European Journal of Operational Research 226 (2013) 32–45

Contents lists available at SciVerse ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

Storage space allocation models for inbound containers in an automatic container terminal

Mingzhu Yu, Xiangtong Qi *

Department of Industrial Engineering and Logistics Management, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong

ARTICLE INFO

Article history:
Received 4 January 2011
Accepted 29 October 2012
Available online 9 November 2012

Keywords:
Container terminal scheduling
Space allocation
Re-marshaling
Optimization

ABSTRACT

This paper studies the problem of improving the operations efficiency for retrieving inbound containers in a modern automatic container terminal. In the terminal, when an external truck arrives to collect a container stored in a specific container block, it waits at one end of the block where an automatic stack crane will retrieve the container and deliver it to the truck. With the aim of reducing the expected external truck waiting time which is determined by how the containers are stored in a block, we propose two correlated approaches for the operations efficiency improvement, (1) by designing an optimized block space allocation to store the inbound containers after they are discharged from vessels, and (2) by conducting overnight re-marshaling processes to re-organize the block space allocation after some containers are retrieved. For the block space allocation problem, we consider three optimization models under different strategies of storing containers, namely, a non-segregation model, a single-period segregation model, and a multiple-period segregation model. Optimal solution methods are proposed for all three models. For the re-marshaling problem with a given time limit, we find that the problem is NP-hard and develop a heuristic algorithm to solve the problem. We then use simulation to validate our models and solution approaches. Simulation results reveal important managerial insights such as the advantage of the multiple-period segregation over the myopic single-period segregation, the possibility of overflow of the segregation model, and the benefit of re-marshaling.

1. Introduction

The operations efficiency in a container terminal is an important issue in the maritime logistics industry. Researchers have been trying to improve the terminal's operations efficiency from different aspects, for example, by optimizing the schedule of quay cranes and block cranes. In this paper, we study the operations efficiency improvement for inbound container retrieval. The problem is new in that we consider a type of new automatic container terminal that has not been well studied in the literature. Instead of optimizing the real-time crane scheduling, we focus on the off-line planning problem, i.e., how to allocate the arriving inbound containers in a block at the beginning of a day so that they can be efficiently retrieved when being collected by customers.

Generally speaking, there are three types of containers being handled in a container terminal: inbound, outbound, and transshipment containers. Inbound containers are discharged from the vessels, temporarily stored in the container yard for several days, and then collected by the external trucks (ET) of the container owners. Outbound containers are sent to the terminal by the customers, also temporarily stored in the yard, and then loaded onto the designated vessels when the vessels arrive at the berth. Transshipment containers are discharged from one vessel, stored in the yard for some time, and loaded onto another vessel.

To strengthen the business competitiveness, a container terminal has to maintain a high service standard to customers. One of the important measures to the service level is the turn-around time incurred when an ET arrives to collect an inbound container. The turn-around time includes two parts: the time of driving in the yard and the time of waiting for the desired container to be retrieved. The average or expected driving time in the yard mainly depends on the physical layout and dimension of the container terminal, which is roughly a constant. On the other hand, the waiting time is determined by the desired container's exact position in the block, which can be reduced if the containers are stored by following certain well-designed distribution. This is the issue to be addressed in this paper.

We study a modern automatic container terminal as shown in Fig. 1. Rail-mounted Automatic Stacking Cranes (ASCs) are deployed in each block as transfer cranes. The container yard has multiple parallel blocks that are perpendicular to the berth (water side) and the gate (land side). Each block has multiple bays (the length of the block), and each bay has multiple rows (the width

* Corresponding author. Tel.: +852 2358 8231.
E-mail addresses: julieyu@ust.hk (M. Yu), ieemqi@ust.hk (X. Qi).

0377-2217/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ejor.2012.10.045

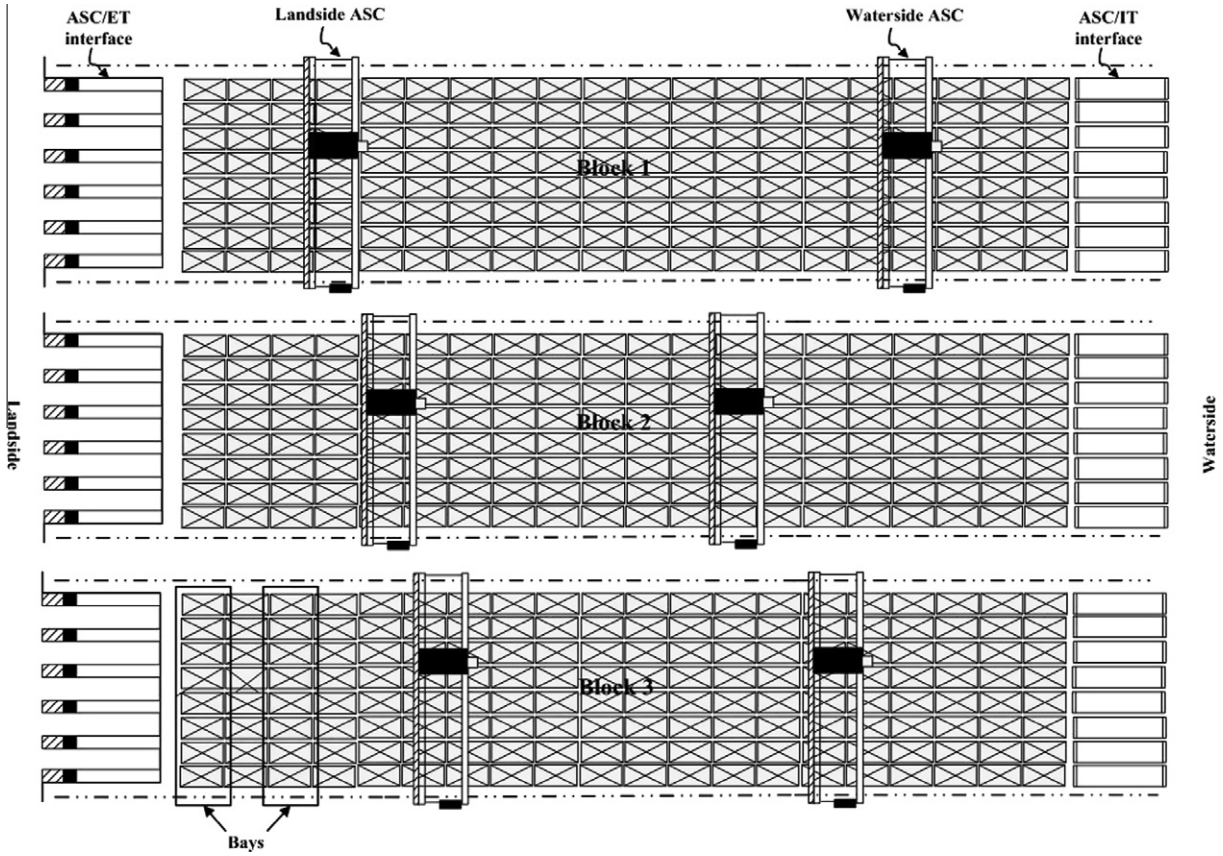

Fig. 1. Planform of a modern automatic container terminal.

of the block) and multiple tiers (the height of the block). In the terminal that directly motivates this research, each block has fifty bays, and each bay has ten rows and five tiers, which is a typical size of a container terminal.

The terminal is new in that the blocks' interfaces to the ETs and internal trucks (ITs) are restricted at the two ends, i.e., the land and water sides, respectively. Trucks cannot go into the lanes between blocks to collect or deliver containers. Traditionally, most terminals allow ETs or ITs to load or unload containers along the lanes between the blocks. In recent years, some new terminals have been built with advanced automatic block cranes. To ensure a high degree of automation, the design only allows ETs and ITs to load or unload containers at the two ends of a container block. There are advantages and disadvantages for each design. The traditional design is flexible for truck scheduling, but it requires more careful traffic control and occupies more space for truck pavements, which results in less space for container storage. The new design restricts the block interfaces at the ends of blocks, leaving more space for container storage and involving fewer manual operations. The use of automation that can increase the efficiency and reduce the operational cost is deemed as the only possible way for the future terminals (Ioannou et al., 2000). However, the operations of such new container terminals have been less studied, compared with the extensive research on traditional container terminals.

In such a container terminal, we define the waiting time of an ET to collect an inbound container as the summation of the container rehandle time that is needed to move the buried container out (if necessary), and the travel time of the ASC from the bay where the container is stored to the land side access point. The ET waiting time in this paper is different from that in the literature. For example, Kim and Kim (1999) considered the ET waiting time only as the expected container rehandle number in the traditional container terminals; Kim and Kim (2002) used a high level M/G/1 queueing model to analyze the ET waiting time which is determined by the ET arrival rate, the expected crane travel time, expected container rehandle time, and container loading/unloading time.

While we concentrate on the ET waiting time, there are other important measures for the operations efficiency in a container terminal, such as crane utilization rate, average container loading/unloading time, and vessel turnaround time, etc. We now discuss how optimizing ET waiting time may affect other operational measures. Firstly, we consider the case where inbound and outbound containers are separately stored in different blocks. The isolated storage of the inbound and outbound containers separates the operations in the inbound and outbound directions and eliminates conflicts between them. Secondly, the inbound containers moved from the quay by the ITs will be unloaded to the block waterside interface area, waiting to be moved to the desired bay. Hence, the inbound container storage space allocation scheme affects only the ASC operation efficiency, independent of other factors, such as IT operation efficiency, vessel turnaround time, and quay crane efficiency. Finally, because the ASC is used to move the inbound containers to their designated positions, the space allocation will directly affect the ASC operations, which will be investigated by numerical experiments later.

The inbound containers are discharged from vessels in groups with a known vessel schedule, but they are collected one by one by the ETs in a random order. In other words, the containers in a block may be collected in any sequence. In practice, it is very difficult, if not totally impossible, to obtain the information on the container collection sequence. Therefore, we assume that, when an ET arrives, each container is to be collected with an identical probability. Such an assumption enables us to estimate the expected number

of rehandles required by collecting a container. Based on such an estimation, we can then reduce the expected inbound container retrieval time through (1) an efficient space allocation scheme for the arriving inbound containers to get a desired block space layout, and (2) a container re-marshaling (house-keeping) operation scheme to re-organize the block space allocation during the night time. Since blocks are operated independently of each other, we focus on the inbound container space allocation and the re-marshaling operation in one block only.

We summarize our major contribution as follows:

1. We develop three optimization models for the block space allocation problem, each with a different strategy of storing containers: a non-segregation model, a single-period segregation model, and a multiple-period segregation model.
2. We provide a more precise estimation for the expected number of rehandles needed to retrieve one container, which improves the results in the literature. Based on the estimation, all the three abovementioned models can be optimally solved either by a convex network flow algorithm or a dynamic programming method.
3. We formally define and formulate the re-marshaling problem with a time constraint. We prove that the problem is NP-hard and develop a heuristic algorithm to solve it.
4. We conduct an extensive simulation study which shows important managerial insights such as the advantage of the multiple-period segregation over the myopic single-period segregation, the possibility of overflow of the segregation model, the performance comparison of non-segregation model and segregation model, and the benefit of re-marshaling.

The rest of the paper is organized as follows. Section 2 reviews the related literature. Section 3 proposes the mathematical formulations for the three allocation models, and Section 4 provides the solution methods. Section 5 focuses on the re-marshaling problem. Section 6 demonstrates the simulation results. The paper is concluded in Section 7.

2. Literature review

In the literature, there exists extensive research for improving the operations efficiency in container terminals from various perspectives. We refer to Günther and Kim (2006), Steenken et al. (2004), and Stahlbock and Voß (2008) for comprehensive reviews. In the following, we only discuss a few results that are directly related to our work.

Most research on container block space allocation has focused on outbound containers (Kim et al., 2000; Kim and Park, 2003; Kim and Lee, 2006; Zhang et al., 2011; Jiang et al., 2012). The main reason lies in the fact that the outbound containers are loaded to vessels according to a predetermined sequence. Therefore, it is easy to define and formulate an optimization problem for outbound containers.

On the contrary, inbound containers are collected by customers randomly. The dynamic characteristic makes the space allocation more challenging. The best that can be done is to reduce the expected retrieval time under the assumption that containers are retrieved with certain probability distribution. For the traditional container blocks, Kim and Kim (1999) studied the inbound container space allocation problem for the segregation allocation policy. They aimed to decide the optimal average stack height so as to minimize the rehandle time. Considering both the fixed investment cost and variable operational cost, Kim and Kim (2002) determined the optimal size of the storage space and the optimal number of handling facilities for inbound containers.

The container re-marshaling research has also mainly focused on outbound containers, overlooking inbound containers for similar reasons. Kim and Bae (1998) studied how to convert the given initial block layout to the desired block layout by moving the fewest containers in the shortest traveling distance. Choe et al. (2009) divided the outbound container re-marshaling problem into two subproblems. They aimed to completely prevent rehandling during loading and minimize the interference between multiple ASCs. Lee and Chao (2009) provided a neighborhood search heuristic to solve the outbound re-marshaling problem by adopting a two-stage approach, where the first stage deals with the movement sequence decision and the second stage makes the final bay layout decision. Lee and Hsu (2007) developed a mathematical network model for the outbound container re-marshaling problem, which is called container pre-marshaling. They provided some strategies to simplify the complicated model, and proposed a heuristic approach to solve this problem. Other methods for the container re-marshaling problem can be found in Bortfeldt (2004), Caserta and Voß (2009), Caserta et al. (2012), and Bortfeldt and Forster (2012). One important missing factor in the above work is the time constraint in doing re-marshaling, which will be included in this paper. With a time constraint, the final layout of a block also becomes a decision variable.

3. Inbound container space allocation models

In this section, we present the mathematical formulations for the block space allocation models. We focus on the problem for a single block because the results for a single block can be used for every block in the yard. Let the number of bays in the block be θ and the bays be indexed from the land side access point. In other words, bay 1 is the closest to the land side, and bay θ is the farthest.

We have the following assumptions on the container retrieval operations.

Assumption 1. The inbound containers stored in any bay are evenly allocated within the bay, i.e., the height difference between rows in the same bay is at most one tier.

Assumption 2. The container rehandling is restricted within a single bay, i.e., a rehandled container is moved onto another stack in the same bay.

Assumption 3. Every container in a block has the same probability of being collected by an ET, regardless of the time when it arrives.

Assumption 1, which will lead to the least expected number of container rehandles, is not restrictive because we can always maintain such a structure when locating and moving the containers. Specifically, we assume that each time after finishing moving an inbound container to the ET, the ASC has enough time to restore the even distribution of containers in the concerned bay. In practice, this can be done in the idle times during the ET arrival process. Although we may not be able to guarantee that can be done in time for all cases, it is an acceptable approximation that facilitates the analysis. Assumption 2 is a well accepted practice for easy administration, partially because moving a rehandled container to another bay will cause extra time for ASC movement. Assumption 3 is reasonable when there is no information from customers for the container collection. Upon the arrival of the inbound containers, terminals basically have no idea when these containers will be collected by the ETs. Based on Assumption 3, the expected number of rehandles to retrieve all containers in a bay can be estimated (Section 3.1).

We have the following notation in the problem definition and formulation.

B: the number of inbound containers to be allocated in the block in a certain period (for non-segregation and single-period segregation allocation models).
B_g: the number of inbound containers to be allocated in period g, $g = 1, \ldots, G$ (for the multiple-period segregation allocation model), where G is the length of the planning horizon.
θ: the number of bays in a block.
E: the set of empty bays in a block.
$\bar{h}$: the maximum number of containers in a bay, that is, the capacity of a bay.
h_k^0: the initial number of containers in bay k, $k = 1, \ldots, \theta$.
s: the number of rows in a bay of a block.
r: the time required for rehandling one container, i.e., the time for moving a container from the top of one stack to the top of another stack within the same bay. This time includes hoist-direction container loading time, traverse-direction moving time and hoist-direction container unloading time. We take average for the traverse-direction moving time between any two stacks in a bay. To eliminate the significant setup time of ASC's movement between bays, we assume that the rehandled container is moved onto another stack in the same bay (refer to Assumption 2).
t: the bi-direction travel time of the ASCs per bay. It is the sum of the ASC's empty movement (move from the landside access point to the target bay, without a container) time per bay, and loaded movement (move from the target bay to the landside access point, with a container) time per bay.
$R(x)$: the expected number of rehandles to pick up all the containers in a bay when the number of containers in the bay is x. The expression of $R(x)$ is determined by the rehandle number estimation (Section 3.1).

Decision variable:
y_k: the number of inbound containers allocated in bay k, $k = 1, \ldots, \theta$.

In practice, inbound containers arrive in batches, where we define "a batch of inbound containers" as the inbound container group unloaded from vessels in the same period. In this paper, we set the planning time as a day. We treat the inbound container allocation problem as an off-line planning problem. At the beginning of a day, the terminal operator could obtain the information of the arriving inbound container number to the block in this day, and the current block layout. According to this information, he decides the allocation of the arriving inbound containers. There are two strategies being used in space allocation for inbound containers from different batches (Castilho and Daganzo, 1993), segregation policy and non-segregation policy. Under the segregation policy, multiple designated empty bays are allocated for each batch of the inbound containers. Under the non-segregation policy, all inbound containers are mixed together such that newly discharged containers are piled on top of existing containers. Although both segregation and non-segregation policies are commonly used in practice and assumed by the academic research, as far as we know, their relative performance comparison has been rarely studied in the literature.

3.1. Rehandle estimation

Before presenting the inbound container space allocation models, we estimate the expected number of rehandles required to retrieve the containers from a bay, assuming all containers have the same probability of being collected as the next container. Such a problem was studied in Kim and Kim (1999), and Kim (1997). Under their analysis, we will have

$$R(x) = \left(\frac{1}{4s} + \frac{1}{16s^2}\right)x^2 + \left(\frac{1}{8s} - \frac{1}{4}\right)x, \tag{1}$$

where x is the number of containers in a bay. Note that (1) looks slightly different from the original equations in the literature because we use different notation here (details in Explanation 1 in Appendix). Numerical experiments show that when a bay includes more than two tiers of containers, i.e., $x \geqslant 2s$, the estimation has a very small relative error (less than 5.1%). However, it is not known how good the estimation is when $x < 2s$. We calculate the expected number of rehandles when there are fewer than two tiers of containers in a bay, and find that the estimation may lead to much larger relative error. Therefore, we need a better estimation when there are fewer than two tiers of containers. We have the following results in Lemma 1.

Lemma 1. *When the number of containers in a bay is less than 2s, the expected number of rehandles to retrieve all containers is given by*

$$R(x) = \begin{cases} 0, & x \leqslant s, \\ \alpha x - \beta, & s+1 \leqslant x < 2s, \end{cases} \tag{2}$$

where $\alpha = \frac{s+2}{2s+2}$ *and* $\beta = \frac{s(s+2)}{2s+2}$.

The proof of Lemma 1 is provided in Appendix.

So we can use the following function $R(x)$ to estimate the expected number of total rehandles to retrieve all containers in a bay, where

$$R(x) = \begin{cases} 0, & x \leqslant s, \\ \alpha x - \beta, & s+1 \leqslant x < 2s, \\ \mu x^2 + \eta x, & x \geqslant 2s, \end{cases} \tag{3}$$

with $\mu = \frac{1}{4s} + \frac{1}{16s^2}$ and $\eta = \frac{1}{8s} - \frac{1}{4}$. It can be verified that $R(x)$ is a convex function of x.

3.2. Non-segregation space allocation model

We now present the formulation of the non-segregation space allocation model. Given the current block layout characterized by $h_1^0, h_2^0, \ldots, h_\theta^0$, and a batch of B inbound containers to be allocated in the block, we need to determine how to allocate the containers in the block, i.e., to decide y_k, the number of containers allocated in each bay k, with the aim of minimizing the expected container retrieval time. Recall that the retrieval time of a container includes two parts, the ASC movement that is determined by the distance from the allocated bay of the container to the land side access point, and the expected rehandling time that is determined by the total number of containers in the bay. The model can be formally defined by the following formulation.

$$\text{(P1)} \quad \min \sum_{k=1}^{\theta} r \cdot \left(R\left(h_k^0 + y_k\right) - R\left(h_k^0\right)\right) + \sum_{k=1}^{\theta} t \cdot k \cdot y_k, \tag{4}$$

$$\text{s.t.} \sum_{k=1}^{\theta} y_k = B, \tag{5}$$

$$h_k^0 + y_k \leqslant \bar{h}, \quad \forall k = 1, \ldots, \theta, \tag{6}$$

$$y_k \in \{0, 1, \ldots, \bar{h}\}, \quad \forall k = 1, \ldots, \theta. \tag{7}$$

The objective function (4) is to minimize the total inbound container retrieval time, where the first term is the increased expected rehandle time caused by the space allocation and the second term is the increased total ASC travel time. To be consistent with the second term, we subtract the constant $R\left(h_k^0\right)$ from $R\left(h_k^0 + y_k\right)$ in the first term so as to describe the "increased" expected rehandle time. Constraint (5)

allocates the B arriving inbound containers to some of the bays in the block, and Constraint (6) is the bay capacity constraint. The difficulty of problem (P1) depends on the exact form of the rehandling number estimation function $R(\cdot)$.

3.3. Single-period segregation space allocation model

Under the segregation policy, inbound containers from different periods are not mixed with each other in the same bay. Hence given the current block layout, only the empty bays in a block, i.e., in the set E, are available for allocation. In the segregation space allocation model, for a batch of arriving inbound containers, we also want to allocate space for them so as to minimize the expected increase of the container retrieval time. The formulation of the segregation space allocation model is as follows:

$$\text{(P2)} \quad \min \sum_{k\in E} r\cdot R(y_k) + \sum_{k\in E} t\cdot k\cdot y_k, \tag{8}$$

$$\text{s.t.} \sum_{k\in E} y_k = B, \tag{9}$$

$$y_k \in \{0,1,\ldots,\bar{h}\}, \quad \forall k\in E. \tag{10}$$

Clearly, (P2) is a special case of (P1).

3.4. Multiple-period segregation space allocation model

The segregation space allocation model (P2) only considers the decision for containers in the current batch (or period). As a consequence, such allocation may be myopic and greedily occupy the "good positions" for the current batch of containers, leaving the "inferior positions" for the future arriving containers. To address such a problem, it is necessary to consider the effects of the current decision on the future arriving containers' space allocation.

As a straightforward solution, one may suggest an approach that makes the space allocation for multiple batches together. Such an idea, however, not only leads to a very complicated model, but also has limited practical value, due to the fact that containers are continually retrieved by ETs with an unknown retrieval schedule. Therefore, what is the optimal allocation for the future arriving containers based on the current layout may not be optimal for the future arriving containers for the future layout.

We propose to allocate the space for multiple periods on a rolling horizon basis. Suppose that the planned horizon contains G periods with B_g containers arriving in period g ($g = 1, \ldots, G$). We first decide the allocation for the B_1 containers that arrive in period 1, by considering the decision effect on periods 2 to G. Based on the solution obtained, we allocate the space for period-1 containers and get a block layout. Such a layout, however, will be continuously and randomly changed due to the container retrievals. Then when the containers in period 2 arrive, there will be a new layout, based on which we can allocate space for period-2 containers, by considering the effect on period 3 to $G+1$, and so on. Hereafter, we call the allocation decision period "current period".

For such a purpose, we define new decision variables x_k as

$$x_k = \begin{cases} 1, & \text{if inbound containers in current period are} \\ & \quad \text{allocated in bay } k, \quad k\in E, \\ 0, & \text{otherwise.} \end{cases}$$

In order to model the "effect" of the current period decision on the future periods, we introduce a new performance metric, the average expected retrieval time (AERT), denoted by τ. The AERT is defined as the total expected retrieval time divided by the number of the total containers to be allocated, i.e.,

$$\tau = \frac{\sum_{k\in E}(r\cdot R(y_k) + t\cdot k\cdot y_k)}{\sum_{k\in E} y_k}.$$

Similarly, we define the AERT of bay k as $\tau_k(y_k) = (r\cdot R(y_k) + t\cdot k\cdot y_k)/y_k$. For a given τ, we can get $\bar{y}_k(\tau)$, the maximum number of containers that can be allocated in bay k such that the AERT of this bay is no more than τ. Now we derive the expression of $\bar{y}_k(\tau)$. From the rehandle number estimation $R(x)$ in (3), we can get the average expected retrieval time for one container in bay k as

$$\tau_k(y_k) = \frac{r\cdot R(y_k) + k\cdot t\cdot y_k}{y_k} = \begin{cases} 0, & y_k = 0, \\ kt, & 0 < y_k \leqslant s, \\ kt + r\alpha - r\beta/y_k, & s < y_k \leqslant 2s, \\ kt + r\mu y_k + r\eta, & 2s+1 < y_k \leqslant \bar{h}. \end{cases} \tag{11}$$

Note that τ has a largest possible value τ_{max}, where $\tau_{max} = \theta t + r\mu\bar{h} + r\eta$, which is achieved at $k = \theta$ and $y_k = \bar{h}$. In addition, τ takes a set of discrete values because y_k's are integers. Based on $\tau_k(y_k)$ and the definition of $\bar{y}_k(\tau)$, we can get the expression of $\bar{y}_k(\tau)$, which is the inverse function of $\tau_k(y_k)$:

$$\bar{y}_k(\tau) = \begin{cases} 0, & 0 < \tau < kt, \\ s, & \tau = kt, \\ \left\lceil \frac{r\beta}{r\alpha + kt - \tau} \right\rceil, & kt < \tau \leqslant r\alpha/2 + kt, \\ \min\left\{ \left\lceil \frac{\tau - r\eta - kt}{r\mu} \right\rceil, \bar{h} \right\}, & \tau > r\alpha/2 + kt. \end{cases} \tag{12}$$

We aim to find the minimum AERT, denoted by τ_{min}, that can be sustained for all G periods. Due to the randomness in the future container retrieval, we require that the allocation of the current period has to lead to an AERT strictly no higher than τ_{min}, and at the same time, there is still enough space leftover for the future arriving containers. For future arrivals, we make an approximate requirement such that the leftover space can guarantee τ_{min} by regarding the containers $\sum_{g=2}^{G} B_g$ as a single-period container. Thus in the current decision process, we only need to know B_1 and $\sum_{g=2}^{G} B_g$. In reality, B_1 can be deterministic, and there may exist larger and larger uncertainties for $B_2, \ldots, B_G$. However, the aggregated batch size, $\sum_{g=2}^{G} B_g$, can be accurate enough because the uncertainty can be pooled together. For example, a vessel is delayed from period 2 to period 3; this will change both B_2 and B_3, but $B_2 + B_3$ is still constant.

Based on the above discussion, we have the formulation for the multiple-period segregation space allocation model as follows:

$$\text{(P3)} \quad \min \tau, \tag{13}$$

$$\text{s.t.} \ \tau = \frac{\sum_{k\in E}(r\cdot R(y_k) + t\cdot k\cdot y_k)}{\sum_{k\in E} y_k}, \tag{14}$$

$$\sum_{k\in E} y_k = B_1, \tag{15}$$

$$B_1\cdot x_k \geqslant y_k, \quad \forall k\in E, \tag{16}$$

$$\sum_{k\in E} \bar{y}_k(\tau)(1-x_k) \geqslant \sum_{g=2}^{G} B_g, \tag{17}$$

$$y_k \in \{0,1,\ldots,\bar{h}\}, \quad \forall k\in E, \tag{18}$$

$$x_k \in \{0,1\}, \quad \forall k\in E. \tag{19}$$

The objective (13) is to minimize τ, the AERT of the inbound containers in period 1. Constraint (15) means the total number of inbound containers from current period is B_1. Constraint (16) states the relation between variable y_k and x_k. Constraint (17) means that, after current period's decision, the leftover empty bays' capacity is enough to keep future arriving containers' AERT at τ. Constraint

(18) is the bay capacity constraint. Note that, (P3) has feasible solutions if and only if $\left\lceil \frac{B_1}{h} \right\rceil + \left\lceil \frac{\sum_{g=2}^{G} B_g}{h} \right\rceil \leqslant |E|$.

3.5. Numerical examples

We now use numerical examples to show the different solutions of (P1), (P2) and (P3) with the same arriving inbound container pattern. The examples involve three-period container allocation. The block is initially empty and the arriving container numbers in the three periods are 54, 150, and 171, respectively. Please refer to Section 6 for the details of the setting for other parameters. The results are shown in Fig. 2. As in Fig. 2a and b, fewer bays are occupied in (P1) compared to (P2). In (P3) (Fig. 2c), containers from the first period are not all allocated in the leftmost bays, but in the 1st, 9th and 10th bays, leaving more "good" positions for the future arriving containers. This is the advantage of the multiple-period segregation allocation model over the single-period one.

4. Space allocation solution procedures

4.1. Convex cost network flow algorithm for (P1) and (P2)

Due to the convexity of $R(x)$, both (P1) and (P2) are convex programming problems. In particular, the structure of the constraints enables us to solve the two problems by network flow techniques.

In the research of network flow problems (Ahuja et al., 1993), people have studied a set of models with separable convex objective functions, known as convex cost network flow problem. The problem is defined on a directed network $G = (N,A)$, where each arc $(i,j) \in A$ has a capacity $u_{i,j}$ and a convex cost $C_{i,j}(x_{i,j})$, and each node $i \in N$ is associated with a number d_i to specify the node's supply or demand, depending on whether $d_i > 0$ or $d_i < 0$. The objective is to minimize the total network flow cost.

In models (P1) and (P2), the objective functions consist of separable convex costs. We can remodel the two problems as convex cost network flow problem as shown in Fig. 3. In the directed network, there is a dummy source node S, a dummy sink node T, and a

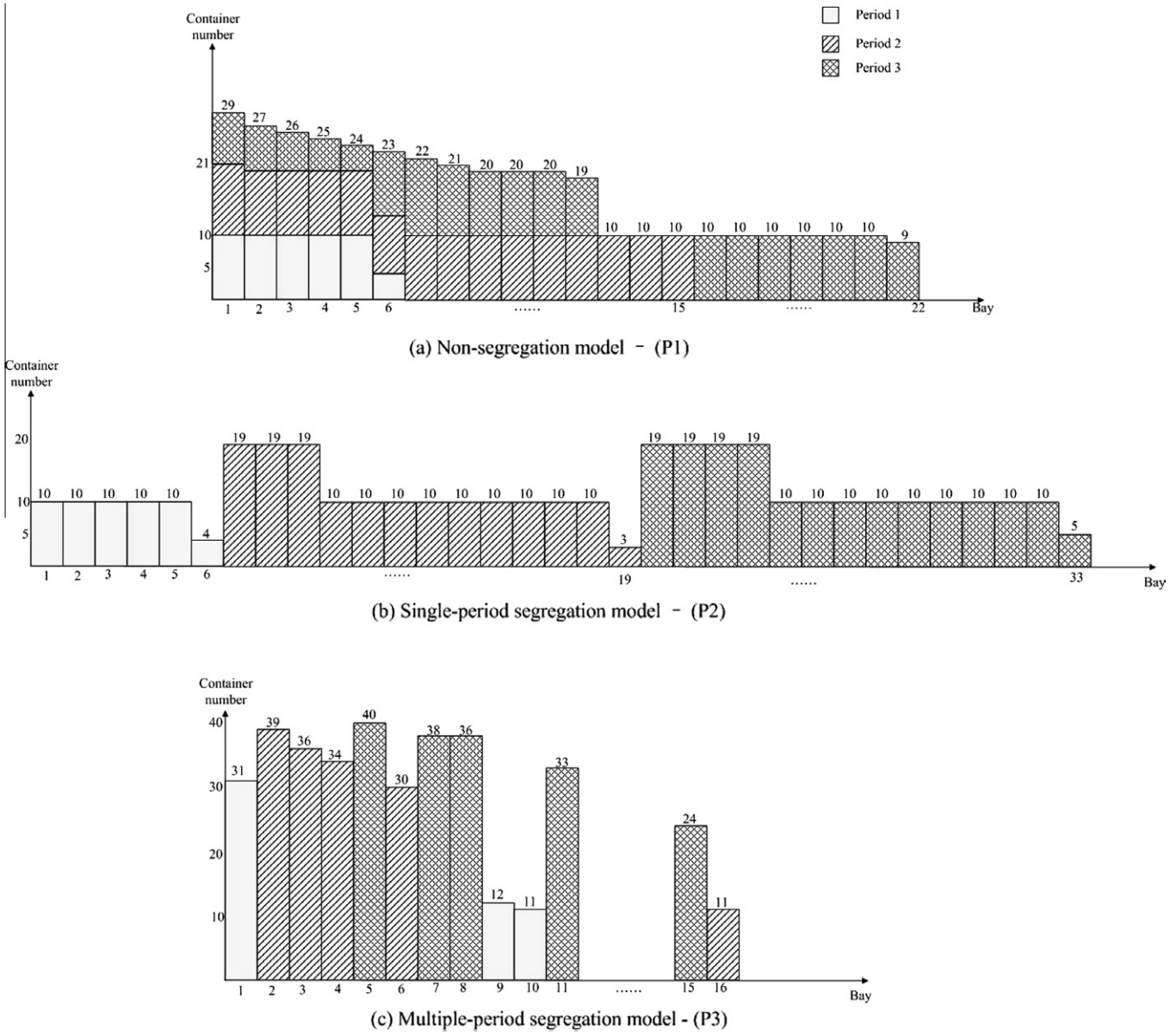

Fig. 2. Numerical example solutions for (P1), (P2) and (P3).

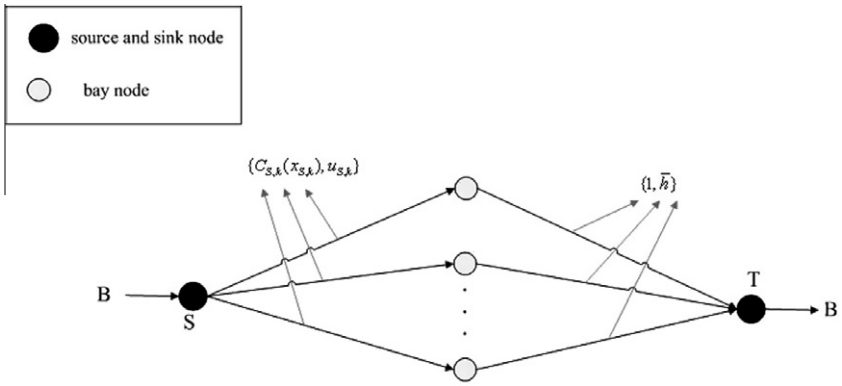

Fig. 3. Convex cost network flow model for (P1) and (P2).

set N' of bay nodes each corresponding to an available bay, where $|N'| = \theta$ for (P1) and $|N'| = |E|$ for (P2). There are B containers flowing from node S, through some of bay nodes in N', and eventually arriving at node T. Let $x_{S,k}$ be the number of containers flowing from the source node S to the bay node k, $k \in N'$. The cost functions and the arc capacities of the convex network flow problem corresponding to (P1) and (P2) are defined as follows:

$$\text{(P1):}\quad C_{S,k}(x_{S,k}) = r \cdot \left(R\left(h_k^0 + x_{S,k}\right) - R\left(h_k^0\right)\right) + k \cdot t \cdot x_{S,k},\quad u_{S,k} = \bar{h} - h_k^0;$$

$$\text{(P2):}\quad C_{S,k}(x_{S,k}) = r \cdot R(x_{S,k}) + k \cdot t \cdot x_{S,k},\ u_{S,k} = \bar{h}.$$

We can use the capacity scaling algorithm (Ahuja et al., 1993) to solve the convex cost network flow problem, and the computational complexity is polynomial. Let U be the upper bound on the largest arc capacity in the convex cost network, and n and m be the number of nodes and arcs. Then the capacity scaling algorithm obtains an integer optimal flow in $O((m \log U)n^2)$. In our models, $n = \theta + 2$, $m = 2\theta$ and $U = \bar{h}$, and all of them are constants. Hence, the computational complexity of the capacity scaling algorithm is independent of the number of arrival containers, B.

4.2. Dynamic programming for (P3)

We present a dynamic programming (DP) algorithm for the multiple-period segregation allocation model (P3). Recall that (P3) is to allocate B_1 containers in the current batch and $\sum_{i=2}^{G} B_i$ containers in future batches into a set E of empty bays, with the aim of minimizing the AERT τ. In the DP algorithm, we consider a subproblem with d_1 containers in the current batch and d_2 containers in future batches to be allocated to the empty bays from bay k to bay θ. With a given AERT τ, we can check the feasibility of meeting the τ requirement for the subproblem.

Specifically, let $f(k, d_1, d_2, \tau) = 1$ if it is feasible to allocate d_1 containers of the current period and d_2 containers of future periods to the empty bays from bay k to bay θ, and $f(k, d_1, d_2, \tau) = 0$ if it is infeasible. Then if bay k is empty, we can allocate up to $\bar{y}_k(\tau)$ containers, either from the current batch or from future batches, to bay k. The feasibility of doing this depends on whether it is feasible to allocate the remaining containers from bay $k+1$ to bay θ. Formally, for $k \in E$, we have the following DP recursion

$$f(k, d_1, d_2, \tau) = \max\{f(k+1, \max\{d_1 - \bar{y}_k(\tau), 0\}, d_2, \tau), f(k+1, d_1, \max\{d_2 - \bar{y}_k(\tau), 0\}, \tau)\}.$$

For $k \notin E$, we cannot allocate any containers to bay k, hence

$$f(k, d_1, d_2, \tau) = f(k+1, d_1, d_2, \tau).$$

We have the initial conditions for $k = \theta$ as follows. If $\theta \notin E$, then

$$f(\theta, 0, 0, \tau) = 1, \text{ and } f(\theta, d_1, d_2, \tau) = 0 \text{ for } d_1 > 0 \text{ or } d_2 > 0.$$

If $\theta \in E$, we have $f(\theta, d_1, d_2, \tau) = 0$ except

$$f(\theta, d_1, 0, \tau) = 1 \text{ if } 0 \leqslant d_1 \leqslant \bar{y}_\theta(\tau) \text{ and } f(\theta, 0, d_2, \tau) = 1 \text{ if } 0 \leqslant d_2 \leqslant \bar{y}_\theta(\tau).$$

Finally, we can solve (P3) by finding the minimal AERT, under which it is feasible to allocate the B_1 containers in the current batch and the $\sum_{i=2}^{G} B_i$ containers in future batches to all empty bays, i.e.,

$$(DP)\qquad \min\left\{\tau : f\left(1, B_1, \sum_{i=2}^{G} B_i, \tau\right) = 1\right\}.$$

The following results show that we do not need to perform the DP for all possible values of τ.

Proposition 1. *$f(k, d_1, d_2, \tau)$ is monotonic with respect to τ.*

Proof. Since $f(k, d_1, d_2, \tau)$ either equals to 1 or 0, to finish the proof, we only need to prove the following two conditions:

(1) if $f(k, d_1, d_2, \tau) = 1$, then $\forall \Delta$, $f(k, d_1, d_2, \tau + \Delta) = 1$;
(2) if $f(k, d_1, d_2, \tau) = 0$ then $\forall \Delta$, $f(k, d_1, d_2, \tau - \Delta) = 0$.

According to the definition of $f(k, d_1, d_2, \tau)$, if it is feasible at τ, then it must be feasible at $\tau + \Delta$. Similarly, if it is infeasible at τ, then it is also infeasible at $\tau - \Delta$. Hence, conditions (1) and (2) are satisfied. □

Based on the monotonic property discussed in Proposition 1 and the discrete characteristic of τ (c.f. (11)), we can use a bisection method to search the best τ. So the time complexity of searching τ is $\log \tau_{max}$, here $\tau_{max} = \theta t + r\mu\bar{h} + r\eta$. Assuming the block size θ and bay size s are constants, τ_{max} is a constant. According to the DP recursion relation, the complexity of the DP depends on the maximum values of k, d_1, d_2 and τ; that is, θ, B_1, $\sum_{i=2}^{G} B_i$, and τ_{max}, in which θ and τ_{max} are constants. Then the complexity of the DP is in $O\left(B_1 \sum_{i=2}^{G} B_i\right)$.

5. The re-marshaling problem

After the inbound container retrieval processes in the daytime, the block layout previously decided by the space allocation model is changed. In order to speed up the retrieval process for the next

day, the re-marshaling operation at night (house-keeping) is a common practice in a container terminal. Existing research on re-marshaling overlooks the re-marshaling time constraint. In practice, re-marshaling often has a time limit during which the containers can be rearranged. Specifically, given an initial block layout and an operation time limit, we aim to do re-marshaling to improve the block layout as much as possible.

5.1. Re-marshaling problem definition

Suppose that a set of F bays, denoted by $H^0 = \{h_1^0, h_2^0 \dots h_F^0\}$, need re-marshaling operations. For the case of the non-segregation allocation policy, the re-marshaling bay set contains all bays in the block. For the case of the segregation allocation policy, it only consists of the bays whose containers arrived in the same period. In particular, we cannot occupy those empty bays that are reserved for the future arrival containers, and hence (17) will not be violated. In such a re-marshaling design, we concentrate on the existing containers to facilitate their retrieval in the future, where we do not directly consider the future arriving containers because they can be handled by formulation (P3).

We want to rearrange the containers in the initial bays and obtain a new bay set, $H = \{h_1, h_2, \dots, h_F\}$, so as to reduce the total expected container retrieval time. Moreover, there is a time limit T for conducting the re-marshaling process. Let $G(H^0, H)$ be the total re-marshaling time needed to change bay set H^0 to H, and $L_k(x) = r \cdot R(x) + t \cdot k \cdot x$, be the expected time to retrieve all containers in bay k when the number of containers in bay k is x. Then we define the re-marshaling problem as follows:

$$(\mathrm{R}) \quad \min \sum_{k=1}^{F} L_k(h_k), \tag{20}$$

$$\text{s.t.} \sum_{j=1}^{F} h_j = \sum_{i=1}^{F} h_i^0, \tag{21}$$

$$G(H^0, H) \leqslant T. \tag{22}$$

The complexity of the re-marshaling problem comes from two parts: deciding the container flows among the bays and obtaining the container moving sequences. We now prove that the re-marshaling problem is NP-hard. Before that we need a lemma to show the following defined Fixed-size Subset-sum Problem is NP-hard.

Fixed-size Subset-sum Problem: Given $s + 2$ positive integers $w_1, \dots, w_s$, e and K, is there a subset $M \subseteq S = \{1, \dots, s\}$ such that $\sum_{i \in M} w_i = e$ and $|M| = K$?

Lemma 2. *The Fixed-size Subset-sum Problem is NP-hard.*

Proof. Consider the following well-known Subset-sum Problem which is NP-hard.

Subset-sum Problem: Given $s + 1$ positive integers $w_1, \dots, w_s$, e, is there a subset $M \subseteq S = \{1, \dots, s\}$ such that $\sum_{i \in M} w_i = e$?

If the Fixed-size Subset-sum Problem can be solved in polynomial time (namely, not NP-hard), then the Subset-sum Problem can also be solved in polynomial time by enumerating $K = 1, \dots, |M|$. Hence, Fixed-size Subset-sum Problem must be NP-hard. □

Theorem 1. *The re-marshaling problem with time constraint is NP-hard.*

Proof. We use a reduction from the Fixed-size Subset-sum Problem.

From an instance of Fixed-size Subset-sum Problem (**S1**), where we assume $w_1 > w_2 > \cdots > w_s$, we define the following instance of the re-marshaling problem (**S2**). As shown in Fig. 4, the capacity of each bay is two tiers, namely, $\bar{h} = 2s$; there are s bays where each of them contains $2s - 1$ containers; there is another bay (bay b) which has s containers. Assume that the initial position of the ASC is at bay b. The distances between bay b and other bays are $w'_1, \dots, w'_s$ bays, where we set $w'_i = w_i + \delta,\ i \in \{1, \dots, s\}$ with the positive number δ satisfying $\delta > w_1(s+3) - w_s(s+4)$. Moreover, we set the rehandle time r and travel time per bay t satisfying $2(w'_1 - w'_s)(s+1) < r/t < 2w'_s(s+1)/(s+3)$. In **S2**, we aim to improve the initial block layout such that the future container retrieval time is reduced by at least $et + K\delta t - Kr(s+3)/2(s+1)$ with the constraint that re-marshaling time limit is at most $T = et + K\delta t$.

Note that in the definition of **S2**, $\delta > w_1(s+3) - w_s(s+4)$ guarantees the existence of r and t satisfying $2(w'_1 - w'_s)(s+1) < r/t < 2w'_s(s+1)/(s+3)$. By $r/t < 2w'_s(s+1)/(s+3)$, we can verify that, moving a container from bay b to any bay $i, i \in \{1, \dots, s\}$, leads to a positive reduction of the retrieval time for the future container retrieval in the amount of $w'_i t - r(s+3)/2(s+1) > 0$. We call such a movement "reducing movement". Hence, we want to make "reducing movements" as much as possible. On the other hand, from $r/t > 2(w'_1 - w'_s)(s+1)$, we can verify that, moving a container from bay i to bay j in the first s bays ($j < i \leqslant s$) will increase the retrieval time for the future container retrieval in the amount of $r/2(s+1) - t\left(w'_j - w'_i\right) > 0$. Therefore, we will never consider such movements.

We now prove that there exists a feasible solution for **S2** if and only if a feasible solution exists for **S1**.

First, if **S1** has a subset $M \subseteq S = \{1, \dots, s\}$ such that $\sum_{i \in M} w_i = e$ and $|M| = K$, then in **S2**, we can do the corresponding "reducing movements" of the numbers in subset M. These "reducing movements" reduce the future container retrieval time by $\sum_{i \in M}(w'_i t - r(s+3)/2(s+1)) = et + K\delta t - Kr(s+3)/2(s+1)$, and the required re-marshaling time is $\sum_{i \in M} w'_i t = et + K\delta t$.

Second, if there is a solution for **S2** with the re-marshaling time being at most $et + K\delta t$ and the reduced container retrieval time being at least $et + K\delta t - Kr(s+3)/2(s+1)$, then the re-marshaling operations must consist of K "reducing movements". These "reducing movements" correspond to a subset $M \subseteq S = \{1, \dots, s\}$ such that $|M| = K$, $\sum_{i \in M}(w'_i t) \leqslant et + K\delta t$ and $\sum_{i \in M}(w'_i t - r(s+3)/2(s+1)) \geqslant et + K\delta t - Kr(s+3)/2(s+1)$. Namely, $\sum_{i \in M} w_i = e$. Therefore, **S1** has a feasible solution. □

5.2. Heuristic algorithm

Due to the NP-hardness of the re-marshaling problem, we develop a greedy algorithm that includes multiple iterations. In each iteration, we move a container between a pair of bays such that the movement saves enough expected container retrieval time in reasonable operation time. The movements continue until the total operation time reaches the re-marshaling time limit or no movement can reduce the expected retrieval time any more.

Before describing the details of the algorithm, we define a so-called weighted reduced expected retrieval time (*WRET*) for moving a container from bay i to bay j, $i, j \in \{1, 2, \dots, F\}$, $i \neq j$, as follows:

$$WRET_{ij} = \frac{[L_i(h_i) + L_j(h_j)] - [L_i(h_i - 1) + L_j(h_j + 1)]}{ASC\ moving\ time + 2LT}.$$

The numerator of $WRET_{ij}$ is the expected container retrieval time saving, which is caused by moving one container from bay i to bay j. The denominator is the operation time of the movement, which consists of the ASC travel time and container loading/unloading time ($2LT$). In each iteration, a container is moved from the top of bay k to the top of bay l if,

$$WRET_{kl} = \max\{WRET_{ij} : WRET_{ij} > 0, i, j \in \{1, 2, \dots, F\}, i \neq j\}.$$

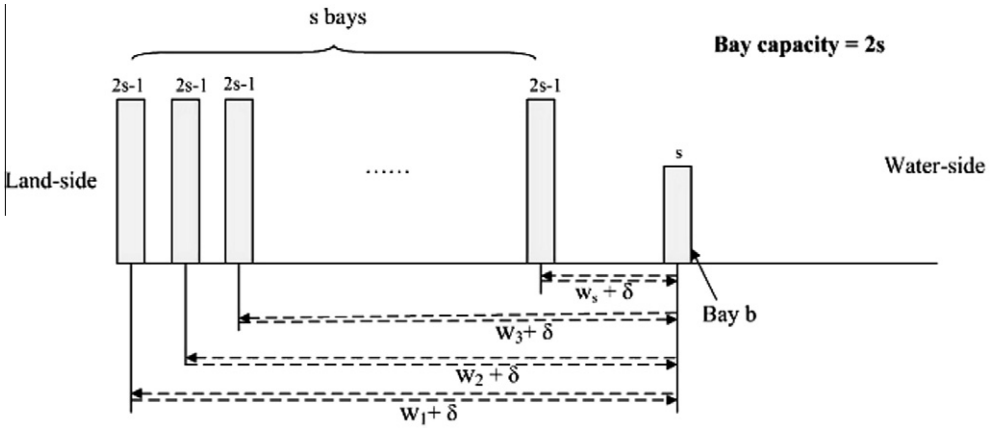

Fig. 4. The instance of the re-marshaling problem in the proof of Theorem 1.

After every movement, the values of all *WRET*s, h_i's and the total operation time are updated. The iteration stops when the total operation time is larger than *T*, or all *WRET*s become negative (the optimal layout is obtained). Moreover, the following optimality property could be utilized to further improve the solution.

Property 1. *In the re-marshaling operation, it is not optimal that there are containers being moved into and out of a bay in the same re-marshaling period.*

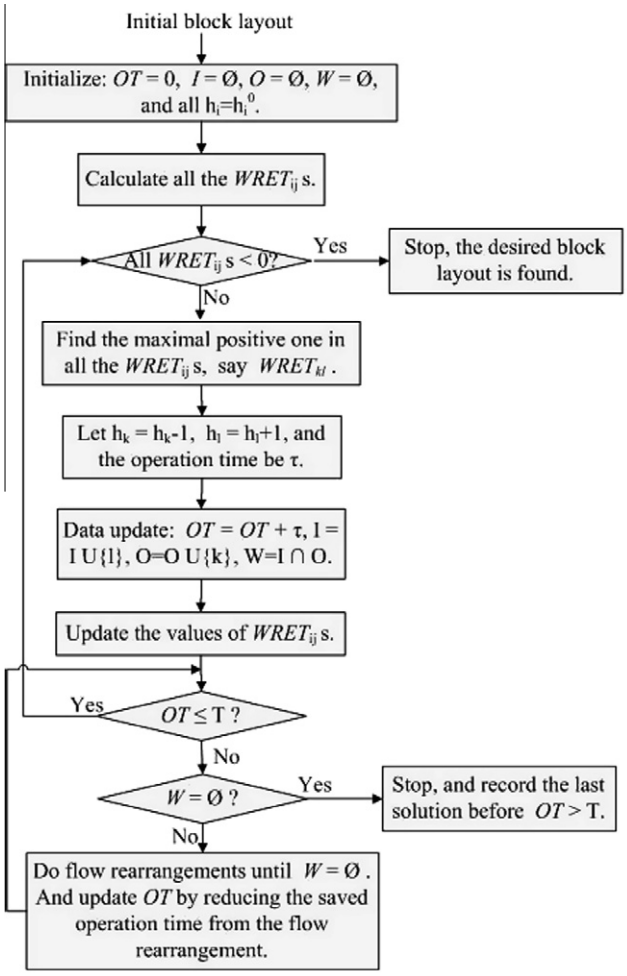

Fig. 5. Outline of the greedy algorithm.

Proof. It is sufficient to prove that it is not optimal to move a certain container into and out of a bay in the same re-marshaling period. Suppose that in one solution, there is a container being moved from a starting bay to bay *i*, and then moved from bay *i* to the desired bay. We can rearrange to move the container directly from the starting bay to the desired bay. The rearrangement will cut the re-marshaling operation time, specifically, by reducing the container loading/unloading time at bay *i*. Hence, whenever there are containers being moved into and out of a bay in the same period, we can do the container flow rearrangement and save more re-marshaling time capacity to improve the block layout. □

Considering Property 1, in the heuristic algorithm, if there exist bays with containers flowing in and out, we can do the flow arrangement to save more time for the re-marshaling operation. The outline of the greedy algorithm is provided in Fig. 5. In the algorithm description, the following variables are used:

OT	: the cumulative re-marshaling operation time.
I	: the set of bays with containers flowing in.
O	: the set of bays with containers flowing out.
$W = I \cap O$	: the set of bays with containers flowing in and out.

6. Computational experiments

In this section, we report our computational experiments. The experiment results validate our model and solution approaches in the following five aspects: (1) we demonstrate how the block utilization level may affect the average expected container retrieval time; (2) we analyze how the inbound container space allocation schemes affect the ASC travel time in the container unloading process; (3) we show the advantage of the multiple-period segregation model (P3) over the single-period segregation model (P2); (4) we illustrate the benefits of re-marshaling; (5) we compare the long-term performance of segregation allocation policy and the non-segregation allocation policy, especially on the possibility of block overflow.

Based on our consulting experience with a local container terminal, we set the parameters in our experiments in Table 1.

6.1. The impact of block utilization

We first demonstrate how the average container retrieval time may be affected by block utilization, or the number of containers in a block. We show this under two models, the non-segregation model (P1) and the single-period segregation model (P2). The experiment is designed to cover different numbers of initial containers, different numbers of empty bays, and different numbers

Table 1
Parameters setting for computational experiments.

Parameters	Values
Number of bays in a block	50
Number of rows in a bay	10
Container length	20 ft = 6.1 m
ASC load time	0.5 min
ASC unload time	0.5 min
ASC gantry travel speed (empty move)	225 m/min
ASC gantry travel speed (loaded move)	146 m/min
ASC operation time of one rehandle	1.1 min

of arriving inbound containers, reflecting the different block utilization rates. For (P1), the initial number of containers is in {700, 900, 1100, 1300}, and for (P2), the number of empty bays is in {24, 22, 20, 18}. The results are shown in Fig. 6.

Fig. 6 clearly shows that the average retrieval time increases with the block utilization, which is quite straightforward. The more interesting observation is that the retrieval time increase tends to be a concave function within a reasonable region of arrival containers. In other words, when the arrival container number is not too large, the retrieval time increases at a rate lower than the increase of the container number. This is evidence of the benefit of taking an optimized block allocation scheme.

Note that Fig. 6 shows both the segregation model and non-segregation model. The purpose is to understand the impact of the block utilization on the container retrieval process, rather than directly compare the relative advantage between the two models. We will address the comparison between them in Section 6.5 via a long-term simulation.

6.2. The impact of space allocation on the ASC operation efficiency

Our model concentrates on the minimization of container retrieval time or ET waiting time. Another important performance factor in container terminal operations is the ASC efficiency in a block. We now investigate how minimizing the retrieval time may affect the ASC operation efficiency. We do this from two perspectives, in the vessel discharge process and in the container retrieval process.

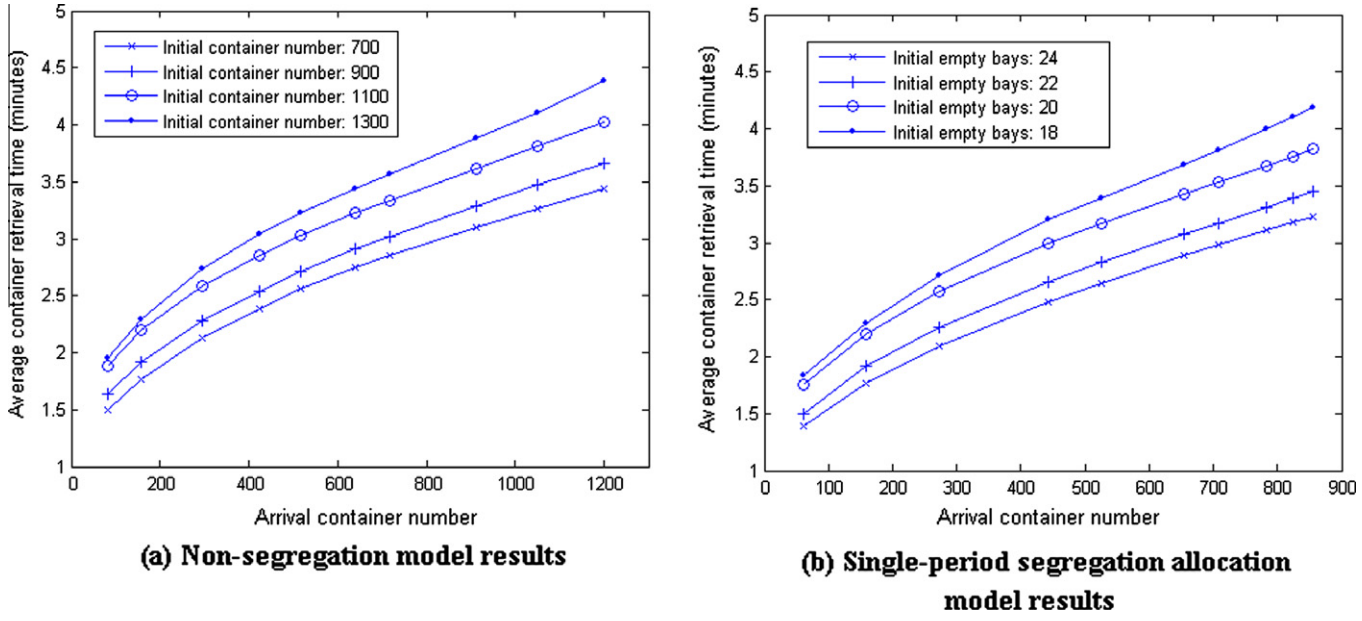

Fig. 6. Average container retrieval time under different block utilizations.

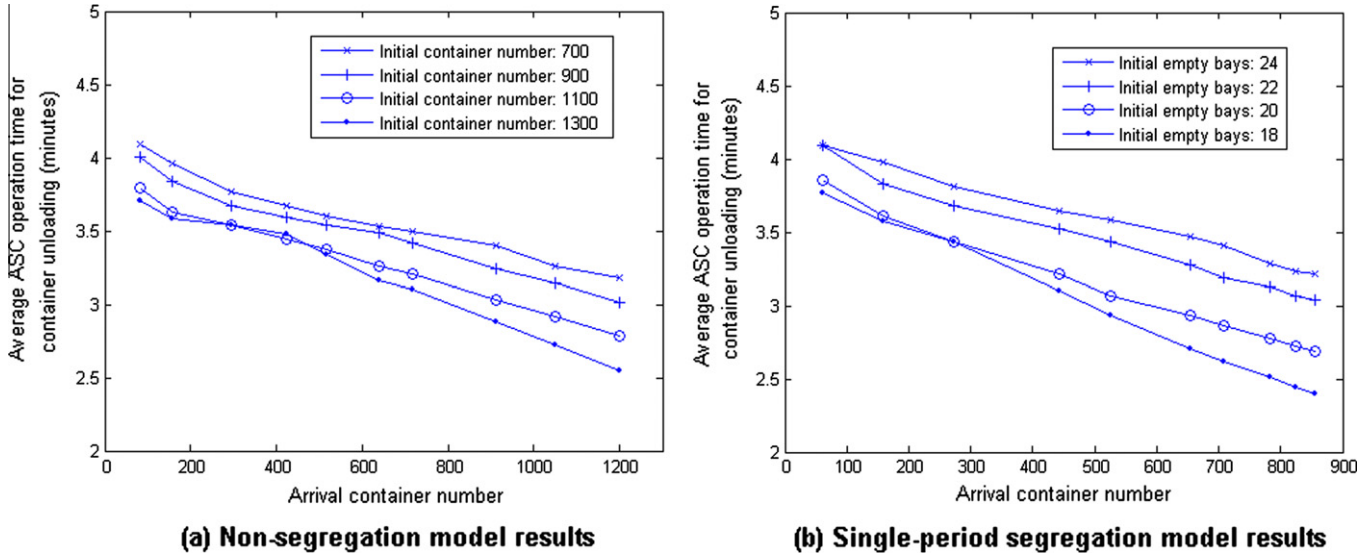

Fig. 7. Average ASC operation time in the vessel discharge process.

In Fig. 7, we report the average ASC operations time in the vessel discharge process, i.e., the average time of putting a container to its designated position, under the same setting as Fig. 6. It can be intuitively argued that putting containers to designated positions may result in a longer ASC operations time in the vessel discharge process, because the positions are designed to facilitate the retrieval process. However, from Fig. 7, we can observe that the average ASC operation time in the vessel discharging process decreases when the number of arrival containers increases. This shows that the layout design for container retrieval may have negative effect on the vessel discharging process, but the impact is limited when there are a large number of containers.

In Fig. 8, we look into the retrieval process, and break down the average ASC operations time into ASC travel time and ASC rehandle time. As shown in the figure, both times increase with the number of containers (the two figures on the left side), which is expected. A less intuitive result can be revealed if we show the percentage time spent on these two operations (the two figures on the right side), where we can see that the ASC will spend relatively more and more time on rehandles but less and less time on travels when the number of containers increase. This shows the tradeoff effect between the ASC travel time and rehandle time when the total ASC operations time is being minimized.

6.3. Importance of multiple-period planning

We now show the importance of multiple-period planning for the segregation policy by comparing the single-period segregation model (P2) and the multiple-period segregation model (P3). We do this by running simulations for 3 months. At the beginning of the simulation, the block is empty, then inbound containers arrive and are collected day by day. The daily container arrival number is a uniform random variable with mean of {80, 100, 120}, and the daily retrieval number is a percentage, in {30%, 40%, 50%}, of the existing containers in the block, regardless how long a container has been in the block. The first three days are a warm-up period to fill up

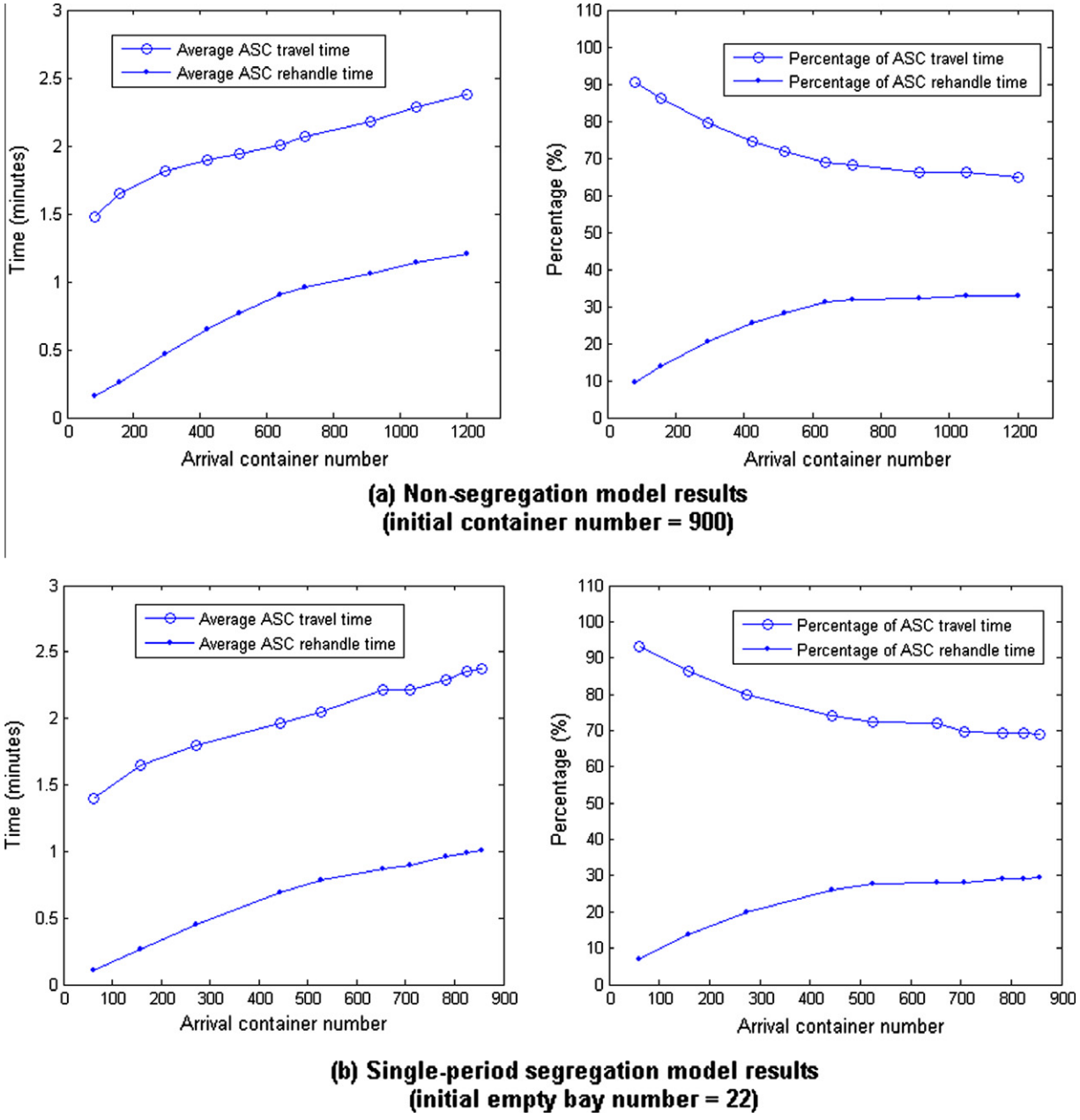

Fig. 8. Average ASC travel time and average ASC rehandle time for container retrieval.

the empty block. After the warm-up period, container retrieval starts.

Table 2 shows that the multiple-period segregation model has a lower average retrieval time than the single-period model in all scenarios. We have also calculated the standard deviations of the retrieval time. With a lower standard deviation, the multiple-period model can provide a more stable service level to the customers.

The column "saving" in the table gives the relative time saving of the multiple-period model, where we can conclude that the multiple-period model has a higher advantage when the block utilization is high, i.e., with more number of arrival container and low retrieval rates.

6.4. The effectiveness of re-marshaling

In order to investigate the effect of the overnight re-marshaling operations, we compare the system performances of the multiple-period segregation models with different re-marshaling operations time limit. We conduct simulation for 3 months. The daily container arrival number is a uniform random variable with mean of 80, and the daily retrieval number is 30% of the existing containers in the block, regardless how long a container has been in the block. The first three days are a warm-up period to fill up the empty block. After the warm-up period, container retrieval starts.

Fig. 9 shows that the re-marshaling operations can improve the system performances. With the increase of the re-marshaling time limit (T), the average container retrieval time decreases. When the re-marshaling time limit is large enough, the block layout achieves optimum and the average container retrieval time becomes stable. Such results can provide some guidance for determining the re-marshaling time to be balanced with the re-marshaling benefit.

6.5. Comparison of segregation and non-segregation allocation policies

Both segregation and non-segregation allocation policies are used in practice (Kim and Kim, 1999; Kim and Kim, 2007). To obtain some insights on their relative advantages and disadvantages in the automatic container terminal, we have done a simulation to compare the two policies. The simulation has considered different scenarios of block utilization, which are characterized by the process of container arrivals and retrievals. Specifically, we assume that daily container arrival number is a uniform random variable with mean of {50, 80, 100, 120}, and the daily retrieval number is a percentage, in {10%, 15%, 20%, 25%}, of the containers in the block, 40%, 50%}, of the existing containers in the block, regardless how long a container has been in the block. At the beginning, the block is empty, and after the warm-up period of three days, the retrieval process starts.

We report the simulation results in Table 3, where we list both the average container retrieval time and the occurrence of block overflow. Note that a block overflow occurs when there is no space for an arriving container. Under the non-segregation policy, overflow occurs only when the number of existing containers is equal to the block capacity, which seldom happens; but under the segregation policy, overflow may occur when there are no new empty bays, which may happen when the block utilization is high.

In general, if the existing containers in the block are randomly retrieved, non-segregation allocation policy outperforms segregation allocation policy. This is because the non-segregation allocation policy produces a smoother block layout. Segregation policy is only a feasible solution of the non-segregation policy, hence the optimal block layout obtained from the non-segregation policy is at least no worse than the one from the segregation policy. The percentage saving in column "saving" is the differences that the non-segregation policy's average container retrieval time is shorter

Table 2
Comparison of single-period and multiple-period segregation models.

Average arrival container number	Retrieval percentage	Average retrieval time (min)			Standard deviation (min)	
		Multiple	Single	Saving (%)	Multiple	Single
80	30	1.92	2.07	7.25	0.08	0.14
80	40	1.74	1.86	6.45	0.09	0.17
80	50	1.63	1.7	4.12	0.10	0.16
100	30	2.05	2.26	9.29	0.10	0.20
100	40	1.84	2.01	8.46	0.11	0.18
100	50	1.72	1.85	7.03	0.12	0.18
120	30	2.16	2.40	10.0	0.10	0.19
120	40	1.93	2.11	8.53	0.11	0.21
120	50	1.78	1.93	7.77	0.10	0.20

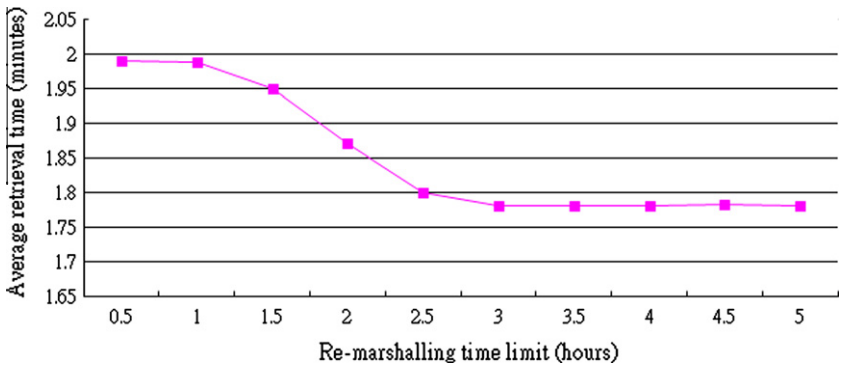

Fig. 9. The benefit of re-marshaling.

Table 3
Comparison of segregation and non-segregation allocation policies.

Average arrival container number	Retrieval percentage	Average retrieval time (min)			Number of overflows	
		Seg.	Non-seg.	Saving	Seg.	Non-seg.
50	10	2.60	1.86	28.5	0	0
50	15	2.21	1.68	24.0	0	0
50	20	1.94	1.57	19.1	0	0
50	25	1.81	1.49	17.7	0	0
80	10	2.84	2.32	18.3	5	0
80	15	2.56	2.11	17.6	0	0
80	20	2.28	1.93	15.4	0	0
80	25	2.06	1.81	12.1	0	0
100	10	2.88	2.43	15.6	14	0
100	15	2.74	2.32	15.3	0	0
100	20	2.42	2.12	12.4	0	0
100	25	2.20	1.97	10.5	0	0
120	10	3.07	2.65	13.7	16	0
120	15	2.94	2.56	12.9	2	0
120	20	2.58	2.29	11.2	0	0
120	25	2.33	2.13	8.6	0	0

than that of the segregation policy. Table 3 reveals that, the more congested the block, the better the non-segregation policy is than the segregation one. Considering the performances of these two policies under different container arrival and retrieval patterns, we suggest using non-segregation allocation policy when the block is congested.

7. Conclusion

The inbound container operation in a terminal has not gained enough attention in the academic area, although it directly affects the efficiency improvement in a container terminal with respect to the service level for customers. In this paper, we focus on the inbound container retrieval efficiency improvement in a modern automatic container terminal, where container blocks are perpendicular to the berth/gate and the blocks' interfaces to the ETs and ITs are restricted at the two ends. This kind of modern automatic container terminal has been used in practice in recent years, but is not yet adequately studied in the literature.

We develop models and algorithms to improve the inbound container retrieval efficiency by (1) optimally allocating the arrival inbound containers so as to minimize the expected container retrieval time, and (2) implementing an overnight re-marshaling operation to improve the block layout after the container retrieval processes during the day time. Both the segregation allocation policy and non-segregation allocation policy are considered. We show that the multiple-period segregation model is better than the myopic
single-period segregation model. In addition, when the block is congested, the non-segregation allocation policy outperforms the segregation allocation policy, where extra re-arrangements are needed to vacate more empty bays and prevent the overflow in the segregation allocation policy. These results may provide some managerial and operational references for the container terminal managers.

This is just a startup for studying the inbound container space allocation problem in such automatic container terminals. Many open issues provide opportunities for future research. We can consider heterogenous inbound containers (size, weight) and relax the assumption that each container has the same probability of being collected by an ET. For the multiple-period segregation allocation problem, a more challenging extension is to treat containers from different periods separately and design an optimal allocation layout so as to minimize the total container retrieval time.

Acknowledgements

The authors are grateful to the editor and three anonymous referees for their constructive comments on an earlier version of this paper. The work described in this paper was partially supported by a grant from the Research Grants Council of the HKSAR, China, T32-620/11.

Appendix A

Explanation 1. In Kim and Kim (1999), they estimate the total number of rehandles to retrieve all containers in a bay as $R(h \cdot s) = h \cdot s \cdot (h-1)/4 + h \cdot (h+2)/16$, where the average height of the bay is h, the number of rows in the bay is s and the total number of containers in the bay is $h \cdot s$. In our paper, we assume that the number of containers in a bay is given as x, where we have $x = h \cdot s$ and $h = x/s$. Hence, the total number of rehandles to retrieve all containers in a bay can be written as $R(x) = \left(\frac{1}{4s} + \frac{1}{16s^2}\right)x^2 + \left(\frac{1}{8s} - \frac{1}{4}\right)x$.

Proof of Lemma 1. When $x \leqslant s$, it is obvious that $R(x) = 0$. So we only consider the case when $s < x < 2s$. A bay can be denoted by (x,y), showing that there are x containers in the first tier and y containers in the second tier ($y \leqslant x$). Let $H(x,y)$ be the expected number of rehandles to retrieve all containers in bay (x,y). As shown in Fig. A.1, when there are s and y containers in the first and second tiers, respectively, there are three possible configurations caused by retrieving one container in (x,y): (1) if the retrieved container is in the first tier and not buried, then the configuration is changed to $(s-1,y)$ and 0 rehandle is needed; (2) if the retrieved container is in the second tier, the configuration is changed to $(s,y-1)$ and 0 rehandle is needed; and (3) if the retrieved container is buried in the first tier, then the configuration is changed to $(s-1,y)$ and 1 rehandle is needed. Hence, we have the expression of $H(s,y)$ as,

$$H(s,y) = \frac{s-y}{s+y}[0 + H(s-1,y)] + \frac{y}{s+y}[0 + H(s,y-1)] + \frac{y}{s+y}[1 + H(s-1,y)] = \frac{y}{s+y} + \frac{s}{s+y}H(s-1,y) + \frac{y}{s+y}H(s,y-1), \quad 1 \leqslant y < s. \tag{A.1}$$

When $1 \leqslant x < s$ and $0 \leqslant y \leqslant x$, the configuration of the bay is shown in Fig. A.2. Similar to (A.1), we have, when $1 \leqslant x < s$ and $0 \leqslant y \leqslant x$,

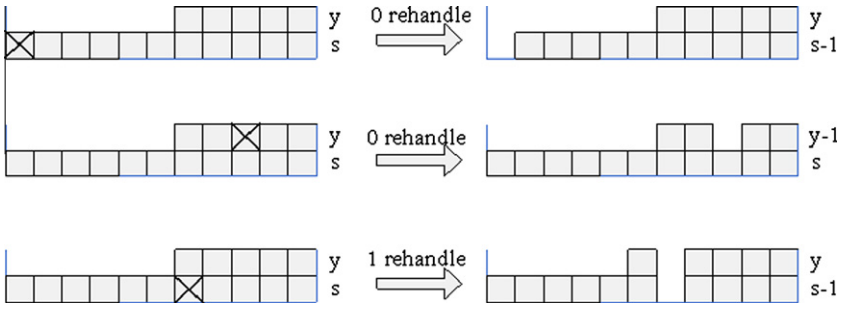

Fig. A.1. Bay configuration change.

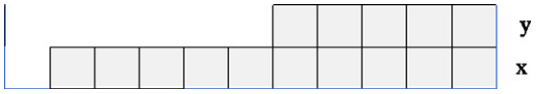

Fig. A.2. Bay configuration when the first tier is not full.

$$H(x,y) = \frac{x-y}{x+y}[0+H(x-1,y)] + \frac{y}{x+y}[0+H(x,y-1)] + \frac{y}{x+y}[1+H(x,y-1)] = \frac{y}{x+y} + \frac{x-y}{x+y}H(x-1,y) + \frac{2y}{x+y}H(x,y-1). \quad (A.2)$$

By substituting $H(x,0) = 0$ into the above recursion relation, we can get

$$H(x,y) = y/2, \quad 1 \leqslant x < s, 0 \leqslant y \leqslant x. \quad (A.3)$$

By (A.1) and (A.3), we can get the recursion equations

$$H(s,y) = \tfrac{y}{s+y} + \tfrac{sy}{2(s+y)} + \tfrac{y}{s+y}H(s,y-1),$$
$$H(s,y-1) = \tfrac{y-1}{s+y-1} + \tfrac{s(y-1)}{2(s+y-1)} + \tfrac{y-1}{s+y-1}H(s,y-2),$$
$$\ldots$$
$$H(s,1) = \tfrac{1}{s+1} + \tfrac{s}{2(s+1)} + \tfrac{1}{s+1}H(s,0),$$
$$H(s,0) = 0.$$

Solving the above $y+1$ equations, we can get the general equation

$$H(s,y) = \frac{y(s+2)}{2s+2}, \quad 1 \leqslant y < s. \quad (A.4)$$

So, when there are x containers in the bay ($s+1 \leqslant x < 2s$), we have $y = x - s$. Substitute $y = x - s$ into (A.4), we get the expected rehandle number to retrieve all containers:

$$R(x) = \frac{s+2}{2s+2}x - \frac{s(s+2)}{2s+2}, \quad s+1 \leqslant x < 2s. \quad \square$$

References

Ahuja, R.K., Magnanti, T.L., Orlin, J.B., 1993. Network Flows – Theory, Algorithms and Applications. Prentice-Hall, Inc., pp. 543–565.

Bortfeldt, A., 2004. A heuristic for the container pre-marshaling problem. In: Proceedings of the 3rd International Conference on Computer and IT Applications in the Maritime Industries 2004, pp. 419–429.

Bortfeldt, A., Forster, F., 2012. A tree search procedure for the container pre-marshaling problem. European Journal of Operational Research 217 (3), 531–540.

Caserta, M., Voß, S., 2009. A corridor method-based algorithm for the pre-marshaling problem. In: Proceeding of the EvoWorkshops 2009, pp. 788–797.

Caserta, M., Schwarze, S., Voß, S., 2012. A mathematical formulation and complexity considerations for the blocks relocation problem. European Journal of Operational Research 219 (1), 96–104.

Castilho, B.D., Daganzo, C.F., 1993. Handling strategies for inbound containers at marine terminal. Transportation Research Part B 27 (2), 151–166.

Choe, R., Park, T., Oh, M.-S., Kang, J., Ryu, K.R., 2009. Generating a rehandling-free intra-block remarshaling plan for an automated container yard. Journal of Intelligent Manufacturing 22 (2), 201–217.

Günther, H.O., Kim, K.H., 2006. Container terminals and terminal operations. OR Spectrum 28 (4), 437–445.

Ioannou, P.A., Kosmatopoulos, E.B., Jula, H., Collinge, A., Liu, C.I., Asef-Vaziri, A., Dougherty, E., 2000. Cargo Handling Technologies. Technical Report, Department of Electrical Engineering, University of Southern California.

Jiang, X., Lee, L.H., Chew, E.P., Han, Y., Tan, K.C., 2012. A container yard storage strategy for improving land utilization and operation efficiency in a transshipment hub port. European Journal of Operational Research 221 (1), 64–73.

Kim, K.H., 1997. Evaluation of the number of rehandles in container yards. Computers and Industrial Engineering 32 (4), 701–711.

Kim, K.H., Bae, J.W., 1998. Re-marshaling export containers in port container terminals. Computers and Industry Engineering 35 (3–4), 655–658.

Kim, K.H., Kim, H.B., 1999. Segregating space allocation models for container inventories in port container terminals. International Journal of Production Economics 59 (1–3), 415–423.

Kim, K.H., Kim, H.B., 2002. The optimal sizing of the storage space and handling facilities for import containers. Transportation Research Part B 36 (9), 821–835.

Kim, K.H., Kim, K.Y., 2007. Optimal price schedules for storage of inbound containers. Transportation Research Part B 41 (8), 892–905.

Kim, K.H., Lee, J.S., 2006. Satisfying constraints for locating export containers in port container terminals. In: Computational Science and Its Application/ICCSA 2006, PT3, 3982, pp. 564–573.

Kim, K.H., Park, K.T., 2003. A note on a dynamic space-allocation method for outbound containers. European Journal of Operational Research 148 (1), 92–101.

Kim, K.H., Park, Y.M., Ryu, K.R., 2000. Deriving decision rules to locate export containers in container yards. European Journal of Operational Research 124 (1), 89–101.

Lee, Y., Hsu, N.Y., 2007. An optimization model for the container pre-marshaling problem. Computers and Operations Research 34 (11), 3295–3313.

Lee, Y., Chao, S.L., 2009. A neighborhood search heuristic for pre-marshaling export containers. European Journal of Operational Research 196 (2), 468–475.

Stahlbock, R., Voß, S., 2008. Operations research at container terminals: a literature update. OR Spectrum 30 (1), 1–52.

Steenken, D., Voß, S., Stahlbock, R., 2004. Container terminal operation and operations research – a classification and literature review. OR Spectrum 26 (1), 3–49.

Zhang, C., Chen, W., Shi, L., Zheng, L., 2011. A note on deriving decision rules to locate export containers in container yards. European Journal of Operational Research 205 (2), 483–485.