European Journal of Operational Research 264 (2018) 165–180

Contents lists available at ScienceDirect

European Journal of Operational Research

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

Production, Manufacturing and Logistics

Solving real-world sized container pre-marshalling problems with an iterative deepening branch-and-bound algorithm

Shunji Tanaka[a], Kevin Tierney[b,*]

[a] Institute for Liberal Arts and Sciences & Department of Electrical Engineering, Kyoto University, Kyotodaigaku-Katsura, Nishikyo-ku, 615-8510 Kyoto, Japan
[b] Decision Support & Operations Research Lab, University of Paderborn, Warburger Straße 100, Paderborn 33098, Germany

ARTICLE INFO

Article history:
Received 21 February 2017
Accepted 24 May 2017
Available online 30 May 2017

Keywords:
OR in maritime industry
Container pre-marshalling
Terminal operations

ABSTRACT

Container terminals around the world regularly re-sort the containers they store according to their retrieval times in a process called pre-marshalling, thus ensuring containers are efficiently transferred through the terminal. State-of-the-art algorithms struggle to find optimal solutions for real-world sized pre-marshalling problems. To this end, we introduce an improved exact algorithm using an iterative deepening branch and bound search, including a novel lower bound computation, a new branching heuristic, new dominance rule and a new greedy partial solution completion heuristic. Our approach finds optimal solutions for 161 more instances than the state-of-the-art algorithm on two well known, difficult pre-marshalling datasets, and solves all instances in three other datasets in just several seconds. Furthermore, we find optimal solutions for a majority of real-world sized instances, and feasible solutions with very low relaxation gaps on those instances where no optimal could be found.

1. Introduction

The container trade has grown over the past decades at a tremendous rate, with 1687 million tons of cargo shipped in containers in 2015, a 69% increase over 2005, and a 355% increase over 1995 (UNCTAD, 2016). The world's busiest container port, in Shanghai, China, had a throughput of 36.5 million TEU[1] in 2015, and the port with the highest throughput in Europe, Rotterdam, Netherlands, transferred 12.3 million TEU. With the global container market estimated to grow 4.4% per year through 2018 (Danish Ship Finance, 2017), effective decision support algorithms for managing containers along their journey are becoming critical to the operations of liner carriers and container terminals.

Efficient operations in container terminals are particularly important for shippers, as delays and uncertainty in the logistics chain force companies and organizations to maintain larger safety stocks to avoid interruptions in production and sales (Vernimmen, Dullaert, & Engelen, 2007). Terminals that perform poorly or have high costs will lose business to competitors, as, according to Psaraftis (2004), shipping lines can relatively easily and quickly switch hubs. Port and terminal operations therefore have a strong interest in decision support tools that can help them maintain or even improve their efficiency in the face of increasing container volumes.

Delays can occur in container ports in a variety of areas and for a number of reasons, including during the transfer of containers between terminals (Tierney, Voß, & Stahlbock, 2014), intra-terminal handling and vehicle dispatching (Grunow, Günther, & Lehmann, 2006), and due to incorrect stacking in the *yard* (see Fig. 1 for an overview of a terminal). Our focus in this work is on delays in the yard, as containers must be quickly moved in and out of the potentially thousands of stacks in this part of the terminal. There are two central problems regarding the retrieval of containers from the yard once containers have been placed through, e.g., the container stacking problem (Dekker, Voogd, & van Asperen, 2006): the container (or blocks) relocation problem, in which containers are extracted from a set of stacks (see, e.g., Kim & Hong, 2006), and the container pre-marshalling problem (CPMP), in which containers are re-sorted in stacks (Lee & Hsu, 2007), the latter of which is the main focus of this article.

In the CPMP, a set of stacks are sorted by a rail-mounted gantry crane (RMGC) according to the time each container is expected to leave the stacks. Solving the CPMP prevents containers with late exit times from blocking containers with early exit times, helping to ensure efficient yard operations. The goal of the CPMP is to find a minimal sequence of container moves from the top of one stack to the top of another stack such that no container blocks the removal of a container that must leave before it. Solving the CPMP

* Corresponding author.
E-mail addresses: tanaka@kuee.kyoto-u.ac.jp (S. Tanaka), tierney@dsor.de (K. Tierney).

[1] TEU stands for *twenty-foot equivalent unit* and represents a 20 foot intermodal container.

http://dx.doi.org/10.1016/j.ejor.2017.05.046
0377-2217/

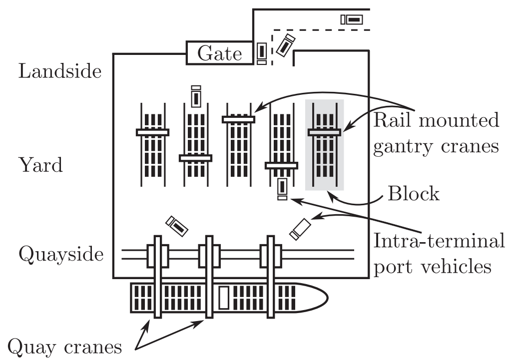

Fig. 1. Overview of a sea container terminal's operations, adapted from Tierney (2015).

is thus an important task for container terminals to perform when the terminal is not busy so that it is prepared for peak throughput periods.

In this paper, we introduce an improved exact algorithm based on an iterative deepening search that includes the following novel components:

1. an improved, tighter version of the lower bound introduced in Bortfeldt and Forster (2012),
2. a new dominance rule addressing containers with the same retrieval time,
3. a greedy heuristic for completing partial CPMP solutions, and
4. an effective branching strategy that uses the improved lower bound along with several tie breaking criteria.

The improved lower bound we introduce focuses on special cases of the CPMP that occur when all (or most) stacks of containers are not correctly sorted. By taking into account extra container movements into the lower bound that must occur in the cases we present, the overall search tree can be significantly pruned. The new dominance rule breaks symmetries that occur when multiple containers have the same retrieval time, which is a common occurrence in container terminals. Our greedy heuristic shares similarities with well-known CPMP heuristics in the literature, such as the lowest priority first heuristic from Expósito-Izquierdo, Melián-Batista, and Moreno-Vega (2012), but is modified to only focus on simple moves and to be fast to execute. Finally, our branching strategy is a well-known one, namely branching on the node with the best lower bound value. However, in the CPMP there are often ties for the best lower bound, thus we introduce a number of tie breaking criteria based on properties of the resulting bays when performing moves.

Our approach solves 161 more instances to optimality than the state of the art on the two main CPMP datasets from Caserta, Voß, Brabazon, and Tarantino (2009) and Bortfeldt and Forster (2012), and solves all instances to optimality on three further CPMP datasets. On instances already solved by state-of-the-art techniques, such as those in Tierney, Pacino, and Voß (2016) and van Brink and van der Zwaan (2014), our algorithm reduces the both the number of nodes and CPU time required to solve instances drastically, in some cases solving problems after only several search nodes that require tens of thousands of nodes with other approaches. Furthermore, our greedy, partial solution completion heuristic allows us to find feasible solutions on 58% of the instances for which we do not find the optimal value. These feasible solutions are in many cases better than the state-of-the-art biased random-key metaheuristic approach from Hottung and Tierney (2016).

We organize this paper as follows. After describing and formalizing the CPMP in Section 2 we provide a comprehensive literature review of the CPMP and related problems in Section 3. We describe our branch and bound approach in Section 4, including a discussion of the new dominance rule and branching strategy. We present our improved lower bound in Section 5 and give detailed computational results in Section 6, finally concluding and discussing future work in Section 7.

2. The pre-marshalling problem

Container terminals consist of several input/output areas where containers are brought into or out of the container terminal and a large buffer area, called the yard, as shown in Fig. 1. Containers are temporarily stored in this area while they are transferred between modes of transportation or are transshipped between liner shipping services. The figure shows one way of organizing the yard using rail mounted gantry cranes, in which the containers are organized into rectangular *blocks* containing multiple rows of container *stacks*.

A *bay* consists of a row of stacks and is shown in Fig. 2. Each bay contains several stacks with a common height restriction measured in *tiers* of containers. No containers may be stacked above the maximum height, which is usually due to the height of the crane.

The *pre-marshalling* of containers involves re-sorting them according to their retrieval times, ensuring efficient terminal operations when containers are removed from the bay. Formally, a bay contains a set of C containers (container 1, container 2, ..., container C), S stacks (stack 1, stack 2, ..., stack S), and a maximum height, modeled as a set T of tiers. The nonnegative integer-valued function $group(s, t)$ gives the exit time or *group*[2] of the container at stack s, tier t, with $group(s,t) = 0$ if no container is located at position s, t.

A stack is said to be *non-misoverlaid* if it has no *misoverlaid* containers, meaning no container is placed above a container with a lower group value. The goal of pre-marshalling is to remove all *misoverlays* from the bay with as few container moves as possible. Thus, each stack should be sorted in non-increasing order from the ground up according to the group of each container. A bay has no misoverlays iff $group(s,t) \geq group(s,t+1)$ for all $s \in S$, $t \in T \backslash \{|T|\}$. Note that multiple containers may have the same group value, and these containers do not misoverlay each other. The CPMP is shown to be NP-hard in Caserta, Schwarze, and Voß (2011).

In Fig. 3, we present an example solution for pre-marshalling a bay with misoverlaid containers. The solution shown is optimal, as at least three moves are required to solve the problem given that there are three misoverlaid containers.

2.1. Assumptions

The model of the CPMP we present is common in the literature (see Caserta, Voß, Brabazon, and Tarantino, 2009; Lee and Chao, 2009; Lee and Hsu, 2007 among others), but we discuss some of the implicit assumptions here anyway. First, the CPMP operates only on a single bay, even though there are many bays in the yard. This assumption follows from the literature (see, e.g., Lee & Hsu, 2007) and is based on several considerations, a primary one being safety constraints within many terminals that prevent inter-bay crane movements with containers. Furthermore, RMGCs can quickly move containers within a single bay, however, moving the RMGC between bays takes a significant amount of time. The single bay assumption therefore avoids move costs in the objective function. In addition, this assumption prevents us from having

[2] A group is called a priority in Expósito-Izquierdo, Melián-Batista, and Moreno-Vega (2012) and Caserta, Voß, Brabazon, and Tarantino (2009) and exit time in Tierney and Malitsky (2015).