Optimization of hinterland container transportation and terminal operations

Zweers, B.G.

2021

Summary

Since the introduction of container transportation in the 1960s, the number of containers being transported worldwide has grown enormously. Moreover, the size of the vessels on which the containers are transported is increasing. Nowadays, the largest deep-sea vessels can carry more than 20,000 TEU, a standard size for a container. Consequently, a large number of containers are transhipped via deep-sea ports. For instance, almost 25,000 containers were being loaded and unloaded every day in the port of Rotterdam in 2019. The transshipment of that many containers is only possible if the containers are efficiently delivered to and picked up from the deep-sea port. This inland transportation, also called *hinterland transportation*, is the focus of this dissertation.

For hinterland transportation three modalities are available: train, barge, and truck. If containers are shipped by a truck, they can be delivered directly to their final destination. Since most companies do not have a rail or water connection, *inland terminals* are essential for the use of trains and barges. The barges and trains bring containers from the deep-sea port to the inland terminal, at which the containers are temporarily stored. Afterward, a truck delivers the container to its final destination. The use of trains and barges has many advantages over transportation by truck. First of all, it is considerably cheaper, and second, the CO_2-emission is much lower. A final advantage is that the use of fewer trucks could lead to a reduction in traffic jams. Therefore, the European Commission aims for a modal shift from trucks to barges and trains.

To utilize the advantages of barge and train transportation, several *operational planning problems* need to be solved. The problems studied in this dissertation are based on challenges faced by an inland container terminal in the port of Amsterdam. However, the methods that are developed are generally applicable. First of all, containers must be on the 'best possible' barge. To determine which barge is optimal, not only the transportation costs are taken into account but also other factors, such as the delay probability or the possibility that a barge cannot be unloaded. These types of problems are discussed in Chapters 2, 3, and 4. An efficient transshipment of containers at an inland terminal is a second aspect that is important to transport more containers per barge and train. Ideally, a container is moved as little as possible, and if it is moved, then only when the workload at the terminal is low. In Chapters 5, 6, and 7, these types of problems are solved.

In this dissertation, these two types of problems are studied from the perspective of an inland terminal. We use the same approach for every problem. First of all, the problem is modeled as a *mathematical optimization problem*. Afterward, an *exact algorithm* is developed that produces the optimal solution. However, calculating the optimal solution often takes too long to apply these algorithms in practice. Therefore, *heuristics* that produce fast solutions that are close to the optimal solution are also developed.

In Chapter 2, the best plan for the transportation of a set of containers located at multiple deep-sea terminals to the inland terminal is determined. The objective of this problem is to ship as many containers as possible per barge. However, at the same time, the barge must not visit too many terminals because that increases the probability of a delay for the barge. A third important aspect that is taken into account are the storage costs at both the deep-sea and inland terminal. This problem is modeled as an *integer linear optimization problem*. When using commercial solvers for this formulation, the optimal solution is obtained, but the running time can be more than a few hours for larger instances. We show that when this formulation is solved in two steps the running time is only a few seconds. The first step determines which terminals are visited, and which containers are shipped on which barge is decided in the second step. The solutions of this are nearly optimal. Finally, a heuristic is developed that simulates the behavior of a human planner. With that algorithm, we show that there are strong improvements possible by implementing the other two algorithms.

The problem of Chapter 3 is similar to that of Chapter 2, but there are two main differences. First, in Chapter 3 containers are transported both to and from the deep-sea terminal. Second, the number of containers that can be loaded and unloaded is unknown at the moment when the planning is done in Chapter 3. As a result, it might be that more containers are loaded on the barge than can be unloaded at the terminal. We treat the number of containers that can be loaded and unloaded at a single terminal as a stochastic variable. Afterward, the problem is modeled as a *stochastic problem with recourse*. Subsequently, this problem is solved using a technique called *Sample Average Approximation*. This method converges to the optimal solution if sufficiently many samples of the stochastic variable are used. However, it is also possible to generate faster solutions. We develop a heuristic in which the original problem is simplified, such that the optimal solution can be calculated using standard techniques of *stochastic programming*. This optimal solution can then be used to replace the stochastic variable by a deterministic value. This heuristic produces solutions that are better than those of other heuristics in which a deterministic value replaces the stochastic variables.

In Chapter 4, the best transportation plan is not determined for a set of containers but a single container. This container is transported through a network in which the travel times are stochastic. Moreover, there is a probability that a leg is overbooked. The goal of the problem is to find the cheapest route for which the shipment arrives at the final destination before a specific deadline. Each route has a certain on-time arrival probability because of the stochasticity. Since determining an acceptable on-time arrival probability beforehand is hard, *Pareto-optimal* solutions are constructed. In these solutions, the costs of a route are compared with the probability of arriving before the deadline. Besides an optimal algorithm based on *dynamic programming*, we also give a heuristic in which a *risk measure* replaces all stochastic variables. This risk measure is a deterministic value that is used in an integer linear optimization problem. By varying the risk acceptance in the risk measure, this heuristic can also be used to construct Pareto-optimal solutions.

In Chapter 5, we change our focus to stacking problems for containers at terminals. When a container needs to leave the terminal, it frequently occurs that other containers are stacked on top of it. These containers need to be relocated to other stacks while a truck is waiting at the terminal. These moves are called *relocation moves*. An alternative is that containers are already positioned in the right order when it is less busy at the terminal, known as *pre-marshalling moves*. The problem of pre-marshalling moves is that more moves are needed than relocation moves. In this chapter, we introduce a new type of movement, namely the *pre-processing moves*. These moves are performed when the terminal equipment is idle, but in contrast to the pre-marshalling moves, the containers do not need to be positioned entirely in the right order. Consequently, fewer pre-processing moves are performed than pre-marshalling moves, and the moment the pre-processing moves are performed is better than for the relocation moves. We formulate two new problems in this chapter, and in Chapters 6 and 7, solution methods are presented for these problems.

In the problem studied in Chapter 6, the weighted sum of the number of pre-processing and relocation moves is minimized. We derive a heuristic for this problem in which the first containers to be positioned correctly are those that leave the terminal the latest. We need to calculate the expected number of relocation moves for the resulting bay to know if these moves result in an improvement. In this chapter, a method is developed that uses a few decision rules to determine the expected number of relocation moves. The optimal solution is calculated using a *branch-and-bound algorithm*, but for problem instances consisting of many containers, this method needs a couple of hours.

In Chapter 7, the number of pre-processing moves that can be performed is limited. The optimal algorithm of the previous chapter can be used for this problem with a few adjustments. Nevertheless, in this chapter, we use the branch-and-bound method also in a heuristic in which the remaining relocation moves are estimated. Another heuristic presented in this chapter tries to position every stack's top container in a correct position. Eight different ways in which this can be done are derived, and the one resulting in the largest decrease in the objective function is chosen. In the problem studied in Chapter 6, we assume the crane that handles the containers to move during the pre-processing phase. Nevertheless, in this chapter we relax this assumption. The solutions for the original problem can be used in an integer linear optimization problem to determine how many moves need to be performed in which part of the terminal.

Finally, we study problems from a different perspective in Chapter 8. In the previous chapters, numerical experiments are used to evaluate the quality of heuristics. In contrast, in this chapter, we derive algorithms for which it can theoretically be proven that a given factor bounds the difference between their solution and the optimal solution. We develop these so-called *approximation algorithms* for optimization problems with two levels of capacity. Well-known problems with a single tier of capacity are the *Multiple Knapsack Problem*, the *Maximum Coverage Problem with Knapsack Constraint*, and the *Capacitated Facility Location Problem*. We extend these problems by partitioning the knapsacks in clusters, which also impose a capacity constraint. We introduce a decision variable that determines how much capacity of the cluster is dedicated to which knapsack. This decision variable can be applied to extend existing approximation algorithms based on linear programming. For specific problems, the approximation algorithm's performance guarantee remains the same for the extensions with clusters, and for other problems, it only becomes slightly worse.