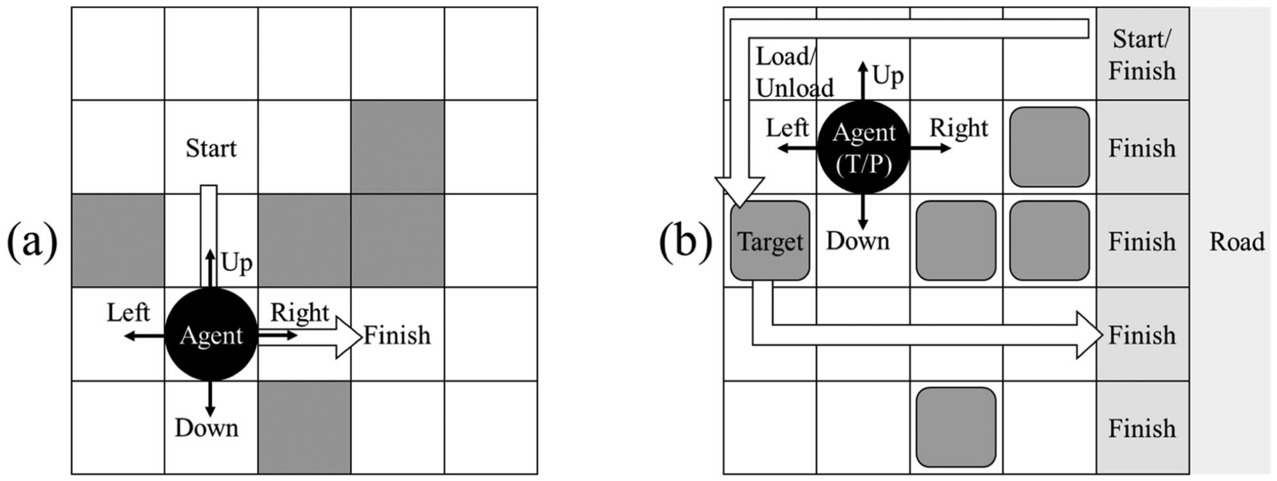

Figure 3. Examples of (a) a grid and (b) rearrangement problem.

to 2 is given as a relative ratio of the remaining period of the already stocked blocks to that of the blocks to be stocked. The maximum value of relative ratio is limited to 2 to prevent differences in scale of the input data. Also, the empty cell is defined as -1 to distinguish the empty cell from the blocks relative period near zero. Therefore, the values in row i and column j of the state can be expressed as $s_{ij} = -1 \text{or} 0 \leq s_{ij} \leq 2$. The arrangement location, which is the choice of the LA, is decided in the action space A, which is the same size as the stockyard. Therefore, it is expressed as $a_t \in A = [a_1, a_2, \ldots, a_{m\times n}]$. At this time, because the rearrangement event may not occur directly after arrangement, the reward for the action of the LA r_t is not directly given. Instead, it is calculated according to the number of block movements that are obtained when the rearrangement event occurs, in which the t^{th} block is shipped out. Each step in LA corresponds to an action that locates a block, and each episode corresponds to an arrangement schedule for a specific period, with 50–100 block schedules being used in the case study.

The TA performs rearrangement, which is the process of arranging the blocks again that will ship out the next day in the direction of the exit after the arrangement events of the current day have been completed. The blocks that are scheduled to be delivered the next day in the production plan must be moved to addresses near the entrance/exit so that they do not cause obstacles when shipping blocks out. The rearrangement problem has aspects that are similar to a grid world, which is a simple and commonly used problem in reinforcement learning. As such, they are defined in a form that uses a grid world, as shown in Figure 3. In a normal grid world problem, a 2D grid space is used as a state, and agents move up, down, left, and right. If an agent arrives at the goal, it receives a reward. According to the problem, there are cells in the grid corresponding to obstacles and the destination, and this information is displayed in the state. Therefore, the agents look at it and decide on actions. The TA does not simply move to the destination. Instead, it must perform the processes of loading and unloading the selected block into the transporter and transporting the blocks to the selected locations. Therefore, each grid cell of the state can be expressed as $S'_t = [s'_{ij}] \in [0,\ 1,\ 2,\ 3,\ 4,\ 5]$ so that it can be identified as a cell that currently has a transporter, an empty cell, a cell that has a block, a cell that has a block that must be transported, or a cell that has a destination. The TA decides on an action from among five actions: moving up, down, left, or right or loading/unloading a block. As such, the actions can be expressed as $a'_t \in A' = [a'_1, a'_2, a'_3, a'_4, a'_5]$. If the block is moved to the selected address near the entrance/exit, a positive reward is received. If the block cannot be moved to the selected location, a negative reward is received. When a positive reward is received, the reward value is reduced according to the number of block movements. To receive the maximum reward, the rearrangement must be performed with minimal block movement. In a block stockyard, there are cases in which the road to the destination is obstructed by other blocks. In such cases, the blocks that are obstructing the road must be moved to different locations before the block that must be transported is moved. However, in this study, the decision process of the TA was not partitioned separately, and it was designed so that the episode finishes and the reward is received only if the specified block is moved to the selected goal.

Reinforcement learning is the process of finding the optimal policy for a given MDP. When acting according to policy π, the value function of each state $v_\pi(s)$ can be expressed in the Bellman equation form, as shown in Equation (1), and the Bellman optimal equation with maximum reward can be expressed as in Equation (2) (Sutton and Barto 2012). Depending on the reinforcement learning algorithm, the method of solving the Bellman equation differs. Typical examples for such methods include Q-learning using action-value functions and policy gradients using policy functions. In this work, the A3C algorithm, which is a type of actor-critic algorithm, was used.

$$v_\pi(s) = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t = s\right]$$
$$= \sum_a \pi(a|s)\sum_{s'} p(s'|s,a)[r(s,a,s') + \gamma v_\pi(s')] \tag{1}$$
$$v_*(s) = \max_{a\in A(s)}\sum_{s'} p(s'|s,a)[r(s,a,s') + \gamma v_*(s')] \tag{2}$$

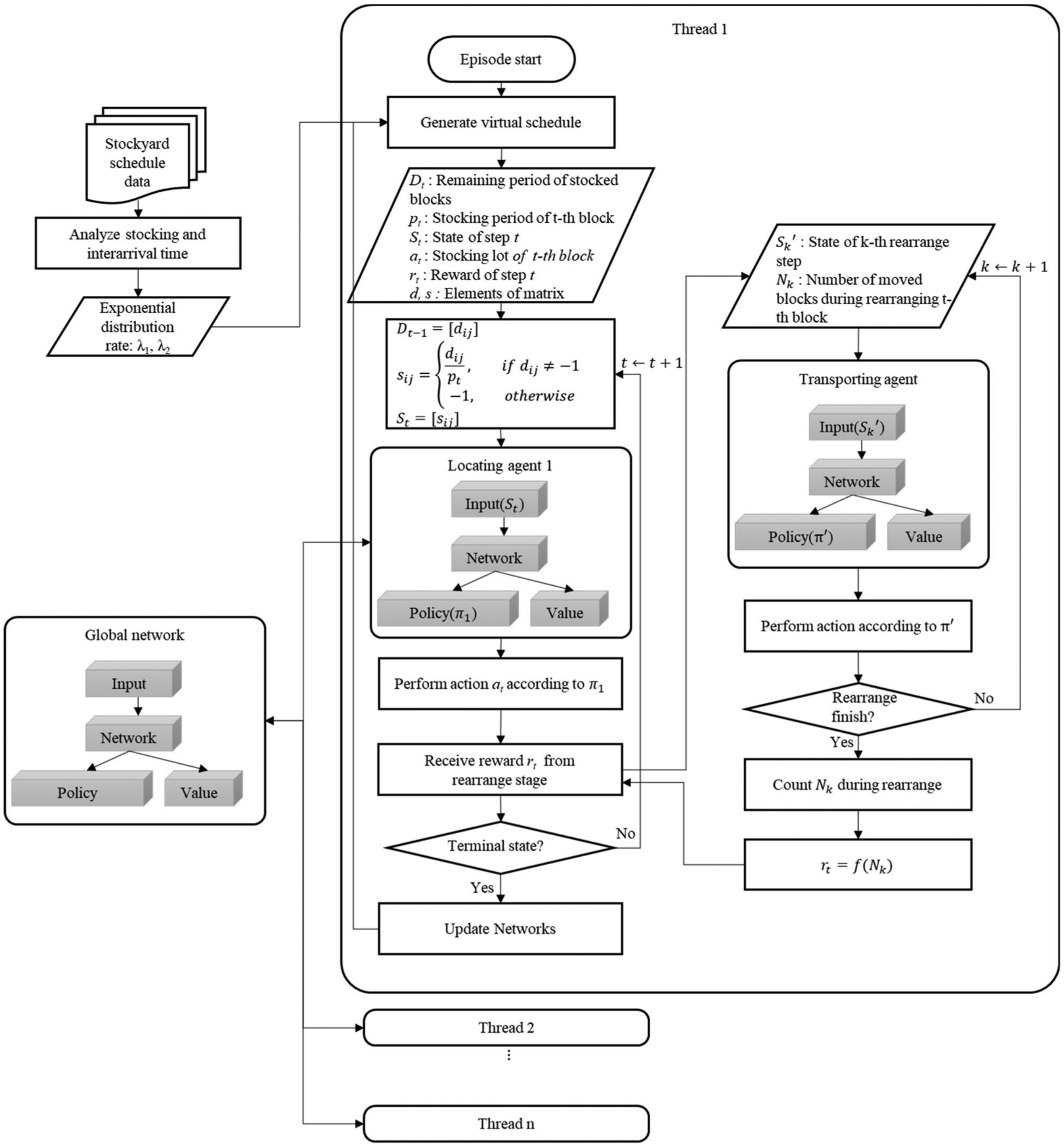
Thread 1
Episode start
Generate virtual schedule
Stockyard schedule data
Analyze stocking and interarrival time
Exponential distribution rate: λ_1, λ_2
D_t : Remaining period of stocked blocks
p_t : Stocking period of t-th block
S_t : State of step t
a_t : Stocking lot of t-th block
r_t : Reward of step t
d, s : Elements of matrix
$D_{t-1} = [d_{ij}]$
$s_{ij} = \begin{cases} \frac{d_{ij}}{p_t}, & \text{if } d_{ij} \neq -1 \\ -1, & \text{otherwise} \end{cases}$
$S_t = [s_{ij}]$
$t \leftarrow t+1$
Locating agent 1
Input(S_t)
Network
Policy(π_1)
Value
Perform action a_t according to π_1
Receive reward r_t from rearrange stage
Terminal state?
No
Yes
Update Networks
Global network
Input
Network
Policy
Value
S_k' : State of k-th rearrange step
N_k : Number of moved blocks during rearranging t-th block
$k \leftarrow k+1$
Transporting agent
Input(S_k')
Network
Policy(π')
Value
Perform action according to π'
Rearrange finish?
No
Yes
Count N_k during rearrange
$r_t = f(N_k)$
Thread 2
Thread n

Figure 4. LA training process based on the A3C approach.

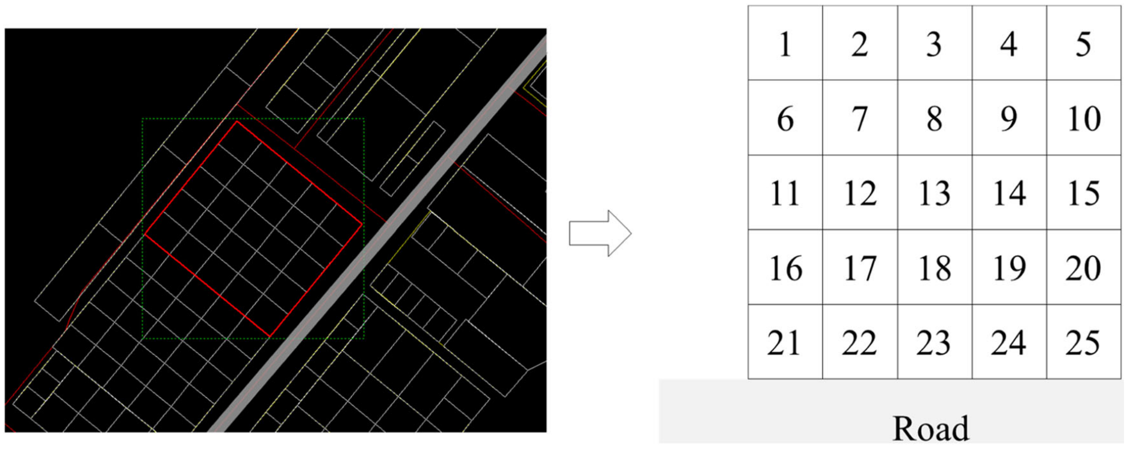

Figure 5. Geographical information and schematic drawing of the target stockyard.

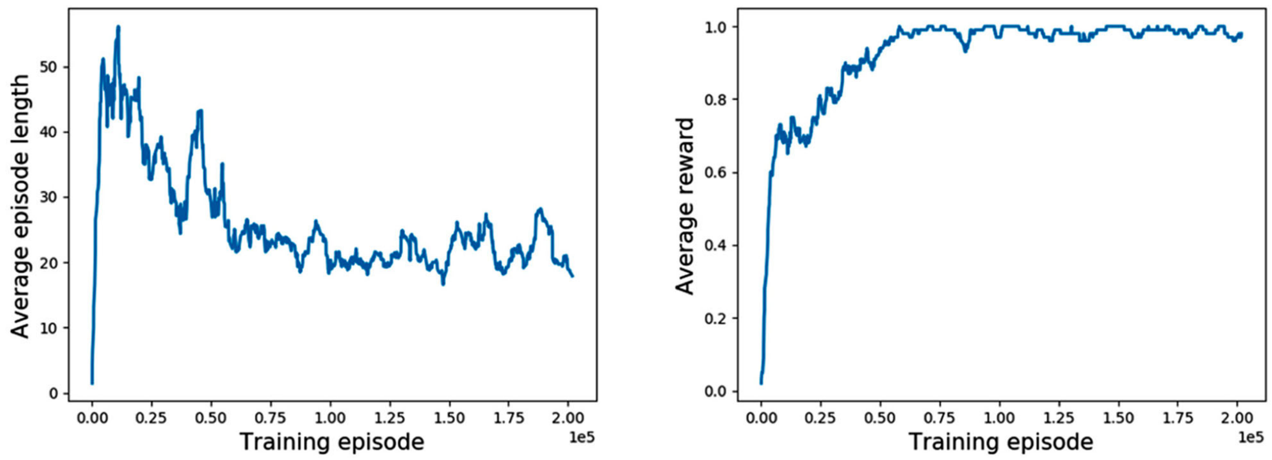

Figure 6. Episode length and reward during a training TA for 200,000 episodes.

3.2. Reinforcement learning algorithm to solve the ship block arrangement problem

It was necessary to use the previously defined MDP to define an algorithm that could train the two agents. In a normal MDP problem, the existing reinforcement learning algorithms such as the DQN, A3C, and policy gradient approaches can be used as-is, but the block stockyard problem have two features to consider during learning process. The first is that the LA cannot receive a reward immediately after an action. The reward of the LA is determined by the number of times obstacles occurred in the block, which corresponds to the action in the rearrangement process. However, because the rearrangement process does not occur for every one of the actions of the LA, the LA can only receive a reward if the next rearrangement is finished. Second, the reward of the LA is determined by the TA, rather than the environment. Therefore, in order to receive a reward, there must a process of interaction with the TA.

Regarding the TA, a single rearrangement process is a single episode. If the initial state of the rearrangement is determined based on the results of the actions of the LA, there is no interaction with the LA until the end of the episode. Also, unlike the LA, the TA can receive a reward immediately at each step. Therefore, if the previously defined components are used, it is possible to use the existing A3C approach as-is. Consequently, a method was used in which the TA was firstly trained sufficiently, and then the trained TA model was used to train the LA.

The process of training LA based on the A3C algorithm is defined such that the two features mentioned above are considered. This training process employs multiple threads in a single machine like the A3C method. The actor-learner in each thread performs learning through the process shown in Figure 4. The reward in each step is obtained from the rearrangement results of the located block. In this rearrangement process, the pre-trained model of the TA is used, and no additional training of the TA is performed.

An agent in a single LA thread selects actions stochastically according to the policy of $\pi_i(a_t|S_t;\theta_i)$, which is calculated using a network composed of parameter vectors θ'. The parameter vectors θ'_v are used in the same network to calculate

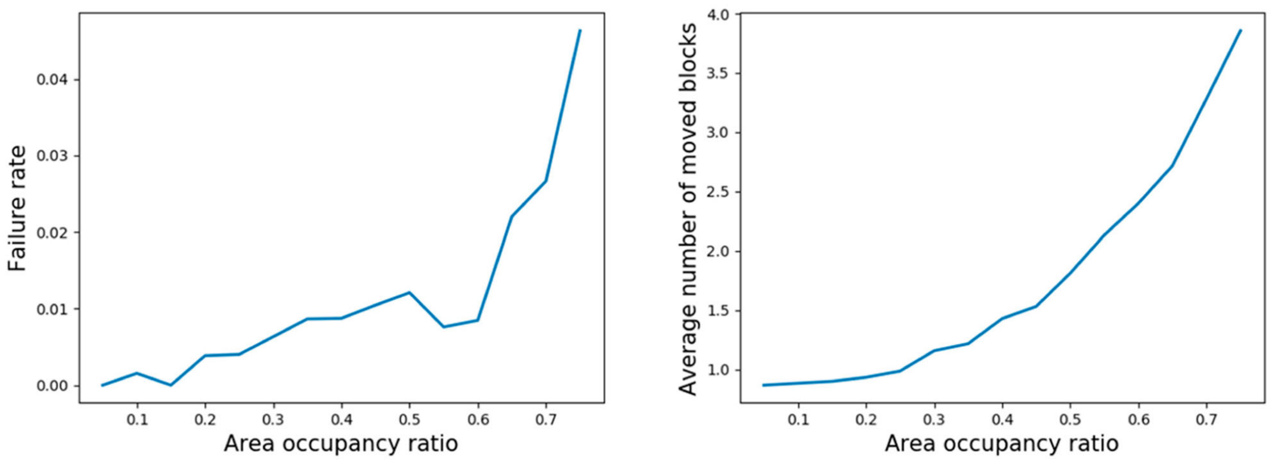

Figure 7. Failure rate and average number of moved blocks according to area occupancy ratio.

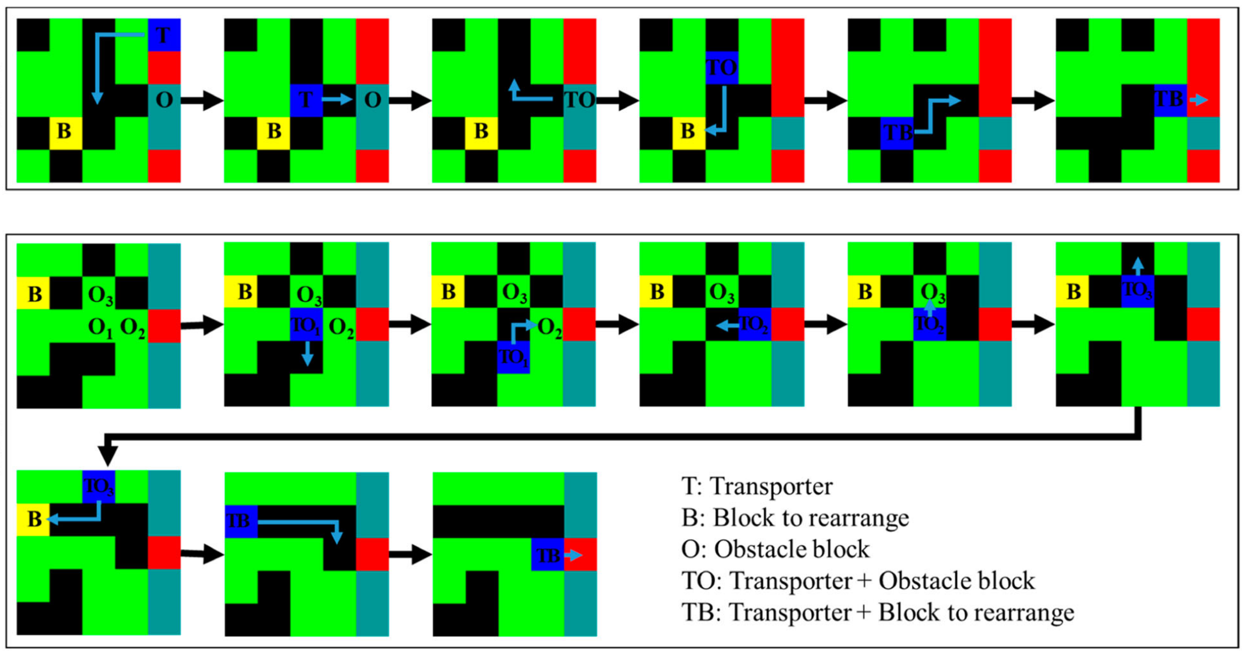
Case 1
Case 2
T: Transporter
B: Block to rearrange
O: Obstacle block
TO: Transporter + Obstacle block
TB: Transporter + Block to rearrange

Figure 8. Examples of rearrangement process using the trained TA.

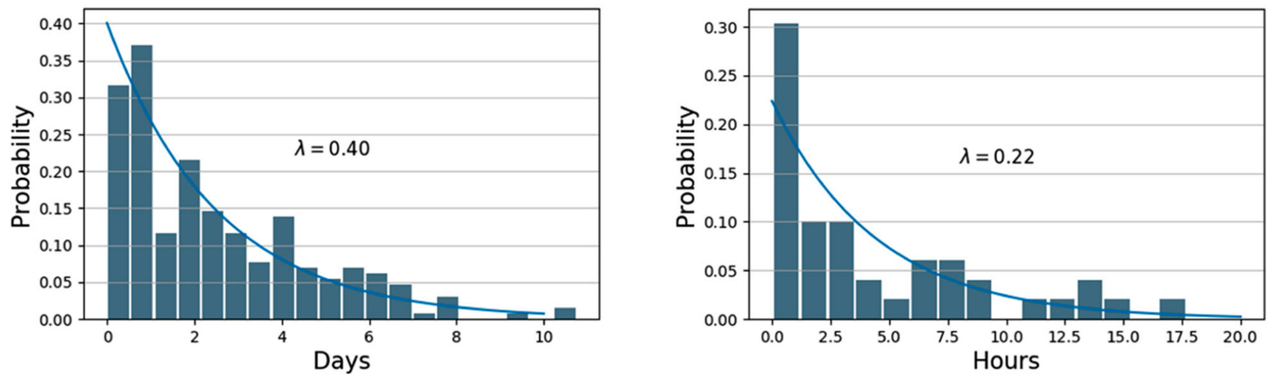

Figure 9. Histograms and probability distribution functions of arrangement period and interarrival time.

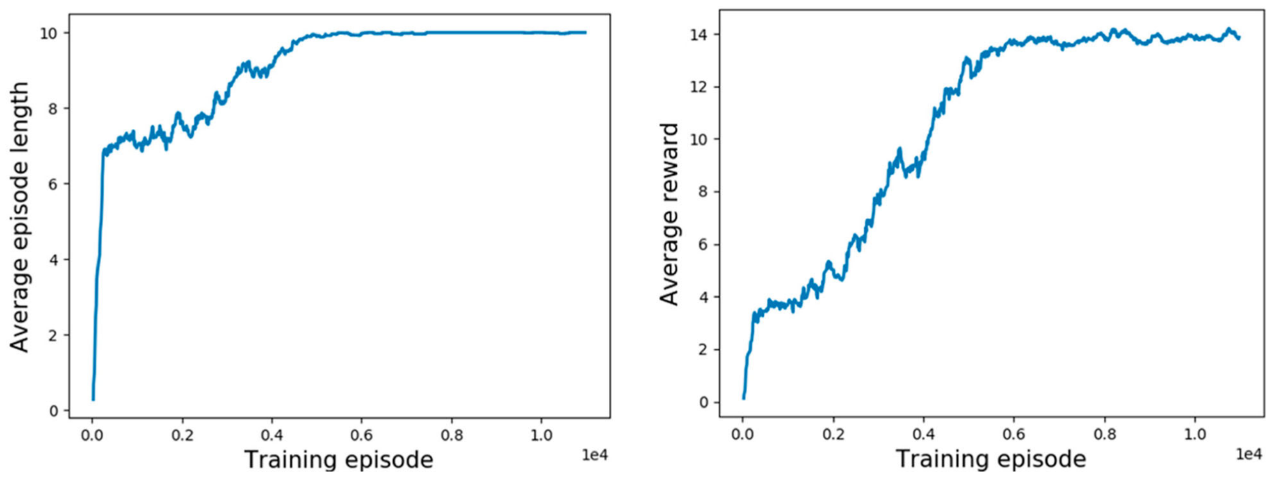

Figure 10. Episode length and reward during a training LA for 11,000 episodes.

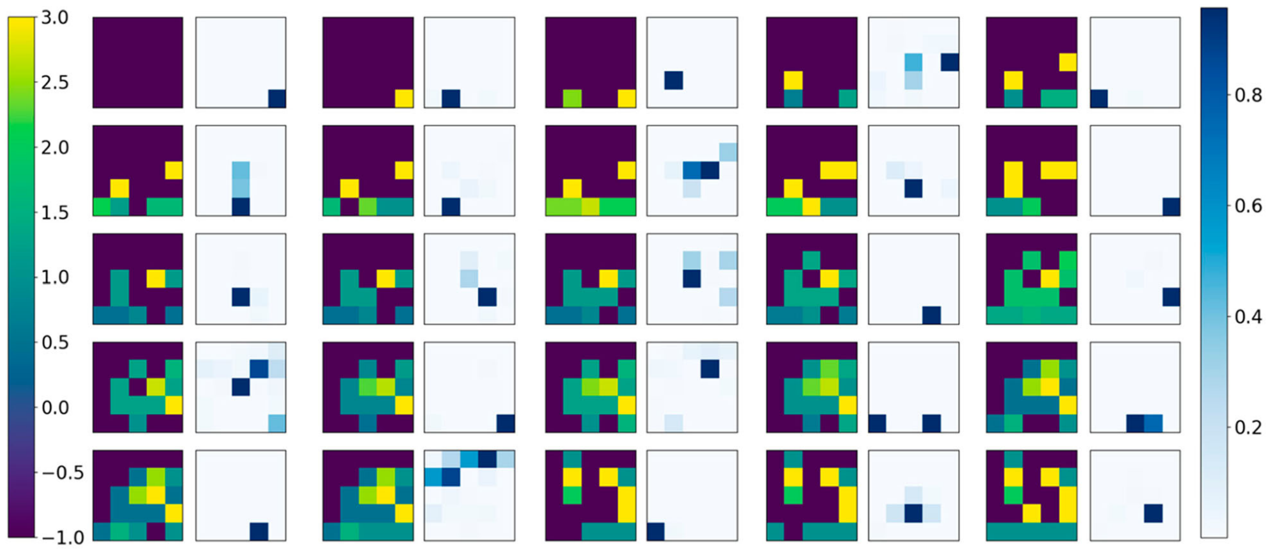

Figure 11. Example policy for the trained LA.

the value of $V(S_t;\theta'_v)$, which corresponds to the expected return of the state. The reward r_t was designed to be large if the number of block movements N_k decreases as the TA completes a given rearrangement. N_k is calculated according to the rearrangement event that occurs at the time of exporting the t-th block and not immediately after execution of a_t. Therefore, each action and its corresponding reward can be completely determined upon the end of the episode. These rewards can then be accumulated to calculate the return $R_t = \sum_{l=0}^{\infty}\gamma^l r_{t+l}$. The advantage function, which is used to update a network, is obtained by subtracting the value function from the action-value function. A network update is performed if an episode is completed by arriving at a terminal state or in a chosen maximum step. Equations (3) and (4) can be used to calculate the gradients of the parameter vectors θ and θ_v, which are subsequently employed to update the parameters of the global network (Mnih et al. 2016).

$$d\theta \leftarrow d\theta + \nabla_{\theta_i}\log\pi_i(a_t|S_t;\theta_i)(R_t - V(S_t;\theta'_v)) \tag{3}$$

$$d\theta_v \leftarrow d\theta_v + \frac{\partial(R_t - V(S_t;\theta'_v))^2}{\partial\theta'_v}. \tag{4}$$

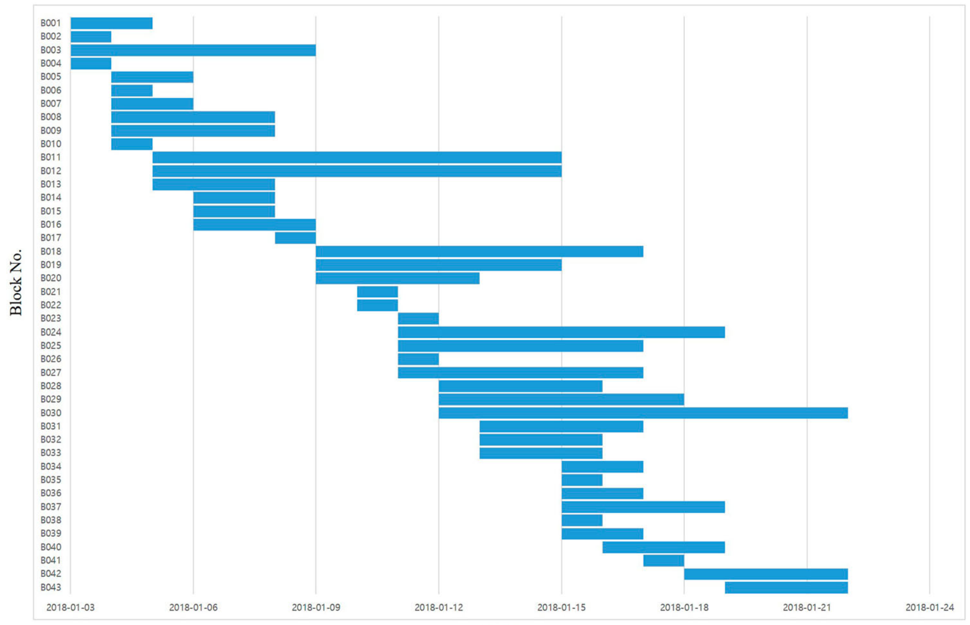

Figure 12. Schedule for a single block stockyard.

4. Experiments

4.1. TA training

The TA was trained first, using the general A3C algorithm. The goal of the TA is to transport stocked blocks that will be delivered on the next day to the addresses adjacent to the main road. Transporting one selected block to its destination is one episode. When the destination is reached, a singlereward is received. The TA was trained independently without the LA. A random number of blocks in a stockyard were arranged at random locations, and one random block out of these was selected to be shipped. This design was chosen so that training could be performed for a variety of cases.

The environment was set up so that 1–20 blocks were arranged at random locations in a rectangular stockyard with the same 5×5 size as the actual target stockyard. Figure 5 shows the geographical information and a corresponding schematic drawing of the target stockyard. Figure 6 shows the changes in episode length and rewards when training was performed for 200,000 episodes. The episode length gradually decreases and converges, and the reward gradually converges to near the value 1.0.

When the transporter transported blocks normally to the destination, a reward of 1 was received regardless of the number of movements. Therefore, it is evident that block transportation was successful in almost all episodes. The ratio of successful transportation by the transporter is related to the number of blocks stocked at the stockyard. Therefore, when the stockyard was almost full of blocks, cases sometimes occurred in which blocks could not be shipped out normally. Figure 7 shows the relationship between this failure rate and the occupancy rate of the stocked blocks. When the occupancy rate of the blocks is less than 60%, the failure rate is very low, at less than 1%. However, when the occupancy rate exceeds 60%, the failure rate increases, although it is less than 5% even at its highest. Thus, when blocks were shipped out, the number of blocks moved by the transporter tended to increase, as evidenced by the fact that the obstacle blocks become more numerous as the occupancy rate increases.

Figure 8 shows the part of the process of the trained TA performing rearrangement in a situation with three obstacle blocks. In this situation, the TA removes the obstacle blocks and generates paths by itself even when there are blocks obstructing the target block rearrangement. Even with several obstacle blocks, the TA was successful at rearrangement without problems in most cases.

4.2. LA training

The LA was trained according to the algorithm described in Section 3.2. In LA training, the arrival and delivery schedule of the blocks and a pre-trained TA to calculate the number of block movements are required. The TA was trained using

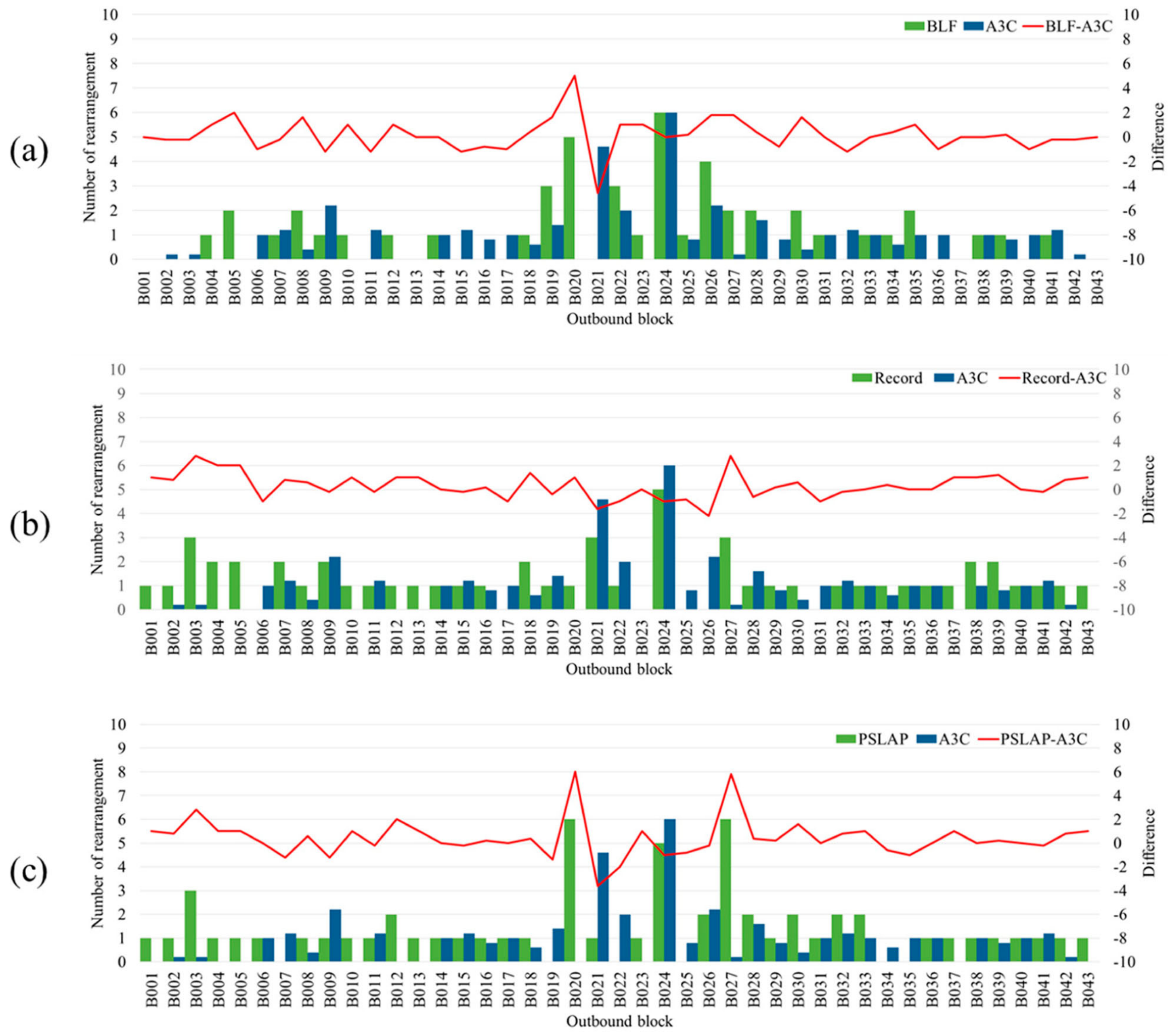

Figure 13. Comparison of rearrangement results obtained using (a) the A3C algorithm and the record data, (b) the A3C and BLF algorithms and (c) the A3C and PSLAP algorithms.

completely random block data, as described in the previous section. Here, when the stockyard occupancy rate was lower than 75%, the blocks were shipped out using the optimal path in almost all cases. In the LA training, the probability distribution obtained from the ship block arrangement records was used to perform training suitable for the ship block arrangement schedule trends of the actual shipyard. An exponential distribution was previously used to model interarrival times when the arrivals were completely random and to model service times that were highly variable (Banks et al. 1984). Figure 9 shows the probability density function corresponding to the records as well as the probability distribution of the interarrival and arrangement period of random blocks. Fifty virtual schedules were created for the block according to this probability density function. These were made into one episode, and training was performed. A new virtual schedule was used in each episode.

Figure 10 shows the average length and average reward results of the 11,000 episodes. The two values converge stably after around 6,000 episodes. The next section describes the use of the agent trained this way to perform predictions about the record data and the evaluation of the results.

Figure 11 shows the LA policy example. The image on the left of each cell shows the current storage state of the blocks. The time period relative to the arrangement period of the block to be stocked is indicated by the colour. The image on the

Table 2. Number of rearrangement for heuristic algorithms and A3C algorithm.

Algorithm		Number of rearrangement	
		5 × 5 cells	5 × 10 cells
Heuristic	BLF	48	132
	PSLAP	47	114
Reinforcement Learning	A3C	41	96

right of each cell shows the probability of a block being located at each address. The dark part means that the address with high probabilities.

4.3. Evaluation of trained LA

The results of arrangement using the trained LA were compared to the results obtained using heuristic algorithms. The actual one-month block schedule data of the actual shipyard as shown in Figure 12 was used for the evaluation. For the heuristic algorithms, the bottom-left-fill (BLF) algorithm and the PSLAP heuristic algorithm (Park and Seo 2010) were used for comparison. In calculating the number of rearrangements, the same pre-trained TA was utilised regardless of the algorithm so that the arrangement locations could be evaluated impartially.

In a similar way, the historical data for blocks stored in the stockyard were compared to the arrangement results obtained when the trained LA was used. Regarding the number of rearrangements, the pre-trained TA was also used in this case to confirm that certain arrangements caused less rearrangement.

Figure 13 compares the numbers of rearrangements for the A3C, BLF and PSLAP algorithms. The LA model determines the arrangement location stochastically according to policy. Therefore, the arrangement locations are not fixed. As such, the average number of rearrangements after five trials was used for the LA. The total number of rearrangements for 43 blocks was 41 when the A3C algorithm was used. This value is 14.6% less than the 48 rearrangements performed by the BLF algorithm, 12.8% less than the 47 rearrangements performed by the PSLAP algorithm and 24.1% less than the 54 rearrangements in the record data.

In order to evaluate the proposed method, we assume a virutal stockyard that is with 5 × 10 cells larger than the target stockyard. At this time, since there is no real block schedule information for 5 × 10 cells, a virtual schedule is created to double the number of blocks in the schedule for the 5 × 5 cells. Table 2 summarises the number of rearrangements required when using the BLF algorithm, PSLAP heuristic algorithm, and A3C algorithm. For both stockyards, the number of rearrangements performed by the A3C algorithm is the smallest. For the 5 × 10 cells, the number of rearrangement performed by A3C algorithm is 15.8% and 27.2% less than the value performed by PSLAP algorithm and the BLF algorithm each.

These results confirm that reinforcement learning is effective for deciding on arrangement locations to minimise rearrangement in a block stockyard. In addition, an agent, which was trained by reinforcement learning, simulated rearrangement and calculated the number of rearrangements, enabling accurate evaluation of the arrangement results.

5. Conclusions

The ship block arrangement problem to minimise transporter usage in a shipyard has been studied by numerous researchers. Metaheuristics approaches have mainly been used before, but these methods have limitations regarding to use at actual sites, due to problems such as long calculation time and difficulties reflecting actual situation. Therefore, the ship block arrangement method using deep reinforcement learning was developed in this study.

Also, the ship block arrangement problem was defined as a two-stage decision process appropriate for actual shipyard operations processes. Two separate agents were defined to make decisions regarding transportation and location. An algorithm based on the A3C approach was employed to train the each agent which is a deep reinforcement learning technology that uses multithreading. It was confirmed that it is possible to perform training effectively for the problems of an actual stockyard in shipyards. The arrangement results from the trained agents were compared to the historical data and the results obtained using a heuristic algorithm. It showed that the trained agents were able to perform the ship block arrangement more effectively.

As reinforcement learning continues to develop and new algorithms are applied, it is expected that it will be possible for the training efficiency to be improved further. However, the actual shipyard operation processes were not fully reflected in this study. It can be developed in future researches. Finally, the problem covered in this research can be applied to various object arrangement problems that consider the spatial resource constraints and production planning schedule. Therefore, it is expected that the method of defining the reinforcement learning problem proposed in this study can be used not only to the arrangement problem of ship block stockyards but also to the various arrangement and allocation problems or logistics problems in the manufacturing industry.

Acknowledgments

The national project (Development of Smart Factory with Artificial Intelligent Forming Machine; No. 10077588), which is supported by the industry core technology development business of the Ministry of Trade, Industry and Energy (Rep. of Korea), supported this research. In addition, 'Development of production strategy for optimising cost of marine ships and execution simulation technology' (No. S1106-16-1020) in the ICT Convergence Industry 4.0S (Naval Architecture & Ocean Engineering) Technology Development Projects supported by the Ministry of Science, ICT and Future Planning (Rep. of Korea) and the national project (development of a simulation-based production management system for middle-sized shipbuilding companies; No. 10050495), which is supported by the Industry Core Technology Development Business of Ministry of Trade, Industry and Energy, supported this research. The Institute of Engineering Research at Seoul National University provided research facilities for this work.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by Ministry of Trade, Industry and Energy: [Grant Number 10050495,10077588]; National IT Industry Promotion Agency: [Grant Number S1106-16-1020].

ORCID

Yongkuk Jeong http://orcid.org/0000-0003-1878-773X

References

Banks, J., J. S. Carson, B. L. Nelson, and D. M. Nicol. 1984. *Discrete-Event System Simulation*. Upper Saddle River, NJ: Prentice Hall.

Bazzazi, M., N. Safaei, and N. Javadian. 2009. "A Genetic Algorithm to Solve the Storage Space Allocation Problem in a Container Terminal." *Computers & Industrial Engineering* 56: 44–52.

Hwang, I., and Y. Jang. 2020. "Q(λ) Learning-Based Dynamic Route Guidance Algorithm for Overhead Hoist Transport Systems in Semiconductor Fabs." *International Journal of Production Research* 58 (4): 1199–1221.

Jeong, Y., S. Ju, H. Shen, D. Lee, J. Shin, and C. Ryu. 2018. "An Analysis of Shipyard Spatial Arrangement Planning Problems and a Spatial Arrangement Algorithm Considering Free Space and Unplaced Block." *International Journal of Advanced Manufacturing Technology* 95: 4307–4325.

Kaelbling, L. P., M. L. Littman, and A. W. Moore. 1996. "Reinforcement Learning: A Survey." *Journal of Artificial Intelligence Research* 4: 237–285.

Littman, M. L. 1994. "Markov Games as a Framework for Multi-Agent Reinforcement Learning." In *Machine Learning Proceedings 1994*, edited by Morgan Kaufmann, 157–163. Amsterdam: Elsevier.

Mnih, V., A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. 2016. "Asynchronous Methods for Deep Reinforcement Learning." *Proceedings of Machine Learning Research* 48: 1928–1937.

Mnih, V., K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. 2013. "Playing Atari with deep reinforcement learning." In Proceedings of NIPS Deep Learning Workshop 2013.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, et al. 2015. "Human-level Control Through Deep Reinforcement Learning." *Nature* 518: 529–533.

Park, C., and J. Seo. 2009. "Assembly Block Storage Location Assignment Problem: Revisited." *Production Planning & Control* 20: 216–226.

Park, C., and J. Seo. 2010. "Comparing Heuristic Algorithms of the Planar Storage Location Assignment Problem." *Transportation Research Part E* 46: 171–185.

Park, C., J. Seo, J. Kim, S. Lee, T.-H. Baek, and S.-K. Min. 2007. "Assembly Block Storage Location Assignment at a Shipyard: A Case of Hyundai Heavy Industries." *Production Planning & Control* 18: 180–189.

Preston, P., and E. Kozan. 2001. "An Approach to Determine Storage Locations of Containers at Seaport Terminals." *Computers & Operations Research* 28: 983–995.

Roh, M., and J. Cha. 2011. "A Block Transportation Scheduling System Considering a Minimisation of Travel Distance Without Loading of and Interference Between Multiple Transporters." *International Journal of Production Research* 49: 3231–3250.

Roh, M., and B. Im. 2011. “Minimization of the Rearrangement of a Block Stockyard Based on the Genetic Algorithm.” *Korean Journal of Computational Design and Engineering* 16 (3): 207–215.

Russel, S., and P. Norvig. 2010. *Artificial Intelligence: A Modern Approach*. 3rd ed. Upper Saddle River, NJ: Prentice Hall.

Shi, D., W. Fan, Y. Xiao, T. Lin, and C. Xing. 2020. “Intelligent Scheduling of Discrete Automated Production Line via Deep Reinforcement Learning.” *International Journal of Production Research*. doi:10.1080/00207543.2020.1717008.

Silver, David, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, et al. 2016. “Mastering the Game of Go with Deep Neural Networks and Tree Search.” *Nature* 529: 484–489.

Silver, D., J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, and A. Bolton. 2017. “Mastering the Game of Go Without Human Knowledge.” *Nature* 550: 354–359.

Sutton, R., and A. Barto. 2012. *A Reinforcement Learning: An Introduction*. 2nd ed. Cambridge, MA: MIT Press.

Tao, N., Z. Jiang, and S. Qu. 2013. “Assembly Block Location and Sequencing for Flat Transporters in a Planar Storage Yard of Shipyards.” *International Journal of Production Research* 51 (14): 4289–4301.

Vinyals, Oriol, Igor Babuschkin, Wojciech M. Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H. Choi, et al. 2019. “Grandmaster Level in StarCraft II Using Multi-Agent Reinforcement Learning.” *Nature* 575 (7782): 350–354.

Wang, J., X. Li, and X. Zhu. 2012. “Intelligent Dynamic Control of Stochastic Economic Lot Scheduling by Agent-Based Reinforcement Learning.” *International Journal of Production Research* 50 (16): 4381–4395.