Dynamic Stochastic Optimization of Relocations in Container Terminals

Setareh Borjian [*] Vahideh H. Manshadi [†] Cynthia Barnhart [‡] Patrick Jaillet [§]

June 25, 2013

Abstract

In this paper, we present a mathematical model and formulation to minimize the number of container relocations in storage systems, like container terminals and warehouses. Container relocations are necessary when a container to be retrieved is not at the topmost position in a stack. We consider a dynamic setting, which allows for continual stacking and retrieving of containers, without restrictive assumptions on container arrival and departure sequences, and on possible relocations. We generalize our objective function to include operational delays due to relocations, and jointly minimize the number of relocation moves and delay. We apply our method to the special case of only container retrievals, and compare our results with those of existing approaches.

Further, we study the impact of uncertainty on the number of relocations, by extending our formulation to the setting with incomplete information on stacking and retrieval times. Using a two-stage stochastic optimization framework, we show that lack of information results in more relocations; and having partial information significantly closes the gap to the optimal solution with complete information.

1 Introduction

A container terminal is a facility where the import/export/transshipment cargo containers arrive and are accommodated before they are distributed to nearby cities or loaded to vessels for onward transportation. As critical international logistics nodes, container terminals facilitate the flow of goods and therefore play an important role in regional and national economies.

An overview of the operations in a container terminal is as follows. On the seaside, containers are unloaded from (or loaded to) vessels using huge berth cranes, and on the landside, they are

[*]CEE, MIT. Email: sborjian@mit.edu.
[†]EECS and ORC, MIT. Email: manshadi@mit.edu.
[‡]CEE, MIT. Email: cbarnhar@mit.edu.
[§]EECS and ORC, MIT. Email: jaillet@mit.edu.

stacked in (or retrieved from) the storage yard using smaller mobile cranes. Internal trucks transfer containers from one point to another inside the terminal, whereas external trucks transfer the containers from storage yard to the city, or vice versa.

In a typical container terminal, the storage yard is the area in which containers are stacked temporarily before they are delivered to external trucks or loaded on a vessel. The storage yard is usually divided into several bays. There are several columns in each bay, and containers are stacked in tiers to form columns. The slot occupied by a container in a bay, can be specified by the tier and the column in which the container is stacked (see Figure 1).

The container relocation problem arises as a result of stacking containers on top of each other; it often happens that the target container, which needs to be retrieved from the bay, is covered by other containers. In this case, some containers need to be relocated, and such relocations are considered non-productive moves, which are costly. In particular, two types of cost stem from relocations. First, there is the cost associated with performing the relocations, which cannot be charged and second, there are the costs associated with resulting delays in delivering the containers to external trucks and a lower quality of customer service.

In a container terminal, the number of relocations compared to the total number of moves is an important measure of the efficiency of the yard management. Thus, decreasing this ratio is of great practical interest, and it is a key objective in yard operations. The question that yard managers face is as follows: given the estimated arrival and departure times of the containers (in a certain period of time), how to stack, relocate, and retrieve the containers to minimize the number of relocations while stacking and retrieving each container within a reasonable level of delay?

The relocation problem referred to as the Container Relocation Problem (CRP) or the Block Relocation Problem (BRP) (e.g Caserta et al. [2012, 2011], Wan et al. [2009]), is defined as follows: A certain number of containers are initially stored in a bay. Given a fixed departure order that is known in advance, the objective is to retrieve all the containers with the minimum number of moves. It takes a single move to retrieve a container if it is on the topmost tier. However, if containers are on top of the target one, they must be relocated to other locations to access the target container. Thus, the problem is to find the sequence of moves (relocations and retrievals) that results in the minimum number of relocations while retrieving containers in the predefined order. Such a setting is "static" in the sense that it assumes the containers have already been stacked and no additional containers arrive (and thus need to be stacked) during the retrieving period. However in practice, stacking and retrieving periods overlap; at the same time that arriving containers are being stacked in a bay, some other containers are claimed by external trucks and are being retrieved. We refer to this practical setting as "dynamic" and to the associated relocation problem as the Dynamic Container Relocation Problem (DCRP). Current mathematical programming formulations Caserta et al. [2012], Wan et al. [2009] do not capture the dynamic aspects. In Wan et al. [2009], the heuristics designed based on the static IP formulation are applied to a dynamic setting.

In this paper, we present the first formulation and solution approach for the DCRP that al-

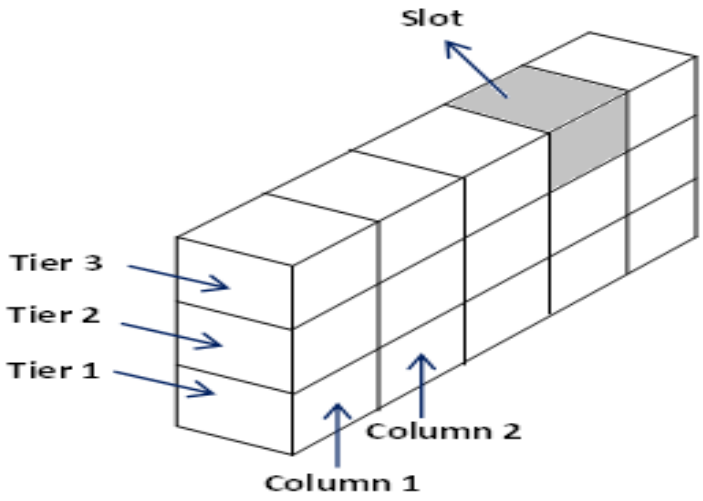

Figure 1: Illustration of a bay

lows for continual container stacking and retrieving at arbitrary times. Our dynamic formulation solves this problem in a general form, i.e., without restrictive assumptions on container arrival and departure sequences, and on possible relocations. (A restrictive assumption in many of the existing models and formulations is that only the containers that are above the target container are allowed to be relocated.). Inputs to our formulation include container arrival times, departure times, service time windows for stacking and retrieving each container, and the initial configuration of the bay. The stacking time window (retrieval time window) is the maximum delay allowed in stacking (retrieving) the container from the time it arrives (is ready for departure). We define discrete time steps, with a time step representing the minimum time needed to complete one move (that is, stacking, relocating, or retrieving). We minimize the number of relocation moves subject to the following constraints: i) each container is stacked or retrieved within its allowed window; ii) at most one move takes place in each time step; and iii) at each step, the block configuration is feasible (e.g., no container is floating, or stacked in an occupied slot, etc.). Our Integer Program (IP) formulation and solution approach is able to solve real-sized instances with continual container arrivals and departures (See Section 2).

Our dynamic formulation is also capable of jointly optimizing the number of relocations and the delays in stacking and retrieving. The motivation to this approach is that most of the time, minimizing the relocations while retrieving the containers according to their departure order, is achieved at the expense of large amounts of delay. Further, our formulation can also solve the static CRP (i.e., when all the containers are stacked before the retrieving period), as a special case with the given flexibility that we can deviate from the predefined retrieving order. We show that our model results in fewer relocations, and less operations delay compared to existing formulations (for instance, Caserta et al. [2012]).

Our other main contribution is to initiate the formalization of the CRP with incomplete information; our dynamic model assumes complete information on container arrival and departure times. However, in practice, we often only have rough estimates of arrival times, and only a few hours ahead of the arrivals. Further, in terminals without an appointment system, the arrival of the external trucks, and hence, the container departure times are highly uncertain. We extend our mathematical formulation to the setting where we only have partial information on the departure times. In particular in a bay with N containers at the start of the retrieving process, we know the departure times of the first N_1 containers that need to be retrieved; but we only have partial information about the departure time of the remaining (See Subsection 3.1 for the uncertainty models). At a given (later) time, the exact departure times of the remaining containers are revealed. For this setting, we present a two-stage stochastic optimization model that jointly minimizes the expected number of relocations and delay by considering all possible departure scenarios with the binding constraint on the decisions made in the first stage. Using this optimization framework, we study the effect of uncertainty (or lack of complete information) on the number of relocations as well as the value of partial information and the effect of the information revelation time. Using the example of a small bay, we show that a lack of information results in a significant increase in the number of relocations. Also we show that partial information such as knowing the departure order of groups of containers (instead of knowing the order of all containers) can be very valuable in the sense that it significantly reduces the number of relocations compared to the case of not having any information.

Further, solution of our IP-based formulation provides a benchmark to evaluate the performance of heuristics for the CRP with incomplete information. The run time of our program exponentially increases with the size of the bay as well as with the lack of information (the number of scenarios in our formulation grows exponentially with the number of containers whose departure times are unknown at the beginning). This motivates us to define heuristics based on our formulations. In particular, we propose the following heuristics: in our two-stage program, we first *ignore* the departure times of the last N_2 containers and only consider different scenarios of departure times for the smaller number of $N - N_1 - N_2$ containers. Then at the time that the exact departures are revealed, we re-solve the problem, as a simple CRP with a given initial configuration. We present the computational results for this heuristic and compare them with the solution to the stochastic CRP.

The structure of our paper is as follows. In the rest of this section, we review the literature on the container relocation problem. In Section 2, we describe the dynamic container relocation problem and present our model, formulation, and numerical results. In Section 3, we formalize the container relocation problem under uncertainty, presenting a stochastic optimization model as well as a heuristic approach for the problem with incomplete information. In Section 4, we present and compare the simulation results of the stochastic model and the heuristic. Finally in Section 5, we summarize the results, mention the possible broader application of our work, and suggest a few

directions for future work.

1.1 Related Work

Literature on the relocation problem in container terminals follows two main directions. The first approach is to design heuristics and to evaluate their performance by applying them to a set of random instances. In this direction, Caserta et al. [2009], develop a binary description of the stacking area and apply a four-step heuristic to retrieve all the containers from the bay. Lee and Lee [2010] present a three-phase heuristic to retrieve all the containers from a given yard according to a given order with the aim of minimizing the number of relocations, as well as the working time of the cranes. Other researchers study the stacking problem, i.e., where to put an arriving container, which can also be extended to the relocation problem. For example, Kim et al. [2000] propose a methodology to determine the stacking location of an arriving export container considering its weight. In Borgman et al. [2010], the stacking location is determined by the estimated departure time and the distance from the exit point, while Dekker et al. [2007] simulate different stacking policies for containers in automated terminals.

The second approach is to develop a mathematical programming formulation for the problem, and either solve the program or use it to design heuristics. Kim and Hong [2006] propose a branch-and-bound algorithm and a decision rule that uses an estimation of additional relocations to solve their program. They compare the performance of the decision rule with that of the branch-and-bound algorithm. The authors in Wan et al. [2009] propose an IP formulation and several IP-based heuristics, and compare them with other heuristics in the literature. Also, they apply heuristics designed based on the static IP formulation to a dynamic setting. In a recent paper, Caserta et al. [2012] propose two IP formulations of the problem and show that the CRP is NP-hard. Their first IP provides an optimal solution to the CRP without any restrictive assumptions on the moves, but its computational time is very long. The second formulation requires less computation time but has the restrictive assumption dictating that only the containers on top of the target container can be relocated. They compare the solution of their second IP with those of the heuristic proposed in Kim and Hong [2006], and show that their solution outperforms the heuristic for many instances.

The literature on CRP with incomplete information is limited. Zhao and Goodchild [2010] use simulation to evaluate the use of truck arrival information to reduce relocations. They generate various initial configurations and different levels of information to study the impact of information quality and bay configuration on the number of relocations. They compare the performance of their heuristics to another simple heuristic rule that if a container is on top of the target container, it is relocated to the nearest unoccupied slot. They show that their heuristics outperform this rule. In this paper, we develop an IP based approach to the CRP with incomplete information. Its solution provides the sequence of moves that minimizes the expected number of relocations and delay, for each possible departure order scenario. Such a solution also provides a benchmark to evaluate the

performance of any heuristic.

Our contribution to the literature is two fold. First, we present the first IP formulation for the dynamic CRP. Our formulation is general, in that it does not contain restrictive assumptions on possible moves. Second, we develop a stochastic optimization framework for the CRP with incomplete information. The solution to the stochastic problem provides an optimal sequence of decisions to the port decision makers ; and is a benchmark for evaluating the performance of heuristics for the CRP with incomplete information.

2 The Dynamic Container Relocation Problem

In this section, we formally define the Dynamic Container Relocation Problem (DCRP) and present a mathematical model for it. Suppose we are given a bay with C columns and P tiers. In a given time horizon (say a day), N containers need to be stacked in and retrieved from this bay. We consider discrete time steps, where one time step is the minimum time needed to complete one move, which can be stacking, relocating, or retrieving one container. For example, if each move takes five minutes and we want to model the process for one hour, there would be 12 time steps. For each container n, we are given its arrival time (a_n), its departure time (d_n), its service time window for stacking (α_n), and its service time window for retrieving (δ_n). We assume that all $a_n, d_n, \alpha_n, \delta_n$ are integer numbers, and $a_n < d_n$. To keep the setting general, we assume that some containers might initially be in the bay at time 0; if container n is initially in the bay, we let $a_n = \alpha_n = 0$. Also we allow some containers to stay in the bay arbitrarily long by setting their departure time to infinity.

The stack and retrieval service time windows are set by the port operators, and are usually prioritized differently. The retrieval and delivery of containers to external trucks are more sensitive to delay than the stacking of arriving containers, mostly due to customer satisfaction issues and the desire to avoid long queues of external trucks.

Let T to be the total number of time steps required to complete the entire set of tasks (stacking the arriving containers and retrieving the departing containers). Because each task has a due time determined by its service time window, we can determine T by:

$$T = \max_{1 \le n \le N,\ 1 \le m \le N,\ d_m < \infty} \{a_n + \alpha_n, d_m + \delta_m\} \tag{1}$$

Our objective is to find the sequence of *feasible* moves at time steps $1, 2, \ldots, T$ that jointly optimizes the number of relocations and delays in stacking and retrieving containers. Note that in each time step t, the crane might be idle, i.e., might perform no move. In the following subsection, we provide an integer Program (IP) formulation for the DCRP in its general form.

2.1 DCRP formulation

We index the slots of the bay by (i, j) where $1 \leq i \leq C$ represents the column and $1 \leq j \leq P$ represents the tier (for relocation moves that involve two slots, we use (k, l) for indexing the second slot). We use index $1 \leq n \leq N$ for the containers, and index $1 \leq t \leq T$ for the time steps. We define the following sets of variables that enable us to formulate the DCRP as an IP. The set of variables b_{ijnt} (defined in (2)) determine the configuration of the bay at time step t; the set of variables x_{ijklnt} (defined in (3)) define the relocation move within the bay at time t; the set of variables s_{ijnt} (defined in (4)) represent the stacking status and location of an arrived container; similarly the set of variables y_{ijnt} (defined in (5)) represent the retrieval status and location of a departing container; finally z_{nt} and v_{nt} (defined in (6) and (7)) define the events of stacking and retrieving container n at any time before t.

$$b_{ijnt} = \begin{cases} 1 & \text{if container } n \text{ is in } (i,j) \text{ at time } t, \\ 0 & \text{otherwise;} \end{cases} \tag{2}$$

$$x_{ijklnt} = \begin{cases} 1 & \text{if container } n \text{ is relocated from } (i,j) \text{ to } (k,l) \text{ at time } t, \\ 0 & \text{otherwise;} \end{cases} \tag{3}$$

$$s_{ijnt} = \begin{cases} 1 & \text{if container } n \text{ is stacked in } (i,j) \text{ at time } t, \\ 0 & \text{otherwise;} \end{cases} \tag{4}$$

$$y_{ijnt} = \begin{cases} 1 & \text{if container } n \text{ is retrieved from } (i,j) \text{ at time } t, \\ 0 & \text{otherwise;} \end{cases} \tag{5}$$

$$z_{nt} = \begin{cases} 1 & \text{if container } n \text{ has been stacked at time } t' \in \{1,...,t\text{-}1\}, \\ 0 & \text{otherwise;} \end{cases} \tag{6}$$

$$v_{nt} = \begin{cases} 1 & \text{if container } n \text{ has been retrieved at time } t' \in \{1,...,t\text{-}1\}, \\ 0 & \text{otherwise;} \end{cases} \tag{7}$$

Given these variables, we compute the total number of relocations as

$$\sum_{i,k=1}^{C} \sum_{j,l=1}^{P} \sum_{n=1}^{N} \sum_{t=1}^{T} x_{ijklnt}$$

Also note that the earlier we stack (retrieve) container n, the larger the summation $\sum_{t=a_n}^{T} z_{nt}$ ($\sum_{t=d_n}^{T} v_{nt}$). Thus we use the negative of these quantities as measures of delay in stacking and retrieving. To jointly minimize the number of relocations and stacking/retrieving delays, we define parameters w_{rel}, w_s and w_r as the weight factors for relocations, stack delays, and retrieval delays,

respectively. The weight factors are determined by the port operator based on the port policies and sensitivity of the port operations to the number of relocations and total amount of delays. Given these weights, our objective is to minimize:

$$w_{rel} \sum_{i,k=1}^{C} \sum_{j,l=1}^{P} \sum_{n=1}^{N} \sum_{t=1}^{T} x_{ijklnt} - w_s \sum_{n=1}^{N} \sum_{t=1}^{T} z_{nt} - w_r \sum_{n=1}^{N} \sum_{t=1}^{T} v_{nt} \tag{8}$$

The following constraints are necessary to ensure that at each time step, valid moves take place, the configuration of the yard is feasible, and the containers are stacked (retrieved) within the allowable time windows. Constraints (9) ensure that at each time step, each container is either inside the yard, outside the bay waiting to be stacked, or has been delivered to an external truck. Constraints (10) and (11) ensure that each slot is occupied by at most one container, and there are no empty slots between containers in a column. Constraints (12) ensure that at each time step, at most one move takes place. Constraints (13) are the critical constraint that update the configuration of the yard at time step t based on the configuration and the move that took place in the time step $t-1$. Constraints (14) and (15) are the relations between stacking variables and retrieval variables, respectively. Constraints (16) and (19) ensure that containers are stacked or retrieved within their allowable time windows. Constraints (17) and (20) ensure that each container is stacked (or retrieved) only after its scheduled arrival (or departure) time. Constraints (18) and (21) require that containers are not stacked (or retrieved) at some time beyond their allowed time window. Constraints (22) ensure that containers are not relocated within the same column, and are mainly added to reduce the number of variables and potentially improve computation time.

$$\sum_{i=1}^{C} \sum_{j=1}^{P} b_{ijnt} + v_{nt} = z_{nt}, \quad n = 1, ..., N, \quad t = 1, ..., T \tag{9}$$

$$\sum_{n=1}^{N} b_{ijnt} \leq 1, \quad i = 1, ..., C, \quad j = 1, ..., P, \quad t = 1, ..., T \tag{10}$$

$$\sum_{n=1}^{N} b_{ijnt} \geq \sum_{n=1}^{N} b_{ij+1nt}, \quad i = 1, ..., C, \quad j = 1, ..., P-1, \quad t = 1, ..., T \tag{11}$$

$$\sum_{i,k=1}^{C} \sum_{j,l=1}^{P} \sum_{n=1}^{N} x_{ijklnt} + \sum_{i=1}^{C} \sum_{j=1}^{P} \sum_{n=1}^{N} y_{ijnt} + \sum_{i=1}^{C} \sum_{j=1}^{P} \sum_{n=1}^{N} s_{ijnt} \leq 1, \quad t = 1, ..., T \tag{12}$$

$$b_{ijnt} = b_{ijnt-1} - \sum_{k=1}^{C} \sum_{l=1}^{P} x_{ijklnt-1} + \sum_{k=1}^{C} \sum_{l=1}^{P} x_{klijnt-1} - y_{ijnt-1} + s_{ijnt-1}$$
$$i = 1, ..., C, \quad j = 1, ..., P, \quad n = 1, ..., N, \quad t = 2, ..., T \tag{13}$$

$$v_{nt} = \sum_{i=1}^{C} \sum_{j=1}^{P} \sum_{t'=1}^{t-1} y_{ijnt'}, \quad n = 1, ..., N, \quad t = 1, ..., T \tag{14}$$

$$z_{nt} = \sum_{i=1}^{C} \sum_{j=1}^{P} \sum_{t'=1}^{t-1} s_{ijnt'}, \quad n = 1, ..., N, \quad t = 1, ..., T \tag{15}$$

$$\sum_{i=1}^{C} \sum_{j=1}^{P} \sum_{t=a_n}^{a_n+\alpha_n} s_{ijnt} = 1, \quad n = 1, ..., N \tag{16}$$

$$\sum_{i=1}^{C} \sum_{j=1}^{P} \sum_{t=1}^{a_n-1} s_{ijnt} = 0, \quad n = 1, ..., N \tag{17}$$

$$\sum_{i=1}^{C} \sum_{j=1}^{P} \sum_{t=a_n+\alpha_n+1}^{T} s_{ijnt} = 0, \quad n = 1, ..., N \tag{18}$$

$$\sum_{i=1}^{C} \sum_{j=1}^{P} \sum_{t=d_n}^{d_n+\delta_n} y_{ijnt} = 1, \quad n = 1, ..., N \tag{19}$$

$$\sum_{i=1}^{C} \sum_{j=1}^{P} \sum_{t=1}^{d_n-1} y_{ijnt} = 0, \quad n = 1, ..., N \tag{20}$$

$$\sum_{i=1}^{C} \sum_{j=1}^{P} \sum_{t=d_n+\delta_n+1}^{T} y_{ijnt} = 0, \quad n = 1, ..., N \tag{21}$$

$$x_{ijilnt} = 0, \quad i = 1, ..., C, \quad j = 1, ..., P, \quad l = 1, ..., P, \quad n = 1, ..., N, \quad t = 1, ..., T \tag{22}$$

To summarize, our optimization problem is:

$$\min \left\{ \sum_{i,k=1}^{C} \sum_{j,l=1}^{P} \sum_{n=1}^{N} \sum_{t=1}^{T} x_{ijklnt} - w_s \sum_{t=1}^{T} \sum_{n=1}^{N} z_{nt} - w_r \sum_{t=1}^{T} \sum_{n=1}^{N} v_{nt} \right\}$$
$$\text{subject to } (9) - (22)$$

Solution to the above optimization problem gives a sequence of moves for stacking and retrieving containers (including the relocation moves) such that the weighted sum of total delays and relocations is minimized. Given that the deterministic schedule of container departure and arrival times are available, the solution to DCRP minimizes the relocations and total delay. Typically, however, there is significant uncertainty in departure and arrival times. In Section 3, we propose a stochastic optimization model to address such uncertainties. Before proceeding to models with uncertainty, we discuss a special case of our DCRP that solves the static CRP, and compare its performance to that of an existing formulation.

2.2 The special case of the static CRP

Our formulation is a generalization of that of Caserta et al. [2012], in that it can be used to solve the CRP in a dynamic setting while that of Caserta et al. [2012] solves only the static problem. Main difference between our approach and that of Caserta et al. is that we define arrival and departure times and service time windows for each container. This allows flexibility in the stacking and retrieval sequence of containers. Caserta et al. specifies the order of container departures a priori and require strict preservation of that order. As a result, our solutions to the static case require fewer relocations and incur less delay for many instances (below we present some numerical comparisons).

To better understand the difference between the two models, consider a small bay with three columns and two tiers and the given initial configuration illustrated in Table 1. Each container number determines its scheduled departure time and also the order of the departures used in the model of Caserta et al. [2012]. For example, container 3 can be discharged at time step 3 or later.

The optimal sequence of moves for this problem as determined by solving the IP of Caserta et al. [2012] is shown in Table 2. During the retrieval process, container 5 must be relocated twice, which results in a total of two relocations.

1	5	
4	2	3

Table 1: Initial configuration

Now let us evaluate the above solution from our model. Container 1 is retrieved at time step 1, implying that the retrieval delay for container 1 is zero. The second container though, cannot be retrieved at its scheduled time (time step 2), because it must wait for container 5 to be relocated first. Container 2 is thus retrieved at the third time step, resulting in one unit of delay for container 2 and at least one unit of delay for containers 3, 4, and 5. Table 3 shows the delay incurred for the five containers.

In the Caserta et al. solution, the total delay and maximum delay for this example is 6 and 2, respectively. From Table 2, we see that at time 5, container 5 must be relocated to allow access to container 4. However, retrieving container 5 at time 5 is more efficient than relocating it to another slot in order to satisfy the rule that container 4 must be retrieved before container 5. By relaxing the pre-determined departure order for containers 4 and 5, both the number of relocations and total delay are reduced and the retrieval process is accomplished faster. In Table 4, we show the delays for the DCRP solution. Total delay and the total number of relocations are reduced by two and one unit, respectively. Moreover, the total time to retrieve all containers is 6 time steps rather than 7. The shorter completion time has two impacts; first, it improves the efficiency of operations; and second, it allows us to find a tighter bound on T (the number of time steps) using

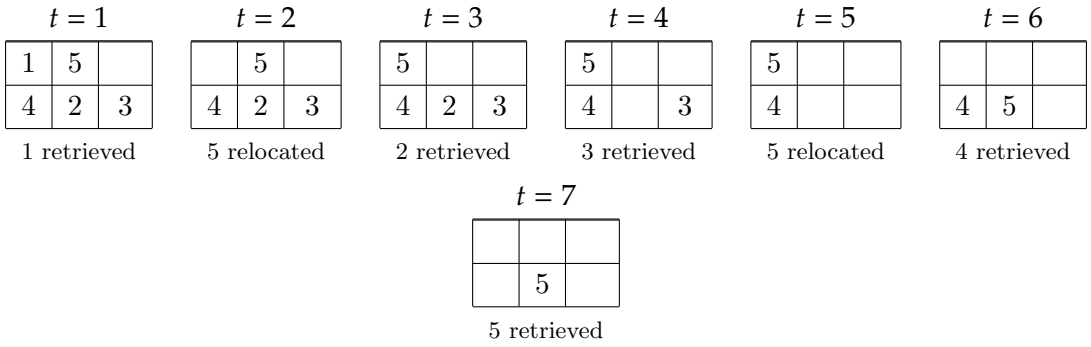

Table 2: Sequence of moves given by IP of Caserta et al. [2012]

Container	Scheduled departure time	Actual departure time	Delay [time units]
1	1	1	0
2	2	3	1
3	3	4	1
4	4	6	2
5	5	6	2

Table 3: Relocations and delay of the solution of IP of Caserta et al. [2012]

Container	Scheduled departure time	Actual departure time	Delay [time units]
1	1	1	0
2	2	3	1
3	3	4	1
4	4	6	2
5	5	5	0

Table 4: Relocations and delay of the solution of DCRP

equation (1).

We use our model to find the number of relocations and amount of delay for a number of instances with initial configurations and container departure times given. We compare our model with the BRP-II model presented by Caserta et al. The solution to BRP-II specifies the minimum number of relocations needed to retrieve all containers in a bay given a specified departure order, and the restrictive assumption that when retrieving a container from the bay, only containers on top of that container can be relocated. This assumption can result in more relocations than necessary (see Example 2 in Caserta et al. [2012]).

We solve the instances presented in Caserta et al. [2012] and summarize in Table 5 the results of our model and compare them to those of BRP-II.

The second, third, and fourth columns in Table 5 show the number of tiers, columns, and total number of containers in the bay, respectively. The first column shows the instance number for the corresponding bay size, as used in Caserta et al. [2012]. The next four columns show the number of relocations and the total delay for each instance, as solved by the DCRP and BRP-II models, respectively. We use the BRP-II formulation in Caserta et al. [2012] to find the number of relocations and delay (note, however, that the number of relocations we report for some instances are different from those obtained in Caserta et al.; see Appendix B for a detailed explanation). We also compute the delay by counting the number of relocations prior to retrieving each container (note that delay is not studied and reported in that paper). For the DCRP model, all weight factors in objective function (8) are set to one. Also for each bay size, the retrieval time window (δ_n) is the same for all containers, and large enough to ensure feasibility. As shown in Table 5, our model (that relaxes the strict ordering of container retrievals) results in fewer relocations and less delay for all instances, compared to the BRP-II model. Our model also results in fewer relocations compared to the heuristic of Kim and Hong [2006] (see Table 1 in Caserta et al. [2012] for the results of Kim and Hong's heuristic).

3 CRP with Incomplete Information

The model and formulation presented in Section 2, as well as those in Kim and Hong [2006], Wan et al. [2009], Caserta et al. [2012], critically rely on the assumption that the arrival and departure times of the containers are known in advance. Such assumptions are quite unrealistic, because in practice terminals often have only rough estimates of container arrival times just a few hours in advance and in terminals without an appointment system, the arrival of the external trucks, and hence, the container departure times are highly uncertain.

In the presence of uncertainty, when making decisions at a time (say time t), we need to consider several possibilities for events happening in the future. For instance, it might happen that at time $t+1$ a new container arrives and needs to be stacked, or an existing container needs to be retrieved. We call each possible set of future events a *scenario*. Given historical data, we construct the set

	Bay setting			DCRP		BRP-II	
instance	tiers	columns	containers	Relocations	Total delay	Relocations	Total delay
1	5	6	18	7	105	11	165
2	5	6	18	4	60	7	98
3	5	6	18	6	101	11	146
4	5	6	18	3	49	7	104
5	5	6	18	1	18	4	46
1	5	7	21	3	47	7	92
2	5	7	21	2	38	10	144
3	5	7	21	6	87	9	126
4	5	7	21	3	51	8	122
5	5	7	21	6	104	12	198
1	5	8	24	6	118	8	165
2	5	8	24	6	112	10	178
3	5	8	24	5	90	9	140
4	5	8	24	5	132	10	185
5	5	8	24	3	60	13	201
1	6	4	16	4	50	10	127
2	6	4	16	3	40	10	103
3	6	4	16	4	53	10	103
4	6	4	16	3	36	7	75
5	6	4	16	5	74	9	113
1	6	5	20	8	141	16	222
2	6	5	20	4	101	10	149
3	6	5	20	8	120	13	181
4	6	5	20	5	77	8	103
5	6	5	20	9	179	16	262

Table 5: Relocations and delay for some instances solved by BRP-II and DCRP

of all possible scenarios and their likelihoods. With these estimates, we can make decisions that jointly minimize the *expected* number of relocations and delay.

In this section, we formalize the relocation problem with uncertainty and, using techniques from stochastic optimization, we extend our optimization framework (presented in Section 2) to find the optimum sequence of moves for each possible scenario. In particular, we consider the special case of retrieving containers that are initially in the bay. At first, we only have partial (or no) information on the departure times of some of the containers. Later in the retrieval process, the exact departure times of those containers are revealed. In the next subsection, we describe a few models of uncertainty on the departure times.

Note that the framework presented in the rest of this section can be generalized to DCRP with uncertainty. For the sake of brevity, however, we present the framework for the static relocation problem with uncertainty. Further, because terminals have limited information about arrivals of external trucks, the uncertainty in container departure times (the retrieval process) is significantly more than the uncertainty in arrival times (the stacking process). Typically, a large number of containers (in a vessel) arrive at the same time, and a single estimation of stacking time is sufficient for the whole group of containers. On the other hand, departing containers are delivered to different external trucks and thus their retrieval times need to be estimated independently. Considering the significant impact of retrieval processes on relocations and operations delays, port operators are highly interested in taking advantage of the partial information on container departure times.

In the following, we assume that N containers are initially stacked in the bay (i.e., $a_n = d_n = 0,\ n \in \{1, \ldots, N\}$), and all need to be retrieved. At time 0, we know the departure time of only a subset S_1 of these containers; for the rest of containers (in the set $S_2 \triangleq N \setminus S_1$) we only have partial (or no) information about their departure times. At a given time step t^*, the exact departure of containers in S_2 is revealed. We assume that the departure times of all containers in S_1 is earlier than those of the containers that belong to S_2. Also for simplicity, we assume that the retrieval service time window is the same for all containers. Finally, note that the departure times of all containers in S_2 is greater than t^*, i.e., we cannot retrieve a container before knowing its departure time.

For this setting, we present a two-stage stochastic optimization framework (for uncertainty models defined later) that jointly minimizes the expected relocations and delays. Investigating the solutions of the stochastic problem enables us to study the value of information and its impact on the efficiency of operations. Furthermore, the solution to the stochastic problem provides the port decision makers an optimal sequence of decisions, for each possible scenario that might be realized at time t^*.

An important challenge associated with stochastic models is how to model the uncertainty, which depends highly on the type of information available at a given time, as well as how frequently the information is updated throughout time. In practice, in a container terminal, the retrieval time of a container is known only after the external truck, which is claiming it, checks in at the entrance

gate. This means that at any given time, the exact departure times of only a limited number of containers are known. For the remaining containers, the departure times are either completely unknown or can only be estimated using partial information. Such information may come from the appointment system or from historical patterns. For example, historical data may suggest that some companies tend to pick up their containers on a particular day of the week and before/after noon. In fact, some terminals apply statistical models to predict the likelihood that a particular container (or group of containers) will be claimed in a given day or within a time horizon. In the next subsection, we introduce two models of uncertainty motivated by the structure of typical information available to port operators.

3.1 Uncertainty models

In the first model, we only have probabilistic information on the departure times of containers in S_2 (we call these containers *uncertain containers*). This means that for the uncertain containers, any departure times might be possible and we only know the probability distribution of the possible departure times (the distribution can be computed using the historical data). When such distributional information is unavailable, we assume that all scenarios are equally likely. To limit the set of possible scenarios (i.e, departure times), we consider a special class of departure times. We assume that containers $1, 2, \ldots, |S_1|$, belonging to set S_1, have departure times $d_1, \ldots, d_{|S_1|}$. Further, containers $|S_1|+1, \ldots, N$ have departure times that are a permutation of $\theta, \theta+1, \ldots, \theta+|S_2|$ for some $\theta \geq t^*$(remember that the departure time of a node in set S_2 cannot be earlier than its revelation time which is t^*). For this class, we have $|S_2|!$ possible departure time scenarios. Note that the number of scenarios grows exponentially with the number of uncertain containers, thereby causing the run time of the corresponding IP to increase with the number of scenarios.

As an example of the first model of uncertainty, consider 10 containers in the bay where 6 of them have been claimed by the trucks that have initially checked in at the gate. In this example, 6 of the departure times are known and 4 of them are unknown, resulting in $4! = 24$ possible scenarios, each with a given probability. At a later time t^*, the departure times of the last 4 containers are also revealed. This uncertainty model might not take advantage of the entire available information; however, it gives us a good benchmark to study the value of information.

In the second model of uncertainty, we build upon our first model and include some deterministic information on the departure times, taking advantage of the available partial information that often comes from the historical data and statistical models. We assume that for uncertain containers, although we do not know the exact departure times, we know that a group of containers that, for example, belong to the same company, are claimed together. Based on such information, we partition the set of uncertain containers into two or more batches, and assume that we know the departure order of the batches. There is still uncertainty on the departure times of the containers within each batch, and we can have distributions of possible departure times within one batch.

This model of uncertainty limits the number of possible scenarios. For instance, suppose the set of uncertain containers S_2, can be partitioned into two sets S_{21} and S_{22}, and we know that containers of set S_{21} depart before those of S_{22}. Here, the total number of scenarios is $|S_{21}|!|S_{22}|!$. However, if we did not know about the order of the two groups, then the number of scenarios would be $|S_2|!$.

As an example, consider 10 containers out of which 6 are certain. Suppose we know that the remaining containers belong to two batches of size two. For each batch, we have $2!$ scenarios, resulting in a total of $(2!)(2!) = 4$ possible scenarios, as compared to $4! = 24$ scenarios associated with the first model of uncertainty.

3.2 Formulation of the stochastic problem

In two-stage stochastic problems, there are two sets of decisions (see, e.g., Birge and Louveaux [1997]). The first set of decisions needs to be made without full information about some random events. These decisions are referred to as *first stage decisions*. In our problem, all moves (relocations and retrievals) before time t^*, are first-stage decisions. Later, full information is revealed on the realization of the random events. Then the *second-stage* actions, which in our case are the relocations and retrievals beyond time t^*, are taken. In the container relocation problem, the randomness is on the departure times and is represented by a set of scenarios as described in Subsection 3.1. We use the notations s and p_s to refer to each possible scenario and its probability. We also denote the set of all possible scenarios by σ. For each scenario s, we define all variables (2)-(7) with superscript s. These variables determine the sequence of decisions for scenario s.

The two-stage stochastic problem is to minimize the expected value of the objective function defined in (8). Note that because we are considering the static case, we can remove the summation of variables z_{nt} from the objective function. Thus the objective function for scenario s is given by

$$Q(s) \triangleq w_{rel} \sum_{i,k=1}^{C} \sum_{j,l=1}^{P} \sum_{n=1}^{N} \sum_{t=1}^{T} x^s_{ijklnt} - w_r \sum_{n=1}^{N} \sum_{t=1}^{T} v^s_{nt} \tag{23}$$

and the objective of our optimization problem is:

$$\sum_{s \in \sigma} p_s Q(s). \tag{24}$$

Finally, because the first stage decisions must be the same for all the scenarios, we introduce the following binding constraints:

$$x^{s}_{ijklnt} = x^{s'}_{ijklnt} \tag{25}$$

$$y^{s}_{ijnt} = y^{s'}_{ijnt} \tag{26}$$

$$s, s' \in \sigma, \quad i = 1, ..., C, \quad j = 1, ..., P, \quad k = 1, ..., C, \quad l = 1, ..., P, \quad n = 1, ..., N, \quad t = 1, ..., t^*$$

constraints (25) and (26) require that all the first-stage decision variables are equal, irrespective of the scenarios in the second stage. After time t^*, the decisions can be different depending on which scenario is realized. Note that the first-stage decisions specified by the stochastic optimization solution are such that the expected value of relocations and delay is minimized over all possible scenarios.

In Section 4, we report the numerical results of applying our IP for many bay sizes, configurations, and information levels. As illustrated later, the runtime of the program gets prohibitively long for even moderate-sized problems. This motivates us to design heuristics based on our IP formulation. In the next subsection, we present one particular heuristic and in Section 4, we compare its performance to that of the optimal solution.

3.3 Heuristic based on the stochastic optimization model

The number of variables in the stochastic model presented above increases exponentially with the size of the problem. As the number of containers in the bay increases, more scenarios and thus, more decision variables are needed to account for the lack of information on uncertain containers. To reduce computation time, we reduce the number of scenarios using the following heuristics.

Suppose that based on our initial information, we can group the uncertain containers into two batches, S_{21} and S_{22}, and we know that batch S_{21} will depart first. Later, at time t^*, the information is updated and the departure times within each batch are revealed. The best strategy is to generate $(|S_{21}|!|S_{22}|!)$ scenarios for all possible departure times within the two batches, and minimize the expected relocations and delay using the stochastic optimization model presented above. However, this requires long computation time. To reduce this, we (sub-optimally) ignore the departure times (and their corresponding scenarios) of the second batch S_{22} and only generate the scenarios of the first batch, which results in $|S_1|!$ scenarios. We solve the stochastic problem until t^*, ignoring the last batch. At t^*, when the information is updated for all uncertain containers, we solve the deterministic problem for the remaining time steps. For this deterministic problem, the initial configuration is given by the stochastic optimization solution at time t^*.

The basic intuition for such a heuristic is that the departure times of the containers that will depart far in the future (the ones in batch S_{22} in our model) might only have a small impact on the decisions made before time t^*. Thus we might ignore the departure order of those containers without significantly increasing the number of relocations.

From a computational perspective, using the above heuristic, we split the original intractable stochastic optimization problem into two subproblems: an easier-to-solve stochastic problem, and a deterministic problem. The computation time is reduced significantly by limiting the number of scenarios and decision variables, at the expense of solutions whose quality might be reduced. The proposed heuristic can be extended to the general case, where there are more than two batches of containers specified by the initially available information. We then decide which batches to consider or to ignore when generating scenarios.

4 Computational results for our stochastic optimization model

In this section, we use the stochastic formulation presented in the previous section to study the effects of information on the number of relocations. We design several experiments in which we incrementally improve the amount of initial information about departure times, and also the time of the information revelation. It should be noted that using our formulation, we can theoretically show that having more information and knowing the information earlier each results in fewer relocations. However, in the following, we quantitatively study the role of information by solving the CRP on many instances and comparing the average number of relocations in different experiments.

In our first set of experiments, we solve the CRP with a small bay containing 3 columns and 3 tiers. Containers $1, 2, \ldots, 6$ are initially stacked in this bay. Note that even for such a small bay, we have 1200 distinct initial configurations. Here, we partition these initial configurations into 3 groups: (1) configurations with equal height columns; (2) configurations with unequal height columns; and (3) configurations with one empty column. Table 6 illustrates these three groups. Note that the symbol 'x' represents a container and the symbol '-' represents an empty slot. Also, note that the number of distinct configurations in each group is different: groups 1, 2, and 3 contain $6!/2! = 360$, $6! = 720$, and $6!/3! = 120$ distinct instances, respectively. We solve for all possible instances in each group and report the corresponding average number of relocations. This way, in addition to the role of information, we evaluate the effect of the initial configuration (which can be another decision made by the port operator). In all experiments for a 3×3 bay, the weight factors in objective function (23) are set to one, and for each container, the retrieval time window (δ_n) is set to four.

For each initial configuration and each set of departure times, we solve the stochastic CRP with three levels of information and three revelation times (t^*), defined in Table 7, using the notation defined in Section 3. As explained in Section 3, the departure times of each container belonging to uncertain set S_2 cannot be earlier than t^*. For the experiment with $t^* = 6$, and $S_2 = \{3, 4, 5, 6\}$, we assume that the set of departure times $d_3, \ldots, d_6$ is a permutation of $6, 7, 8, 9$. To keep the instance the same for other information levels and revelation times, we assume similar departure times for containers $3 - 6$ in other experiments as well. For example, even when $t^* = 2$, we assume that the departure time of containers $3 - 6$ is a permutation of $6 - 9$, (following the notation defined

equal height columns	-	-	-
	X	X	X
	X	X	X
unequal height columns	X	-	-
	X	X	-
	X	X	X
one empty column	X	X	-
	X	X	-
	X	X	-

Table 6: Initial configuration groups for a 3x3 bay

Information level		t^*
Little	$S_1 = \{1, 2\}$, $S_2 = \{3, 4, 5, 6\}$	6
Moderate	$S_1 = \{1, 2\}$, $S_{21} = \{3, 4\}$, $S_{22} = \{5, 6\}$	4
High	$S_1 = \{1, 2, 3, 4\}$, $S_2 = \{5, 6\}$	2

Table 7: levels of information and revelation times for bay 3x3.

in Section 3, $\theta = 6$). Finally, we assume that in each experiment all possible scenarios are equally likely.

The average of the optimal number of relocations for the above experiments on the three groups of initial conditions (shown in Table 6) are summarized in Figures 2, 3, and 4.

Let us consider Figure 2. First note that these numerical results confirm that as we increase the level of information initially available, the average number of relocations decreases. Further, revealing the missing information earlier improves the performance. However, missing even a small amount of information, results in an increase in the number of relocations: consider, for example, the last bar in the figure (the rightmost blue bar). In this experiment, the only missing information at time step 1 is d_5 and d_6, which can be either $d_5 = 8, d_6 = 9$ or $d_5 = 9, d_6 = 8$. At time step 2, we have complete information. As shown, the average number of relocations for this experiment is slightly higher than that with complete information. This implies that there are instances for which not knowing the exact departure order of the last two containers results in a mistake in deciding the first move at time 1.

Another important observation is that knowing the departure times of batches of containers (which can often be estimated using the historical data) is quite valuable. Figures 2, 3, and 4 show that the gap between the number of relocations for the case of moderate information (knowing that batch of containers 3 and 4 will be retrieved before the batch of 5 and 6) and high information (complete information on d_3, d_4) is always about 20%. This gap is inversely correlated with the

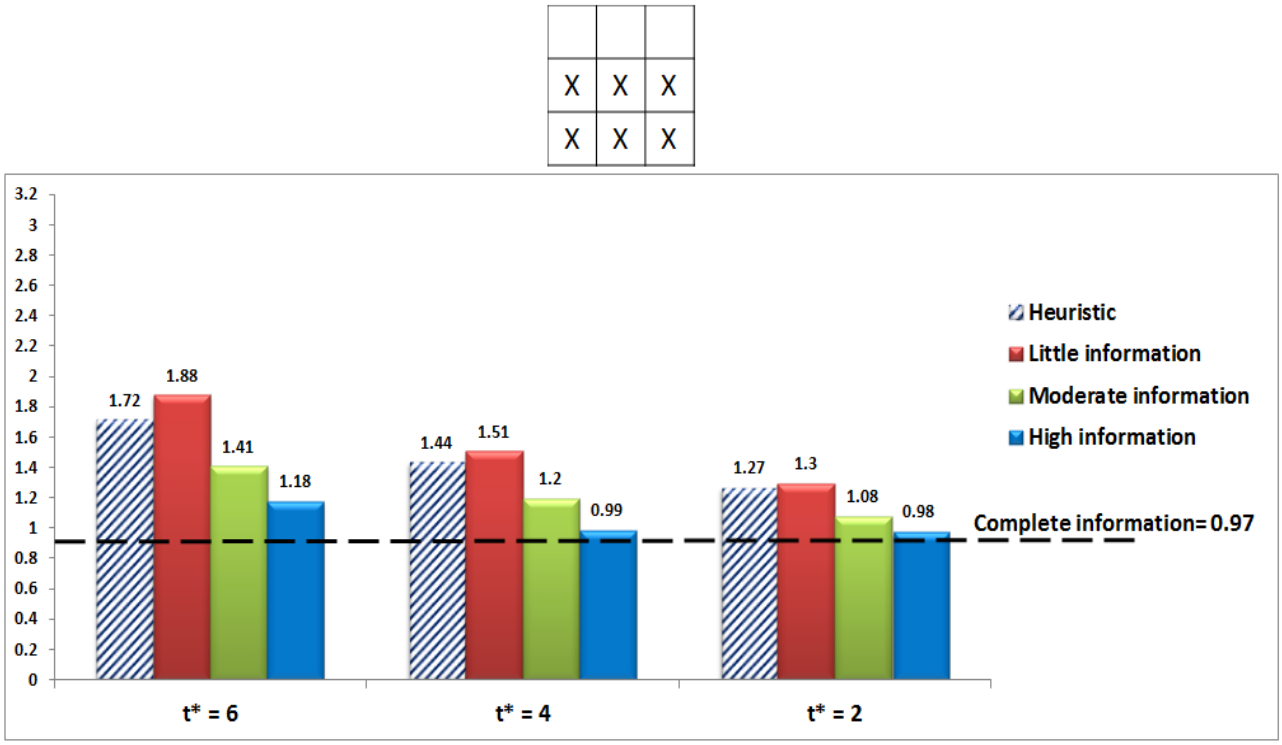

Figure 2: Average number of relocations for a 3×3 bay with equal height columns. Levels of information are: (i) Little: two certain and four uncertain containers; (ii) Moderate: two certain containers and two batches of size two; (iii) High: four certain and two uncertain containers. The heuristic corresponding to the moderate information level is to ignore the second batch of containers before $t = 4$ and generate only two scenarios for the containers in the first batch.

information update time: for $t^* = 2$, the gap is less than 10% for all configurations and reaches 20% for $t^* = 6$. Moreover, moderate information results in fewer relocations by up to 40% compared to little information (no information about d_3, d_4, d_5, d_6).

Further, the figure suggests that even though having high levels of information always results in better performance compared to the experiments with moderate levels of information, if we reveal the moderate information two steps earlier, its corresponding optimal solution is quite close to that of the high information case. Compare, for example, the blue bar of $t^* = 4$ and the green bar of $t^* = 2$.

Similar observations can also be made for the other groups of initial configurations (see Figures 3 and 4). Comparing the results of these three groups for all experiments, we observe that having an initial configuration with equal height columns results in fewer relocations compared to the other initial configurations. On the other hand, having an initial configuration with one empty column results in more relocations compared to the others. Intuitively, to retrieve each container, we need to relocate (at least) all the containers on top of it. For the containers that are scheduled to be

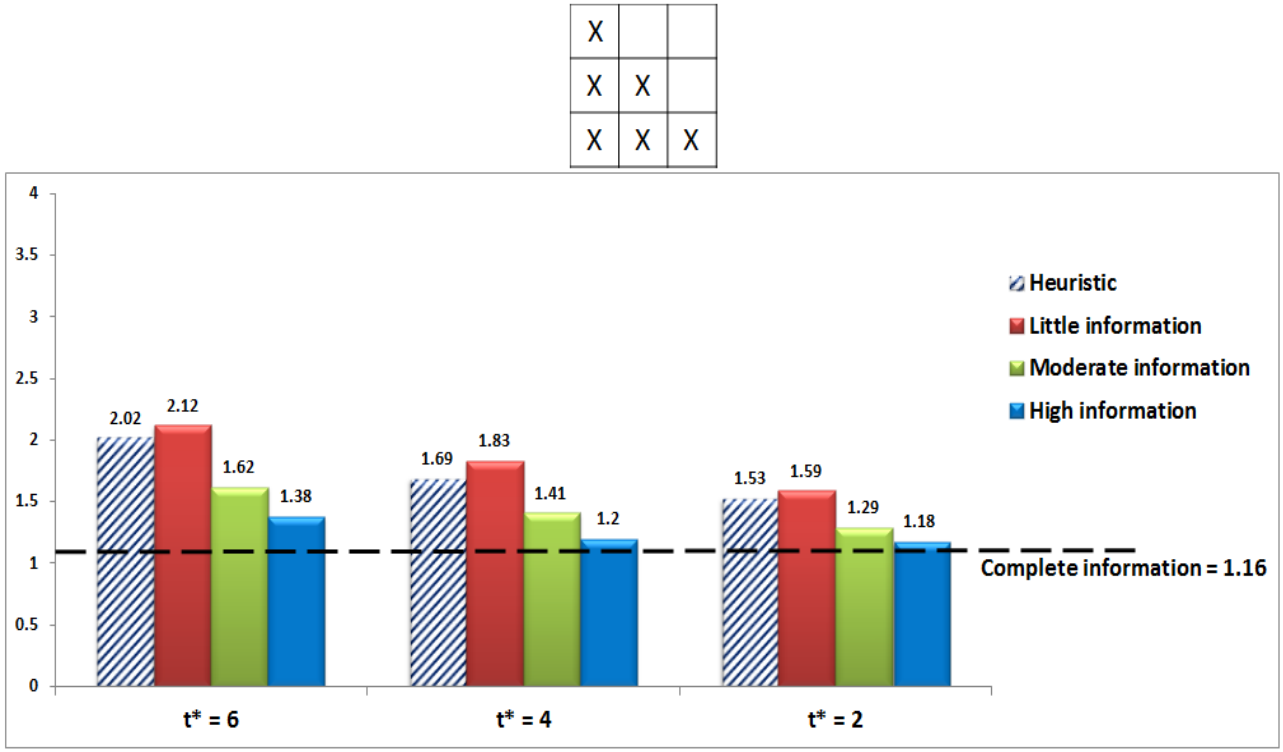

Figure 3: Average number of relocations for a 3×3 bay with unequal height columns. Levels of information are: (i) Little: two certain and four uncertain containers; (ii) Moderate: two certain containers and two batches of size two; (iii) High: four certain and two uncertain containers. The heuristic corresponding to the moderate information level is to ignore the second batch of containers before $t = 4$ and generate only two scenarios for the containers in the first batch.

retrieved early, the initial height gives a rough estimate of the number of relocations needed to retrieve them. Thus, in the average case, initial configurations with equal height columns result in fewer relocations.

As discussed in Subsection 3.3, the computation time of our IP increases exponentially with the size of the bay and the number of containers. To address this problem, we introduced the heuristic that initially ignores the retrieval of (some or all) containers scheduled to be retrieved after t^*. To examine the performance of this heuristic, we apply it to the experiments in which we have moderate information at time 0. In particular, at time steps $1, \ldots, t^* - 1$, we ignore the retrieval of containers 5 and 6. Then later, at time t^* when complete information is revealed, we re-solve the problem. The dashed bars in Figures 2, 3, and 4 show the performance of this heuristic. In all cases, the heuristic is suboptimal compared to the corresponding optimal moderate-information solutions. However, the average number of relocations is increased by a factor less than 1.25, and in all cases, the heuristic performs better than having little initial information (only knowing d_1 and d_2).

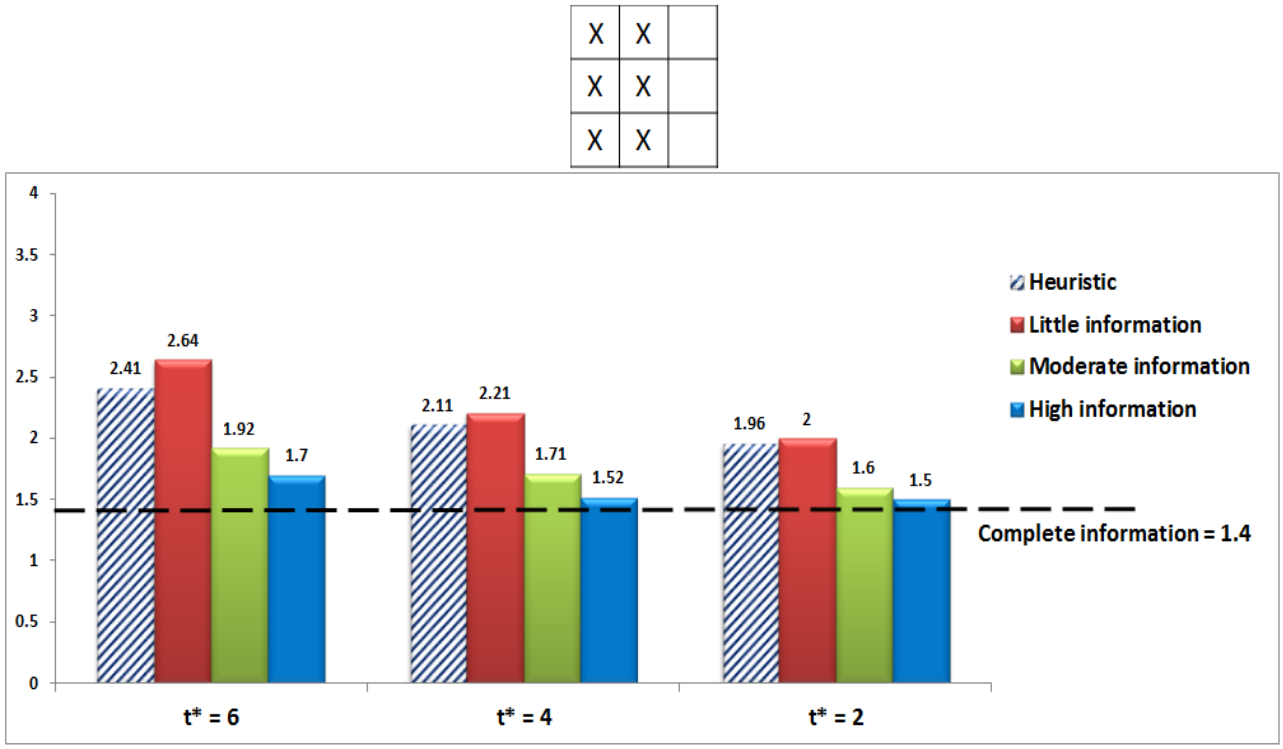

Figure 4: Average number of relocations for a 3×3 bay with one empty column. Levels of information are: (i) Little: two certain and four uncertain containers; (ii) Moderate: two certain containers and two batches of size two; (iii) High: four certain and two uncertain containers. The heuristic corresponding to the moderate information level is to ignore the second batch of containers before $t = 4$ and generate only two scenarios for the containers in the first batch.

In our second set of experiments, we solve the problem for a 4×4 bay with 12 containers. For such a large bay, we compare the results of the heuristic with the optimal solution of our stochastic model. For the 4×4 bay, we only consider an initial configuration with equal height columns. This is because the results of the smaller bay suggest that such initial configurations perform best in terms of the number of relocations. We design the following four experiments:

Experiment1 We initially know $d_1, d_2, \ldots, d_6$. Further we know the remaining containers belong to two batches. Batch 1 includes $7, 8, 9$, and batch 2 includes $10, 11, 12$. We initially know that batch 1 is scheduled to be retrieved after containers $1, 2, \ldots, 6$ and earlier than batch 2. The exact departure times $d_7, d_8, \ldots, d_{12}$ are revealed at time $t^* = 14$.

Experiment2 We initially know $d_1, d_2, \ldots, d_6$. Further we know the remaining containers belong to two batches. Batch 1 includes $7, 8, 9$, and batch 2 includes $10, 11, 12$. We initially know that batch is scheduled to be retrieved after containers $1, 2, \ldots, 6$ and earlier than batch 2. The exact departure times $d_7, d_8, \ldots, d_{12}$ are revealed at time $t^* = 7$.

Experiment3 We initially know $d_1, d_2, \ldots, d_6$. Further we know the remaining containers belong to three batches. Batch 1 includes $7, 8$, batch 2 includes $9, 10$, and batch 3 includes $11, 12$. We initially know that containers in all batches will depart after containers $1, 2, \ldots, 6$. Also, we know that batch 1 is scheduled to be retrieved earlier than batch 2, and batch 2 is to be retrieved before batch 3. The exact departure times $d_7, d_8, \ldots, d_{12}$ are revealed at time $t^* = 14$.

Experiment4 We initially know $d_1, d_2, \ldots, d_6$. Further we know the remaining containers belong to three batches. Batch 1 includes $7, 8$, batch 2 includes $9, 10$, and batch 3 includes $11, 12$. We initially know that containers in all batches will depart after containers $1, 2, \ldots, 6$. Also, we know that batch 1 is scheduled to be retrieved earlier than batch 2, and batch 2 is to be retrieved before batch 3. The exact departure times $d_7, d_8, \ldots, d_{12}$ are revealed at time $t^* = 7$.

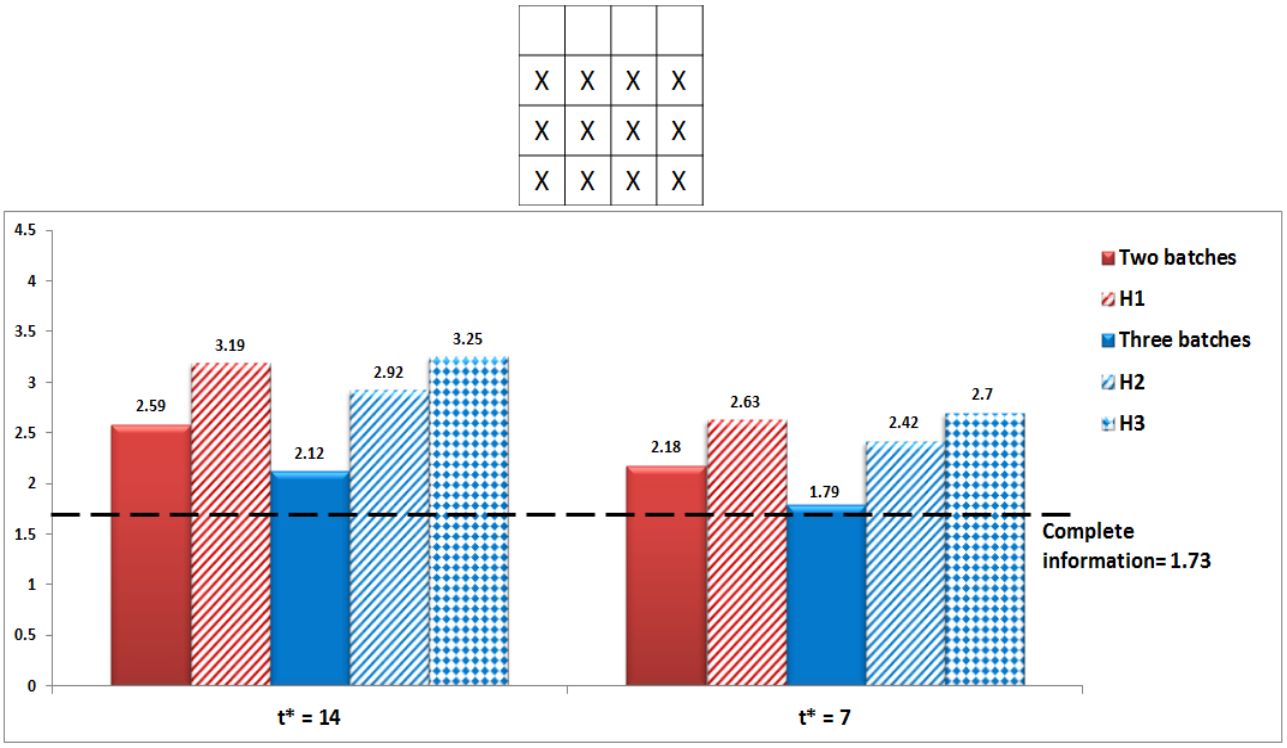

Figure 5: Average number of relocations over 30 instances for a 4×4 bay with equal height columns. Levels of information are: (i) Two batches: two certain containers and two batches of size three; (ii) Three batches: two certain containers and three batches of size two; Three heuristics: H1, H2, and H3.

In all four experiments for the 4×4 bay, the weight factors in objective function (23) are set to one, and for each container, the retrieval time window (δ_n) is set to eight.

Finding the optimal solution for the above experiments (using our IP formulation) is computationally intensive. In fact, for most instances of the problem, it takes more than 2 hours to find the optimal solution using a state-of-the-art commercial computer. Thus, we only solve for a limited number of instances sampled randomly. The average of the optimal number of relocations (over 30 random samples) are summarized in Figure 5 (for the number of relocations in each instance, see Appendix A). The characteristics of the optimal solutions in different experiments are similar to that of the related experiments in the smaller bay (the 3x3 bay).

We also use the following heuristics to generate solutions for the experiments defined above

H1 In the first two experiments, at time steps $1, 2, \ldots, t^* - 1$, we ignore the retrieval of the second batch that includes containers $10, 11, 12$.

H2 In the last two experiments, at time steps $1, 2, \ldots, t^* - 1$, we ignore the retrieval of the third batch that includes containers $11, 12$.

H3 In the last two experiments, at time steps $1, 2, \ldots, t^* - 1$, we ignore the retrieval of the second and third batches that include containers $9, 10, 11, 12$.

The dashed bars in Figure 5 present the average number of relocations found by the heuristics for the same set of instances. In all the experiments, H2 performs better than H3, and H1 performs better than H3. These numerical results are consistent with our basic intuition that initially ignoring (the retrieval of) fewer containers should only improve the number of relocations, but it increases the runtime as more scenarios must be included in the formulation.

5 Discussion

In most container terminals, containers continually arrive at (depart from) the storage yard and thus need to be stacked (retrieved) continually. Further, information on stacking and retrieval times are not available far in advance, and perhaps only estimates of the future stacking/retrieval times are available. Yard operations (and in particular layout planning of the bays) is highly affected by these dynamic and uncertain features. In the bays, because containers are stacked on top of each other, it often happens that the container that needs to be retrieved is covered by other containers. In this case, containers must be relocated, and such relocations are considered non-productive moves, which are costly. Minimizing the relocation moves in a yard with continual arrivals and departures, and the effects of having incomplete information on the number of such relocations, have been poorly understood. In this paper, we first provide an IP-formulation, the solution of which is an optimal sequence of moves for the dynamic container relocation problem. For the special case of only retrieving containers in a predefined order (as previously studied by Caserta et al. [2012], Kim and Hong [2006], and Wan et al. [2009]), we compare our results with those of Caserta et al. and Kim and Hong, and show that defining a window within which each

container is retrieved and relaxing the strict ordering of container retrievals result in a realistic model and a solution with fewer relocations and less delay.

Further, to study the value of: 1) information; and 2) estimates on the arrival/departure times, we develop a two-stage stochastic optimization model and approach that yields the optimal sequence of moves for each possible outcome, minimizing the expected number of relocations. Through experiment, we confirm the value of estimating departure times using historical data.

In this work, we introduce a two-stage stochastic optimization framework, in which all information are initially unavailable, and revealed at once (at a later time). In general, a multi-stage formulation or a rolling horizon methodology would be closer to the reality of port operations. Because the running times of the two-stage stochastic program are prohibitively long, the design of different IP-based heuristics to provide quality solutions is warranted.

The relocation problem also arises in other storage systems such as steel plate stacking and warehousing systems (see Kim et al. [2011], Zäpfel and Wasner [2006] for the former, and Chen et al. [2011] for the latter). These systems also face the challenge of continual stacking and retrieving with incomplete information. We believe our methodology might be applicable to the operations of these systems as well.

References

J.R. Birge and F.V. Louveaux. *Introduction to stochastic programming*. Springer series in operations research and financial engineering. SPRINGER VERLAG GMBH, 1997. ISBN 9780387982175.

Bram Borgman, Eelco Asperen, and Rommert Dekker. Online rules for container stacking. *OR Spectrum*, 32(3):687–716, 2010. ISSN 0171-6468.

Marco Caserta, Silvia Schwarze, and Stefan Voß. A new binary description of the blocks relocation problem and benefits in a look ahead heuristic. In Carlos Cotta and Peter Cowling, editors, *Evolutionary Computation in Combinatorial Optimization*, volume 5482 of *Lecture Notes in Computer Science*, pages 37–48. Springer Berlin Heidelberg, 2009. ISBN 978-3-642-01008-8.

Marco Caserta, Stefan Voß, and Moshe Sniedovich. Applying the corridor method to a blocks relocation problem. *Or Spectrum*, 33(4):915–929, 2011.

Marco Caserta, Silvia Schwarze, and Stefan Voß. A mathematical formulation and complexity considerations for the blocks relocation problem. *European Journal of Operational Research*, 219 (1):96 – 104, 2012. ISSN 0377-2217.

Lu Chen, Andr Langevin, and Diane Riopel. A tabu search algorithm for the relocation problem in a warehousing system. *International Journal of Production Economics*, 129(1):147 – 156, 2011. ISSN 0925-5273.

Rommert Dekker, Patrick Voogd, and Eelco Asperen. Advanced methods for container stacking. In KapHwan Kim and Hans-Otto Gnther, editors, *Container Terminals and Cargo Systems*, pages 131–154. Springer Berlin Heidelberg, 2007. ISBN 978-3-540-49549-9.

Byung-In Kim, Jeongin Koo, and Hotkar Parshuram Sambhajirao. A simplified steel plate stacking problem. *International Journal of Production Research*, 49(17):5133–5151, 2011.

Kap Hwan Kim and Gyu-Pyo Hong. A heuristic rule for relocating blocks. *Comput. Oper. Res.*, 33(4):940–954, April 2006. ISSN 0305-0548.

Kap Hwan Kim, Young Man Park, and Kwang-Ryul Ryu. Deriving decision rules to locate export containers in container yards. *European Journal of Operational Research*, 124(1):89–101, July 2000.

Yusin Lee and Yen-Ju Lee. A heuristic for retrieving containers from a yard. *Comput. Oper. Res.*, 37(6):1139–1147, June 2010. ISSN 0305-0548.

Yat-wah Wan, Jiyin Liu, and Pei-Chun Tsai. The assignment of storage locations to containers for a container stack. *Naval Research Logistics (NRL)*, 56(8):699–713, 2009. ISSN 1520-6750.

Günther Zäpfel and Michael Wasner. Warehouse sequencing in the steel supply chain as a generalized job shop model. *International Journal of Production Economics*, 104(2):482–501, 2006.

Wenjuan Zhao and Anne V. Goodchild. The impact of truck arrival information on container terminal rehandling. *Transportation Research Part E: Logistics and Transportation Review*, 46 (3):327–343, May 2010.

Appendix A

	$t^* = 14$					$t^* = 7$				
instance	2 batches	H1	3 batches	H2	H3	2 batches	H1	3 batches	H2	H3
1	3	3	2	3	3	3	3	2	3	3
2	2.25	2	2	2	2	2	2	1.75	2	2
3	2.5	3	2.5	3	3.5	2	2	1.75	2	2
4	3	3	3	3.5	3.5	2.5	2.5	2.5	3	3.25
5	2.5	3.5	2.375	3.5	4	2	2.5	2	2	3
6	2.5	3	2	3	3	2.5	3	2	3	2.25
7	2.75	3	2	3	4	2.25	2.5	1.5	2	2
8	2	2	2	2.5	2.5	1	2.5	1	1	1.5
9	1	2.75	1	2	3	1	1.75	1	2	2
10	3	4	3	3	3	1.75	2	1.5	2	2.5
11	2.25	3.5	2	3	3.75	2	2	2	3	3.5
12	3	4.5	3	4	4.5	2	2.5	2	3	3.5
13	3	4	1	3.5	3.5	1	2	1	2	2
14	2.75	3.375	2	2	2.5	2.75	3	2	3	2
15	3	3.375	2	3	3	2	2.75	1.75	2	2.75
16	3	4	3	3.25	4	3	3	3	3	4
17	2.5	3.25	2	2.5	3	2.5	3	2	2	4
18	3.5	4	3.25	3.75	4	3	3.5	3.25	3	3.5
19	2	3	2	3	2.75	1	1.25	0	1	1.5
20	2	3	1.5	3	3	2	3	1.5	2.5	3
21	3.5	4	3	4	5	3.5	3.5	3	3.5	3.5
22	1.5	2.5	1	2.625	3	1.3	1.5	1	3	2.25
23	2	2.5	1	2.75	3	2	2.76	1	3	2
24	2.5	3	2.5	2.5	2.5	2	3	1.75	2	2
25	2.5	2.5	1.75	2.25	2.5	2.5	3	1.625	1.75	2
26	2	3	2	3	3	2	2.75	2	2	3
27	3	3.5	2	2.5	3	2.75	3	2	2.5	3
28	3.25	3	2.75	3	3.75	2.25	2.25	2	2	2
29	2.5	2.5	2	2	3	2.5	3.75	2	3	4
30	3.5	4	2	3.5	3.5	3.5	3.75	1.75	3.5	4

Table 8: Number of relocations for 30 randomly generated instances for a 4×4 bay

Appendix B

The logical constraint (constraint (8) in Caserta et al. [2012]) ensures that the LIFO order in a column is not violated, when two or more containers are relocated (this constraint is necessary in the BRP-II model because multiple relocations can be handled at each time period). For example if container m is below container n, and container m is relocated in time period t, then container n cannot be above container m in the next time period . This constraint, as stated by Caserta et al., is as follows:

$$1 - \sum_{n=1}^{N} x_{ijklnt} \geq \sum_{n=1}^{N} \sum_{j'=j+1}^{P} \sum_{l'=l+1}^{P} x_{ij'kl'nt} \quad i,k = 1,...,C, \quad j,l = 1,...,P, \quad t = 1,...,N-1 \tag{27}$$

Constraint (27) does impose this required condition, but is in fact overly restrictive: it does not allow the relocation of two containers (that are initially in the same column) to another same column. The left hand side of the constraint equals zero if no container is relocated from position (i, j) to position (k, l) in time period t. The resulting inequality forces that at most one container may be relocated from a position above (i, j) to a position above (k, l). To relax this restriction, we replace constraint (27) with constraint (28) in our implementation of the BRP-II model, which allows for $H-1$ relocations if $\sum_{n=1}^{N} x_{ijklnt} = 0$.

$$(H-1)(1 - \sum_{n=1}^{N} x_{ijklnt}) \geq \sum_{n=1}^{N} \sum_{j'=j+1}^{P} \sum_{l'=l+1}^{P} x_{ij'kl'nt} \quad i,k = 1,...,C, \quad j,l = 1,...,P, \quad t = 1,...,N-1 \tag{28}$$

Because of the change in this constraint, some of the results in Table 5 are different from those of Caserta et al. [2012].