European Journal of Operational Research 238 (2014) 155–165

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Production, Manufacturing and Logistics

Conservative allocation models for outbound containers in container terminals

Canrong Zhang [a], Tao Wu [b], Kap Hwan Kim [c,*], Lixin Miao [a]

[a] Logistics Engineering and Simulation Laboratory, Graduate School at Shenzhen, Tsinghua University, Shenzhen 518055, China
[b] Business Analytics and Optimization, University of Phoenix, Apollo Group, Inc., Phoenix, AZ 85040, USA
[c] Department of Industrial Engineering, Pusan National University, Jangjeon-dong, Geumjung-gu, Busan 609-735, Republic of Korea

ARTICLE INFO

Article history:
Received 31 March 2013
Accepted 31 March 2014
Available online 13 April 2014

Keywords:
Container terminals
Dynamic programming
Location assignment
Simulation

ABSTRACT

This paper examines location assignment for outbound containers in container terminals. It is an extension to the previous modeling work of Kim et al. (2000) and Zhang et al. (2010). The previous model was an "optimistic" handling way and gave a moderate punishment for placing a lighter container onto the top of a stack already loaded with heavier containers. Considering that the original model neglected the stack height and the state-changing magnitude information when interpreting the punishment parameter and hid too much information about the specific configurations for a given stack representation, we propose two new "conservative" allocation models in this paper. One considers the stack height and the state-changing magnitude information by reinterpreting the punishment parameter and the other further considers the specific configurations for a given stack representation. Solution qualities for the "optimistic" and the two "conservative" allocation models are compared on two performance indicators. The numerical experiments indicate that both the first and second "conservative" allocation models outperform the original model in terms of the two performance indicators. In addition, to overcome computational difficulties encountered by the dynamic programming algorithm for large-scale problems, an approximate dynamic programming algorithm is presented as well.

1. Introduction

In container terminals, seeking a proper yard location for a newly arriving outbound container is an important issue. It is a common practice that the heavier containers are loaded earlier than the lighter ones during the loading operations. Therefore, a relocation movement generally occurs when a lighter container is stacked on the top of a heavier one. A proper location-assignment strategy or policy can avoid burying the heavier containers underneath and reduce relocation movements for the upcoming loading operations.

To place a newly arriving container onto a stack, the following three decisions at different levels need to be made: (1) allocating yard blocks or sub-blocks to the outbound containers destined for each arriving vessel; (2) allocating yard bays to the containers of the same group (i.e., the containers of the same length, destined for the same destination port and the same vessel); (3) assigning a yard location, within the range of a single yard bay, to each newly arriving container. This paper is concerned about the lowest-level decision. For the higher-level decisions, please refer to Kim and Park (2003), Zhang, Liu, Wan, Murty, and Linn (2003), Lee, Chew, Tan, and Han (2006), Han, Lee, Chew, and Tan (2008), Zhang, Zhang, Zheng, and Miao (2011), Jiang, Lee, and Chew (2012), Yu and Qi (2012) and the papers therein.

Kozan and Preston (2006) integrated container transfer model with container location model. In that paper, the models were evaluated by the transfer time between the storage location and the destined vessel. The related research can be found in Preston and Kozan (2001) and Kozan and Preston (1999). Dekker, Voogd, and Asperen (2006) used a simulation method to compare random stacking with category stacking by the number of relocation movements. Kang, Ryu, and Kim (2006) attempted to accommodate all realizations of container arrivals by a stacking strategy, which was also evaluated by the number of relocation movements via simulation. Chen and Lu (2012) proposed a hybrid sequence stacking algorithm and compared it with a random stacking algorithm and a vertical stacking algorithm. Jang, Kim, and Kim (2013) suggested a genetic algorithm to determine the storage location for each arriving container in order to minimize the expected number of relocation movements.

* Corresponding author.
E-mail addresses: crzhang@sz.tsinghua.edu.cn (C. Zhang), danielwu9999@gmail.com (T. Wu), kapkim@pusan.ac.kr (K.H. Kim), lxmiao@tsinghua.edu.cn (L. Miao).

http://dx.doi.org/10.1016/j.ejor.2014.03.040
0377-2217/

Similar to the above papers, Kim, Park, and Ryu (2000) and Zhang, Chen, Shi, and Zheng (2010) also used the number of relocation movements to evaluate the stacking strategy but the difference is that an analytical model rather than (meta-) heuristic algorithms were used to generate the feasible yard location (or the stacking priority sequence called in the referenced papers) for the newly arriving container. This paper is an extension to Kim et al. (2000) and Zhang et al. (2010). The difference between them is that a punishment coefficient, for placing a lighter container onto the top of a stack already loaded with heavier containers, was treated "optimistically" in the previous work while it is treated relatively "conservatively" in this study. In particular, considering that the original model neglected the stack height and the state-changing magnitude information when interpreting the punishment parameter, and hid too much information about the specific configurations for a given stack representation, we propose two new "conservative" allocation models in this paper. One considers the stack height and the state-changing magnitude information by reinterpreting the punishment parameter and the other further considers the specific configurations for a given stack representation. These two new allocation models may lead to better solutions. That is the motivation of this paper. In addition, to overcome computational difficulties encountered by the dynamic programming algorithm for large-scale problems, an approximate dynamic programming algorithm is also presented.

In addition, container pre-marshaling (reposition), which is closely pertaining to the location assignment for arriving containers, is another hot research topic. Given a yard layout and a sequence that containers are loaded onto a ship, Lee and Hsu (2007) formulated an integer programming model to reposition the containers to ensure that no extra relocations are needed during the loading operation. For the similar problem, Lee and Chao (2009) proposed a heuristic which consists of a neighborhood search process, an integer programming model and three minor subroutines; Kim and Hong (2006) proposed a branch-and-bound algorithm and a heuristic rule based on dynamic programming results; Caserta, VoB, and Sniedovich (2011) presented a corridor method inspired algorithm; Zhu, Qin, Lim, and Zhang (2012) investigated iterative deepening A* algorithms using new lower bound measures and heuristics; and Huang and Lin (2012) proposed two labeling algorithms for two relocation problems.

The remainder of this paper is organized as follows: The original model is presented in Section 2; the two "conservative" allocation models are presented in Section 3, followed by an approximate dynamic programming algorithm in Section 4; the comparisons among models are presented in Section 5; and the conclusions are drawn in the last section.

2. Optimistic allocation model

2.1. The original model

The definitions related to the model are presented as follows:

Stage: the total number of empty slots in a yard bay under consideration. The example as shown in Fig. 1 is in stage 7, as its total number of empty slots is 7.

State: the state of a yard bay which consists of the combination of the number of empty slots in each stack and the combination of the representation of each stack. In particular, in the original model the representation of a stack is represented by the heaviest weight group of its loaded containers. When a stack is full, the stack is conventionally represented as "0" no matter what containers it contains. When a stack is totally empty, it is represented as "*". Stacks are sorted in the decreasing order of the number of empty slots. When several stacks have the same number of empty slots, the one with the heavier weight group is placed in front of the others. For example, the yard bay as shown in Fig. 1, which has a bay pattern with 6 stacks, 4 tiers and 3 weight groups (heavy (H), medium (M) and light (L)), has a state represented as (221110 LLHHM0) according to the aforementioned rules.

The notations related to the model are presented as follows:

s: the number of stacks in a yard bay.
t: the number of tiers in a stack.
N: the total number of stages (not including the stage with zero empty slot) which is equal to $s \times t$.
n: the stage number, that is, the number of empty slots.
G: the set of weight groups. The elements in G are indexed in the decreasing order of their weight groups.
X_n: the input state of the nth stage.
k_n: the weight group of an arriving container at stage n.
$p_n(k_n)$: the probability that a newly arriving container is with weight group k_n. It is assumed that the probability does not change during the whole receiving process.
D_n: the stack number assigned to an arriving container at stage n (a decision variable).
$r_n(X_n, D_n, k_n)$: the punishment coefficient for assigning a newly arriving container with the lighter weight group k_n to a heavier stack D_n when the input state is X_n. In particular, in the original model this punishment coefficient is equal to 1 when a lighter container is placed onto a heavier stack and 0 otherwise.
$t_n(X_n, D_n, k_n)$: the state transfer function that maps X_n to X_{n-1} when a newly arriving container with weight group k_n is assigned to stack D_n.

The decision process is shown in Fig. 2, from which we see that a decision (D_n) is made at each input state (X_n) after we know the weight group (k_n) of the newly arriving container, with consideration of the impact of the output state (X_{n-1}) on the subsequent stacking for the remaining empty slots. The objective function is to minimize the total punishment of $r_n(X_n, D_n, k_n)$, which is formulated as follows:

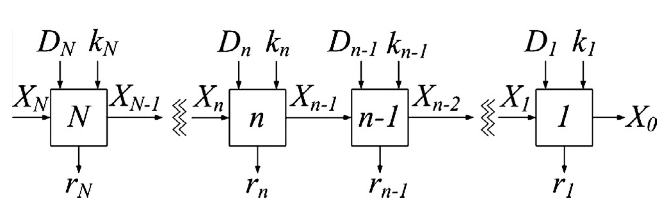

Fig. 2. The decision process.

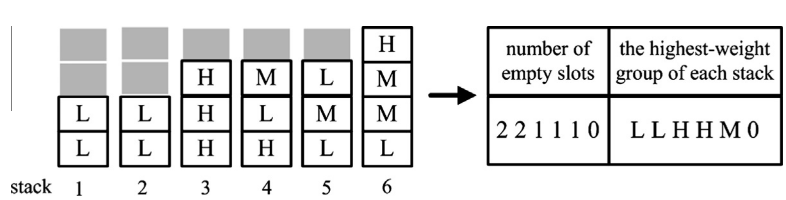

Fig. 1. An illustration of bay representation.