PhD Thesis	Developing New Methods for Efficient Container Stacking Operations
PhD Student	Amir Hossein Gharehgozli
Promoter	Prof.dr. René de Koster
Co-Promoter	Prof. Yugang Yu
PhD Committee	Prof. Gilbert Laporte, Prof.dr. Jan Tijmen Udding, Prof.dr. Iris Vis, Prof.dr. Rommert Dekker, Prof.dr. Rob Zuidwijk, Dr. Bart Kuipers
PhD Period	01 September 2008 - 27 November 2012
Institute	Rotterdam School of Management, Erasmus University, Rotterdam, Netherlands
E-mail	AGharehgozli@rsm.nl, YYugang@rsm.nl, RKoster@rsm.nl

Abstract

This dissertation proposes, develops, and tests optimization methods to support the decisions of container terminal operators in the stacking area of a terminal. We focus on operational stacking problems, which are treated in four chapters. The first three chapters focus on sequencing a given set of container storage and retrieval requests in a single block. All three problems are complex and modeled as continuous time integer programming models. The objective is to minimize the makespan to carry out all requests. We try to optimally solve the problems as long as the complexity allows it. Otherwise, heuristic algorithms are developed to obtain near-optimal solutions. The last chapter links indirectly to the first three and will be discussed later. the models described in this thesis are new and have the ability to improve container stacking operations in practice. In the meantime, the model of chapter 4 has been applied to a real container terminal, where reductions of about 8% in makespan were obtained, compared to algorithms currently used in the Terminal Operating System (see the next Section).

In Chapter 2, we study an ASC, stacking and retrieving containers from a single block of containers. We formulate the problem as an asymmetrical traveling salesman problem (ATSP) and propose a two-phase solution method to optimally solve it. In the first phase, using the I/O points, we develop a new merging algorithm to patch subtours of an optimal solution of the assignment problem (AP) relaxation of the problem. We show that the first phase runs in a polynomial time, and often finds an optimal solution. Otherwise, a branch-and-bound (B&B) algorithm is used in the second phase to find an optimal solution of the problem.

In Chapter 3, we extend the problem by considering sets of open locations for stacking storage containers. We formulate the problem as a generalized ATSP (GATSP). In this problem, locations in the intersection of multiple sets make the problem more complex. Extra constraints are necessary to stack at most one container in such a location. We propose a three-phase solution method to optimally solve the problem. The main idea of the algorithm is to simplify the problem to an ATSP, by redefining the original travel time matrix. An optimal ATSP solution can be quickly obtained using the solution method studied in Chapter 2. An optimal ATSP solution provides an optimal GATSP solution either directly or through a B&B algorithm.

In Chapter 4, we extend the problem further. In this problem, two ASCs carry out the requests, which have different priorities. We model the problem as a multiple GATSP with precedence constraints (mGATSP-PC). Several extra practical constraints are considered: (1) the ASCs cannot pass each other and, for safety reasons, the ASCs must be separated by a safety distance; (2) each storage container must be stacked in a location selected from a set of available open locations; and (3) containers have different priorities. We develop an adaptive large neighborhood search (ALNS) heuristic capable of solving instances of realistic sizes.

In Chapter 5, we study a different problem which is indirectly related to the previous ones. The problem is how to stack incoming containers in a real-size container block shared by containers of multiple ships, one at a time. The objective is to minimize the expected number of reshuffles. We use a stochastic dynamic programming (DP) model and develop a decision-tree (DT) heuristic. The heuristic uses the results of the DP model for small-scale problems to generate generalized decision trees that can solve large-scale problems.