A Reinforcement Learning with Group-based Candidate-extraction for Container Marshalling at Marine Ports

Yoichi Hirashima

***Abstract*—In container yard terminals, containers brought by trucks in the random order. Containers have to be loaded into the ship in a certain order, since each container has its own shipping destination and it cannot be rearranged after loading. Therefore, containers have to be rearranged from the initial arrangement into the desired arrangement before shipping. In the problem, the number of container-arrangements increases by the exponential rate with increase of total count of containers, and the rearrangement process occupies large part of total run time of material handling operation at the terminal. In this paper, a reinforcement learning method considering the *desired position* of containers for a marshalling in the container yard terminal is proposed. In the proposed method, only the apropriate candidates are extracted for optimization of the rearrangement order of container, so that the learning performance of the method can be improved as compared to the conventional method. In order to show effectiveness of the proposed method, computer simulations for several examples are conducted.**

***Index Terms*—Scheduling, Container Marshalling, Container Transfer Problem, Q-Learning, Block Stacking, Reinforcement Learning**

I. Introduction

A Container terminal at marine port has an important role that shifts containers arriving by inland carriers to ocean vessels. Recently, the number of shipping containers grows rapidly, and operations for container-shifting occupy a large part of the total run time of shipping at container terminals. Since containers are moved by a transfer crane, containers in the yard are stacked in a specific area called bay, and the yard consists several bays. The initial arrangement of containers in a bay is random, since the containers are stacked by the arraiving order. Commonly, materials are packed into containers and each container has its own shipping destination. Containers have to be loaded into a ship in a certain desired order because they cannot be rearranged in the vessel. Thus, containers must be rearranged before loading if the initial layout is different from the desired layout. When the number of containers for shipping is large, the rearrangement operation is complex and takes long time to achieve the desired layout of containers. Therefore the rearrangement process occupies a large part of the total run time of shipping, and generating operations that can reduce cost, run time, and environmental burden of material handling systems is important[1]. The rearrangement process conducted within a bay is called marshalling. In the problem, the number of stacks in each bay is predetermined and the maximum number of containers in a stack is limited. Containers are moved by a transfer crane and the destination stack for the container in a bay is selected from the stacks being in the same bay. In this case, a long series of movements of containers is often required to achieve a desired arrangement, and results that are derived from similar arrangements can be quite different. Problems of this type have been solved by using techniques of optimization, such as genetic algorithm (GA) and multi agent method [3], [4]. These methods can successfully yield some solutions for block stacking problems. However, they adopt the environmental model different from the marshalling process, and do not assure to obtain the desired arrangement of containers.

The reinforcement learning [5], [6] is known as an effective unsupervised learning method. In the reinforcement learning for generating marshalling plan, for all the referred pairs of the layout and movement of containers, evaluation-values are calculated according to the principle of optimality. These values reflect "total cost" required to achieve the objective of the addressed problem and are stored in lookup tables. The input of a lookup table is the yard state and the output is an evaluation value corresponding to the input. A movement is selected with a certain probability that is calculated by using the magnitude of evaluation values. Then, the evaluation value corresponding to the selected movement is updated based on the result of the movement. The optimal pattern of container movements can be obtained by selecting the movement that has the best evaluation value at each state-movement pair. However, the conventional lookup table has to store evaluation-values for all the state-movement pairs. Therefore, the conventional reinforcement learning method, has great difficulties for solving the marshalling problem, due to its huge number of learning iterations and states required to obtain admissible operation of containers [7]. Recently, a Q-learning method that can generate marshalling plan has been proposed [8], [12]. Although these methods were effective for several cases, many constraints are required to assure the achievement of desired layout for every marshalling, so that the early-phase performances of learning process as well as marshalling plans can be degraded.

In this paper, a new reinforcement learning system to generate a marshalling plan is proposed. The learning process in the proposed method is consisted of two stages: ① determination of rearrangement order, ② selection of destination for removal containers. Learning algorithms in these stages are independent to each other and evaluation values in one stage are referred from the other stage. That is, evaluation values are discounted according to the transfer distance of

Faculty of Information Science and Technology, Osaka Institute of Technology, 1-79-1, Kita-yama, Hirakata City, Osaka, 573-0196, Japan. Tel/Fax: +86-72-866-5187 Email: yoichi.hirashima@oit.ac.jp

ISBN: 978-988-14047-8-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

containers and lockup table for rearrangement is constructed by using evaluation values for movements of container, so that evaluation values reflect the total transfer distance of containers to obtain a desired arrangement. Moreover, in the end of stage ①, selected container is rearranged into the desired position so that every trial can achieve the desired layout. In addition, in the proposed method, each container has several desired positions in the final arrangement, and the feature is considered in the learning algorithm. Thus, the early-phase performances of the learning process can be improved. Finally, effectiveness of the proposed method is shown by computer simulations.

II. PROBLEM DESCRIPTION

Fig.1 shows an example of container yard terminal. The terminal consists of containers, yard areas, yard transfer cranes, auto-guided vehicles, and port crane. Containers are carried by trucks and each container is stacked in a corresponding area called bay and a set of bays constitutes a yard area. Each bay has n_{y} stacks that m_{y} containers can be laden, the number of containers in a bay is k, and the number of bays depends on the number of containers. Each container is recognized by an unique name $\mathrm{c}_i\,(i = 1, \cdots, k)$. A position of each container is discriminated by using discrete position numbers, $1, \cdots, n_{\mathrm{y}} \cdot m_{\mathrm{y}}$. Then, the position of the container c_i is described by $x_i\,(1 \leq i \leq k, 1 \leq x_i \leq m_{\mathrm{y}} \cdot n_{\mathrm{y}})$, and the state of a bay is determined by the vector $\boldsymbol{x} = [x_1, \cdots, x_k]$.

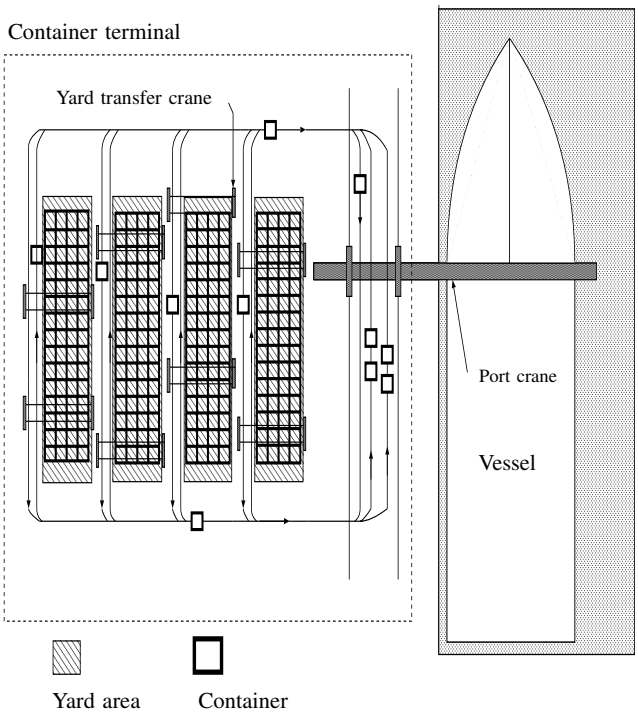

Fig. 1. Container terminal

A. Grouping

The desired layout in a bay is generated based on the loading order of containers that are moved from the bay to a vessel. In this case, the container to be loaded into the vessel can be anywhere in the bay if it is on top of a stack. This feature yields several desired layouts for the bay.

B. Horizontal group (Original group)

In the addressed problem, when containers on different stacks are placed at the same height in the bay, it is assumed that the positions of such containers can be exchanged. Fig.2 shows an example of desired layouts, where $m_{\mathrm{y}} = n_{\mathrm{y}} = 3, k = 9$. In the figure, containers are loaded in the ship in the descendent order. Then, containers $\mathrm{c}_7, \mathrm{c}_8, \mathrm{c}_9$ are in the same group (group_1), and their positions are exchanged because the loading order can be kept unchanged after the exchange of positions. In the same way, $\mathrm{c}_4, \mathrm{c}_5, \mathrm{c}_6$ are in the group_2, and $\mathrm{c}_1, \mathrm{c}_2, \mathrm{c}_3$ are in the group_3 where positions of containers can be exchanged. Consequently several candidates for desired layout of the bay are generated from the original desired-layout.

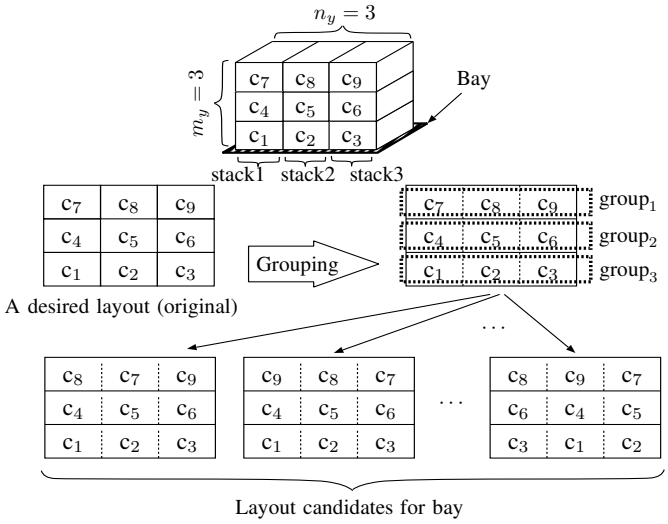

Fig. 2. Layouts for bay

1) Heap shaped group: A "heap shaped group" for n_{y} containers at the top of stacks in original the desired-layout (group_1) is generated as follows:

1) Each of them can be stacked on other $n_{\mathrm{y}}-1$ containers when both of followings are satisfied:
   a) They are placed at the top of each stack in the original disired-layout,
   b) The container to be stacked is loaded into the ship before other containers being under the container.

Other groups are the same as ones in the original grouping, so that the grouping with heap contains all the desired layout in the original grouping.

Consequently, several candidates for desired arrangement of containers are generated. In the proposed method, in order to avoid rearrangements without candidate positions, following strategy for extracting candidates of goal positions is :

1) The priority of each stack in the buffer is determined by the loading order of the container on top of the stack. That is, the earlier loading has higher priority
2) For each stack in the buffer, make a list of containers that can be rearranged (Initially, all the list has the same members.)
3) Exclude containers that must be loaded before the top container of processing stack

ISBN: 978-988-14047-8-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

4) Fill stacks of higher priorities virtually by using containers that must be loaded earlier. Defining the number of containers that can be placed on the stack as a, from the list, discard $a-1$ containers that must be loaded earlier
5) Candidates for rearrangement are containers remained in the list

Fig.3 depicts an example of heap grouping for $k = 9, n_{\mathrm{y}} = 3$. In the figure, containers are loaded into a vessel by the order $\mathrm{c}_9, \mathrm{c}_8, \mathrm{c}_7, \cdots$. Then, c_9 can be placed on c_7 and c_8, c_8 can be placed on c_7, so that the number of desired layouts is incresed.

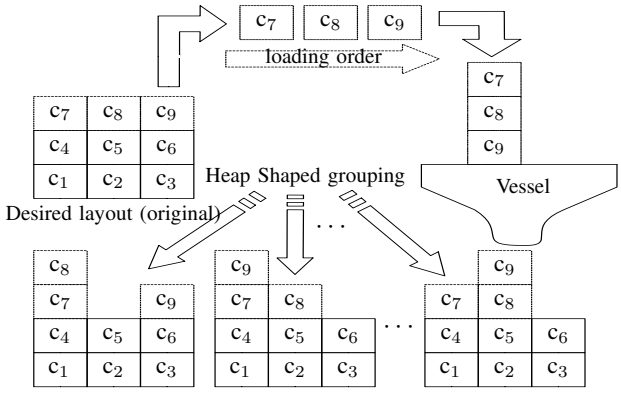

Fig. 3. Heap shaped group

C. Marshalling process

The marshalling process consists of 2 stages: ① selection of a container to be rearranged, and ② removal of the containers on the selected container in ①. After these stages, rearrangement of the selected container is conducted. In the stage ②, the removed container is placed on the destination stack selected from stacks being in the same bay. When a container is rearranged, n_{y} positions that are at the same height in a bay can be candidates for the destination. In addition, n_{y} containers can be placed for each candidate of the destination. Then, defining t as the time step, $c_{\mathrm{a}}(t)$ denotes the container to be rearranged at t in the stage ①. $c_{\mathrm{a}}(t)$ is selected from candidates $\mathrm{c}_{y_{i_1}}$ $(i_1 = 1, \cdots, n_{\mathrm{y}}^2)$ that are at the same height in a desired layout. A candidate of destination exists at a bottom position that has undesired container in each corresponding stack. The maximum number of such stacks is n_{y}, and they can have n_{y} containers as candidates, since the proposed method considers groups in the desired position. The number of candidates of $c_{\mathrm{a}}(t)$ is thus $n_{\mathrm{y}} \times n_{\mathrm{y}}$. In the stage ②, the container to be removed at t is $c_{\mathrm{b}}(t)$ and is selected from two containers $\mathrm{c}_{y_{i_2}}$ $(i_2 = 1, 2)$ on the top of stacks. c_{y_1} is on the $c_{\mathrm{a}}(t)$ and c_{y_2} is on the destination of $c_{\mathrm{a}}(t)$. Then, in the stage ②, $c_{\mathrm{b}}(t)$ is removed to one of the other stacks in the same bay, and the destination stack $u(t)$ at time t is selected from the candidates u_j $(j = 1, \cdots, n_{\mathrm{y}}-2)$. $c_{\mathrm{a}}(t)$ is rearranged to its desired position after all the $\mathrm{c}_{y_{i_2}}$s are removed. Thus, a state transition of the bay is described as follows:

$$\boldsymbol{x}_{t+1} = \begin{cases} f(\boldsymbol{x}_t, c_{\mathrm{a}}(t)) & (\text{stage ①}) \\ f(\boldsymbol{x}_t, c_{\mathrm{b}}(t), u(t)) & (\text{stage ②}) \end{cases} \quad (1)$$

where $f(\cdot)$ denotes that removal is processed and $\boldsymbol{x}_{t+1}$ is the state determined only by $c_{\mathrm{a}}(t), c_{\mathrm{b}}(t)$ and $u(t)$ at the previous state $\boldsymbol{x}_t$. Therefore, the marshalling plan can be treated as the Markov Decision Process.

Additional assumptions are listed below:

1) The bay is 2-dimensional.
2) Each container has the same size.
3) The goal position of the target container must be located where all containers under the target container are placed at their own goal positions.
4) $k \leq m_{\mathrm{y}} n_{\mathrm{y}} - 2m_{\mathrm{y}} + 1$

The maximum number of containers that must removed before rearrangement of $c_{\mathrm{a}}(t)$ is $2m_{\mathrm{y}} - 1$ because the height of each stack is limited to m_{y}. Thus, assumption (4) assures the existence of space for removing all the $c_{\mathrm{b}}(t)$, and $c_{\mathrm{a}}(t)$ can be placed at the desired position from any state $\boldsymbol{x}_t$.

Figure 4 shows 3 examples of marshalling process, where $m_{\mathrm{y}} = 3, n_{\mathrm{y}} = 5, k = 8$. Positions of containers are discriminated by integers $1, \cdots, 15$. The first container to be loaded is c_8 and containers must be loaded by descendent order until c_1 is loaded. In the figure, a container marked with a □ denotes c_1, a container marked with a ◯ is removed one, and an arrowed line links source and destination positions of removed container. Cases (a),(b) have the same order of rearrangement, $\mathrm{c}_2, \mathrm{c}_7, \mathrm{c}_6$, and the removal destinations are different. Whereas, case (c) has the different order of rearrangement, $\mathrm{c}_8, \mathrm{c}_2, \mathrm{c}_7$. When no groups are considered in desired arrangement, case (b) requires 5 steps to complete the marshalling process, and other cases require one more step. Thus, the total number of movements of container can be changed by the destination of the container to be removed as well as the rearrangement order of containers.

If heap shaped grouping is considered in desired arrangement, cases (a), (b) achieve a goal arrangement at step2, and case (c) achives at step4, so that equivalent or better marshalling plans can be generated by using the grouping as compared to plans generated without grouping.

The objective of the problem is to find the best series of movements which transfers every container from an initial position to the goal position. The goal state is generated from the shipping order that is predetermined according to destinations of containers. A series of movements that leads a initial state into the goal state is defined as an episode. The best episode is the series of movements having the smallest number of movements of containers to achieve the goal state.

III. REINFORCEMENT LEARNING FOR MARSHALLING PLAN

A. Update rule of Q-values

In the selection of c_{a}, the container to be rearranged, an evaluation value is used for each candidate $\mathrm{c}_{y_{i_1}}$ $(i_1 = 1, \cdots, r$, where r is the number of candidates. In the same way, evaluation values are used in the selection of the container to be removed c_{b} and its destination u_j $(j = 1, \cdots, n_{\mathrm{y}} - 2)$. Candidates of c_{b} is $\mathrm{c}_{y_{i_2}}$ $(i_2 = 1, \cdots, n_{\mathrm{y}})$. The evaluation value for the selection of $\mathrm{c}_{y_{i_1}}$, $\mathrm{c}_{y_{i_2}}$ and u_j at the state $\boldsymbol{x}$ are called Q-values, and a set of Q-values is called Q-table. At the *l*th episode, the Q-value for selecting $\mathrm{c}_{y_{i_1}}$ is defined as $Q_1(l, \boldsymbol{x}, \mathrm{c}_{y_{i_1}})$, the Q-value for selecting $\mathrm{c}_{y_{i_2}}$ is defined as $Q_2(l, \boldsymbol{x}, \mathrm{c}_{y_{i_1}}, \mathrm{c}_{y_{i_2}})$ and the Q-value for selecting u_j is

ISBN: 978-988-14047-8-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

defined as $Q_3(l, \boldsymbol{x}, \mathrm{c}_{y_{i_1}}, \mathrm{c}_{y_{i_2}}, u_j)$. The initial value for both Q_1, Q_2, Q_3 is assumed to be 0.

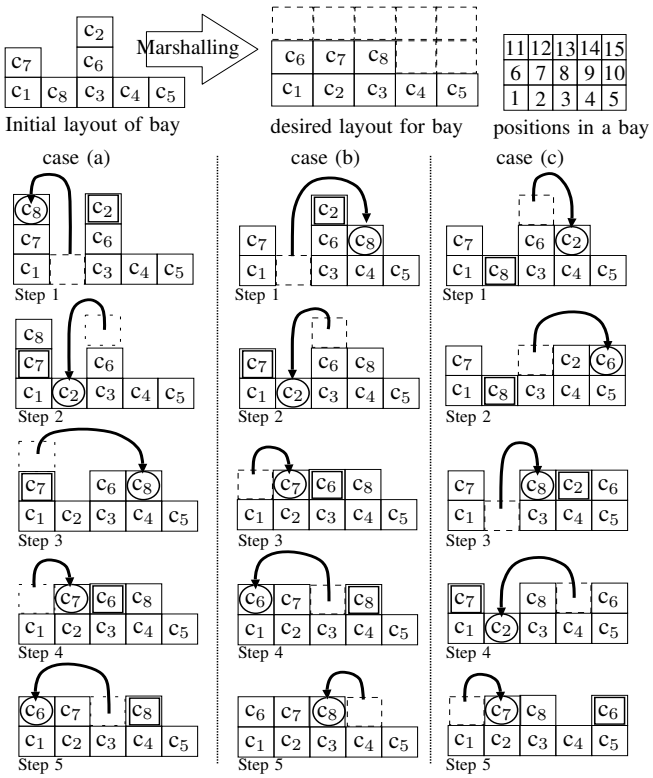

Fig. 4. Marshalling process

In this method, a large amount of memory space is required to store all the Q-values referred in every episode. In order to reduce the required memory size, the length of episode that corresponding Q-values are stored should be limited, since long episode often includes ineffective movements of container. In the following, update rule of Q_3 is described. When a series of n movements of container achieves the goal state $\boldsymbol{x}_n$ from an initial state $\boldsymbol{x}_0$, all the referred Q-values from $\boldsymbol{x}_0$ to $\boldsymbol{x}_n$ are updated. Then, defining L as the total counts of container-movements for the corresponding episode, L_{min} as the smallest value of L found in the past episodes, and s as the parameter determining the threshold, Q_3 is updated when $L < L_{min} + s\,(s > 0)$ is satisfied by the following equation:

$$\begin{aligned} &Q_3(l, \boldsymbol{x}_t, c_{\mathrm{a}}(t), c_{\mathrm{b}}(t), u(t)) = \\ &\quad (1-\alpha) Q_3(l-1, \boldsymbol{x}_t, c_{\mathrm{a}}(t), c_{\mathrm{b}}(t), u(t)) \\ &\quad +\alpha[\mathrm{R} + V_{t+1}] \\ &V_t = \begin{cases} \gamma \max_{y_{i_1}} Q_1(l, \boldsymbol{x}_t, \mathrm{c}_{y_{i_1}}) & (\text{stage } ①) \\ \gamma \max_{y_{i_2}} Q_2(l, \boldsymbol{x}_t, c_{\mathrm{a}}(t), \mathrm{c}_{y_{i_2}}) & (\text{stage } ②) \end{cases} \end{aligned} \tag{2}$$

where γ denotes the discount factor and α is the learning rate. Reward R is given only when the desired layout has been achieved. L_{min} is assumed to be infinity at the initial state, and updated when $L < L_{min}$ by the following equation: $L = L_{min}$.

In the selection of $c_{\mathrm{b}}(t)$, the evaluation value $Q_3(l, \boldsymbol{x}, c_{\mathrm{a}}(t), c_{\mathrm{b}}(t), u_j)$ can be referred for all the $u_j\,(j = 1 \cdots n_{\mathrm{y}} - 2)$, and the state $\boldsymbol{x}$ does not change. Thus, the maximum value of $Q_3(l, \boldsymbol{x}, c_{\mathrm{a}}(t), c_{\mathrm{b}}(t), u_j)$ is copied to $Q_1(l, \boldsymbol{x}, c(t))$, that is,

$$\begin{aligned} &Q_2(l, \boldsymbol{x}, c_{\mathrm{a}}(t), c_{\mathrm{b}}(t)) = \\ &\quad \max_j Q_3(l, \boldsymbol{x}, c_{\mathrm{a}}(t), c_{\mathrm{b}}(t), u_j). \end{aligned} \tag{3}$$

In the selection of $c_{\mathrm{a}}(t)$, the evaluation value $Q_1(l, \boldsymbol{x}, c_{\mathrm{a}}(t))$ is updated by the following equations:

$$\begin{aligned} &Q_1(l, \boldsymbol{x}_t, c_{\mathrm{a}}(t)) = \\ &\begin{cases} \max_{y_{i_1}} Q_1(l, \boldsymbol{x}_t, \mathrm{c}_{y_{i_1}}) + \mathrm{R} & (\text{stage } ①) \\ \max_{y_{i_2}} Q_2(l, \boldsymbol{x}_t, c_{\mathrm{a}}(t), \mathrm{c}_{y_{i_2}}) & (\text{stage } ②) \end{cases} \end{aligned} \tag{4}$$

In order to select actions, the "ϵ-greedy" method is used. In the "ϵ-greedy" method, $c_{\mathrm{a}}(t), c_{\mathrm{b}}(t)$ and a movement that have the largest $Q_1(l, \boldsymbol{x}, c_{\mathrm{a}}(t))$, $Q_2(l, \boldsymbol{x}, c_{\mathrm{a}}(t), c_{\mathrm{b}}(t))$ and $Q_3(l, \boldsymbol{x}, c_{\mathrm{a}}(t), c_{\mathrm{b}}(t), u_j)$ are selected with probability $1 - \epsilon (0 < \epsilon < 1)$, and with probability ϵ, a container and a movement are selected randomly.

B. Calculation of discount factor

In order to reflect the total transfer distance of containers to each evaluation value, the discount factor γ is calculated by the following equation:

$$\gamma = \delta \frac{D_{\max} - \beta D}{D_{\max}}, 0 < \gamma < 1, 0 < \beta < 1 \tag{5}$$

where D is the distance between positions that the corresponding container moves, $D_{\max}$ is the maxmum value of D, δ and β are parameters that determine the range of γ.

C. Learning algorithm

By using the update rule, restricted movements and goal states explained above, the learning process is described as follows:

[1]. Count the number of containers being in the goal positions and store it as n
[2]. If $n = k$, go to [10]
[3]. Select $c_{\mathrm{a}}(t)$ to be rearranged
[4]. Store $(\boldsymbol{x}, c_{\mathrm{a}}(t))$
[5]. Select $c_{\mathrm{b}}(t)$ to be removed
[6]. Store $(\boldsymbol{x}, c_{\mathrm{a}}(t), c_{\mathrm{b}}(t))$
[7]. Select destination position u_j for $c_{\mathrm{b}}(t)$
[8]. Store $(\boldsymbol{x}, c_{\mathrm{a}}(t), c_{\mathrm{b}}(t), u_j)$
[9]. Remove $c_{\mathrm{b}}(t)$ and go to [5] if another $c_{\mathrm{b}}(t)$ exists, otherwise go to [1]
[10]. Update all the Q-values referred from the initial state to the goal state according to eqs. (2), (3)

A flow chart of the learning algorithm is depicted in Figure 5.

IV. SIMULATIONS

Computer simulations are conducted for the following 2 methods, and learning performances are compared:

(A) proposed method using heap shaped grouping for every containers,
(B) proposed method using heap shaped grouping only for the horizontal group located on top of stacks in the original disired arrangement [12].

In methods (A)(B), parameters in the yard are set as $k = 18, m_{\mathrm{y}} = n_{\mathrm{y}} = 6$ that are typical values of marshalling environment in real container terminals. Containers are assumed to be loaded in a vessel in ascending order from c_1

ISBN: 978-988-14047-8-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

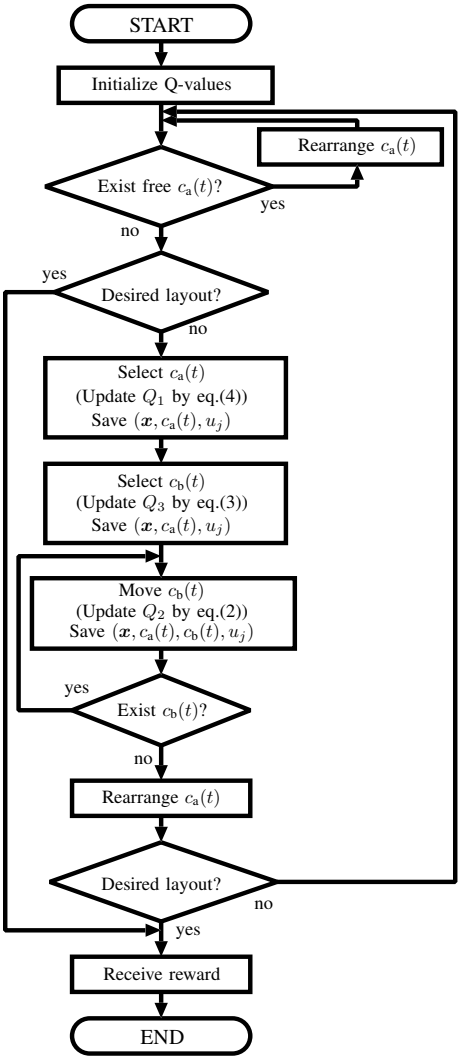

Fig. 5. Flowchart of the learning algorithm

to c_{18}. Figure 7 shows the original desired layout, and figure 8 shows the initial layout. Other parameters are put as $\alpha = 0.8, \beta = 0.9, \gamma = 0.8, \delta = 0.9, R = 1.0, \epsilon = 0.8, s = 15$. For simplicity, the transfer distance between adjacent containers is defined by 1. Figure 6 shows the definision of the transfer distance of containers and position index for the state vector $\boldsymbol{x}$.

Results are shown in Fig.9. In the figure, horizontal axis shows the number of trials, and vertical axis shows the minimum transfer distance of containers found in the finished trials. Each result is averaged over 20 independent simulations. A final arrangement of container and a solution obtained by method (A) is shown in Figs.11,12. Also, a fineall arrangement of container obatined by method (B) is shown in Fig.10. In Fig.9, the learning performance of method (A) is better than that of method (B) since the candidate of disired ararngement in method (A) is extended from the one in method (B). In Fig.11, the desired position of c_1 is located at bottom left by method (A), which is avoided to decrease "dead space" in the arrangement by method (B) dipicted in Fig.10. In method (B), the heap shaped group is generated only with $c_1 - c_6$, the members in group$_1$ of the original desired arrangement, and is located above the group$_2$ of the original desired arrangement[12]. Moreover, at 10000th trail the number of movements of containers in method (A) is smaller as compared to that in method (B) because, among the extended layouts, method (A) obtained better desired layouts for improving the marshalling process as compared to the layout generated by method (B). In addition, at 1000th trial, all the simulations in method (A) found the solution that has the same value for the total transfer distanse of containers. Whereas, simulations in method (B) requires 20100 trials in order to find the same best solutions, as shown in table.I. A solution generated by mehtod (A) is depicted in Fig.12. The sequence of movements for containers consists of 2 removals and 5 direct rearrangements, which achieves a desired arrangement by 7 steps. The total transfer distance of containers is 32 containing transitions that can be improved. This means, as well as the arrangement of containers in desired layout, removal order and removal destination are important to reduce the total transfer distance of containers.

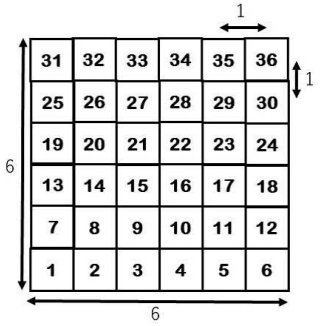

Fig. 6. Position index and transfer distance

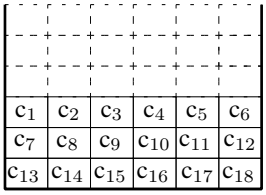

Fig. 7. A desired layout

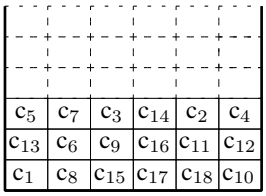

Fig. 8. Initial layout

V. CONCLUSIONS

A new reinforcement learning system for marshalling plan at container terminals has been proposed. Each container has several desired positions that are in the heap shaped groups, and the learning algorithm is designed to considering the feature.

ISBN: 978-988-14047-8-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

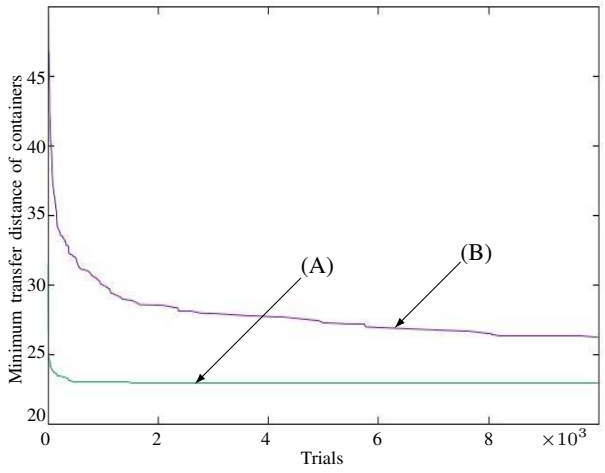

Fig. 9. Performance comparison

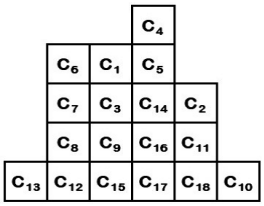

Fig. 10. A final arrangement of a solution obtained by method (B)

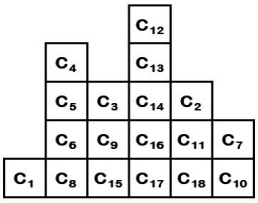

Fig. 11. A final arrangement of a solution obtained by method (A)

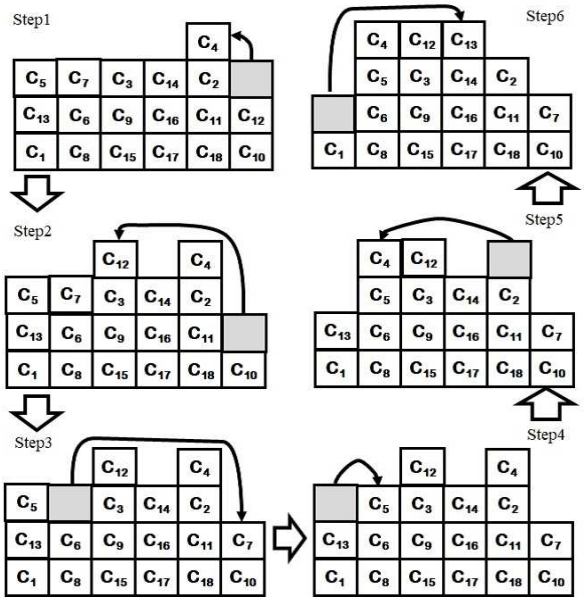

Fig. 12. Marshalling plan generated by method (A)

TABLE I
THE BEST SOLUTION OF EACH METHOD

Method	Trial	best	ave.	worst
(A)	1000	23.0	23.05	24.0
↑	10000	23.0	23.00	23.0
(B)	1000	26.0	30.10	36.0
↑	10000	26.0	26.35	30.0
↑	20100	26.0	26.00	26.0

In simulations, the proposed method could find solutions that had smaller number of movements of containers as compared to conventional methods. Moreover, since the proposed method assures the achievement of the desired layout in each trial with appropriate candidates in each selection stage as well as learns the desirable layout for heap shaped grouping, the method can generate better solutions with the smaller number of trials as compared to the conventional method. Since all the arrangements of group layouts in the conventional method[12] are contained as a part of candidates in the proposed method, the improvement for solution is assured in the proposed mehotd. In addition, the arrangement of containers in the desired laytout, the rearrange order of containers, and the position of each removal container has been obtained simultaneously so that the learning performance of the proposed method has been improved.

REFERENCES

[1] Siberholz, M. B., Golden, B. L., Baker, K., "Using Simulation to Study the Impact of Work Rules on Productivity at Marine Container Terminals", *Computers Oper. Res.*, V.18, N.5, pp433–452, 1991.
[2] Günther, H.-O., Kim, K. H., Container Terminals and Automated Transport Systems, Springer, pp184–206, 2005.
[3] Koza, J. R., Genetic Programming : On Programming Computers by means of Natural Selection and Genetics, MIT Press, 1992.
[4] Minagawa, M., Kakazu, Y., "An Approach to the Block Stacking Problem by Multi Agent Cooperation", *Trans. Jpn. Soc. Mech. Eng. (in Japanese)*, V.C-63, N.608, pp231–240, 1997.
[5] Watkins, C. J. C. H., Dayan, P., "Q-learning", *Machine Learning*, V.8, pp279–292, 1992.
[6] Sutton, R., Barto, A., "Reinforcement Learning", MIT Press (1999).
[7] Baum, E. B., "Toward a Model of Intelligence as an Economy of Agents", *Machine Learning*, V35, pp155–185, 1999.
[8] Hirashima, Y., Iiguni, Y., Inoue, A., Masuda, S., "Q-Learning Algorithm Using an Adaptive-Sized Q-table", *Proc. IEEE Conf. Decision and Control*, pp1599-1604, 1999.
[9] Hirashima, Y., Takeda, K., Furuya, O., Inoue, A., Deng, M., "A New Method for Marshaling Plan Using a Reinforcement Learning Considering Desired Layout of Containers in Terminals", *Preprint of 16th IFAC World Congress*, We-E16-TO/2, 2005.
[10] Hirashima, Y., Ishikawa, N., Takeda, K., "A New Reinforcement Learning for Group-Based Marshaling Plan Considering Desired Layout of Containers in Port Terminals", *International Conference on Networking, Sensing, and Control*, pp670-675, 2006.
[11] Motoyama, S., Hirashima, Y., Takeda, K., Inoue A., "A marshalling plan for container terminals based on reinforce-ment learning", *Proc. of Inter. Sympo. on Advanced Control of Industrial Processes*, pp631–636, 2001.
[12] Hirashima, Y., "A Q-learning system for container marshalling with group-based learning model at container yard terminals," *Lecture Notes in Engineering and Computer Science: Proceedings of the International MultiConference of Engineers and Computer Scientist 2009, Hong Kong*, pp18–20.

ISBN: 978-988-14047-8-7
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)