A BRANCH-AND-BOUND ALGORITHM FOR THE BLOCK RELOCATION PROBLEM TO MINIMIZE TOTAL CRANE OPERATION TIME

Y. Inaoka[(a)], S. Tanaka[(b)]

[(a),(b)] Kyoto University

[(a)]inaoka@rotary.kyoto-u.ac.jp, [(b)] tanaka@kuee.kyoto-u.ac.jp

ABSTRACT

Containers transferred to a sea port are stacked and stored in container yards of a container terminal. In order to retrieve a container on which other containers are stacked by a crane, these interfering containers should be relocated to some places beforehand, where they may cause further relocations. The aim of the block (container) relocation problem is to retrieve all blocks in a specified order with such unproductive relocation being minimized. Most of existing studies on this problem aim to minimize the total number of relocations. However, it is more desirable from a practical point of view to consider actual crane operation time. In this research, we propose a branch-and-bound algorithm for the block relocation problem to minimize total crane operation time. Its effectiveness is examined by computational experiments.

Keywords: container terminal, block relocation problem, branch-and-bound algorithm, total crane operation time

1. INTRODUCTION

Container transport plays an important role in the global logistics system. Containers transferred to a sea port by vessels or trucks are stored temporarily in a container terminal. Due to limitation of space, they are in general piled up in container yards as Figure 1. Containers are then transferred to their next destinations from there. Since this order is determined by their departure time, destinations, weight, contents and so on, it in general does not coincide with the stacked order. Therefore, relocation or reshuffling inevitably occurs to retrieve a container stacked in a lower tier by a crane. Such relocated containers may interfere another container if they are stacked on it, meaning that a careful and intelligent decision of relocations can improve the throughput of container handling in a container terminal. For the purpose of reducing unproductive relocations, the container relocation problem, which is also known as the block relocation problem, has been studied in the literature. Its objective is to retrieve stacked containers (blocks) in a specified order with the least effort. For the

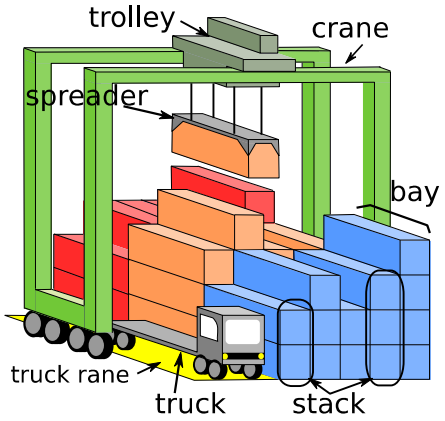


Figure 1: Container Yard

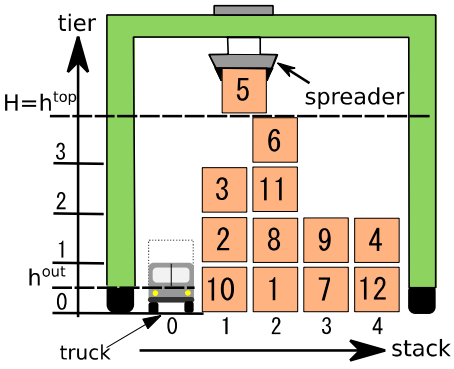


Figure 2: Blocks in the Target Bay

sake of generality, this problem is referred to as the block relocation problem (BRP), and a container as a block accordingly. Most of previous studies on this problem aim to minimize the total number of relocations. To the best of the authors' knowledge, studies that try to minimize crane operation time are limited in spite of its practical importance. Lee and Lee (2010) proposed a heuristic algorithm for the BRP where horizontal travel time of a block is considered. Ünlüyurt and Aydın (2012) also treated the BRP with horizontal travel time, and proposed a branch-and-bound algorithm as well as heuristic ones. Lin et al. (2015) further considered

vertical travel time and constructed a heuristic algorithm. They also treated a crane capable of handling multiple blocks at a time. Recently, Schwarze and Voß (2016) solved the problem with horizontal and vertical travel time using the ILP (integer linear programming) formulation by Zehendner et al. (2015), which was originally proposed to minimize the total number of relocations. However, the size of instances that can be solved to optimality is still restrictive. The purpose of this study is to construct an efficient branch-and-bound algorithm for the BRP to minimize total crane operation time where not only horizontal travel time but also vertical travel time is taken into account. For this purpose, we will propose dominance properties for suppression of unnecessary nodes and two types of lower bound of the objective value for bounding. The effectiveness of the algorithm will be demonstrated by numerical experiments.

2. PROBLEM DESCRIPTION

In this section, we will provide an explicit description of the BRP considered in this study. Suppose that blocks (containers) are stored in a container yard as illustrated in Figure 1. A single row of blocks is called a bay, and a bay is composed of stacks where blocks are piled up vertically. A gantry crane is used to move blocks. Since the travel time of the crane across bays is time-consuming, we concentrate on a single bay and consider retrieving all blocks therein (Figure. 2).

The stacks in the bay are numbered as stack 1, stack 2, stack S. N blocks of the same size are stored in the bay, which are numbered as block 1, block 2, …, block N. Due to the height of the crane, the maximum number of blocks in each stack is limited to H. The slot in the h-th tier of stack s is denoted by (s, h), where the ground level is $h = 0$. We are to retrieve all the blocks from the bay in the ascending order of block numbers. To do this, the blocks should be moved onto the bed of a truck at $(0, h^{\mathrm{out}})$ one by one. The crane can access only topmost blocks, so that the crane performs the following two operations.

- Relocation: A block on the top of a stack is moved to the top of another stack that does not reach the height limit.
- Retrieval: The block with the smallest number (target block) is moved to the bed of the truck if it is on the top of a stack.

Our objective is to minimize the total crane operation time. Here, we make the following assumption.

Only blocks above the target block can be relocated.

The BRP with this assumption is often referred to as the restricted BRP in the literature.

When the crane relocates a block, it moves the spreader horizontally to the stack where the block is placed. Next, it winds the spreader down onto the block, and grasps the block. Then the crane winds them up together, moves them horizontally to the destination stack, winds them down, and releases the block. We assume that the spreader can move horizontally only along the h^{top}–th tier. In addition, the initial position of the spreader is assumed to be $(0, h^{\mathrm{out}})$. The detailed crane operation time is summarized in Table 1. It follows from Table 1 that when the initial position of the spreader is $(0, h^{\mathrm{out}})$, the time necessary for retrieving a block from (s, h) is given by

Table 1: Crane Operation Time

t_{sg}[s]	The block grasp time
t_{sr}[s]	The block release time
t_{t} [s/stack]	The trolley speed for horizontal move
t_{hu} [s/tier]	The hoisting speed (unloaded)
t_{hl}[s/tier]	The hoisting speed (loaded)

$$t_{\mathrm{s}} + 2t_{\mathrm{t}}s + t_{\mathrm{h}}(2h^{\mathrm{top}} - h - h^{\mathrm{out}}), \quad (1)$$

where $t_{\mathrm{s}} = t_{\mathrm{sr}} + t_{\mathrm{sg}}$ and $t_{\mathrm{h}} = t_{\mathrm{hu}} + t_{\mathrm{hl}}$. If, before retrieving the block from (s, h), n blocks on it are relocated from $(s, h + k)$ to $(s_k^{\mathrm{d}}, h_k^{\mathrm{d}})$, $(k = 1, 2, \dots, n)$, respectively, the total crane operation time becomes:

$$\begin{aligned} t_{\mathrm{s}} + 2t_{\mathrm{t}}s + t_{\mathrm{h}}(2h^{\mathrm{top}} - h - h^{\mathrm{out}}) \\ + \sum_{k=1}^{n} \{t_{\mathrm{s}} + 2t_{\mathrm{t}}|s_k^{\mathrm{d}} - s| \\ + t_{\mathrm{h}}(2h^{\mathrm{top}} - h_k^{\mathrm{d}} - h - k)\}. \end{aligned} \quad (2)$$

Suppose that block i $(i = 1, 2, \dots, N)$ is retrieved from (s_i, h_i), which causes M relocations from $(s_k^{\mathrm{s}}, h_k^{\mathrm{s}})$ to $(s_k^{\mathrm{d}}, h_k^{\mathrm{d}})$ $(k = 1, 2, \dots, M)$. Noting that every block is relocated from the stack where the current target block is placed as in (2), we can see that the total crane operation time is given by

$$\begin{aligned} \sum_{k=1}^{M} \{t_{\mathrm{s}} + 2t_{\mathrm{t}}|s_k^{\mathrm{d}} - s_k^{\mathrm{s}}| + t_{\mathrm{h}}(2h^{\mathrm{top}} - h_k^{\mathrm{s}} - h_k^{\mathrm{d}})\} \\ + \sum_{i=1}^{N} \{t_{\mathrm{s}} + 2t_{\mathrm{t}}s_i + t_{\mathrm{h}}(2h^{\mathrm{top}} - h_i - h^{\mathrm{out}})\}. \end{aligned} \quad (3)$$

This equation provides the objective function of the BRP that should be minimized. If we ignore the horizontal and vertical travel times by setting $t_{\mathrm{t}} = t_{\mathrm{h}} = 0$, (3) reduces to $t_{\mathrm{s}}(M + N)$, so that the problem becomes equivalent to the problem of minimizing M, the total number of relocations.

3. BRANCH-AND-BOUND ALGORITHM

To solve the BRP explained in the preceding section to optimality, we apply a branch-and-bound algorithm. Since a solution of the BRP can be expressed by a sequence of relocations by assuming that blocks are retrieved as soon as they become retrievable, the

algorithm searches for an optimal sequence of relocations. Subproblems are generated by fixing the sequence from the start one by one. Thus, a node at depth k in the search tree represents a bay configuration obtained by applying k relocations (and retrieving as many blocks as possible). The search tree is traversed in the depth-first manner. The initial solution is calculated by a constructive heuristic PR4 by Zhu et al. (2012) for simplicity, although it aims to minimize the total number of relocations.

4. DOMINANCE OF COLUMNS

We will derive two dominance properties, which are employed to suppress generation of unnecessary nodes in the search tree of the branch-and-bound algorithm. A (partial) sequence of relocations is said to dominate another sequence of relocations if the former yields at least as good a solution as the latter. Here, we show that the same bay configuration is obtained by two sequences of relocations under some conditions. Then, the one with a longer crane operation time is dominated by the other, so that the former can be forbidden in the search tree.

Let us denote by C_0 the bay configuration obtained by retrieving as many blocks as possible from the initial configuration without any relocation. Let us also denote by a triplet $(i, s^{\mathrm{s}}, s^{\mathrm{d}})$ the relocation of block i from stack s^{s} to stack s^{d}. In the following, we prove two theorems that provide conditions for a sequence of relocations $(i_1, s_1^{\mathrm{s}}, s_1^{\mathrm{d}}), (i_2, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}), \dots, (i_n, s_n^{\mathrm{s}}, s_n^{\mathrm{d}})$ to be dominated by another sequence when applied to C_0. Throughout this section, the bay configuration obtained by applying $(i_1, s_1^{\mathrm{s}}, s_1^{\mathrm{d}}), \dots, (i_k, s_k^{\mathrm{s}}, s_k^{\mathrm{d}})$ to C_0 (and retrieving all retrievable blocks) is denoted by C_k. Furthermore, the number of blocks and the smallest block number in stack s of a bay configuration C are denoted by $N_s(C)$ and $Q_s(C)$, respectively. The stack where the target block is placed is referred to as the target stack: it is given by $\underset{1 \le s \le S}{\operatorname{argmin}} Q_s(C)$.

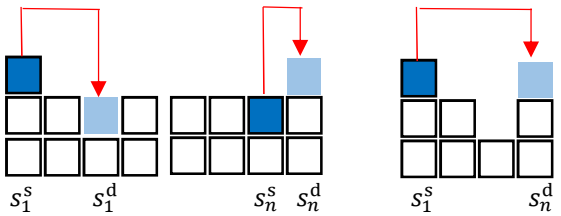
4.1. Transitivity of two relocations

The first dominance property concerns transitivity of two relocations. If some block is relocated from stack s_1 to stack s_2 and then stack s_2 to stack s_3, these two relocations can be combined into one relocation from stack s_1 to stack s_3 without increasing the total crane operation time (Figure 3) as long as it does not affect block retrieval.

Theorem 1

The sequence $(i_1, s_1^{\mathrm{s}}, s_1^{\mathrm{d}}), (i_2, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}), \dots, (i_n, s_n^{\mathrm{s}}, s_n^{\mathrm{d}})$ for C_0 is dominated by a sequence $(i_1, s_1^{\mathrm{s}}, s_n^{\mathrm{d}}), (i_2, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}), \dots, (i_{n-1}, s_{n-1}^{\mathrm{s}}, s_{n-1}^{\mathrm{d}})$, if all the following conditions are satisfied:

1. $i_n = i_1$.
2. $\{s_1^{\mathrm{d}}, s_n^{\mathrm{d}}\} \cap \{s_2^{\mathrm{s}}, s_2^{\mathrm{d}}, \dots, s_{n-1}^{\mathrm{s}}, s_{n-1}^{\mathrm{d}}\} = \emptyset$.
3. $N_{s_n^{\mathrm{d}}}(C_0) = N_{s_n^{\mathrm{d}}}(C_{n-1})$.



(a)Transitive Relocations (b) Combined Relocation

Figure 3: Combining Two Transitive Relocations into One

Proof. Condition 2 ensures that no block is relocated from or to stack s_1^{d} by the sequence $(i_2, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}), \dots, (i_{n-1}, s_{n-1}^{\mathrm{s}}, s_{n-1}^{\mathrm{d}})$. Moreover, condition 1 implies that block i_1 is not retrieved by this sequence, so that $s_n^{\mathrm{s}} = s_1^{\mathrm{d}}$ holds. Furthermore, stack s_1^{d} is not the target block in $C_0, C_1, \dots, C_{n-1}$ because no block is relocated from there by this sequence. It follows that this stack becomes the target block for the first time after relocation $(i_{n-1}, s_{n-1}^{\mathrm{s}}, s_{n-1}^{\mathrm{d}})$. Now, assume that block i_1 is relocated from s_1^{s} not to stack s_1^{d} but to stack s_n^{d}. From conditions 2 and 3, it does not make block i_1 interfere any retrieval from stack s_n^{d}. Let us denote by $\hat{C}_{n-1}$ the block configuration obtained by sequence $(i_1, s_1^{\mathrm{s}}, s_n^{\mathrm{d}}), (i_2, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}), \dots, (i_{n-1}, s_{n-1}^{\mathrm{s}}, s_{n-1}^{\mathrm{d}})$ for C_0. Then, the differences between C_{n-1} and $\hat{C}_{n-1}$ are:

(a) block i_1 is on the top of stack s_1^{d} in C_{n-1}, while it is on the top of stack s_n^{d}, unless it is already retrieved in $\hat{C}_{n-1}$,

(b) some block may be retrieved from stack s_1^{d} in $\hat{C}_{n-1}$, which may cause further retrieval.

The retrieval from stack s_1^{d} in (b), which is interfered by block i_1 in C_{n-1}, should be after relocation $(i_{n-1}, s_{n-1}^{\mathrm{s}}, s_{n-1}^{\mathrm{d}})$ because this stack does not become the target stack until then. Therefore, this retrieval should also be enabled by relocating block i_1 to stack s_n^{d} in C_{n-1}. In other words, the block should be retrieved in C_n. If, as in (a), block i_1 is already retrieved in $\hat{C}_{n-1}$, it should be caused by the retrieval from s_1^{d} in (b) because otherwise, block i_1 should already be retrieved also in C_{n-1}. Hence, relocating it from s_n^{s} in C_{n-1} makes it retrievable. From these observations, $\hat{C}_{n-1}$ and C_n are exactly the same. Since it is obvious that the crane operation time of sequence $(i_1, s_1^{\mathrm{s}}, s_1^{\mathrm{d}}), (i_2, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}), \dots, (i_n, s_n^{\mathrm{s}}, s_n^{\mathrm{d}})$ is not shorter than that of sequence $(i_1, s_1^{\mathrm{s}}, s_n^{\mathrm{d}}), (i_2, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}), \dots, (i_{n-1}, s_{n-1}^{\mathrm{s}}, s_{n-1}^{\mathrm{d}})$, the former is dominated by the latter.

4.2. Dominance on retrieval

The second dominance properties covers the situation when a block is retrieved regardless of which stack it is relocated to. In such a case, the destination stack with a shorter crane operation time is preferred (Figure 4).

Theorem 2

The sequence $(i_1, s_1^{\mathrm{s}}, s_1^{\mathrm{d}}), (i_2, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}), \ldots, (i_n, s_n^{\mathrm{s}}, s_n^{\mathrm{d}})$ for C_0 is dominated by a sequence $(i_1, s_1^{\mathrm{s}}, s_1^{\mathrm{d}'}), (i_2, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}), \ldots, (i_n, s_n^{\mathrm{s}}, s_n^{\mathrm{d}})$, if all the following conditions are satisfied:

1. $i_1 \in C_{n-1} \wedge i_1 \notin C_n$.
2. $\{s_1^{\mathrm{d}}, s_1^{\mathrm{d}'}\} \cap \{s_1^{\mathrm{s}}, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}, \ldots, s_n^{\mathrm{s}}, s_n^{\mathrm{d}}\} = \emptyset$.
3. $Q_{s_1^{\mathrm{d}'}}(C_0) > i_1$.
4. $N_{s_1^{\mathrm{d}'}}(C_0) < H$.
5. $N_{s_1^{\mathrm{d}'}}(C_0) \geq N_{s_1^{\mathrm{d}}}(C_0)$.
6. $s_1^{\mathrm{d}'} < s_1^{\mathrm{d}}$.

Proof. From condition 1, block i_1 is retrieved after the relocation of block i_n. Block i_1 stays on the top of stack s_1^{d} in $C_1, \ldots, C_{n-1}$ from condition 2. Since conditions 2 and 3 claim that stack $s_1^{\mathrm{d}'}$ is unchanged before block i_1 is retrieved, this block does not interfere any retrieval even if it is relocated not to s_1^{d} but to $s_1^{\mathrm{d}'}$, whose feasibility is guaranteed by condition 4. In addition, this relocation does not enable any retrieval from stack s_1^{d} before that of block i_1 because block i_1 is retrieved before the blocks in stack s_1^{d} in C_0 from condition 1, meaning that it never interferes their retrieval. Therefore, sequence $(i_1, s_1^{\mathrm{s}}, s_1^{\mathrm{d}'}), (i_2, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}), \ldots, (i_n, s_n^{\mathrm{s}}, s_n^{\mathrm{d}})$ yields exactly the same configuration as C_n. The total crane operation time of this sequence is not longer than that of sequence $(i_1, s_1^{\mathrm{s}}, s_1^{\mathrm{d}}), (i_2, s_2^{\mathrm{s}}, s_2^{\mathrm{d}}), \ldots, (i_n, s_n^{\mathrm{s}}, s_n^{\mathrm{d}})$ due to conditions 5 and 6: stack $s_1^{\mathrm{d}'}$ is nearer from the truck than stack s_1^{d}, and the former is at least as tall as the latter.

5. LOWER BOUND COMPUTATION

In this section, we will propose two types of lower bound of the objective value, which are employed in the branch-and-bound algorithm. Hereafter, a block below which a block with a smaller number is placed is referred to as a blocking block. Every blocking block should be relocated at least once.

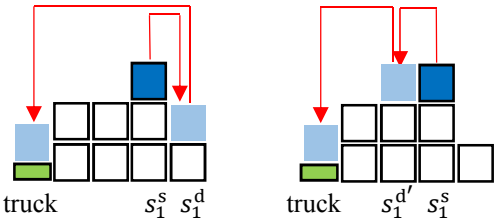
5.1. LB-A

Let us consider a bay configuration C where n blocks are placed at $(s_1, h_1), (s_2, h_2), \ldots, (s_n, h_n)$. If all these blocks can be retrieved without any relocations, the total crane operation time is, as the second term of (3), given by

$$\sum_{i=1}^{n} \{t_{\mathrm{s}} + 2t_{\mathrm{t}} s_i + t_{\mathrm{h}}(2h^{\mathrm{top}} - h_i - h^{\mathrm{out}})\}. \quad (4)$$

If the block placed at (s_i, h_i) is relocated to (s_i', h_i') from where it is retrieved, the increase of the total crane operation time from (4) is:

$$t_{\mathrm{s}} + 2t_{\mathrm{t}}(|s_i' - s_i| + s_i' - s_i) + 2t_{\mathrm{h}}(h^{\mathrm{top}} - h_i'). \quad (5)$$



(a) Longer Operation Time (b) Shorter Operation Time

Figure 4: Difference of Retrieval due to Preceding Relocation

The second term of (5) takes a minimum value 0 at $s_i' < s_i$ if $s_i > 1$, and $4t_{\mathrm{t}}$ at $s_i' = 2$ if $s_i = 1$. The minimum of the third term is achieved by relocating the block to the tallest stack and is given by $2t_{\mathrm{h}}(h^{\mathrm{top}} - h^{\max})$, where $h^{\max} = \max_{\substack{1 \leq s \leq S \\ N_s(C) < H}} N_s(C)$. Taking into account the increase of the stack height from $h^{\max}$ to $h^{\max} + 1$ after this relocation, we obtain the following lower bound of the total crane operation time for C:

$$\sum_{i=1}^{n} \{t_{\mathrm{s}} + 2t_{\mathrm{t}} s_i + t_{\mathrm{h}}(2h^{\mathrm{top}} - h_i - h^{\mathrm{out}})\} + t_{\mathrm{t}} m_1$$

$$+ \sum_{i=1}^{m} [t_{\mathrm{s}} + t_{\mathrm{h}}\{h^{\mathrm{top}} - \max(h^{\max} + i - 1, H - 1)\}]. \quad (6)$$

Here, m denotes the total number of relocations and m_1 that from stack 1. We choose m_1 as the number of blocking blocks in stack 1, and m as a lower bound of the total number of relocations. We refer to this lower bound as LB-A1 and LB-A2 when m is chosen as the total number of blocking blocks (the lower bound by Kim and Hong (2006)), and the lower bound by Tanaka and Takii (2016), respectively.

5.2. LB-B

To derive another lower bound LB-B, we start from (5) as LB-A, which provides the increase of the total crane operation time caused by a relocation. Unlike LB-A, we further take into consideration the situation when this block becomes a blocking block again and thus should be relocated once more. Let (s_i'', h_i'') be the destination slot of this relocation. Then, the total crane operation time further increases by

$$t_{\mathrm{s}} + 2t_{\mathrm{t}}(|s_i'' - s_i'| + s_i'' - s_i') + 2t_{\mathrm{h}}(h^{\mathrm{top}} - h_i''). \quad (7)$$

A lower bound of this increase is computed as that of (5) in LB-A, except for that h_i'' is chosen simply as $H - 1$. Thus it is given by $t_{\mathrm{s}} + 2t_{\mathrm{h}}(h^{\mathrm{top}} - H + 1)$ if $s_i' \neq 1$, and $t_{\mathrm{s}} + 4t_{\mathrm{t}} + 2t_{\mathrm{h}}(h^{\mathrm{top}} - H + 1)$ if $s_i' = 1$. In summary, a lower bound of the increase caused by relocating a block is provided as follows.

1. If the block does not become a blocking block: (5).

Table 2: Three Types of Crane Specification in the Literature

Setting	t_s[s]	t_t[s/stack]	t_h[s/stack]
1	30	1.2	0
2	5	1	0
3	0	1.2	7.77

1 (Lee and Lee, 2010)
2 (Ünlüyurt and Aydın, 2012)
3 (Lin et al, 2015)

2. If the block becomes a blocking block after the relocation:

$$\text{If } s_i' \neq 1,$$
$$2t_s + 2t_t(|s_i' - s_i| + s_i' - s_i) + 2t_h(2h^{top} - h_i' - H + 1). \quad (8)$$
$$\text{If } s_i' = 1,$$
$$2t_s + 2t_t(|s_i' - s_i| + s_i' - s_i + 2) + 2t_h(2h^{top} - h_i' - H + 1). \quad (9)$$

To obtain LB-B for C, we compute this lower bound for every blocking block and add them to (4). Inspired by the lower bound of the total number of relocations proposed by Zhu et al. (2012), the destination stack s_i' in (5), (8) and (9) is determined in the following manner. First, we relocate the topmost block above the target block, and its destination is determined so as to minimize the lower bound of the increase given by (5), (8) or (9). It is done by computing (5), (8) or (9) for every candidate stack whose height is less than H. Then, this block is removed from the bay, and the destination stack of the second topmost block (the topmost block in the current bay configuration) is determined in the same way. After the destination stacks of all blocking blocks above the target block are determined, the target block itself is removed as well, and the new target block in the current bay configuration is identified. Then, the destination stacks of blocking blocks above it are determined. This procedure is repeated until the destination stacks of all blocking blocks are determined. With regard to the height h_i' of the destination stack, we should consider the influence of removed blocking blocks that in practice are relocated to some stacks. Let b be the number of blocks removed so far in the above procedure (it includes removed target blocks). Without loss of generality, the target block is block 1 in C, and block c in the current block configuration. Then, the number of ignored blocks is given by $b - (c - 1)$. Therefore, h_i' is assumed to be the current height of the destination stack plus $b - (c - 1)$ (the maximum is $H - 1$).

6. COMPUTATIONAL EXPERIMENTS

We applied the proposed algorithm to the set of benchmark instances used in Caserta et al. (2011) in order to examine its effectiveness. This benchmark set is composed of 40 randomly generated instances for each combination of S and T, where T is the number of blocks in each stack (the total number of blocks is ST). The stack height limit H was set to $T + 2$. As the specification of the crane, we considered three settings in the literature, which are summarized in Table 2. In every setting, h^{top} and h^{out} were chosen as $h^{top} = H$ and $h^{out} = 0.5$, respectively. The computation was conducted using a desktop computer with an Intel Core i7-6700K CPU (4.00GHz) and 64GB RAM. The time limit for one instance was set to 1800s.

Table 3: Computational Results under Setting 1

T	S	LB-A1		LB-A2		LB-B	
		opt	time[s]	opt	time	Opt	Time
3	3	40	0.00	40	0.00	40	0.00
	4	40	0.00	40	0.00	40	0.00
	5	40	0.00	40	0.00	40	0.00
	6	40	0.00	40	0.00	40	0.00
	7	40	0.00	40	0.00	40	0.00
	8	40	0.00	40	0.00	40	0.00
4	4	40	0.00	40	0.00	40	0.00
	5	40	0.00	40	0.00	40	0.00
	6	40	0.01	40	0.01	40	0.00
	7	40	0.05	40	0.05	40	0.02
5	4	40	0.00	40	0.00	40	0.00
	5	40	0.14	40	0.01	40	0.02
	6	40	1.88	40	0.28	40	0.39
	7	40	49.31	40	11.73	40	9.28
	8	37	74.64	39	56.11	39	34.01
	9	29	302.03	32	404.13	37	141.73
	10	20	511.13	16	203.59	27	220.53

Table 4: Computational Results under Setting 2

T	S	LB-A1		LB-A2		LB-B	
		opt	time[s]	opt	time	Opt	time
3	3	40	0.00	40	0.00	40	0.00
	4	40	0.00	40	0.00	40	0.00
	5	40	0.00	40	0.00	40	0.00
	6	40	0.00	40	0.00	40	0.00
	7	40	0.00	40	0.00	40	0.00
	8	40	0.00	40	0.01	40	0.00
4	4	40	0.00	40	0.00	40	0.00
	5	40	0.00	40	0.01	40	0.00
	6	40	0.03	40	0.06	40	0.01
	7	40	0.42	40	0.91	40	0.04
5	4	40	0.00	40	0.00	40	0.00
	5	40	0.38	40	0.35	40	0.04
	6	40	16.49	40	29.35	40	0.71
	7	37	128.17	36	67.49	40	64.49
	8	27	225.70	25	332.92	38	65.95
	9	14	538.09	14	675.19	26	106.93
	10	6	511.48	4	304.39	14	366.22

Table 5: Computational Results under Setting 3

T	S	LB-A1		LB-A2		LB-B	
		opt	time[s]	opt	Time	Opt	Time
3	3	40	0.00	40	0.00	40	0.00
	4	40	0.00	40	0.00	40	0.00
	5	40	0.01	40	0.01	40	0.00
	6	40	0.69	40	2.20	40	0.34
	7	40	2.73	40	8.21	40	1.13
	8	40	60.64	39	105.65	40	21.26
4	4	40	0.01	40	0.02	40	0.01
	5	40	1.80	40	5.46	40	1.29
	6	39	170.40	33	134.64	40	88.36
	7	17	430.01	11	498.18	22	402.28
5	4	40	0.59	40	1.77	40	0.51
	5	29	234.27	25	344.92	31	210.05
	6	4	863.87	2	1507.20	4	515.74

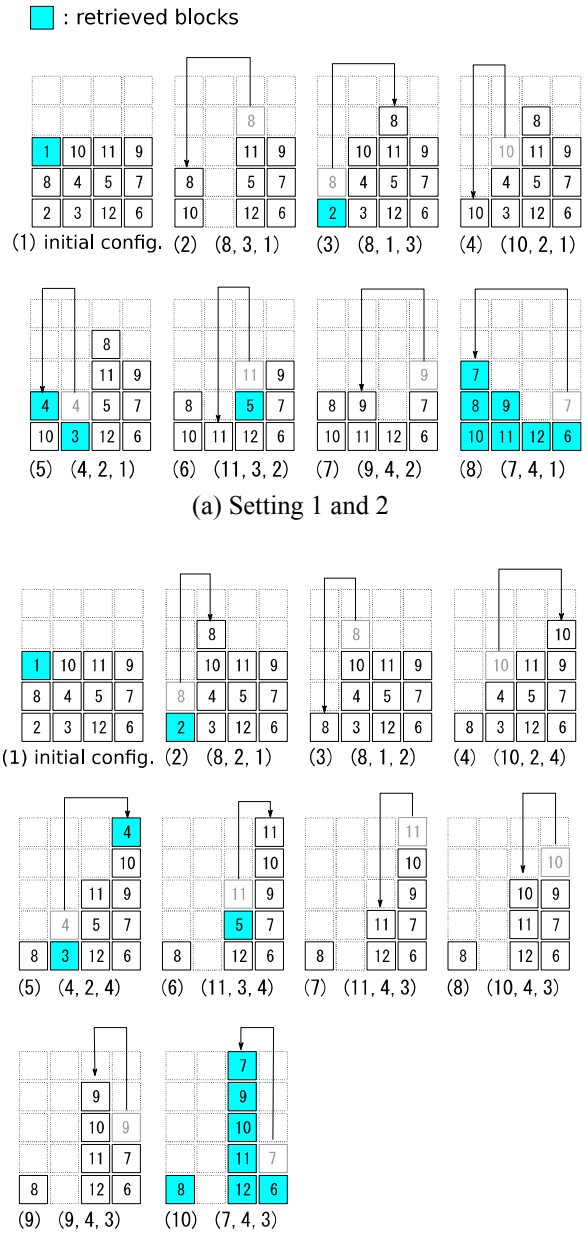
The computational results are summarized in Tables 3-5 for settings 1-3, respectively. In the tables, 'opt' denotes the number of instances out of 40 solved to optimality within the time limit, and 'time' the average CPU time in seconds over instances solved to optimality.

LB-A2 is not smaller than LB-A1 because the lower bound of the total number of relocations by Tanaka and Takii (2016) used in LB-A2 always dominates the total number of blocking blocks used in LB-A1. On the other hand, the former takes a longer computation time than the latter. Therefore, it depends on the crane specification which lower bound yields a better result.

Indeed, the algorithm with LB-A2 is faster than that with LB-A1 under setting 1, whereas the converse is true under settings 2 and 3. It will be because the impact of the number of relocations on the objective value is smaller in setting 1 than in settings 2 and 3. Among the three types of lower bound, LB-B yields the best results under all the settings. Although it requires a longer computation time than LB-A1, its tightness seems to contribute to improving the efficiency of the algorithm further.

Schwarze and Voß, (2016) solved the same instances under settings 1 and 3 using an ILP formulation for the BRP to minimize the total number of relocations in Zehendner et al. (2015). Although a direct comparison is not possible due to differences in CPUs (their CPU is slower than ours), they failed in solving to optimality within a time limit of 3,600s in a multi-thread environment, six instances with $(T, S) = (5, 5)$ under setting 1, and three instances and one instance with $(T, S) = (4, 6)$ and $(5, 4)$, respectively, under setting 3. Because all these instances were solved to optimality by the proposed algorithm with LB-B, it seems that our algorithm outperforms their approach.

Next, we examine the effect of the crane specifications on a solution. Figure 5 provides optimal solutions of the same instance under different settings. In this example,



(a) Setting 1 and 2

(b) Setting 3

Figure 5: An Example of Solutions under Different Crane Specification

settings 1 and 2 yield the same solution (a), whereas a difficult solution (b) is obtained under setting 3. We should also note that solution (a) also minimizes the total number of relocations. We can observe from this figure that in solution (b), the right most two stacks are more likely to be used, although they are far from the truck. It is due to the fact that the hoisting time (vertical travel time) is relatively large compared to the trolley time (horizontal travel time) under setting 3, so that taller stacks are preferred in order to reduce the hoisting time. In solution (b), the total number of relocations increases by 2 from the optimal value. However, it does not affect the total crane operation time directly because $t_s = 0$ is assumed under setting 3.

7. CONCLUSION

In this study we proposed a branch-and-bound algorithm for the block relocation problem to minimize the total crane operation time. For this purpose, we proposed dominance properties and two types of lower bound. Numerical experiments showed that the algorithm is capable of solving benchmark instances efficiently although its performance depends highly on the crane specification. In the three types of settings considered in this study, the crane travel speed (trolley speed and hoisting speed) linearly depends on the travel distance. However, it is not the case in practice due to acceleration and deceleration. Hence it will be necessary to extend the proposed lower bounds for such situations. In real-world container yards, it is often the case that 5 or 6 containers are piled up in each stack. Since instances of this size is still hard to solve for the proposed algorithm, constructing good heuristic algorithms is also an important future research topic.

This work was supported by JSPS KAKENHI Grant Number JP 15K01187.

8. REFERENCES

Caserta M. and Voß S., 2011. Applying the corridor method to a blocks relocation problem. OR Spectrum, 33, 915-929.

Kim K. H. and Hong G.-P., 2006. A heuristic rule for relocating blocks. Computers & Operations Research, 33, 940-954.

Lee Y. and Lee Y. J., 2010. A heuristics for retrieving containers from a yard. Computers & Operations Research, 37, 1139-1147.

Lin D.-Y. et al., 2015. The container retrieval problem with respect to relocation. Transportation Research Part C, 52, 132-143.

Schwarze S. and Voß S., 2016. Analysis of alternative objectives for the blocks relocation problem, working paper.

Tanaka S. and Takii K., 2016. A faster branch-and-bound algorithm for the block relocation problem. IEEE Transactions on Automaton Science and Engineering, 13, 415-422.

Ünlüyurt T. and Aydın C., 2012. Improved rehandling strategies for the container retrieval process. Journal of Advanced Transportation, 46, 378-393.

Zehendner E. et al., 2015. An improved mathematical formulation for the blocks relocation problem. European Journal of Operational Research, 245, 181-190.

Zhu W., Qin H., Lim A. and Zhang H., 2012. Iterative deepening A* algorithms for the container relocation problem. IEEE Transactions on Automation Science and Engineering, 9, 710-722.