NTNU
Norwegian University of
Science and Technology

A Heuristic Approach to the Two-Dimensional Roll-on Roll-off Liner Shipping Stowage Problem

Jone Hansen
Ivar Hukkelberg

Industrial Economics and Technology Management
Submission date: June 2016
Supervisor: Kjetil Fagerholt, IØT
Co-supervisor: Magnus Stålhane, IØT
Jørgen Glomvik Rakke, Wallenius Wilhelmsen Logistics

Norwegian University of Science and Technology
Department of Industrial Economics and Technology Management

Problem Description

In this master thesis, the Roll-on Roll-off ship stowage problem (RSSP) is considered. A problem that occurs when transporting the cargo is shifting, which means temporarily moving some vehicles to make an entry/exit route for the vehicles that are to be loaded/unloaded at a given port. Therefore, it is important to develop a good stowage plan that brings as much optional cargo as possible, utilizing all available area, and at the same time keep the cost spent on shifting as low as possible.

Practical stowage planning problems in Roll-on Roll-off (RoRo) Liner Shipping is difficult to solve to optimality within acceptable time limits. Our aim is to use heuristics to solve the RoRo stowage planning problem, providing good feasible solutions within reasonable time. The objective is to maximize the sum of revenue from optional cargoes, minus the penalty costs incurred when having to shift cargoes.

Preface

This master thesis is the final result of the work for our Master of Science degree at the Norwegian University of Science and Technology (NTNU). The master thesis was written during the spring of 2016, in the course Managerial Economics and Operations Research at the Department of Industrial Economics and Technology Management. The thesis is a continuation of our project thesis, done in the fall semester 2015.

The purpose of the thesis has been to develop a solution method for the operational stowage plan decisions within RoRo transportation. We would like to thank our supervisors Kjetil Fagerholt and Magnus Stålhane for their guidance, contributions and valuable feedback throughout the semester. We would also like to thank our contact in the case company, Jørgen G. Rakke, for useful input about the RoRo liner shipping industry and contribution of real case data.

Trondheim, June 08, 2016.

Jone R. Hansen

Ivar Hukkelberg

Cover photo: Wallenius Wilhelmsen Logistics (2016a)

Abstract

Roll-on/Roll-off (RoRo) ships represent the primary source for transporting vehicles and other types of rolling material over long distances. In this master thesis, we focus on operational decisions related to stowage of cargoes for a RoRo ship voyage visiting a given set of loading and unloading ports. Decisions, such as which cargoes to carry and how to stow the vehicles carried during the voyage, are considered.

We focus on stowage of one single deck, which is an essential building block in solving the problem for multiple decks, i.e. for the whole ship. By focusing on stowage on one deck, this can be viewed as a special version of a 2-dimensional packing problem with a number of additional considerations. One important consideration is the placement of the vehicles, where one wants to place vehicles that belong to the same shipment close to each other to ease the loading and unloading. Another important consideration in this problem is shifting, which means temporarily moving some vehicles to make an entry/exit route for the vehicles that are to be loaded/unloaded at the given port.

Our approach to solving the 2DRSSP splits the problem into two phases. First, we solve the stowage problem for a given deck. Then, the shifting evaluation problem is solved to evaluate the quality of the stowage plan with respect to shifting cost. Two mathematical models are made, describing these problems. In addition, a 2-phase heuristic is developed using components from the greedy randomized adaptive search procedure (GRASP) and the adaptive large neighborhood search (ALNS) procedure. The heuristic first determines which cargoes to carry, in order to maximize the revenue generated from carrying these cargoes, while ensuring a feasible stowage plan exists for the given cargoes. Then, vehicles are relocated in order to ease the loading and unloading. Several problem-specific destroy and repair operators are built in order to improve the heuristic's performance.

Computational results show that the mixed integer programming (MIP) solver is able to solve small instances. The proposed heuristic provides close to optimal solutions to these same instances and is able to handle realistically sized problem instances.

Sammendrag

Roll-on/Roll-off (RoRo) skip er det foretrukne valget for transport av kjøretøy og andre typer last som kan rulles av og på over lange avstander. I denne masteroppgaven fokuserer vi på operasjonelle beslutninger knyttet til stuing av last på et RoRo skip, for en seilas der havnebesøkene er gitt. Beslutninger som hvilke laster som skal tas med og hvordan bilene skal stues under seilasen blir vurdert.

Vi fokuserer på stuing av ett dekk, som er et viktig sub-problem for stuasjeproblemet for flere dekk. Ved å fokusere på ett dekk kan problemet betraktes som en spesialversjon av et todimensjonalt pakkeproblem, med ytterligere hensyn. En viktig betraktning er plasseringen av kjøretøyene. Her ønsker man å plassere kjøretøy som tilhører den samme forsendelsen nær hverandre for å forenkle lasting og lossing i havnene. Et annet viktig aspekt ved dette problemet er shifting, som betyr å midlertidig flytte noen kjøretøy for å få andre kjøretøy ut av skipet, eventuelt sørge for at kjøretøy når frem til sin fastsatte plassering på dekk under lasting.

Vår tilnærming deler problemet inn i to faser. Først løser vi stuasjeproblemet for et gitt dekk. Deretter blir shifting-evalueringsproblemet løst for å vurdere kvaliteten av stuasjeplanen med hensyn til shifting-kostnaden. To matematiske modeller er laget for å beskrive problemene. I tillegg er en to-fase heuristikk utviklet, basert på en grådig søkeprosedyre (GRASP) og et nabolagssøk (ALNS). I første fase beslutter heuristikken hvilke laster som skal tas med for å maksimere inntekten på seilasen, samtidig som heuristikken sjekker at en gyldig stuasjeplan eksisterer. I fase to blir bilene flyttet rundt på dekket, i håp om å finne en stuasjeplan som gjør lastingen og lossingen enklere. Flere problemspesifikke operatorer er laget for å øke sjansene for smarte replasseringer av bilene i fase to.

Resultatene viser at programvaren brukt for å løse problemene er i stand til å løse små instanser av heltallsproblemene eksakt. Den foreslåtte heuristikken gir nær optimale løsninger på de små instansene, og er i stand til å løse problemer av realistisk størrelse.

Contents

List of Figures

List of Tables

1. Introduction

Today, maritime transportation is the most important mode of transportation in international trade. Over 90% of the world's trade is carried by sea, according to IMO (2016). Figure 1.1 shows how transportation by sea has increased steadily during the last years, with an exception during the financial crises, and how the trade historically has exceeded the growth in GDP. In 2014, the total shipments by sea in terms of volume reached 9.86 billion tons and had a 3.4% growth (UNCTAD, 2015). Figure 1.1 gives no indication that this trend is about to turn, and from this a continuous growth in the maritime transportation segment is expected to come in the following years. However, even though the market is increasing, the competition in maritime transportation is growing, and new laws regarding sustainability are making this segment tougher for the companies to compete in (Christiansen et al., 2007).

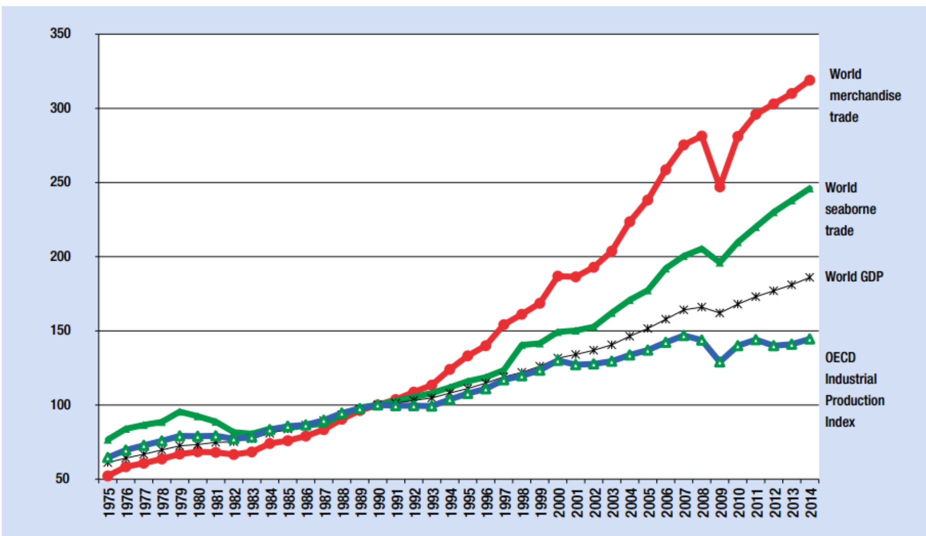
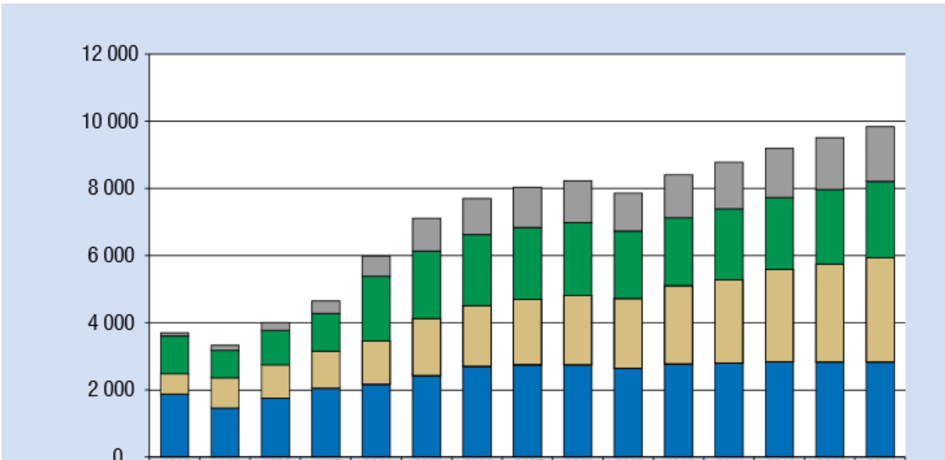


Figure 1.1: The OECD Industrial Production Index and indices for world GDP, merchandise trade and seaborne shipments (1975-2014) (base year 1990=100). Source: UNCTAD (2015)

UNCTAD (2015) divide the total seaborne trade into different categories, according to the type of cargo transported. Figure 1.2 shows the total volume traded during 2014 and the amount each of the different vessel types transported. The largest segment was dry cargo, i.e. container, major bulks and other dry cargo, and accounted for over two-thirds (71.3%) of the total volume. The increase in containerized trade was 5.6% compared to only 2.4% for general cargo and break bulk together during 2014 (UNCTAD, 2015). Due to the increased competition in and between the shipping segments, cost reducing measures are important in order to stay competitive. The use of operation research is proposed to help making better decisions, in order for the companies to continuously improve their fleet utilization.



	1980	1985	1990	1995	2000	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014
Container	102	152	234	371	598	969	1 076	1 193	1 249	1 127	1 280	1 393	1 464	1 544	1631
Other dry cargo	1 123	819	1 031	1 125	1 928	2 009	2 112	2 141	2 173	2 004	2 022	2 112	2 150	2 218	2272
Five major bulks	608	900	988	1 105	1 295	1 709	1 814	1 953	2 065	2 085	2 335	2 486	2 742	2 923	3112
Oil and gas	1 871	1 459	1 755	2 050	2 163	2 422	2 698	2 747	2 742	2 642	2 772	2 794	2 841	2 829	2 826

Figure 1.2: International seaborne trade for different categories of vessel groupings for selected years in millions of tons loaded. Source: UNCTAD (2015)

1.1 Shipping Segments and Planning Decisions

The maritime transportation industry can be divided into three different modes according to Lawrence (1972): Industrial, tramp and liner shipping. In industrial shipping, the company both owns the cargoes transported and the ships transporting them, and the aim is to minimize the cost of the voyages. In tramp shipping, the ships do not follow a predefined route, but instead sails to different ports based on available cargoes, i.e. similar to a taxi. Besides the mandatory contracted cargoes it needs to carry, available spot cargoes are carried in order to maximize profit. At last there is liner shipping, where ships follow fixed routes with a public schedule and aim to maximize profit on the given routes, illustrated in Figure 1.3. By carrying optional cargoes available at the ports along a fixed route, the revenue of these voyages increases. As long there is available space on the ship's decks, as many optional cargoes as possible are carried.

Figure 1.3: Two possible voyages for a Roll-on/Roll-off liner shipping company. Source: Wallenius Wilhelmsen Logistics (2016b)

The increasing competition in maritime transportation forces the shipping companies to utilize the fleet in the best possible way, and good planning decisions are of that reason crucial. According to Crainic and Laporte (1997), the common approach in most industries is to divide planning decisions into different levels, which is also the case for the maritime shipping industry (Christiansen et al., 2007). Planning decisions are divided into three levels: Strategic, tactical and operational. Strategic planning is concerned with a time horizon of several years, and typically involves decisions such as determining the fleet size and mix, e.g. (Pantuso et al., 2015). Tactical planning problems have a time horizon from a month up to a year. A typical tactical problem is the fleet deployment problem, e.g. (Andersson et al., 2015). Operational planning is short-term decisions, i.e. day-to-day, where the time factor plays an important role (Crainic and Laporte, 1997), and typical problems are stowage planning and sailing speed decisions.

It is common to accept the decisions made in higher planning levels when making decisions at lower levels. Thus, one will usually take the strategic and tactical plans as given, and use these decisions as input data when developing plans on the operational level. An example is the stowage problem in liner shipping, where a ship's voyage and which mandatory cargoes to carry are given, and used as input data to create a good stowage plan.

1.2 The Roll-on/Roll-off Industry

According to UNCTAD (2015), Roll-on/Roll-off (RoRo) ships are classified as general-cargo ships and belong to the liner shipping mode. The RoRo shipping industry has a rather small fleet, in comparison to containerized shipping (Lindstad et al., 2012). Figure 1.4 illustrates different types of RoRo ships with various abilities. Two examples are ConRo and RoPax, both categorized as RoRo ships, where the former is a combination of a RoRo and a container ship, and the latter refers to cruise ferries. The most common RoRo vessels within the liner shipping segment are the ordinary Roll-on Roll-off carriers, PCC (Pure Car Carrier), which only carry cars, and PCTC (Pure Car and Truck Carrier), which carries both cars and trucks (DanishShipFinance, 2016). The difference between an ordinary RoRo carrier and a PCC/PCTC is that the RoRo vessels are constructed to carry both ordinary vehicles and larger cargoes with irregular sizes and shapes, known as breakbulk cargo. Even though shipments done by RoRo ships are small compared to the total transportation done by sea, it is still the preferred choice when transporting vehicles and other types of rolling material around the globe. However, due to more efficient short sea feeder traffic in and out of main ports, the containerized fleets are becoming more and more of a threat to the RoRo segment. Therefore, it is important for the RoRo industry to continuously improve and become more effective, maintaining the position as the leading maritime transportation method for this type of cargo.

RoRo ships transport a wide variety of cargoes. Many of them are not standardized and of that reason not suited for transportation by container ships, as

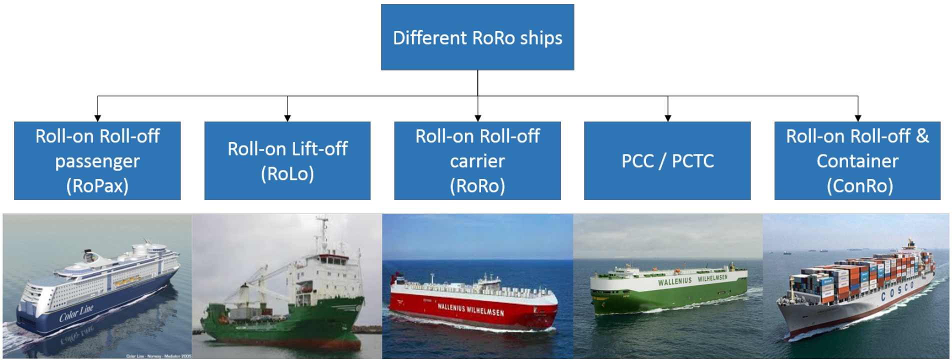


Figure 1.4: Different types of RoRo ships, each with their unique ability and characteristic. Sources (from left to right): Medialon (2014), Ship World Brokerage (2014), Wallenius Wilhelmsen Logistics (2016c), Wallenius Wilhelmsen Logistics (2016c), and Port Technology (2014)

they do not fit into the standardized containers. The cargoes transported usually consist of vehicles, such as cars and trucks, but also larger cargo, i.e. breakbulk, such as heavy rolling machinery, trains and yachts, as illustrated in Figure 1.5. Because of the large variety of the cargoes transported, the design of a RoRo vessel is rather complex. A typical ship has multiple decks, each with different requirements in both height and carrying capacity. During loading, the vehicles typically enter the ship through a ramp placed at the stern or the ship's port side and are placed on one of the ship's decks. A major problem that occurs when loading or unloading the cargoes is shifting. Shifting means temporarily moving some cargoes to make an entry/exit route for the cargo that is to be loaded/unloaded at a given port. This forces the ship to stay longer in the port and increases the cost for the voyage, as dock workers may work overtime, delivery dates may not be kept, etc. Therefore, it is important to develop a good stowage plan which brings as many cargoes as possible, utilizes the available space on the decks, and at the same time keeps the cost and time spent on shifting as low as possible.

Due to the increased competition, the RoRo providers have to make smart planning decisions in order to stay competitive. Looking at the operational level of planning, the greater part of research regarding RoRo ships focuses on safety and stability, such as Kreuzer et al. (2007). Despite its importance, research within stowage on RoRo ships is scarce, and to the authors' knowledge, only the research conducted by (Øvstebø et al., 2011a,b) exists on stowage on board RoRo ships.

Figure 1.5: RoRo vessel. Source: Wallenius Wilhelmsen Logistics (2014)

1.3 The Stowage Problem

One of the operational planning decisions liner shipping faces is the creation of stowage plans. The aim with a stowage plan is to find the optimal placement of cargoes for a ship, so that the decks are utilized in the best possible way and the most profitable set of cargoes is chosen (Øvstebø et al., 2011a). Due to limited capacity on a ship, all cargoes available on a given voyage cannot always be carried. One usually distinguishes between mandatory and optional cargoes. A mandatory cargo is contractual bounded and needs to be carried. Optional cargoes are additional cargoes that the ship can pick up at a loading port and deliver to an unloading port it visits during the voyage, if there is available space on the decks. Carrying optional cargoes will increase the revenue generated on the voyage. Therefore, it is an important decision for the RoRo providers to decide which set of optional cargoes to carry in order to be most profitable while not exceeding the ships capacity.

Another goal with stowage plans is to reduce the shifting done in each port. Since cargoes are picked up and delivered at different ports, a good operational plan on where to stow each cargo is crucial in order to ease the loading/unloading process and to reduce the time spent in a port.

In addition to select which cargoes to carry and reduce shifting, a stowage plan should consider the stability of a ship. Stability calculation is crucial for a ship so that it does not heel. There have been incidents concerning heeling because of poor stowage planning. One of the latest one to date was the Roll-on Roll-off vessel Osaka, see Figure 1.6, which started tilting because the center of gravity on the ship was higher than recommended (MAIB, 2016). The reason for this was that the real weight of some cargoes placed on the upper decks was higher than what the pre-generated stowage plan showed. An updated stowage plan with correct data was never made, and caused the ship to heel. The accident was quite expensive and it shows the importance of having a proper stowage plan.

Figure 1.6: The RoRo vessel "Osaka" stranded on Bramble Bank, south of Southampton. Source: MarineTraffic (2016)

1.4 Scope and Thesis Outline

This master thesis is a collaboration with one of the world's largest RoRo liner shipping companies, operating more than 50 RoRo ships all over the world. There were pointed to several reasons for why solving the stowage problem is important for the RoRo liner shipping companies. Increased competition and safety, i.e. correct stability calculations, are two key points that show the importance of stowage plans. A good stowage plan has many benefits, such as reduced amount of shifts and optimal utilization of the decks, so they can bring as much optional cargo as possible. A plan like this would help companies to increase the revenue generated on the voyage.

In this master thesis, we focus on stowage of one single deck, which is an essential building block in solving the problem for multiple decks, i.e. for the whole ship. The problem we aim to solve has two objectives, one main objective, and one secondary objective. The main objective is to maximize the revenue generated for a voyage, by choosing the best combination of optional cargoes to bring. The secondary objective is to reduce the number of shifts. According to the company we collaborate with, the revenue generated for bringing an extra optional cargo always outweighs the shifting. Therefore, one does always bring an extra cargo if there is room for it on the deck, no matter how much the number of shift increases.

As mentioned in Section 1.2, very little research has been conducted on stowage problems for RoRo ships. The only ones to date, to the authors' knowledge, are Øvstebø et al. (2011a,b). Øvstebø et al. (2011a) look at operational decisions for a RoRo ship with a predefined route and the aim is to decide upon a deck configuration with respect to height, which optional cargoes to carry, and how to stow vehicles carried during the voyage. However, some of the assumptions made in Øvstebø et al. (2011a) may be too restricting, limiting the practical use of the resulting stowage plan. Of that reason, in this master thesis, we aim to solve the problem by not relying on those assumptions.

The master thesis is a continuation of the work done in the project thesis by Hansen and Hukkelberg (2015), and the same problem is considered. Our contributions in this thesis are an improved mathematical models, focusing on describing the real problem as good as possible, and a new solution method that has the ability solve realistically sized instances. Our solution method uses a heuristic approach, combining a greedy randomized adaptive search procedure (GRASP) heuristic and an adaptive large neighborhood search (ALNS) heuristic, that takes both the objectives into account, i.e maximize the revenue and reduce the shifting as much as possible.

The remaining outline of this thesis is as follows: Chapter 2 introduces a literature review which describes relevant problems related to the RoRo ship stowage problem. In Chapter 3, a detailed problem description of the two-dimensional packing problem with respect to RoRo liner shipping is given. Further, in Chapter 4, we present our mathematical formulation to the problem, both the stowage model and the shifting evaluation model. In Chapter 5, a heuristic solution method is proposed. In Chapter 6, the test instance creation procedure is given. A computational study is conducted in Chapter 7. Finally, concluding remarks and proposed future research are given in Chapters 7 and 8, respectively.

2. Literature Review

In this chapter, relevant literature to the RoRo stowage problem is addressed. In Section 2.1, stowage problems that arise in different maritime transportation segments are discussed. Then, in Section 2.2, a review of cutting and packing problems is presented, as stowing vehicles on a deck is a packing problem. Finally, a summary of the most important findings is given in Section 2.3. Section 1.1 briefly presented some research within RoRo transportation segment, both at a strategic, tactical and operational level. In this chapter, only research connected to the RoRo stowage problem is presented. Thus, literature regarding other operational decisions within RoRo transportation is excluded.

2.1 Stowage in Maritime Transportation

All vessels used in maritime transportation have a limited storage capacity, whether it is a RoRo vessel, container ship, or a bulk carrier. Thus, the stowage problem arises as an operational decision that must be solved for every voyage. The complexity of the stowage problem depends on the type of vessel used, the number of cargoes, stability restrictions, and many other problem specific restrictions. In this section, we review published literature regarding stowage problems in three segments of maritime transportation. First, literature on stowage problems in the RoRo segment are thoroughly reviewed in Section 2.1.1, as this is the stowage problem we address in this thesis. Then, stowage problems in containerized transportation and bulk shipping are reviewed in Sections 2.1.2 and 2.1.3, respectively. Even though these stowage problems are different from the RoRo stowage problem, the literature may provide valuable insights and ideas, transferable to the RoRo stowage problem.

2.1.1 Roll-on/Roll-off Vessels

The RSSP presented by Øvstebø et al. (2011a), aims at deciding a deck configuration with respect to height, which optional cargoes to carry, and how to stow the vehicles carried during the voyage, given a predefined route. Øvstebø et al. (2011a) propose a mixed integer programming (MIP) model and a heuristic method for solving this problem, where the objective is to maximize the sum of revenue from optional cargoes, minus the penalty costs incurred when having to move cargoes when loading/unloading along the route. For modeling purposes, Øvstebø et al. (2011a) divide each deck into several logical lanes, into which the vehicles are lined. The vehicles enter the ship at the stern and are unloaded according to the last in-first out (LIFO) principle. However, dividing the decks into lanes may be too restricting, limiting the possibilities of finding good solutions.

The mentioned paper by Øvstebø et al. (2011a), is to the authors' knowledge the only paper explicitly dealing with the RoRo stowage problem. However, there do exist several publications regarding fleet deployment decisions for RoRo ships, see for example Fagerholt et al. (2009); Chandra et al. (2015); Andersson et al. (2015). The common approach is to measure a ship's storage capacity in car equivalent units (CEU), where one CEU equals the area usage of a Toyota Corona. This capacity measure is then used to ensure that the carried cargoes do not exceed the ship's capacity. However, as discussed in the following section, this does not necessarily ensure that a feasible stowage plan exists. An extreme case could be that all carried cargoes are breakbulks, thus some of the decks cannot be used to stow these cargoes. Even though the capacity constraints are satisfied, a feasible stowage plan does not exist and a voyage carrying these cargoes is infeasible. Fischer et al. (2016) suggest to introduce capacity classes in order to reduce the probability of creating infeasible voyages. Here, each deck is classified according to which types of cargoes it can handle. While this certainly reduces the probability of infeasibility, a stowage problem still needs to be solved in order to prove feasibility.

Øvstebø et al. (2011b) aim to solve a combined routing and stowage problem for one ship, which in terms of planning level is placed between the tactical fleet deployment decisions and the operational stowage decisions. The single-vessel RoRo ship routing and stowage problem (1-RSRSP) presented by Øvstebø et al. (2011b), considers a RoRo ship that is set to sail between two geographical regions, picking up cargoes in the first region and delivering the same cargoes in the second region. Decisions must be made regarding which cargoes to pick up, how much to carry from each cargo, the sequence and timing of pick-ups and deliveries, and the stowage of the cargoes onboard the ship (Øvstebø et al., 2011b). The objective is to maximize the revenue minus penalty for mandatory vehicles not carried, travel cost, and cost of ship usage. Here, the ability to select cargo quantities within an interval in included, which was fixed in Øvstebø et al. (2011a). Øvstebø et al. (2011b) give a MIP model and a tabu search heuristic for solving the 1-RSRSP. Because of the increased size of the problem, in comparison with the RSSP, the optimization software was only able to find feasible solutions in a reasonable amount of time for the tiny problem instances. For larger instances, a problem-specific tabu search heuristic was used, providing feasible and practical stowage plans.

2.1.2 Container Ships

The vast majority of literature regarding stowage in maritime transportation focuses on stowage problems for container ships. The containers are stacked on top of one another, and when dispatching a certain container, containers stacked on top of it need to be removed. The objective in container stowage problems is therefore often to minimize the loading/unloading time of all containers (Ambrosino et al., 2004) or the number of container movements (Avriel et al., 1998).

The early contributions regarding the container packing problem are given by Martin et al. (1988), and Aslidis (1989). Avriel and Penn (1993) give a 0-1 binary linear programming formulation to find the optimal solution for stowage of a finite number of containers. It was shown that finding the optimal solution using the model was difficult, due to a large number of binary variables. Kang and Kim (2002) present a mathematical formulation of the problem and solve it using a heuristic algorithm combining a greedy heuristic and tree search. Other contributions to the container packing problem includes (Wilson and Roach, 2000), (Avriel et al., 2000), (Dubrovsky et al., 2002), and (Ambrosino et al., 2004). More recently, Pacino et al. (2011) presented a 2-phase IP-model for solving the container stowage problem. They were able to solve the majority of the industry test instances to near optimality, within a time limit of 330 seconds. For especially hard instances, a heuristic was proposed developed.

Recently, a paper considering stowage of general cargo onto platform supply vessels was published, given by Seixas et al. (2016). For a given route assigned to a supply vessel they seek to determine the optimal two-dimensional positioning of deck cargoes so that the overall profit is maximized, while ensuring that several safety and operational constraints are respected. What makes this paper particularly interesting is that due to safety regulations, stacking the containers is not allowed, unlike the typical container packing problems. Thus, in terms of mathematical modeling, the resulting problem can be seen as a rich variation of the two-dimensional knapsack problem, since some cargoes may wait for a later trip (optional cargoes), according to the authors. Packing containers on a deck without stacking has many similarities with stowing vehicles on a deck, as they both result in a two-dimensional packing problem, reviewed in Section 2.2. However, the loading and unloading procedure still differs. Where a container is lifted straight up from its position, a vehicle's entry/exit route needs to be calculated for each vehicle in the RoRo ship stowage problem (RSSP).

Seixas et al. (2016) state that the papers devoted to oil supply operations consider a simplification regarding deck occupation, where each cargo has an area attribute, and any solution whose total area does not exceed the available deck area is feasible. In their approach, they are concerned with the exact cargo positioning on the deck. For instance, a 16 m^2 (4x4 m) cargo cannot be assigned to a 42 m^2 (3x14 m) deck area, if the actual dimensions are taken into consideration (Seixas et al., 2016). In addition, several other considerations are taken, such as packing constraints, weight limitations, the adjacency of delivery/pick-up cargoes, positioning of dangerous and refrigerated cargoes. Despite the substantial difference in the loading and unloading procedure, this problem has many similarities with the RoRo stowage problem.

2.1.3 Bulk Carriers

Hvattum et al. (2009) present the tank allocation problem (TAP), which considers a ship with a given number of tanks, sailing a predefined route. The problem is then to decide which tanks to allocate the cargoes to along the voyage. Constraints such as capacity, stability, and hazardous materials regulation must be satisfied. Hvattum et al. (2009) state that the crucial issue is to identify a feasible solution, as any feasible tank allocation plan is as good as any other. This is due to the fact that the cost of tactical decisions, such as where to sail and which loads to carry, by far outweighs the cost of operational decisions, such as tank allocation (Hvattum et al., 2009). Different objective functions are proposed due to this fact, such as finding a feasible stowage plan, to minimize the inconvenience of cleaning tanks, or to maximize the average capacity of vacant tanks during the ship's route. A mathematical model that computes feasible solutions within a reasonable time limit is given.

Fagerholt and Christiansen (2000a) address a multi-ship pickup and delivery problem with time windows and multi-allocation problem, named a combined Ship Scheduling and Allocation Problem (SSAP). Here, each ship in the fleet is equipped with a flexible cargo hold that can be partitioned into several smaller holds. The SSAP is to decide upon routing, partitioning of cargo holds, and allocation of cargo for each ship. Fagerholt and Christiansen (2000a) solve the problem in two phases. Candidate schedules for each ship are generated in the first phase, and a set partitioning problem is solved in the second phase. Fagerholt and Christiansen (2000b) further examine procedures for finding routes with optimal allocation of cargoes to ship. A dynamic programming (DP) approach is used to solve the Traveling Salesman Problem with Allocation, Time Windows and Precedence Constraints (TSP-ATWPC). The computational results show that the proposed algorithm works for the TSP-ATWPC, and several instances of the real ship scheduling problem are optimally solved within a reasonable timeframe (Fagerholt and Christiansen, 2000b).

2.2 Two-Dimensional Packing Problems

Wäscher et al. (2007) present a typology of cutting and packing problems, partially based on the original ideas of Dyckhoff (1990). According to this typology, the RoRo ship stowage problem is classified as either a two-dimensional knapsack problem (2KP) or a multiple heterogeneous large object placement problem (MHLOPP). In both of these problems, a fixed number of small items has to be allocated to a lower number of large objects, where each item increases the profit by a specified value if placed. This is transferable to the RSSP, where all vehicles (small items) from the mandatory cargoes and the carried spot cargoes have to be allocated on one of the ship's decks (large objects). Even though all vehicles naturally have a given height, which should imply a third dimension, 3D-packing is usually interpreted as a packing problem where the items are stacked in height. We limit the literature study by only reviewing paper regarding 2KP or MHLOPP, as the RSSP is classified under these terms. The reviewed literature do not distinguish between these two types of packing problems, as the difference between them are marginal. The 2KP considers strongly heterogeneous small objects, and the MHLOPP considers weakly heterogeneous small objects.

Hadjiconstantinou and Christofides (1995) present an exact tree-search procedure for solving the 2KP, where the algorithm limits the size of the tree search using a bound derived from a Lagrangian relaxation of a binary formulation of the problem. For large problems, a heuristic approach is necessary to provide good solutions. Hopper and Turton (2001) suggest two types of hybrid algorithms to solve the 2KP. One of the methods is based on a bottom-left rule. Starting with a large empty item, the first item is inserted in the bottom left corner. Then, the next selected item is placed on the upper right corner and slid as far down as possible, and then moved as far to the left as possible. This procedure is repeated until there exist no possible insertions. The quality of the layout, which is constructed by using the bottom-left rule, depends on the sequence in which the rectangles are presented to the routine (Hopper and Turton, 2001). Thus, several metaheuristics are given in order to improve the selection sequence.

Leung et al. (2012) give another approach to solving the 2KP. For some given possible insertion locations, the item with the best fitness value is inserted in one of the corresponding locations. The fitness value is based on the length, width, and value of the object, in addition to the trim loss and height of the large object. The procedure is more computationally expensive than the bottom-left rule, but yields better results. Gonçalves (2007) present a genetic algorithm-heuristic for the orthogonal packing problem, identical to the 2KP. Genes are used to obtain the insertion sequence, using an evolutionary strategy to improve the insertion sequence in each iteration. Each item is placed in the location with the largest possible empty rectangular space.

In the reviewed papers, the insertion sequence of the items seems to be a decisive factor in order to provide good packing results. Thus, this is taken into consideration when developing our stowage heuristic. However, the proposed solution methods are not suitable for the RSSP as it is now. In RoRo transportation, we are not only concerned with maximizing the value of the packed cargoes, as is the objective of the presented papers. When stowing vehicles on a RoRo ship, the quality of the stowage plan is also important, in order to ensure smooth loading and unloading in each port. In addition, constraints such as stability and compactness must be considered when solving the RSSP, where the latter corresponds to placing vehicles from the same cargo next to each other. The additional constraints and other problem-specific aspects limit the use of existing 2KP solution methods.

2.3 Summary

The operational decision of where to stow the cargoes on board the ship is an important decision, regardless of the ship type. For some vessel types, such as bulk carriers with tanks, the stowage problem is a feasibility issue. However, for RoRo transportation, the time spent in port is highly dependent on the loading and unloading efficiency. Thus, the quality of the implemented stowage plan for each voyage is important for the RoRo providers. Despite its importance, literature on the stowage problem for RoRo ships is scarce.

The reviewed literature shows that usually, only the ship's capacity is used when performing tactical decisions. As the stowage problems are often complex, this is a natural simplification when solving RoRo fleet deployment problems. However, this induces the need for solving the stowage problem for each voyage, both to prove that the voyages are feasible with respect to stowage and in order to reduce the time spent at a port.

Øvstebø et al. (2011a) bring important insights and modeling components to the RoRo ship stowage problem. However, the representation of the decks is highly simplified, neglecting pillars and other blocking obstacles. As Seixas et al. (2016) point out, the actual dimensions have to be considered when stowing the cargoes, thus, the deck layout is an important factor. The logical lanes and LIFO principle used by Øvstebø et al. (2011a) may also be questioned. In practice, the vehicles are not reversed out of a logical lane and unloaded at the stern, but take the least inconvenient path to the exit ramp.

3. Problem Description

In this chapter, the RoRo ship stowage problem is presented and discussed in detail. First, a short introduction to the general RoRo ship stowage problem (RSSP) is given. Then, a detailed description of the two-dimensional RoRo ship stowage problem for one deck (2DRSSP) is presented, which is the problem we aim to solve.

The RSSP considers a RoRo ship carrying a number of cargoes along a voyage with a predefined set of loading and unloading ports to visit. At each port, a given number of mandatory cargoes is present. There is also a given number of optional cargoes the ship may load at each loading port. In the RSSP, the decisions that need to be made are which optional cargoes to bring and where to place the cargoes on the ship. This gives the RSSP two objectives to fulfill: (1) to utilize the decks on a RoRo ship in such a way that it can carry as much optional cargo as possible, and (2) to place the cargoes on the deck in such a way that it minimizes the number of shifts during the voyage. As mentioned in Section 1.4, the primary objective is to maximize the revenue from optional cargoes while the secondary objective is to keep the number of shifts to a minimum. The number of shifts determines the quality of a stowage plan on a given voyage, and when a shift is made, the cost of this shift is referred to as the *shifting cost*. The shifting costs reflect the costs and/or time used to move cargoes in order to access other cargoes that are to be loaded/unloaded at a given port.

For the RSSP, there are a couple of factors complicating the problem. A RoRo ship has different weight limits for different decks, which implies that not all cargoes can be placed on all decks. Some decks are adjustable, and for the RSSP one has to decide what height the adjustable decks should have. Based on these decisions, a cargo may or may not be placed on this deck, due to the height limitation. Since it is not allowed to adjust the decks while cargo is placed on them, it has to be decided at the start of the voyage. Finally, the stability constraints need to be fulfilled. There are two angles to consider, the roll angle when the ship is heeling to one of the sides, and trim angle when the ship is leaning forwards or backwards. As is seen from Figure 3.1, placing too much of the cargoes on one side of the ship will force it to start heeling to that side. The same goes for placing too much cargo at the front or back of the ship. In addition, a RoRo ship usually has two exit/entry points, one at the stern and one at the side of the ship, and one needs to find which cargoes should be loaded/unloaded at each of these ramps.

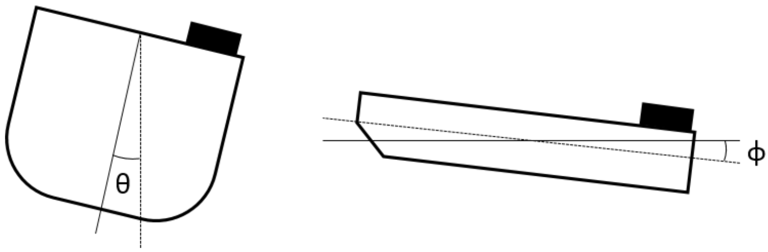


Figure 3.1: Roll (θ) and trim (ϕ) angles.

3.1 Two-Dimensional RoRo Ship Stowage Problem

A simplification of the RSSP is addressed in this master thesis, namely the 2DRSSP which arises if only one deck is considered. The problem is then reduced to a two-dimensional packing problem. In the 2DRSSP, the aim is to decide which optional cargoes to carry and where to place each vehicle on the deck for a given voyage in order to maximize the revenue and minimize the shifting cost.

As mentioned in Section 1.4, focus on stowage of one single deck is an essential building block in solving the problem for multiple decks, i.e. the RSSP. Solving the 2DRSSP gives valuable information to use in the RSSP in terms of methods to use when creating stowage plans for decks, and it is also an important sub-problem to the RSSP. In addition, solving the 2DRSSP gives an efficient way to evaluate the shifting cost for one deck.

As for the RSSP, the 2DRSSP problem has two objectives we need to consider when deciding where to place cargoes on the deck, respectively maximization of revenue generated from optional cargoes and minimization of the shifting cost. According to the case company, the revenue generated for carrying an additional optional cargo always outweighs the shifting cost, as mentioned in Section 1.4. This implies that one will always bring another optional cargo if there is available space on deck, no matter what the new shifting cost becomes. Further, because of the complexity of the problem and in order to better describe the 2DRSSP, we split it into two sub-problems. We define the two sub-problems as (1) stowage problem and (2) shifting evaluation problem. The decisions to be made in the stowage problem are what optional cargoes to bring and where to place each vehicle on the deck. Then, the shifting evaluation problem aims to estimate the shifting cost in order to be able to evaluate how good the created stowage plan is. By solving this shifting evaluation problem we can compare the quality of two or more stowage plans carrying the same optional cargoes.

3.1.1 The Stowage Problem

In the stowage problem, the goal is to decide which cargoes to carry and where to place each single unit on the deck. The complexity of the problem depends on several factors, such as the number of ports to visit, size, shape and number of the cargoes, and the deck's size and shape. Here a detailed description of the stowage problem for a RoRo ship's deck is presented.

Voyage

During a voyage, the RoRo ship visits a given set of ports in a given sequence. In most cases the ship visits all the loading ports first, before traveling to all the unloading ports, due to geographical reasons. Therefore, we assume in this problem that the ship first visits every port in the loading area, and then all the ports in the unloading area. At the start of the voyage, the RoRo operator gets access to a cargo list which includes the mandatory cargoes the ship needs to carry and the optional cargoes it can choose to carry. The cargo list includes information about the cargoes, such as weight, height, width, length, number of vehicles, and which sailing legs a cargo shall be transported. A sailing leg is defined as the part of the voyage between two subsequent ports. This information is used to select the set of optional cargoes to transport, and at each loading port the selected optional cargoes must be loaded together with the mandatory cargoes. In the 2DRSSP we only consider the stowage on one deck, and evaluating the ship's stability gives no real value. Therefore, we are disregarding stability evaluation in this master thesis. However, in the case where all decks are considered, the stability calculation becomes an essential part of the problem.

To illustrate the problem, a small example of a given voyage is shown in Figure 3.2. Here, there are three mandatory cargoes of different sizes, each consisting of four vehicles. There is also one optional cargo, with two vehicles. There are four ports along the voyage, first two loading and then two unloading ports. The four ports indicate that the problem has a total of three sailing legs. The figure shows a feasible stowage plan for each sailing leg. It should be noted that a vehicle in a cargo cannot change position on deck from one sailing leg to the next. This means that the vehicle is located at the exact same place on the deck during the whole voyage. This is common practice on a RoRo ship since each vehicle is attached to the deck during the voyage and moving them would include extra work, increasing the cost. From the solution, it is evident that even though it is enough free area on the deck to bring the optional cargo, the outline of the deck makes it impossible to include it. This example illustrates that allocating vehicles based only on a deck's area capacity could give infeasible solutions.

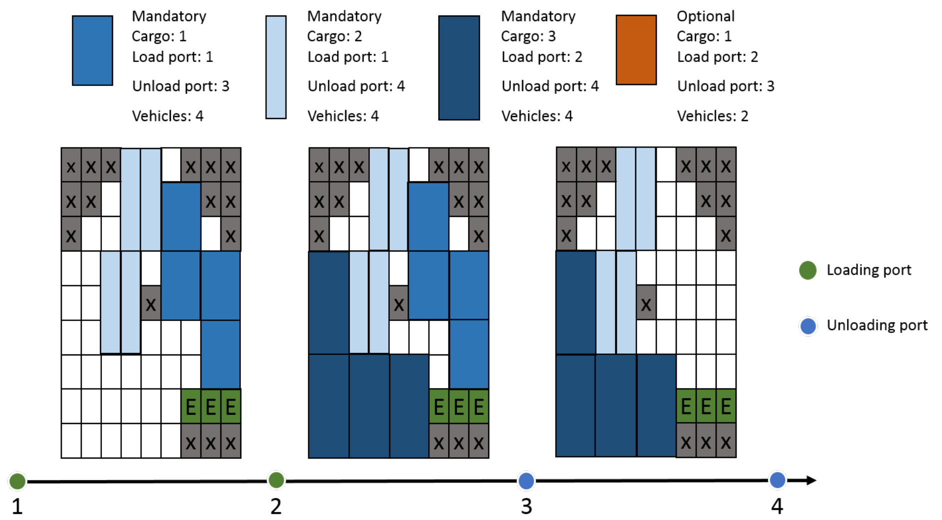


Figure 3.2: A possible solution to the packing problem for each sailing leg during the voyage. Grey squares marked X are unusable space, and squares marked E represent the entry/exit point.

Cargo

The cargo is an important part of the stowage problem, and as described in Section 1.2, it may come in different forms. In this thesis, the term *vehicle* is used when referring to one or more units in a cargo, no matter the cargo type, i.e. car, yacht, trains, heavy rolling machinery, etc. One usually assigns different types of cargoes to different types of decks in a RoRo ship. Decks with low weight and height capacities usually store ordinary cars, while decks with high weight and height capacities typically store breakbulk cargoes, as can be seen in Figure 3.3. In this master thesis, the focus is mainly on stowage plans for decks that hold breakbulk cargoes. The reason for this is that the most profitable optional cargoes are the ones called project cargoes, and they usually consist of breakbulk units. It is a lot harder to know where to place breakbulk cargoes on the deck, since they have irregular sizes and shapes, compared to the standardized ordinary vehicles. In Figure 3.3 one can see the difference in the utilization of the area on the deck when loaded with respectively ordinary cars and breakbulk. For decks loaded with

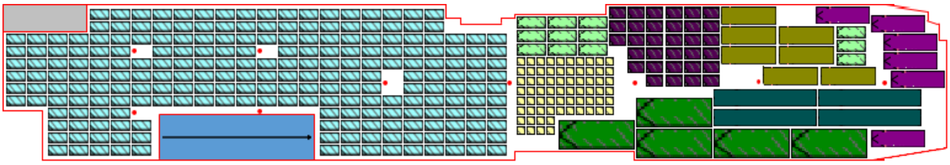


Figure 3.3: To the left on the deck cargoes with ordinary cars are placed, and to the right, breakbulk cargoes are placed. The stowage plan shows how the area utilization of the deck varies for ordinary cars and breakbulk cargoes.

breakbulk cargoes, the modeling approach with lanes presented by Øvstebø et al. (2011a) would not be a good fit, due to the vast different sizes that exist for such cargoes. A lane approach for breakbulk cargoes would most likely lead to poor utilization of the deck's area. The modeling approach would probably be a better fit with ordinary cars, but the project cargoes are the most profitable cargoes to carry, and therefore a stowage plan that is suited for breakbulk cargoes is the focus in the stowage problem in this thesis. In addition, it is not allowed to transport only parts of a cargo. The whole cargo, i.e. all vehicles, needs to be transported.

A consideration to be done when stowing cargoes on the deck is the way they are oriented. There are no physical laws preventing them from being oriented in any direction possible, but following the standards of the case company, the cargoes are almost always oriented in the longitudinal direction on the deck. Therefore, a good approximation in this problem is to assume that all the cargoes need to have either the front or the rear facing the bow of the ship.

On some voyages, the amount of cargo a ship may carry exceeds the capacity of the ship, and then an important part of the stowage problem would be to decide which optional cargoes to carry. In other cases the ship can carry all the available cargoes on a voyage. On these voyages the complexity of the stowage problem is reduced, as we do no longer need to choose which cargoes to carry. Although, we still need to decide where to put the cargoes in order to minimize the shifting cost.

Deck

The deck is an important part of the stowage problem and it limits how many cargoes that can be transported and the maximum weight, length and width of the cargoes. A typical RoRo deck has what we in this thesis call *unusable space*. Unusable space is certain areas of the deck where one cannot stow cargoes, such as pillars and entry/exit ramps, as seen in Figure 3.3. In Øvstebø et al. (2011a) these unusable areas are neglected in order to make the simplification about dividing the space on the ship into lanes work when modeling it mathematically. This assumption might be reasonable when looking at cargoes consisting of regular cars, due to their small size. However, when looking at breakbulk cargoes, this assumption is too rough, because pillars, ramps, etc, may prevent placing a vehicle of this type of cargo at a given spot on the deck. Thus, unusable space is an important consideration.

Besides the unusable space, a deck has both height and weight limitations. For the stowage problem in the 2DRSSP, height limitations are quite simply handled. If the vehicles in a cargo have a height greater than the height limitation, the cargo cannot be carried. However, when considering multiple decks, the cargoes may be placed on some of the other decks on the ship. Weight limitations are important to consider, since we are dealing with breakbulk cargoes that may weigh many tons. The strength is not necessary equal across the whole deck, which means that there may be a certain limitation to where the heaviest cargoes can be placed.

In addition, even though it has been stated that a RoRo ship usually has two entry/exit points, we have in this thesis only considered one entry/exit point for a given deck. This is because breakbulk cargoes use the largest loading/unloading ramp on the ship since this ramp is specialized to cope with the heavy load. Decks that do not have entry/exit points to land have two ramps, i.e. one leading to the upper decks and one to the lower decks. For this case, it is also safe to assume only one entry/exit point, since the cargoes are only using the ramp leading to the main entry/exit point.

3.1.2 The Shifting Evaluation Problem

The secondary objective of the 2DRSSP is to evaluate the stowage plan created from solving the stowage problem. Given a feasible solution for the 2DRSSP, as illustrated in Figure 3.2, the shifting evaluation problem is to find a way to evaluate the quality of stowage plan, given by the shifting cost.

For each vehicle both an entry and exit route need to be set. An entry route is defined as the path for a given vehicle from the entry/exit point to the location it is to be placed on the deck, and vice versa for an exit route. Each possible route a vehicle can take is associated with a shifting cost depending on how many other vehicles that are placed on the route. Therefore, in the shifting evaluation problem the goal is to decide the optimal route each vehicle must take in order to minimize the shifting cost for a given stowage plan. The cost of shifting a vehicle varies. Larger vehicles that need to be shifted usually means higher shifting cost, as there is normally more work related to moving larger vehicles. For example, one needs more workers to move a large vehicle for safety reasons, i.e. need one docker to drive the vehicle off and others to visually ensure that the vehicle does not hit anything due to its size. The total shifting cost of a given stowage plan along a voyage is given by the cost of shifting all blocking vehicles.

There are quite a few considerations to be made when evaluating the total shifting cost for a voyage. In Figure 3.4, an example of how the shifting cost is evaluated for different vehicles in different settings and how they add up to the total shifting cost for the voyage is shown. In this example the two loading ports are evaluated, and it shows the shortest and most shifting cost efficient route for some of the vehicles that are loaded. First, for an entry route for a given vehicle, having a route going through other vehicles on the deck that are loaded at the same port, induces no shifting cost. This is because it is possible to load the vehicles in an order making which make it possible to avoid shifting. In the stowage plan to the left in Figure 3.4, no shift occurs when loading a vehicle from cargo 2, even though the entry route moves through a vehicle from cargo 1. This is because both are loaded at port 1, and by loading the vehicle from cargo 2 first, we can avoid shifting completely. Due to this, we get a shifting cost of 0 for this route.

Another important concept is when two or more vehicles use the same, or partly the same, entry/exit route. Vehicles that need to be shifted on a route used by more than one vehicle is only shifted once at each port. For example, in the stowage plan to the right in Figure 3.4, two vehicles from cargo 3 use the same route, indicated by the gray and black route. They both move through a vehicle from cargo 1, which is loaded at a previous port. As the vehicle already is moved to make way for one of the vehicles from cargo 3, for example the one following the black arrow, there is no reason to place it back on the deck before the other vehicle from cargo 3, i.e. gray route, also has been placed on the deck. Therefore, the shifting cost from the two routes add up to 4, and not 8.

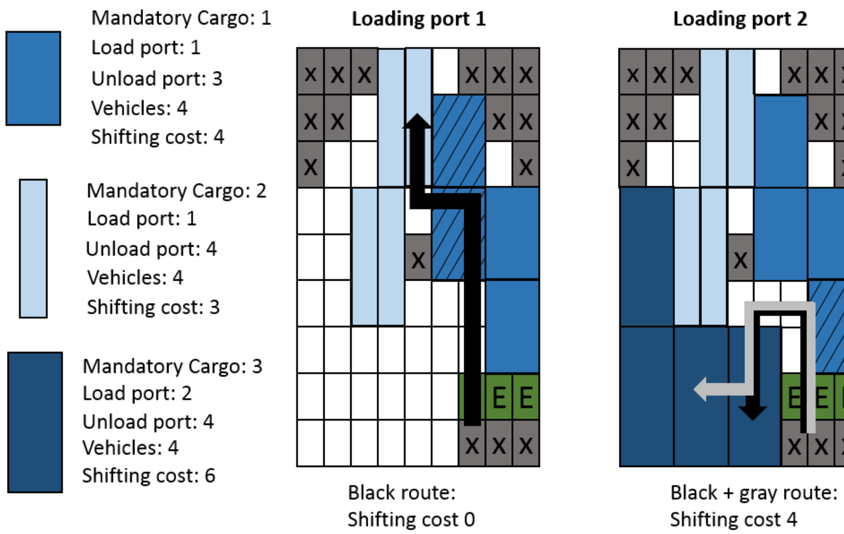


Figure 3.4: Examples on how the shifting cost is evaluated in different cases for the stowage plan presented in Figure 3.2, when loading cargoes. The stowage plan to the left shows how the shifting cost is evaluated for an entry route when vehicles from different cargoes, but the same loading port, are loaded on the deck. The stowage plan to the right shows how the shifting cost is evaluated when two vehicles use the same (or partly the same) entry route.

3.2 Summary

In this thesis, we focus on the 2DRSSP problem, which is a stowage problem for one deck on a RoRo ship and an important building block for the RSSP problem. The objective in the 2DRSSP is to decide which optional cargo to carry and where to place each vehicle on the deck in order to maximize extra profit from optional cargoes and minimize the shifting cost. Due to the problem's complexity it is split into two sub-problems. In the stowage problem we decide which optional cargoes to carry and where to place the vehicles on the deck. In the shifting evaluation problem we decide which entry and exit route each vehicle should choose in order to minimize the shifting cost, such that we can evaluate the quality of the stowage plan. In some cases there may exist equally good stowage plans that have the same amount of revenue generated from the optional cargoes it has chosen to carry along the voyage. Then the results from the shifting evaluation can be used to decide which of the stowage plans that has the lowest shifting cost and is the best choice overall.

4. Mathematical Models

In this chapter, we propose two MIP models, each serving a distinct purpose when solving the 2DRSSP. First, we present our modeling approach, followed by a description of the 2DRSSP Stowage Model. Then, in Sections 4.3 and 4.4, our approach in solving the shifting cost evaluation problem is described, and a MIP model for the problem is given. The models presented in this chapter are based on the work by Hansen and Hukkelberg (2015). However, in order to increase the computational performance of the models, new models are developed based on an improved modeling approach.

4.1 Modeling Approach and Assumptions

Our approach to solving the 2DRSSP splits the problem into two phases. First, we solve the stowage problem for a given deck. Then, we evaluate the number of shifts needed when applying the resulting stowage plan for the voyage. This results in two models: A stowage model and a shifting evaluation model. It is a reasonable approach to deal with these two problems in sequence, since the results of the stowage, i.e. the extra revenue from optional cargoes that can be transported, are assumed more important than the shifting costs, as stated in Section 1.4.

Still the results from the first model influence the shifting cost for a stowage plan. Therefore, to implicitly take into account the shifting when determining a stowage plan, different objective functions are proposed and tested. Two concepts are introduced with the expectation to reduce the shifting cost, namely *grouping* and *placement*. Placing vehicles from the same cargo next to each other is denoted as grouping. By grouping vehicles together, the shifting cost may decrease as vehicles from the same cargo can use the same entry/exit route. Placing cargoes which are on the ship for the most number of sailing legs farther away from the entry/exit than cargoes with shorter time on the vessel is known as placement. This is introduced based on the expectation that vehicles placed nearest the entry/exit are probably more exposed to shifting, and therefore, those squares should be more costly to use.

Instead of dividing the deck into lanes such as Øvstebø et al. (2011a), we suggest a grid representation of the deck, as illustrated in Figure 4.1. This enables us to represent real deck layouts in a better way, and the resulting stowage plan becomes more realistic. This is done by defining a set of rows $\mathcal{I}$ and columns $\mathcal{J}$. Square $(1, 1)$ is defined as the square located at the stern on the ship's port side (bottom left corner in Figure 4.1). All squares are set to be of the same size.

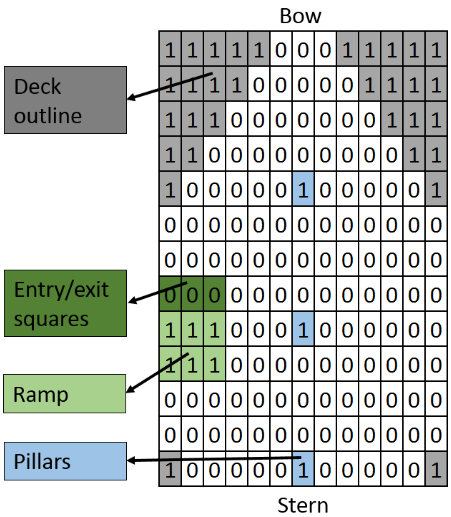


Figure 4.1: Illustration of the grid representation of a deck. The 1's indicate that the corresponding square is unusable

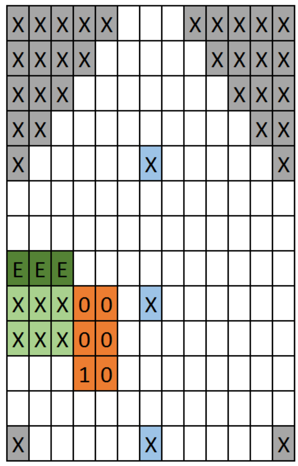


Figure 4.2: Orange squares represent a vehicle placed in these squares. The 1 indicate that $x_{ijc} = 1$ for this square and cargo.

Each cargo $c \in \mathcal{C}$ consists of N_c identical vehicles. If in practice, one is to carry a cargo consisting of heterogeneous vehicles, this cargo is split into several cargoes consisting of identical vehicles. When all vehicles in a cargo are identical, the number of squares needed to place a vehicle from that cargo is equal for all vehicles in the cargo. By doing this, it is possible to assign squares to cargoes, instead of vehicles. Hence, there is no reason to include a vehicle index in the model, which almost eliminates symmetrical solutions. Symmetrical solutions could occur if the deck is symmetrical, which rarely occur due to the ramp placement, or if two or more cargoes are identical. However, the absence of symmetrical solutions does not exclude practical identical solutions. For a given stowage plan, a practical identical solution could be to slightly move a vehicle to another place on the deck, e.g. one square to the right, with no change in the revenue and the shifting cost. The solutions are mathematical different, but for all practical purposes, the solutions are identical.

All vehicles are modeled as rectangular objects, as most of the vehicles have a close to rectangular outline. A drawback to this approach is that the actual area usage of some vehicles is overestimated, as vehicles with irregular shape are modeled as rectangular objects. However, this usually only applies to the minority of the carried cargoes. Thus, for a given grid resolution, each vehicle in a cargo needs S_c^L length squares, and S_c^W width squares to be placed on the deck. These parameters will vary with the actual size of the vehicles in the cargo, the minimum clearance required for the vehicles and the chosen grid resolution, given by the number of rows times the number of columns ($|\mathcal{I}||\mathcal{J}|$). The minimum distance required between the cargoes is here set as a constant length B. In practice, this could vary with the content of the cargo, where expensive cars may require more safety distance, due to contractual terms. The area of the resulting square usage always gives an overestimation of the actual area usage. Increased resolution will give a more detailed representation of the deck and the vehicles, but increases the number of variables in the model.

When a vehicle from cargo c is placed somewhere on the deck, the square where the vehicle's lower left corner is placed is recorded by the model. The variable $x_{ijc} = 1$ if the lower left corner of a vehicle from cargo c is placed in square (i, j), 0 otherwise. As a vehicle in a cargo often has length and width greater than the size of one square, a vehicle placed in square (i, j), uses more than only this square. However, the variable is only set equal to one for the square where the lower left corner is placed.

As the comprehension of this modeling choice is essential for the understanding of the model, an example is shown in Figure 4.2. Here, a vehicle from cargo 2 is placed in squares $(3, 4), (3, 5), (4, 4), (4, 5), (5, 4), (5, 5)$. The vehicle occupies six squares, as it requires three squares in length ($S_c^L = 3$), and two squares in width ($S_c^W = 2$). As the lower left corner of the vehicle is placed in square $(3, 4)$, the variable $x_{342} = 1$. None of the other variables associated with the used squares are set to one. However, it should be noted that the constraints presented in the next section ensure that no other vehicles use any of these squares.

Some squares are unusable due to ramp placement, deck outline, pillars, etc. These constraints are handled in the variable declaration of the model. For all squares (i, j), if the corner of a vehicle from cargo c cannot be placed in that square due to unusable space ($U_{ij} = 1$) or entry/exit squares ($E_{ij} = 1$), then x_{ijc} is fixed to zero for the given cargo and square. This applies for all squares a vehicle spans over. Using the example from last paragraph, if any of the squares $(3, 4), (3, 5), (4, 4), (4, 5), (5, 4), (5, 5)$ is either unusable or an entry/exit square, x_{342} is fixed to zero.

Each square on the deck has a weight limitation and this will reduce the number of possible placements of some cargoes. The weight limitations vary across the deck and typically do spots near pillars and the deck outline have a higher weight capacity than other squares. Parameter Q_{ij} gives the weight limit for square (i, j). The variable x_{ijc} is fixed to zero if the weight of a vehicle (C_c^{We}) in cargo c exceeds the weight limits of the squares that the vehicle is placed upon. These constraints are handled in the variable declaration of the model, as for the unusable space.

The ports are assumed to be separated into two regions, one supply region and one demand region, where the loading ports are visited before the unloading ports. This is how most voyages are in RoRo shipping. Also following common practice, it is assumed that once a vehicle is placed, it stays in the same location during the whole voyage. From this, it follows that all carried vehicles are to be placed on the deck on the sailing leg between the last loading port and the first unloading port. Hence, by generating a stowage plan for this sailing leg, the vehicle placements for all other sailing legs can be derived from this stowage plan, as the vehicles are placed in the same location during the whole voyage.

4.2 2DRSSP Stowage Model

Indices

c : cargo
i : row
j : column

Sets

$\mathcal{C}$: set of all cargoes
$\mathcal{C}^{\mathcal{M}}$: set of all mandatory cargoes
$\mathcal{C}^{\mathcal{O}}$: set of all optional cargoes
$\mathcal{I}$: set of all rows
$\mathcal{J}$: set of all columns
$\mathcal{I}_c$: set of all rows where the corner of a vehicle in cargo c can be placed
$\mathcal{I}_c = \{1, ...|\mathcal{I}| - S_c^L + 1\}$
$\mathcal{J}_c$: set of all columns where the corner of a vehicle in cargo c can be placed
$\mathcal{J}_c = \{1, ..., |\mathcal{J}| - S_c^W + 1\}$

Parameters

L^D : length of deck
W^D : width of deck
C_c^L : length of one vehicle in cargo c
C_c^W : width of one vehicle in cargo c
C_c^{We} : weight of one vehicle in cargo c
B : minimum clearance between vehicles
N_c : number of vehicles in cargo c
S_c^L : number of length squares needed to place one vehicle from cargo c
$S_c^L = \lceil \frac{(C_c^L + B)|\mathcal{I}|}{L^D} \rceil$
S_c^W : number of width squares needed to place one vehicle from cargo c
$S_c^W = \lceil \frac{(C_c^W + B)|\mathcal{J}|}{W^D} \rceil$
P_c^L : loading port of cargo c
P_c^U : unloading port of cargo c, $P_c^U > P_c^L$
R_c : revenue earned if optional cargo c is taken
U_{ij} : 1 if square (i, j) is unusable, 0 otherwise
E_{ij} : 1 if square (i, j) is an exit square, 0 otherwise
Q_{ij} : Weight limit for the deck in position (i, j)
D : A small positive number that will increase the value of the objective function if vehicles from the same cargo are grouped together
C_{ij}^S : The artificial cost of placing a vehicle from cargo c in square (i, j)

Decision variables

x_{ijc} : 1 if the lower left corner of a vehicle from cargo c is placed in square (i, j), 0 otherwise

y_c : 1 if optional cargo c is taken, 0 otherwise

u_{ijc} : Number of vehicles from the same cargo c placed next to a vehicle from cargo c placed in square (i, j)

Objective functions

The objective of the 2DRSSP is to maximize the revenue from optional cargoes, minus the penalty costs incurred when shifting vehicles. Since the stowage model does not explicitly evaluate shifting cost, four objective functions are proposed and tested in an effort to place vehicles in a way that reduces the need for shifting. The objective function (4.1) maximizes the revenues from optional cargoes. The objective function (4.2) maximizes the revenues from optional cargoes and the artificial value of placing vehicles from the same cargo together. The objective function (4.3) maximizes the sum of revenues from optional cargoes minus the placement cost of each vehicle in all carried cargoes. The placement cost for each vehicle is a function of the number of sailing legs a vehicle is placed on the ship, multiplied by the cost of using the chosen square placement. The cost of using a square should reflect the square's probability of being exposed to shifting, and is further elaborated in Figure 5.3, Section 5.3.1. The objective function (4.4) combines objectives (4.2) and (4.3).

$$\max z = \sum_{c \in \mathcal{C}^O} R_c y_c \tag{4.1}$$

$$\max z = \sum_{c \in \mathcal{C}^O} R_c y_c + \sum_{c \in \mathcal{C}} \sum_{i \in \mathcal{I}_c} \sum_{j \in \mathcal{J}_c} D u_{ijc} \tag{4.2}$$

$$\max z = \sum_{c \in \mathcal{C}^O} R_c y_c - \sum_{c \in \mathcal{C}} \sum_{i \in \mathcal{I}_c} \sum_{j \in \mathcal{J}_c} \sum_{i'=i}^{i+S_c^L-1} \sum_{j'=j}^{j+S_c^W-1} (P_c^U - P_c^L) \frac{C_{i'j'}^S x_{ijc}}{S_c^L S_c^W} \tag{4.3}$$

$$\max z = \sum_{c \in \mathcal{C}^O} R_c y_c + \sum_{c \in \mathcal{C}} \sum_{i \in \mathcal{I}_c} \sum_{j \in \mathcal{J}_c} (D u_{ijc} - \sum_{i'=i}^{i+S_c^L-1} \sum_{j'=j}^{j+S_c^W-1} (P_c^U - P_c^L) \frac{C_{i'j'}^S x_{ijc}}{S_c^L S_c^W}) \tag{4.4}$$

Common constraints

$$\sum_{i \in \mathcal{I}_c} \sum_{j \in \mathcal{J}_c} x_{ijc} = N_c, \qquad c \in \mathcal{C}^{\mathcal{M}} \tag{4.5}$$

$$\sum_{i \in \mathcal{I}_c} \sum_{j \in \mathcal{J}_c} x_{ijc} = N_c y_c, \qquad c \in \mathcal{C}^{\mathcal{O}} \tag{4.6}$$

$$\sum_{i'=i}^{i+S_c^L-1} \sum_{j'=j}^{j+S_c^W-1} x_{i'j'c} \leq 1, \qquad c \in \mathcal{C}, i \in \mathcal{I}_c, j \in \mathcal{J}_c \tag{4.7}$$

$$\sum_{i'=max(i-S_{c'}^L+1,1)}^{min(i+S_c^L-1,|\mathcal{I}_{c'}|)} \sum_{j'=max(j-S_{c'}^W+1,1)}^{min(j+S_c^W-1,|\mathcal{J}_{c'}|)} x_{i'j'c'} \leq M_{cc'}(1 - x_{ijc}),$$
$$c \in \mathcal{C}, c' \in \mathcal{C} \backslash \{c\}, i \in \mathcal{I}_c, j \in \mathcal{J}_c \tag{4.8}$$

$$x_{ijc} \in \{0,1\}, \qquad c \in \mathcal{C}, i \in \mathcal{I}_c, j \in \mathcal{J}_c \tag{4.9}$$

$$y_c \in \{0,1\}, \qquad c \in \mathcal{C}^{\mathcal{O}} \tag{4.10}$$

Constraints (4.5) guarantee that all vehicles in the mandatory cargoes are placed on the deck. Constraints (4.6) ensure that all vehicles in an optional cargo are placed on the deck if the optional cargo is taken. Constraints (4.7) guarantee that at most one vehicle from the same cargo uses the same place on the deck. Constraints (4.8) make sure that different cargoes do not use the same place on the deck. Min and max expressions are included to ensure that the constraints do not include squares outside the deck area. An upper bound on $M_{cc'}$ is given by $(S_c^L + S_{c'}^L - 1)(S_c^W + S_{c'}^W - 1)$, which is the maximum number of squares where a vehicle from cargo c' could be placed that would use some of the same squares used by a vehicle from cargo c placed in square (i,j). Constraints (4.9) and (4.10) force the variables to take binary values.

Grouping constraints

$$u_{ijc} \leq x_{i+S_c^L,jc} + x_{i-S_c^L,jc} + x_{i,j+S_c^W,c} + x_{i,j-S_c^W,c}, \qquad c \in \mathcal{C}, i \in \mathcal{I}_c, j \in \mathcal{J}_c \tag{4.11}$$

$$u_{ijc} \leq M x_{ijc}, \qquad c \in \mathcal{C}, i \in \mathcal{I}_c, j \in \mathcal{J}_c \tag{4.12}$$

$$\sum_{i \in \mathcal{I}_c} \sum_{j \in \mathcal{J}_c} u_{ijc} \geq 2N_c - 2, \qquad c \in \mathcal{C}^{\mathcal{M}} \tag{4.13}$$

$$\sum_{i \in \mathcal{I}_c} \sum_{j \in \mathcal{J}_c} u_{ijc} \geq (2N_c - 2) y_c, \qquad c \in \mathcal{C}^{\mathcal{O}} \tag{4.14}$$

$$u_{ijc} \geq 0, \qquad c \in \mathcal{C}, i \in \mathcal{I}_c, j \in \mathcal{J}_c \tag{4.15}$$

Constraints (4.11) force u_{ijc} to take a value equal to the number of vehicles from the same cargo placed next to the vehicle in square (i, j). Constraints (4.12) ensure that the number of neighboring vehicles is only calculated for the squares where a vehicle is placed. The upper bound on M is 4, which is the maximum number of neighboring vehicles, defined as a vehicle placed *exactly* in front, behind, left or right of a vehicle. A neighboring vehicle placed exactly to the right of a vehicle placed in $x_{ijc} = 1$, is mathematically given by $x_{i+S_c^W,jc} = 1$. Thus, a vehicle placed in $x_{i+S_c^L,j+1,c} = 1$ is not defined as a neighbor, even though this could in practice be interpreted as a neighbor. Constraints (4.13) and (4.14), in addition to (4.11) and (4.12) enforce vehicles from same cargo to have a total number of neighbors greater than or equal to the weakest form of compactness. Given that every vehicle is placed next to a vehicle from the same cargo, the weakest form of compactness is a line. In this case, all vehicles would have two neighbors, except the vehicles at each end of the line, which will only have one neighbor. The lower bound on the total number of neighboring vehicles for a cargo, in this case, is given by $2N_c - 2$. Finally, non-negativity requirements for the variables related to grouping are given in (4.15).

Model versions

The different objectives presented aim at influencing the vehicle placements so that the shifting cost is reduced. From this, five versions of the stowage model are presented in Table 4.1. Common for all the model versions is that the objective is to maximize the revenue generated from optional cargoes. For the basic model version N, this is the only objective. Model version P additionally influences the vehicles placement by introducing square costs. This results in placing vehicles carried for the most sailing legs farthest away from the exit, where the probability of being exposed to shifting is less. Model version H enforces a weak form of compactness to each cargo, placing the vehicles together. Model version S rewards grouping of vehicles. For each vehicle in a cargo, a higher number of neighboring vehicles from the same cargo increases the objective value. Finally, model version SP penalizes placement and rewards grouping.

Table 4.1: Model versions.

Model version	Objective	Constraints
Normal (N)	(4.1)	(4.5)-(4.10)
Placement (P)	(4.3)	(4.5)-(4.10)
Hard grouping (H)	(4.1)	(4.5)-(4.15)
Soft grouping (S)	(4.2)	(4.5)-(4.12), (4.15)
Placement + Soft grouping (SP)	(4.4)	(4.5)-(4.12), (4.15)

4.3 Shifting Evaluation

Based on a given feasible solution from the stowage model, it is desirable to evaluate the solution with respect to the shifting cost. Even though a feasible stowage plan is created, the shifting cost associated with this given stowage plan is not known and needs to be calculated. This is a hard problem, as the combined optimal entry and exit routes for all vehicles need to be calculated. The term *combined* constitutes the main challenge in this problem, explained using an example given in Figure 4.3. Here, exit routes for the vehicles V1 and V2 are to be decided, for a given unloading port. Evaluating the exit routes individually for each vehicle, the optimal routes are shown on the deck to the left. This results in a shifting cost of 6 for V1, and 2 for V2, which gives a total shifting cost of 8. The 2DRSSP Shifting Evaluation Model presented in the following section provides a better result. By taking into account that each vehicle only is shifted once, both V1 and V2 could use the squares where the shifted vehicle was placed, as shown on the deck to the right in the figure. This gives an optimal solution of shifting cost equal to 6. For this instance, the optimal route for V2 is actually a suboptimal route in the isolated case. As showed, in order to evaluate the shifting cost of a stowage plan exact, the combined problem must be solved.

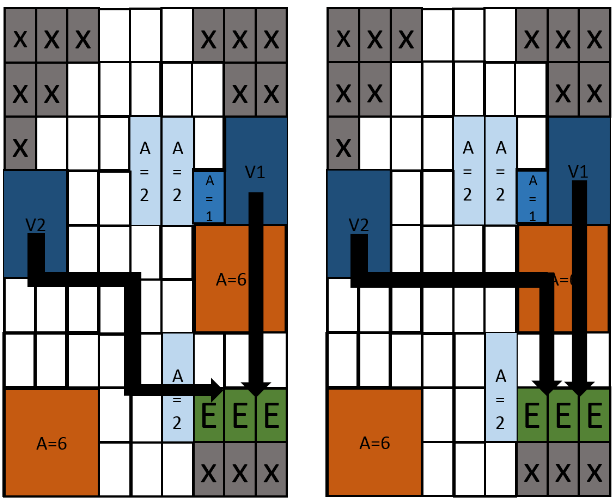


Figure 4.3: Optimal exit route for each vehicle in the isolated case to the left, and the optimal solution from the shifting evaluation model to the right. V1 and V2 indicate the vehicles that are to be unloaded at this given port. A is the number of squares the other vehicles are occupying, and it indicates the cost to move these vehicles.

The model presented in this section aims to give an exact solution to the shifting evaluation problem, i.e. evaluating the entry and exit routes for all vehicles combined. The objective of the shifting model is to find an optimal entry and exit path for each vehicle v in cargo c for the related loading and unloading ports of the cargoes, in order to minimize the total shifting cost. The problem is solved for every port, and the sum of the shifting cost for all ports along the given voyage is reported as the objective value. This shifting cost is then used to evaluate the actual performance of the stowage plan provided by the stowage model.

The cost of shifting a given vehicle is set as a function of the area of the vehicle, since the cost of moving a large vehicle, e.g. a semi-trailer, is assumed higher than the cost of moving a small vehicle, e.g. a 3-door car. The shifting cost could also be set on other considerations than the area, e.g. expected time usage or shifting distance. It is assumed that a vehicle that is shifted is moved out of the deck during the port call and returned to the exact same square when the loading/unloading is done. In practice, vehicles have a given turning radius and can therefore not move sideways. However, as the inclusion of turning radius would drastically increase the modeling complexity of the shifting evaluation, sideways movement is allowed in the model. Thus, when evaluating an exit route for a vehicle, the vehicle is allowed to move one square forward, backward, left, or right in each step from its location on the deck, to the exit ramp.

For each vehicle at its loading and unloading port, the entry/exit route needs to be calculated. The variable $d_{ijcvp} = 1$ if the corner of vehicle v from cargo c uses square (i, j) in port p on its entry/exit route. Thus, for a given vehicle, its exit route is represented as a chain of variables $d_{ijcvp} = 1$ from the vehicle's location of the deck to the entry/exit point.This is illustrated by the example given in Figure 4.4. Here, the suggested exit route for the light blue vehicle positioned in the lower right corner is marked with orange squares. Here, only the squares visited by the lower left corner of the vehicle are recorded by the model, represented by $d_{ijcvp} = 1$ for the given square. The light blue vehicles are to be unloaded at this given port, while the dark blue vehicles are to be unloaded at a subsequent port. The squares marked with red borders are squares visited by the exiting vehicle. As seen from the figure, the exiting vehicle passes through both a light and a dark blue vehicle on its exit route. However, since the light blue vehicle is to be unloaded at the same port, no shifting is imposed on this vehicle. As the dock workers in practice would unload the vehicles near the exit first, this blocking vehicle would not be present on the deck when the current vehicle is unloaded. Using this fact, the model allows an unloading vehicle to pass through other vehicles that are to be unloaded in the same port without the need for shifting the vehicle. Hence, only the dark blue vehicle is shifted in this example. The new indices, sets, parameters, and decision variables used in the 2DRSSP Shifting Evaluation Model are presented in the following section together with a mathematical description of the model.

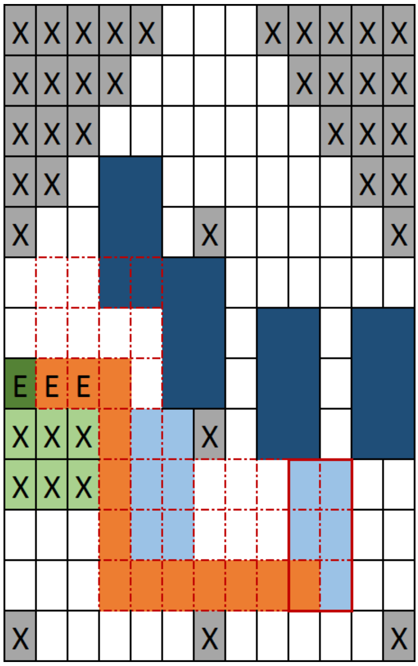


Figure 4.4: Exit route for a vehicle in a port.

4.4 2DRSSP Shifting Evaluation Model

Indices

p : port
v : vehicle

Sets

$\mathcal{C}^{\mathcal{T}}$: set of all mandatory cargoes and the carried/taken optional cargoes
$\mathcal{P}$: set of all ports
$\mathcal{P}_c$: set of ports where a cargo c is on the deck, except loading port and unloading port $\mathcal{P}_c = \{P_c^L + 1, ..., P_c^U - 1\}$
$\mathcal{P}_c^{LU}$: set of loading port and unloading port of cargo c, $\mathcal{P}_c^{LU} = \{P_c^L, P_c^U\}$
$\mathcal{V}_c$: set of all vehicles in cargo C

Parameters

C_c^M : shifting cost for a vehicle in cargo c
E_{ijc}^C : 1 if the square (i, j) is an entry/exit square for cargo c, 0 otherwise.
V_{ijcvp}^P : 1 if vehicle v in cargo c is placed in square (i, j) in port p, 0 otherwise.
V_{ijcvp}^C : 1 if the corner of vehicle v in cargo c is placed in square (i, j) in port p, 0 otherwise.

Decision variables

d_{ijcvp} : 1 if the corner of vehicle v from cargo c uses square (i, j) in port p, on its entry/exit route, 0 otherwise
f_{cvp} : 1 if vehicle v from cargo c is shifted in port p, 0 otherwise

Objective function

$$\text{SM}(p \in \mathcal{P} \backslash \{1, |\mathcal{P}|\}) \ \min z = \sum_{c \in \mathcal{C}^{\mathcal{T}}} \sum_{v \in \mathcal{V}_c} C_c^M f_{cvp} \tag{4.16}$$

The objective function (4.16) is to minimize the cost of shifting vehicles for a given port. The model is solved for every port except the first and the last port, since no vehicles are shifted in these ports. The resulting shifting cost for a voyage is the sum of the shifting costs for each port.

Common constraints

$$d_{i-1,jcvp} + d_{i+1,jcvp} + d_{i,j-1,cvp} + d_{i,j+1,cvp} \geq 2d_{ijcvp},$$
$$c \in \mathcal{C}^{\mathcal{T}}, i \in \mathcal{I}_c, j \in \mathcal{J}_c, v \in \mathcal{V}_c \mid p \in \mathcal{P}_c^{LU} \cap V_{ijcvp}^C = 0 \cap E_{ijc}^C = 0 \quad (4.17)$$

$$d_{i-1,jcvp} + d_{i+1,jcvp} + d_{i,j-1,cvp} + d_{i,j+1,cvp} \geq d_{ijcvp},$$
$$c \in \mathcal{C}^{\mathcal{T}}, i \in \mathcal{I}_c, j \in \mathcal{J}_c, v \in \mathcal{V}_c \mid p \in \mathcal{P}_c^{LU} \cap (E_{ijc}^C + V_{ijcvp}^C) \geq 1 \quad (4.18)$$

$$(d_{ijcvp} - f_{c'v'p'}) \sum_{i'=i}^{i+S_c^L-1} \sum_{j'=j}^{j+S_c^W-1} V_{i'j'c'v'p'}^P \leq 0,$$
$$c \in \mathcal{C}^{\mathcal{T}}, c' \in \mathcal{C}^{\mathcal{T}} \backslash \{c\}, i \in \mathcal{I}_c, j \in \mathcal{J}_c, v \in \mathcal{V}_c, v' \in \mathcal{V}_{c'} \mid (p \in \mathcal{P}_c^{LU}) \cap \ (p' \in \mathcal{P}_{c'} \cap \{p\}) \quad (4.19)$$

$$\sum_{i \in \mathcal{I}_c} \sum_{j \in \mathcal{J}_c} E_{ijc}^C d_{ijcvp} \geq 1, \qquad c \in \mathcal{C}^{\mathcal{T}}, v \in \mathcal{V}_c \mid p \in \mathcal{P}_c^{LU} \quad (4.20)$$

$$\sum_{i'=i}^{i+S_c^L-1} \sum_{j'=j}^{j+S_c^W-1} U_{i'j'} d_{ijcvp} \leq 0, \qquad c \in \mathcal{C}^{\mathcal{T}}, i \in \mathcal{I}_c, j \in \mathcal{J}_c, v \in \mathcal{V}_c \mid p \in \mathcal{P}_c^{LU} \quad (4.21)$$

$$d_{ijcvp} \geq V_{ijcvp}^C, \qquad c \in \mathcal{C}^{\mathcal{T}}, i \in \mathcal{I}_c, j \in \mathcal{J}_c, v \in \mathcal{V}_c \mid p \in \mathcal{P}_c^{LU} \quad (4.22)$$

$$d_{ijcvp} \in \{0,1\}, \qquad c \in \mathcal{C}^{\mathcal{T}}, i \in \mathcal{I}_c, j \in \mathcal{J}_c, v \in \mathcal{V}_c \mid p \in \mathcal{P}_c^{LU} \quad (4.23)$$

$$f_{cvp} \in \{0,1\}, \qquad c \in \mathcal{C}^{\mathcal{T}}, v \in \mathcal{V}_c \mid p \in \mathcal{P}_c \quad (4.24)$$

Constraints (4.17) ensure that if a vehicle uses a square (i,j) on its entry or exit route, it has to visit at least two neighboring squares. This applies for all squares except the origin square and the entry/exit squares. Constraints (4.18) force each vehicle to visit at least one neighboring square to its origin/destination and entry/exit square on the entry and exit route. Constraints (4.19) ensure that if a vehicle uses a given square on its entry/exit route, then a vehicle placed in that square is shifted, given the vehicle is not to be loaded/unloaded in that port. Constraints (4.20) guarantee that each vehicle visits at least one entry/exit square for both loading and unloading port. Constraints (4.21) guarantee that the route a vehicle uses does not pass through an unusable space. Constraints (4.22) force the entry and exit route for each vehicle to visit the square where the vehicle is to be placed. Constraints (4.23) and (4.24) force the given variables to take binary values.

Subtour-eliminating constraints

$$d_{i-1,jcvp} + d_{i+1,jcvp} + d_{i,j-1,cvp} + d_{i,j+1,cvp} + 2d_{ijcvp} \leq 4,$$
$$c \in \mathcal{C}^{\mathcal{T}}, i \in \mathcal{I}_c, j \in \mathcal{J}_c, v \in \mathcal{V}_c \mid p \in \mathcal{P}_c^{LU} \cap V_{ijcvp}^{C} = 0 \cap E_{ijc}^{C} = 0 \quad (4.25)$$

$$d_{i-1,jcvp} + d_{i+1,jcvp} + d_{i,j-1,cvp} + d_{i,j+1,cvp} + 3d_{ijcvp} \leq 4,$$
$$c \in \mathcal{C}^{\mathcal{T}}, i \in \mathcal{I}_c, j \in \mathcal{J}_c, v \in \mathcal{V}_c \mid p \in \mathcal{P}_c^{LU} \cap V_{ijcvp}^{C} = 1 \quad (4.26)$$

In addition to Constraints (4.17)-(4.24), subtour-eliminating constraints are needed. Constraints (4.17) and (4.18) ensure that a route connected to the vehicle's placement and a route connected to an entry/exit square for every loading and unloading vehicle is provided. This can result in two routes; one starting and ending at the vehicle's placement at the deck, and one starting and ending at an entry/exit square, see Figure 4.5. In order for a solution to be feasible, there has to exist exactly one route for an entering/exiting vehicle. By including Constraints (4.25) and (4.26), one ensures that exactly one route is provided. Constraints (4.18) and (4.26) ensure that exactly one neighboring square of the vehicle's placement is used on the entry and exit route. Constraints (4.17) and (4.25) force all squares used on an entry/exit route to have exactly two neighbors, except for the start and end point of the route. The combination of Constraints (4.17), (4.18), (4.25), and (4.26) restrict the creation of subtours, and the only possible way to create a route is to start/end at the vehicle's placement on the deck and end/start at an entry/exit square, as seen in Figure 4.6.

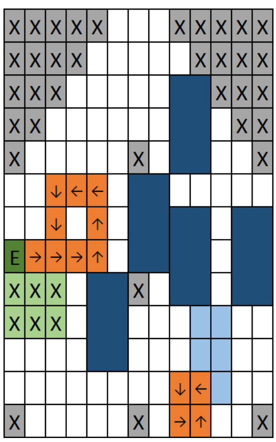


Figure 4.5: Route with subtours

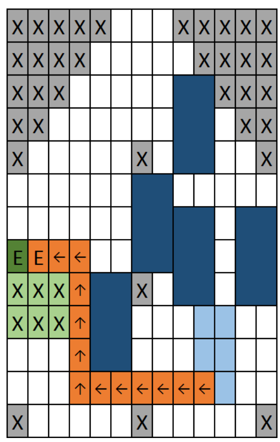


Figure 4.6: Route without subtours

5. 2DRSSP Heuristic

This chapter presents our proposed solution method for solving the 2DRSSP. The approach combines a greedy randomized adaptive search procedure (GRASP) heuristic and an adaptive large neighborhood search (ALNS) heuristic in order to maximize the revenue generated from carrying optional cargoes and minimize the shifting cost. Section 5.1 gives an overview of the 2DRSSP Heuristic. Then, in Section 5.2, a GRASP heuristic is used to find an initial stowage plan. This initial stowage plan is then improved by an ALNS heuristic in Section 5.3. Finally, in Section 5.4, we describe how the shifting cost of a stowage plan is estimated.

5.1 Heuristic Overview

The 2DRSSP Stowage Model, presented in Section 4.4, aims to decide which optional cargoes to carry and where to place each vehicle. This is done simultaneously, i.e. in one phase. The 2DRSSP Heuristic takes another approach, using two phases. The objective of the first phase is to decide which optional cargoes to carry in order to maximize the revenue. The objective of the second phase is to place the chosen optional cargoes and mandatory cargoes on the deck in order to minimize the shifting cost. A pseudocode for the 2DRSSP Heuristic is presented in Algorithm 1.

In the first phase, we disregard the shifting cost of the resulting stowage plan. As the revenue generated by carrying an extra cargo is assumed greater than the additional shifting cost associated with carrying an extra cargo, the main objective is still to achieve the highest revenue possible from optional cargoes. The objective of the first phase is therefore to find the set of optional cargoes that generates the most revenue given the existence of a feasible stowage plan, where $\mathcal{O}_n$ denotes the n'th set of unique optional cargo combinations. The best set of optional cargoes is denoted by $\mathcal{O}^{best}$, initially set to an empty set in line 2. The number of possible cargo combinations is given by $2^{|\mathcal{C}^{O}|}$, which equals the number of sets n. An example with two optional cargoes is used to elaborate this. Four combinations are possible, resulting in four possible sets: $\mathcal{O}_1 = \{0, 0\}$, $\mathcal{O}_2 = \{1, 0\}$, $\mathcal{O}_3 = \{0, 1\}$, and $\mathcal{O}_4 = \{1, 1\}$, where 1 denotes that a cargo is carried, 0 otherwise. These sets are sorted by revenue in descending order, given in line 5. A binary search algorithm is used to identify the optional set that has the highest possible revenue, given the

Algorithm 1 Pseudocode for the 2DRSSP Heuristic

1: Input: Instance data
2: $\mathcal{O}^{best} = \emptyset$
3: Generate all possible optional cargo combinations: $\mathcal{O}_1..\mathcal{O}_n$
4: $\mathcal{P} = \{\mathcal{O}_1, \mathcal{O}_2, .., \mathcal{O}_n\}$
5: Sort($\mathcal{P}$) by revenue in descending order
6: **while** Binary search (BS) of $\mathcal{P}$ not finished **do**
7:     Select next optional cargo set, $\mathcal{O}^{temp}$, according to BS procedure
8:     **run** GRASP($\mathcal{O}^{temp}$) (Algorithm 2)
9:     **if** GRASP($\mathcal{O}^{temp}$) returns a feasible stowage plan **then**
10:         $\mathcal{O}^{best} = \mathcal{O}^{temp}$
11:         $\mathcal{X}$ = the returned feasible stowage plan
12:     **end if**
13: **end while**
14: **while** until stopping criteria met **do**
15:     **run** ALNS($\mathcal{X}$) (Algorithm 4)
16: **end while**
17: **return** Best solution from ALNS($\mathcal{X}$)

existence of a feasible stowage plan. The algorithm starts by selecting the middle element of $\mathcal{P}$ in the first iteration in line 7. Then, in line 8, the problem of finding a feasible stowage plan given this set choice is solved by a GRASP heuristic, provided and described in detail in Section 5.2. If this is successful, the elements in the lower half of $\mathcal{P}$ are eliminated, and the optional set is saved as $\mathcal{O}^{best}$. If the GRASP does not provide a feasible stowage plan, the upper half of $\mathcal{P}$ is eliminated. The lines 7-12 are repeated until $\mathcal{P}$ only consists of one element. From this point on, the choice of which optional cargoes to carry is fixed by $\mathcal{O}^{best}$. The feasible stowage plan generated by the GRASP when evaluating $\mathcal{O}^{best}$ is used as a starting point for the second phase.

The binary search algorithm used in this implementation requires few iterations, as the total number of optional cargoes gives the maximum number of optional set evaluated by the GRASP heuristic. However, this procedure does not guarantee that the optimal optional set is chosen. An example is given by the three new sets, $\mathcal{O}_1$, $\mathcal{O}_2$, $\mathcal{O}_3$, where the revenue of carrying $\mathcal{O}_1 > \mathcal{O}_2 > \mathcal{O}_3$. In the first iteration, $\mathcal{O}_2$ is rejected, as no feasible stowage solution was created by the GRASP. Then, $\mathcal{O}_1$ is removed from $\mathcal{P}$, together with $\mathcal{O}_2$. However, it could be possible to carry $\mathcal{O}_1$, as the total area usage of $\mathcal{O}_1$ could in practice be less than $\mathcal{O}_2$, despite that it generates more revenue. The customers cost of shipping a cargo could in practice be set on multiple factors as area, number of sailing legs carried, weight, height, loading complexity, contractual terms, rush order, etc. A simple way to handle this issue is to do a greedy search, evaluating the optional sets in a decreasing order based on revenue. The downside of this approach is that the maximum number of iterations is given by $2^{|\mathcal{C}^{\mathcal{O}}|}$, rapidly increasing with the number of optional cargoes available. Due to limited information on how the shipping costs are set in practice, it is assumed in this thesis that the revenue of carrying a cargo is linearly dependent on the total area usage of the cargo. This assumption seems reasonable, as there probably is a positive correlation between the area usage and the cost of shipping a cargo for the customer. It would also be logical to include the shipping distance in the equation. However, as the cargoes are transported from ports in a loading region to ports in an unloading region, the total transportation distance is roughly equal for all cargoes. Thus, the shipping distance is omitted. Based on these assumptions, the binary search makes a good fit. However, it is recommended that other search algorithms are considered in cases where the relation between revenue and area usage is more complex.

As the choice of which optional cargoes to bring was fixed by phase one, all selected cargoes are assumed mandatory to carry in phase two. The objective of the heuristic's second phase (lines 14-16) is to minimize the shifting cost for the mandatory cargoes and the chosen optional cargoes. Since a feasible stowage plan had to be generated in phase one in order to prove that $\mathcal{O}^{best}$ is possible to carry, an initial stowage plan $\mathcal{X}$ is available. Our proposed method for minimizing the shifting cost is to use the ALNS heuristic, presented in Section 5.3. Using the initial solution as a starting point, the method rearranges the vehicle placements in order to reduce the shifting cost. By rearranging for a number of iterations and evaluating the shifting cost for each promising rearrangement, the method tries to gradually reduce the shifting cost associated with the stowage plan.

5.2 GRASP Applied to the 2DRSSP

The greedy randomized adaptive seach procedure (GRASP) was initially proposed by Feo and Resende (1989). Feo and Resende (1989) developed GRASP in order to solve computationally difficult set covering problems, but the search procedure has been applied to many problems in different areas in recent time. Literature concerning the use of GRASP in 2D packing are provided by e.g. Alvarez-Valdes et al. (2005), Alvarez-Valdés et al. (2008), and Parreño et al. (2010). The common conclusion is that GRASP is well suited for packing and cutting problems, due to its fast computational time and high quality solutions. From Section 5.1, it is known that the first phase of the heuristic is to determine which optional set to carry. For a given optional set, a packing problem has to be solved in order to prove feasibility. As the reviewed literature recommend the use of GRASP on packing problems, we expect this framework to work well with the heuristic's first phase. Even though the GRASP provides a feasible solution, there is no guarantee that the vehicles are placed in a smart way, considering the shifting cost evaluation. For instance, a huge vehicle could be placed right in front of the exit ramp, despite that it is to be loaded in the first port and unloaded in the last. However, at this point, the quality of the solution is not important as long as it is feasible.

GRASP is a metaheuristic, in which each iteration consists of two phases: Construction and local search (Resende and Ribeiro, 2014). The construction phase builds a solution. Once a feasible solution is obtained, its neighborhood is investigated until a local minimum is found during the local search phase. Though, for the 2DRSSP Heuristic, only the first phase of the GRASP framework is used, i.e. the construction phase. Instead of using the local search phase to improve the solution, the ALNS heuristic is used, presented in Section 5.3. Due to this, only the construction phase is further elaborated in this section.

The heuristic is a greedy construction heuristic, with some random elements. Here, the term greedy describes the procedure of sampling the element with the best contribution to the objective value first, until solution construction is done. However, if the best element is always sampled, the exact same solution would be found in every iteration. Therefore, a restricted candidate list (RCL) is included in order to add some randomization to the search. This list consists of the n best insertion elements, where the chosen element is sampled randomly from the RCL. When an element is sampled from the list, the list is updated with the new best insertion elements, which is the adaptive aspect of the GRASP. Here, the length of the list defines the degree of randomization. If the length of the RCL is only one element, the best element would always be sampled, and the heuristic is pure greedy. With increasing list length, diversification is added to the search due to the increasing possible element choices.

In order for the GRASP to benefit from this RCL list length property, the GRASP must be able to change the list length throughout the search. Prais and Ribeiro (2000) suggest to implement a reactive element to the GRASP, which self-tunes the parameter based on its performance. The probability of selecting a given list length then depends on the previously obtained solutions for this same list length. This reactive feature is implemented for all search guiding parameters in the GRASP, presented in the following section.

The outline of this section is as follows: In Section 5.2.1, we introduce the GRASP framework applied to the 2DRSSP in addition to the search guiding parameters. Section 5.2.2 describes the procedure of creating the RCL.

5.2.1 The GRASP Framework

In this section, the GRASP Algorithm used in the 2DRSSP Heuristic is presented and explained. The objective of the algorithm is to create a feasible stowage plan, consisting of all vehicles in the mandatory cargoes and all vehicles in the given input set of optional cargoes $\mathcal{O}$. The GRASP algorithm is guided by four parameters: Search direction, RCL length, insertion objective, and random insertion. As the comprehension of the GRASP algorithm is highly dependent on these parameters, the parameters are explained first, followed by an elaboration of the GRASP algorithm.

The search direction is to decide which square in the grid to evaluate next. The algorithm is given eight search directions to choose from, that is two directions for each of the four corners of the deck layout. This is explained by using the square at the stern on the deck's port side as a starting point. From this square $(1,1)$, the search direction could either be $\{(1,1),(2,1)..(|\mathcal{I}|,1),(1,2),(2,2),..(|\mathcal{I}|,|\mathcal{J}|)\}$ or $\{(1,1),(1,2)..(1,|\mathcal{J}|),(2,1),(2,2),..(|\mathcal{I}|,|\mathcal{J}|)\}$. This same logic is used for all four corners of the grid, illustrated by Figure 5.1.

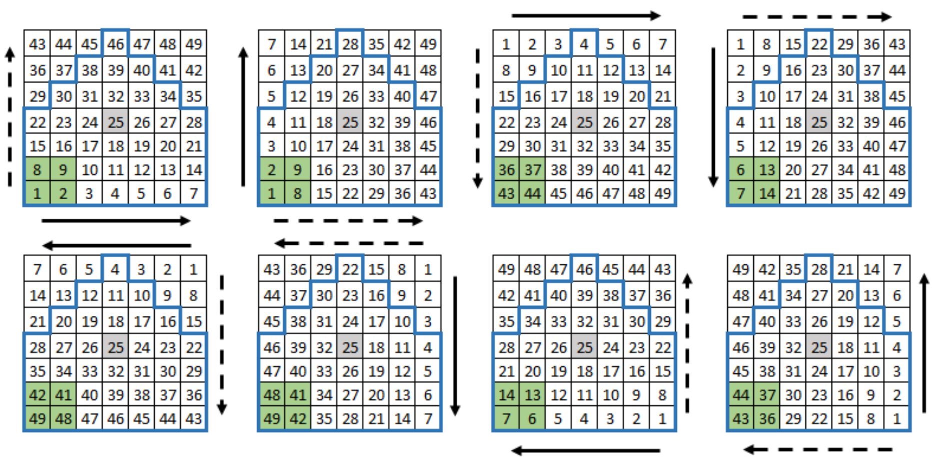


Figure 5.1: The eight search directions. The value of the squares gives the evaluation order, where the squares with the lowest values are evaluated first.

The reasoning for using a fixed search direction is that one should expect the resulting stowage plan to have less unused squares between the vehicles, compared to randomly drawing the next square. This will increase the success rate of the construction phase, as most of the remaining unused squares are located in the same region after each iteration.

The length of the restricted candidate list is an essential part of the GRASP, as stated in the previous section. For the 2DRSSP, the RCL consists of the n best cargoes which has a vehicle that can be placed in the evaluated square. This list is updated for each square evaluated, and which vehicle to place in this square is randomly chosen from the list. If no upper limit is set on this list length, the search becomes purely random. By defining an upper limit on the RCL length, only the best evaluated vehicles are considered.

As the RCL should consist of the n best cargoes, we have developed two objectives in order to evaluate which cargoes that are the best candidates in each iteration. A natural choice is to insert vehicles from the cargoes that have the highest area usage. By inserting the largest vehicles first, more flexibility is given in the later stages of the construction phase, as inserting small vehicles is easier in limited space. The second objective considers the number of possible locations for each cargo. Some cargoes may consist of heavy vehicles, which can only be placed at certain spots on the deck due to their heavy weight. The vehicles may also be abnormally wide and/or long, reducing the possible placements on the deck. Thus, the second objective tries to place vehicles from these cargoes first, reducing the probability of these cargoes not being placed.

Finally, the random insertion parameter is added to the GRASP algorithm in order to impose diversification to the search. For each square evaluated, the chosen vehicle to place on the deck is selected from the RCL. The random insertion parameter is set in the interval $0 \leq r \leq 1$. If the random insertion is accepted with probability r, the length of the RCL is set to unlimited for this square. By not limiting the size of the RCL, all cargoes that have a vehicle that can be placed in the evaluated square are included in the RCL, instead of only the l best cargoes. Hence, the vehicle selected is drawn randomly from all possible cargoes.

A pseudocode for the GRASP is presented in Algorithm 2. The vectors w_d, w_l, w_r, and w_o have a length equal to the number of different parameter choices for each search guiding parameter type. This means that w_d consists of eight elements, equal to the number of different search directions. For the random parameter, a discretization of the interval is done, in order to apply weights to each value. In line 2, initial weights are given to the search guiding parameters. For the search directions, a possible initial weighting could be $w_d = \{0.15, 0.15, 0.1, 0.1, 0.1, 0.1, 0.15, 0.15\}$. Then, the probability of choosing search directions 1, 2, 7, and 8, are slightly higher in each iteration. The chosen values of the weight parameters are set in Section 7.1.1.

Algorithm 2 Pseudocode for the 2DRSSP GRASP

procedure GRASP($\mathcal{O}$)
 Initialize weights: w_d, w_l, w_r, w_o
 while until stopping criteria met **do**
  $\mathcal{S} =$ Set of all squares $(i, j)\ \forall\ i \in \mathcal{I}, j \in \mathcal{J}$
  $\mathcal{X} = \{\}$
  $d =$ SelectSearchDirection(w_d)
  $l =$ SelectRCLLength(w_l)
  $r =$ SelectRandom(w_r)
  $o =$ SelectObjective(w_o)
  Sort($\mathcal{S}$) according to direction d
  while $\mathcal{S} \neq \emptyset$ **do**
   $(i, j) = \mathcal{S}_{first}$, and remove $\mathcal{S}_{first}$ from $\mathcal{S}$
   $RCL =$ makeRCL($(i, j), l, r, o$) according to Algorithm 3
   $x =$ SelectElementAtRandom(RCL)
   $\mathcal{X} = \mathcal{X} \cup \{x\}$
  end while
  if CheckFeasibility($\mathcal{X}$) = true **then**
   return $\mathcal{X}$
  end if
  UpdateWeights($w_d, w_l, w_r, w_o, \delta$)
 end while
 return $\{\}$
end procedure

The stopping criteria are typically set to a maximum running time or a maximum number of iterations. The chosen method to select the guiding parameters in lines 6-9 is a roulette wheel selection method, given in Equation 5.1. Here, the probability of selecting a given value for the parameters is given as the relative weight of the parameter. Using this equation on w_d from the last paragraph, the probability of choosing a search direction d is $p_d = \{15\%, 15\%, 10\%, 10\%, 10\%, 10\%, 15\%, 15\%\}$. However, only one of the four parameters is changed in each iteration, which is selected randomly. This is technically imposed by only allowing one out of the four Select-functions to change their choice.

$$(p_x)_i = \frac{(w_x)_i}{\sum_{i=1}^{N}(w_x)_i} \ \forall \ x = d, l, r, o \ , i = 1, 2, ...N, \text{ where } N = \text{length}(w_x) \quad (5.1)$$

The actual construction phase of the GRASP is given in lines 11-16. A sorted list consisting of all squares is given as input, and the construction continues until all squares are evaluated. For each iteration, the first element in the set including the squares, $\mathcal{S}$, is selected. For this square, an RCL is generated by the procedure presented in Section 5.2.2. Then, in line 14, an element from this list is sampled. This element refers to the variable x_{ijc} being equal to 1, for the drawn cargo from the RCL. In other words, the element states that in the selected square from $\mathcal{S}$, the lower left corner of a vehicle from cargo c is to be placed. As in the 2DRSSP Stowage Model, when a vehicle's corner is placed in a square, several of the nearby squares are also used for this same vehicle. Thus, these squares are removed from the set $\mathcal{S}$.

In the previous paragraph, we stated that the lower left corner of a vehicle is placed in the evaluated square. However, which corner of the vehicle that is placed in the evaluated square depends on the search direction. If a search direction starting in the lower left corner is used, the lower left corner of the vehicle is placed in the evaluated square. Following this logical structure, a vehicle's upper right corner is placed in the evaluated square if the search direction starts at the upper right square. The reason for this is to ensure that the best vehicles are inserted first. In order to explain this, we use a search direction starting in the lower left corner. Two vehicles are to be placed, one sized 3×3, and one sized 1×1. The first square evaluated is (1,1). Then, we check if it is possible to insert the lower left corner of the vehicles in this square. Assuming all other squares are empty, both vehicles can be inserted, and we select the largest vehicle, due to the insertion objective. However, if we check the possibility to insert the vehicles' upper right corner in this square, only the 1×1 sized vehicle can be inserted. The largest vehicle would use the squares (1,1), (0,1), (-1,1), (1,0), (0,0), (-1,0), (1,-1), (0,-1), and (-1,-1), where only (1,1) actually exists. Thus, it is essential to evaluate the vehicle's corner corresponding to the start square of the search direction, in order for the insertion objective to serve its purpose. Unless otherwise is stated, the default search direction is set to start in the lower left corner, and the lower left corner is used without further explanation if the following sections.

When all squares are evaluated, a feasibility check of $\mathcal{X}$ is performed. This feasibility check simply counts the number of vehicles placed for each cargo. If all vehicles in every carried cargo are placed, the stowage plan is feasible. Then, the objective of the GRASP is achieved, i.e. to create a feasible stowage plan, consisting of all vehicles in the mandatory cargoes and all vehicles in the given input set of optional cargoes $\mathcal{O}$. In this case, the stowage plan is returned to the 2DRSSP Heuristic, and the GRASP is terminated. However, if any vehicle from any cargo is missing, the feasibility check fails, and the weights are updated. For every iteration, the total area usage of the remaining vehicles is calculated. The weights are then updated by comparing the missing vehicles' area usage for the current iteration, to the previous iteration. If the sum of the missing vehicles area usage in the current stowage plan is less than the previous stowage plan, the updated parameter is rewarded a positive score δ. For the opposite outcome, a score of $-\delta$ is imposed to the parameter's weight. Since every stowage plan construction is affected by some randomness, this scoring system can seem unfair to the evaluated parameter. However, as only one parameter is changed for every iteration, this changed parameter is most likely the reason for the change in the missing vehicles' area usage. Hence, this simple scoring mechanism appears righteous and is therefore included as the reactive part of the GRASP. This inclusion of parameter weighting is done in order to increase the GRASP's success rate for each iteration. The computational performance of this reactive property is evaluated in Section 7.1.1.

5.2.2 Restricted Candidate List

This section describes how the Restricted Candidate List is created. The objective of the RCL is to give the GRASP a better basis for selecting the next vehicle to place in a square. This is done for a given square, by creating a list of the most promising cargoes to place a vehicle from. A pseudocode for the RCL creation procedure is presented in Algorithm 3.

The procedure starts by taking a square (i, j) as input. Then, for all cargoes carried, the algorithm checks if it is possible to insert a vehicle from these cargoes in this square. For a given cargo, the possible insertion function counts the number of vehicles from this cargo, which already has been placed on the deck. If this number equals the total number of vehicles belonging to this cargo, the function returns false, as there is no more vehicles from this cargo to place. However, if

Algorithm 3 Pseudocode for the Restricted Candidate List

```
procedure MAKERCL((i, j), l, r, o)
    RCL = {}
    for all c ∈ C^carried do
        if PossibleInsertion((i, j), c) = true then
            RCL = RCL ∪ {c}
        end if
    end for
    if Random number [0,1) < r then
        return RCL
    end if
    if o = 1 then
        Sort(RCL) by area in descending order
    end if
    if o = 2 then
        Sort(RCL) by number possible squares in ascending order
    end if
    RCL = {RCL_1, RCL_2, .., RCL_l}
    return RCL
end procedure
```

there still are remaining vehicles to place from this cargo, the function checks if it is allowed to place a vehicle in the evaluated square. Here, all nearby squares need to be empty and usable, in order to place the vehicle. Since it is desirable to place the lower left corner of the vehicle in square (i, j), all squares (i, j) for $i = i..i + S_c^L, j = j..j + S_c^W$ must be empty and usable, where S_c^L and S_c^W are respectively the numbers of squares a vehicle uses in the length and width direction. Because a logical search direction is used, these squares are often available, as they have yet to be evaluated by the GRASP. However, unusable space as pillars etc. placed in one or more of these squares may hinder a possible placement of a vehicle. An important aspect of this possible insertion function is that cargoes consisting of larger vehicles are less likely to be accepted. As large vehicles span over more squares, the probability of some of these squares being occupied increases.

When reaching line 8, the RCL consists of all possible cargoes, where a vehicle's corner can be placed in square (i, j). With the probability of r, the list may be returned as it is, consisting of all possible cargo insertions, according to the intent of the random parameter. If this is not the case, the list is sorted according to the objective o, discussed in Section 5.2.1. Finally, the list is reduced to the desired length l, by removing all elements after element l, if any. The returned RCL could be empty at this stage, meaning that no vehicles from the carried cargoes can be placed in the evaluated square.

5.3 ALNS Applied to the 2DRSSP

The adaptive large neighborhood search (ALNS) was initially proposed by Ropke and Pisinger (2006), where they applied it to a pickup and delivery problem with time windows. The search heuristic is based on a large neighborhood search (LNS), proposed by Shaw (1997). The main difference between ALNS and LNS is that the ALNS applies multiple removal and insertion operators with adaptive weights, whereas the LNS uses only one removal operator and one insertion operator (Gharehgozli et al., 2014).

The ALNS takes an initial solution and replaces it every time a better solution is found. This is done by sequentially destroying and repairing the solution. The destroy operators take a feasible solution and destroy a part of the solution. In order to create a feasible solution, a repair operator is used, which typically is a form of construction heuristic. The repaired solution is then compared to the best solution found so far and is accepted according to some criteria. If accepted, this new solution is destroyed and repaired in the next iteration, otherwise, the same feasible stowage plan is used, i.e. the stowage plan present before the destroy and repair procedure.

In contrary to the GRASP heuristic, there have to the authors' knowledge not been published any literature on ALNS applied on problems similar to the 2DRSSP, i.e. packing problems. However, ALNS has yielded promising results in other contexts including scheduling problems (Gharehgozli et al., 2014), and vehicle routing problems (Ropke and Pisinger, 2006; Ribeiro and Laporte, 2012). Due to the large neighborhoods and diversity of the neighborhoods, the ALNS algorithm explores large parts of the solution space in a structured way (Gendreau and Potvin, 2010). Based on our knowledge of the 2DRSSP, the ALNS framework seems like a promising improvement heuristic. By rearranging the placement of some of the vehicles on the deck according to placement and grouping principles, stowage plans with lower shifting cost may emerge. As stated in Section 5.2, the local search phase of the GRASP heuristic was discarded in favor of the ALNS. One of the main drawbacks of using ALNS instead of a typical local search heuristic is that local search heuristics are able to investigate a huge number of solutions within a short time limit in contrast to the ALNS. However, for the 2DRSSP Heuristic, it is the shifting cost evaluation of a solution that is time-consuming, as shown in Section 7.3.2. Actually, the time used destroying and repairing a solution is negligible, compared to the shifting cost evaluation of the stowage plan. Thus, as the ALNS explores larger neighborhoods in virtually the same time as local search heuristics explore small neighborhoods, the ALNS seems like the better choice for the 2DRSSP.

The outline of this section is as follows: In Section 5.3.1, we introduce the ALNS framework applied to the 2DRSSP. Sections 5.3.2 and 5.3.3 present the destroy and repair operators, respectively. Then, the adaptive weight adjustment procedure is given in Section 5.3.4. Finally, in Section 5.3.5 the selected acceptance criteria is discussed.

5.3.1 The ALNS Framework

In this section, we present the ALNS algorithm included in the 2DRSSP Heuristic. The objective of the ALNS is to minimize the shifting cost of the feasible stowage plan provided by the GRASP heuristic, by sequentially destroying and repairing the solution. The flowchart presented in Figure 5.2 gives an overview of the algorithm, intended to complement the pseudocode given in Algorithm 4. The algorithm is explained based on the pseudocode in the following paragraphs.

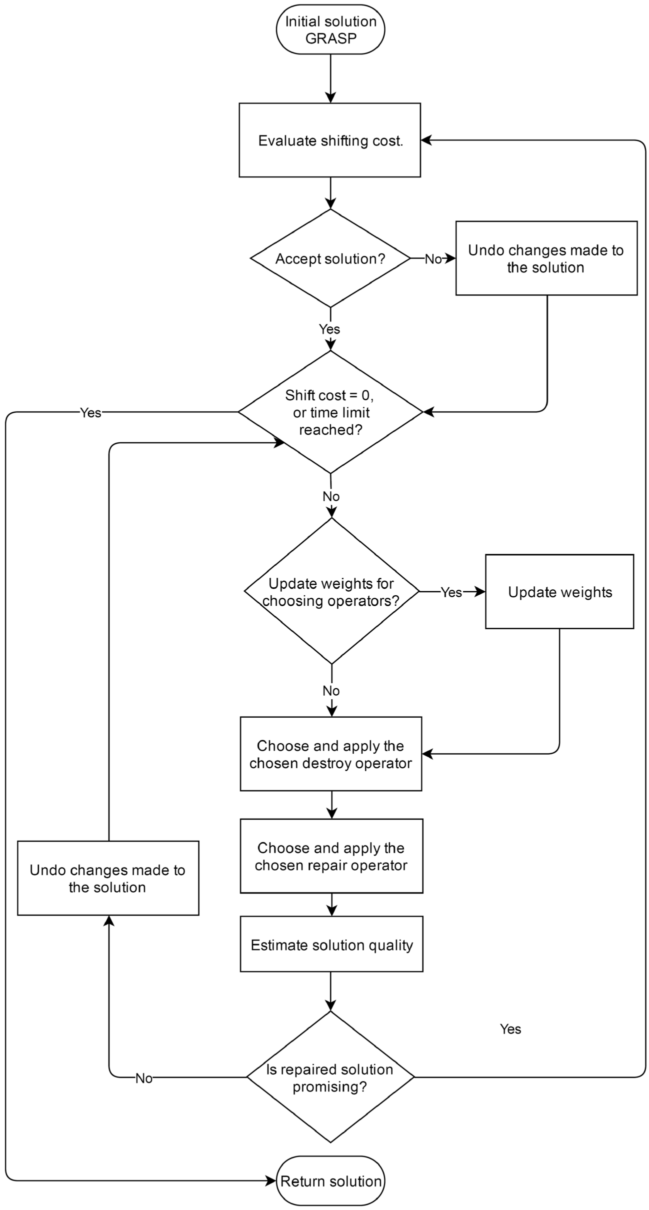


Figure 5.2: ALNS flowchart

Algorithm 4 Pseudocode for the 2DRSSP ALNS

procedure ALNS($\mathcal{X}$)
  Initialize weights: w^{dest}, w^{rep}
  Set segment size $= M$, iteration counter $i = 0$
  Start simulated annealing, $T = T_{start}, c =$ cooling rate
  $\mathcal{X}_{best} \leftarrow \mathcal{X}$
  Calculate shifting cost of initial solution, $Z(\mathcal{X}_{best}) = Z(\mathcal{X})$ (Section 5.4)
  while until stopping criteria met **do**
    $\mathcal{X}' \leftarrow \mathcal{X}$
    Select destroy and repair operators $d \in \mathcal{D}, r \in \mathcal{R}$ using w^{dest}, w^{rep}
    Apply destroy operator d to $\mathcal{X}'$
    Apply repair operator r to $\mathcal{X}'$
    if $(P(\mathcal{X}') \leq P(\mathcal{X})(1+p)$ or $G(\mathcal{X}')(1+g) \geq G(\mathcal{X}))$ and $\mathcal{X}' \neq \mathcal{X}$ **then**
      Calculate shifting cost of the current solution, $Z(\mathcal{X}')$
      if $Z(\mathcal{X}') < Z(\mathcal{X}_{best})$ **then**
        $\mathcal{X}_{best} \leftarrow \mathcal{X}'$
        $\mathcal{X} \leftarrow \mathcal{X}'$
      else if $Z(\mathcal{X}') < Z(\mathcal{X})$ **then**
        $\mathcal{X} \leftarrow \mathcal{X}'$
      else if accept($\mathcal{X}', \mathcal{X}, T$) according to Section 5.3.5 **then**
        $\mathcal{X} \leftarrow \mathcal{X}'$
      end if
    end if
    Award scores $\sigma_1, \sigma_2, \sigma_3$
    if i % $M = 0$ **then**
      Update weights w^{dest}, w^{rep} according to Equation 5.3
    end if
    $T = cT, i = i + 1$
  end while
  return $\mathcal{X}_{best}$
end procedure

The weights w^{dest} and w^{rep} give the probability of selecting the repair and destroy operators. Both of the vectors are handled in the same way as the weight parameters of the search guiding parameters in the GRASP. In line 3, the segment size is set. A segment is defined as M iterations, where the weights w^{dest} and w^{rep} are fixed in each segment, and updated every new segment (lines 24-26). As seen in line 4, the chosen acceptance criteria are based on simulated annealing. At this point, it is sufficient to note that a start temperature is set in addition to a cooling rate, as the acceptance method is discussed in details in Section 5.3.5. In lines 5-6, the shifting cost of the initial solution is calculated. Here, $Z(\mathcal{X})$ gives the shifting cost of the solution $\mathcal{X}$. At this stage, the initial solution is also stored as the overall best solution, as it is the only solution available.

The iterative ALNS search loop is given in lines 7-28. The stopping criteria are typically set to a maximum running time but are also terminated if a solution with zero shifting cost is found. The search loop starts by making a temporary stowage plan $\mathcal{X}'$, which is a copy of the current stowage plan $\mathcal{X}$. Then, a destroy and a repair operator are selected by the roulette wheel selection method, based on the operator weights w^{dest} and w^{rep}. The temporary stowage plan is then destroyed and repaired, using the selected operators.

In line 12, the algorithm decides whether to evaluate the shifting cost of the repaired stowage plan. Only promising stowage plans are evaluated, as the shifting evaluation procedure is quite time-consuming, as shown in Section 7.3.2. First, if $\mathcal{X}'$ is identical to $\mathcal{X}$, the stowage plan is not evaluated, as $Z(\mathcal{X}') = Z(\mathcal{X})$. If $\mathcal{X}'$ is unique and different from $\mathcal{X}$, we have chosen to use the grouping and placement objectives from the mathematical model to decide whether a stowage plan should be evaluated. As preliminary testing has shown that stowage plans with a high number of grouped vehicles often have lower shifting cost, this seems like a reasonable factor to base this decision upon. However, as a stowage solution with less neighboring vehicles may have better shifting cost, the grouping parameter g is included. The total number of neighboring vehicles in $\mathcal{X}'$ is denoted $G(\mathcal{X}')$. If $G(\mathcal{X}')(1 + g) \geq G(\mathcal{X})$, where g is the grouping parameter, greater or equal than zero, the shifting cost of the stowage plan is evaluated. Thus, if $G(\mathcal{X}')$=100, $G(\mathcal{X})$ = 105, and $g = 10\%$, a shifting evaluation is conducted, even though there are less neighboring vehicles in the temporary solution.

Like for the grouping objective, the placement objective is used to decide if the shifting cost of a stowage plan is to be evaluated. If $P(\mathcal{X}') \leq P(\mathcal{X})(1 + p)$, the shifting cost evaluation is conducted. Here, $P(\mathcal{X})$ is given by Equation 5.2, where lower is better, and p is the placement parameter.

$$P(\mathcal{X}) = \sum_{c \in \mathcal{C}} \sum_{i \in \mathcal{I}_c} \sum_{j \in \mathcal{J}_c} \sum_{i'=i}^{i+S_c^L-1} \sum_{j'=j}^{j+S_c^W-1} (P_c^U - P_c^L) \frac{C_{i'j'}^S x_{ijc}}{S_c^L S_c^W} \tag{5.2}$$

The equation is based on objective function (4.3) from Section 4.2. The placement cost for each vehicle is a function of the number of sailing legs a vehicle is placed on the ship, multiplied with the cost of using the chosen square placement. The cost of using a square represents the distance to the exit square, where squares near the exit have a high cost of use. This is based on the expectation that vehicles placed near the exit have a higher probability of blocking other vehicles on their entry/exit route, which induces the need for shifting. Thus, the cost of using a square is set equal to one for the square farthest away from the exit, and the number of moves to reach the exit square from this square is calculated. Then, for all other squares, the cost of using a given square is equal to the difference between the cost of the exit square and the number of moves to the given square, illustrated in Figure 5.3. Note that when evaluating the square cost, the unusable squares are treated as ordinary squares, in order to reduce the computational time of the square cost calculation. In Figure 5.3, the usable squares in the lower right corner have the lowest cost of use, i.e. cost of four and five. Thus, placing the vehicles with the highest number of sailing legs in these squares will keep the placement cost at a minimum, which seems reasonable in order to reduce the shifting cost.

If $\mathcal{X}'$ is considered as a promising stowage plan, by the evaluation criteria in line 12, the shifting cost of the stowage plan is evaluated in line 13, resulting in four possible outcomes. First, the shifting cost of $\mathcal{X}'$ could be better than the best-known stowage plan at this stage. If so, the temporary stowage plan replaces the best-known stowage plan $\mathcal{X}_{best}$ and the current stowage plan $\mathcal{X}$. Another possible outcome is that the temporary stowage plan is better than the current stowage plan, and thus the current stowage plan is replaced. This may occur if the subsequent outcome has occurred in a previous iteration. The third possible outcome is that the temporary stowage plan is accepted, even though it is worse than the current stowage plan. The reason for accepting worse solutions is discussed in Section 5.3.5. Finally, in the last possible outcome, the temporary stowage plan is simply rejected due to worse shifting cost than the current stowage plan.

In line 23, the selected destroy and repair operators are awarded scores based on the outcome in lines 14-22. These scores are used to update the weights of the destroy and repair operators in line 25, for every new segment. The scoring mechanism is described in Section 5.3.4. Finally, the temperature is updated in every iteration, based on the cooling rate, and the next iteration of the ALNS search loop starts. When a stopping criterion is met, the best stowage plan is

13	12	11	10	9	8	7	6	5	4	3	2	1
14	13	12	11	10	9	8	7	6	5	4	3	2
15	14	13	12	11	10	9	8	7	6	5	4	3
16	15	14	13	12	11	10	9	8	7	6	5	4
17	16	15	14	13	12	11	10	9	8	7	6	5
18	17	16	15	14	13	12	11	10	9	8	7	6
19	18	17	16	15	14	13	12	11	10	9	8	7
20	19	18	17	16	15	14	13	12	11	10	9	8
19	18	17	16	15	14	13	12	11	10	9	8	7
18	17	16	15	14	13	12	11	10	9	8	7	6
17	16	15	14	13	12	11	10	9	8	7	6	5
16	15	14	13	12	11	10	9	8	7	6	5	4
15	14	13	12	11	10	9	8	7	6	5	4	3

Figure 5.3: The number in each square gives the cost of using this square, C_{ij}^{S}

returned to the 2DRSSP Heuristic, marking the end of both the the ALNS part of the heuristic and 2DRSSP Heuristic.

5.3.2 Destroy Operators

In this section, the destroy operators used in the ALNS are presented. The main objective of the destroy operators is to destroy a part of the solution so that the repair operators are able to build new and better stowage plans. Thus, by developing well functioning destroy operators, the repair operators' success rate are assumed to increase. Therefore, we present six different destroy operators in this section, where the majority are motivated by some sort of problem specific characteristics.

When applying a certain destroy operator, the degree of destruction will influence the performance of the repair operators. Therefore, a destruction parameter ξ is set in the interval $0 < \xi \leq 1$, which represents the maximum share of the total number of vehicles removed from the solution. As stated in Ropke and Pisinger (2006), if ξ is too low, the heuristic is not able to move very far in each iteration, and it has a higher chance of being trapped in a local optimum. For the 2DRSSP, this translates to removing too few vehicles, where the repair operators could find it difficult to create improving solutions. On the other hand, if ξ is too high, the search space increases, but the repair operators' performance may suffer. In addition, the time used by the destroy and repair operators will increase with increasing ξ (Ropke and Pisinger, 2006). The ξ parameter is used by all destroy operators presented in the following paragraph unless otherwise stated.

Grouping Removal

This operator removes all vehicles that do not have a neighboring vehicle from the same cargo, based on the grouping principle. A neighboring vehicle is defined as in Section 4.2, where the neighboring vehicle is placed exactly to the left, right, in front or behind a vehicle from the same cargo. The purpose of this operator is to increase the number of neighboring vehicles, which is assumed to reduce the shifting cost of the stowage plan. What makes this destroy operator promising, is that if the repair operator finds a feasible solution, there will be at least as many grouped vehicles in the new stowage plan as it was in the previous one. The destroy operator is not affected by the destruction parameter ξ, hence all vehicles without a neighbor is removed.

Area Removal

The purpose of the Area Removal operator is to remove all vehicles placed in the selected area of the deck. The operator randomly chooses a start square (i_s, j_s), and an end square (i_e, j_e), where $i_s < i_e$, $j_s < j_e$. Then, all vehicles with the lower left corner placed in square (i, j), for all $i \in i_s..i_e, j \in j_s..j_e$, are removed from the deck. The size of the selected area is randomly drawn, with an upper limit of ξ. By removing all vehicles from a certain area, the reinsertion procedures try to improve the vehicles placement on a subsegment of the deck, which could reduce the overall shifting cost of the stowage plan.

Random Removal

The Random Removal operator removes randomly selected vehicles from the deck. The performance of this removal heuristic is expected to be poor, as no information from the problem is used. Thus, the inclusion is purely based on the fact that the operator induces diversification to the search (Pisinger and Ropke, 2007) .

Port Removal

This destroy operator randomly selects a port, which could be either a loading or an unloading port. Then, all vehicles loaded or unloaded at the selected port are removed from stowage plan. Additionally, the operator removes a random number of other vehicles placed on the deck. This is done in order to increase the possible locations of each vehicle in the repair phase.

Route Removal

The Route Removal operator removes vehicles based on the shifting evaluation results of the current stowage plan, using the *route costs*. There is a route cost associated with each vehicle in a cargo, at the cargo's loading and unloading port. For a given vehicle in its loading port, the route cost is defined as the sum of the costs of shifting all vehicles blocking the given entry path. In other words, the route cost reflects how expensive the vehicle is to load. An example is now given by a vehicle loaded at a given port. To reach its assigned destination on the deck, four vehicles need to be shifted. The shifted vehicles have areas of 16, 30, 10, and 18. The route cost of the entry route for the vehicle is, therefore, $R_{cvp} = 74$, as the cost of shifting a vehicle is given by its area. A route cost is calculated for all vehicles v in all carried cargoes c, in the loading and unloading port of cargo c when evaluating the shifting cost of a stowage plan. The Route Removal operator then uses these route costs when destroying the solution. The main objective of this procedure is to remove the vehicles with the highest route cost from the deck. A pseudocode for the Route Removal operator is given in Algorithm 5.

The previous paragraph describes the procedure in lines 2-6. In line 7, a list consisting of all R_{cvp} is sorted by cost in decreasing order. Then, in line 11, the first entry from the list is removed, and the vehicle corresponding to this element is removed from the deck. This is repeated until a randomly selected share κ of the vehicles is removed, with an upper limit of τ. The if-statement in line 13 is included as each vehicle has two entries in $\mathcal{R}$, one for the entry route and one for the exit route. If a vehicle is already removed due to the route cost of the entry route, no action is taken when the exit route for this same vehicle is drawn from $\mathcal{R}$. The sum of the area of all these removed vehicles is denoted A^R. In order to give the repair operator more deck area to work with, all vehicles placed in a certain area around the exit square are also removed. The motivation for this choice is that vehicles with high route cost are expected to contribute less to the total shifting cost of the stowage plan if they are moved closer to the exit.

In order to remove vehicles from the area around the exit square, the Area Removal operator is called in lines 19-24. The start square (i_s, j_s), and an end square

Algorithm 5 Pseudocode for the Route Removal destroy operator

procedure ROUTEREMOVAL($\mathcal{X}'$)
    $\mathcal{R} = \{\}$ (Set of route costs)
    for all $c \in \mathcal{C}^{carried}, v \in \mathcal{V}_c, p \in \mathcal{P}_c^{LU}$ **do**
        Calculate the route cost, R_{cvp} based on the shifting evaluation results
        $\mathcal{R} = \mathcal{R} \cup \{R_{cvp}\}$
    end for
    Sort($\mathcal{R}$) by route cost in descending order
    $\kappa =$ random number $(0,\tau]$
    $A^R = 0, A^T =$ Total area of all vehicles on deck
    while $A^R < \kappa A^T$ **do**
        Pick and remove first element from $\mathcal{R}$
        Get (i, j, c) of the corresponding vehicle in this element
        if $x_{ijc} = 1$ **then**
            $x_{ijc} = 0$, $A^R = A^R + S_c^L S_c^W$
        end if
    end while
    $A^A = 0$, $i_s = i_e = i_{exit}$, $j_s = j_e = j_{exit}$
    while $\rho A^A < A^R$ and unsearched squares exists **do**
        for all $i \in i_s..i_e, j \in j_s..j_e$, $c \in \mathcal{C}^{carried}$ **do**
            if $x_{ijc} = 1$ **then**
                $x_{ijc} = 0$, $A^A = A^A + S_c^L S_c^W$
            end if
        end for
        $i_s = i_s - 1, j_s = j_s - 1, i_e = i_e + 1, j_e = j_e + 1$
    end while
    return $\mathcal{X}'$ (destroyed solution)
end procedure

(i_e, j_e) are both set equal to the exit square. Then, all vehicles that use a square (i, j), for all $i = i \in i_s..i_e, j \in j_s..j_e$, are removed from the deck, and the sum of all these vehicles is denoted A^A. While $\rho A^A < A^R$, i_s and j_s are reduced by one, and i_e and j_e are increased by one in each iteration, expanding the area around the exit square.

The parameters τ and ρ are only used in the Route Removal and the Shift Removal operators. As there are a lot parameters connected to the ALNS, these parameters are discussed and set in this paragraph, hence left out of the parameter tuning section. τ sets the upper bound on κ, which denotes the share of vehicles removed due to high route cost. Here, $\tau = 0.4$, and $\rho = 1.5$. If κ is selected randomly at its upper bound τ, then approximately 40% of the vehicles are removed from the deck based on the route costs. Additionally, as $\rho = 1.5$, approximately 60% of the total number of vehicles is removed by the area removal, resulting in a total of 40%+60% = 100% destruction, i.e an empty deck. Thus, the degree of destruction is approximately given by $\kappa + \rho\kappa = 2.5\kappa$. As κ is drawn randomly in the interval $(0, 0.4]$, the average degree of destruction is 50%. ρ is set to 1.5 because we want to give all vehicles removed due to high route cost the opportunity to be placed in the area around the exit square by a repair operator. The upper bound on κ then follows naturally, as $\kappa > 0.4$ result in an empty deck. The operator is not affected by the destruction degree ξ.

Shift Removal

The Shift Removal operator works in a similar way as the Route Removal. However, instead of removing the vehicles with the highest entry/exit route cost, the vehicles that contribute the most to the shifting cost are removed. This contribution is denoted *vehicle shifting cost*, which is given by the number of ports where a vehicle is shifted times the cost of shifting the vehicle. A consequence of this is that a large vehicle shifted in only one port may have a higher vehicle shifting cost than a small vehicle shifted in two ports.

As the removal procedure is similar to the Route Removal, a pseudocode for this operator is not provided. The procedure is explained using Algorithm 5, where only the changes are commented upon. Here, $\mathcal{S}^V$ is used instead of $\mathcal{R}$, which denotes the set of vehicle shifting costs. In line 4, the vehicle shifting cost is calculated instead of the route cost for all $c \in \mathcal{C}^{carried}, v \in \mathcal{V}_c$. In line 17, the center of the destroyed area is moved. As it is expected that the vehicles with the highest vehicle shifting cost are placed near the exit, it is desirable to enable the repair operators to move these vehicles away from the exit. Of that reason, a square on the opposite side of the deck is used as the Area Removal center, where $i_s = i_e = i_{opposite}$, $j_s = j_e = j_{opposite}$, $i_{opposite} = |I| - i_{exit}$, $j_{opposite} = |J| - j_{exit}$.

If the exit square is placed near the center of the deck, the start square of the Area Removal should be adjusted, as the opposite square is placed in the same area. However, as a typical deck often has the entry/exit ramp placed on the side of the deck, the use of an opposite square seems reasonable. The operator is not affected by the destruction degree ξ.

5.3.3 Repair Operators

After using a destroy operator to destroy the solution, the aim of the repair operator is to repair the solution, and hopefully provide a new and better stowage plan. As the degree of destruction varies each iteration, the work needed in order to repair a solution will vary. A stowage plan can always be repaired, regardless of the destruction degree, but we are only interested in the new solution if an improved stowage plan is created. Thus, the development of well-performing repair operators are essential in order to create improved stowage plans. The reader should note that in order for a solution to be considered repaired, it has to be classified as a feasible stowage plan, i.e. all vehicles removed from the solution by the destroy operator has to be reinserted on the deck. This section describes the five proposed repair operators for the 2DRSSP.

GRASP Repair

The GRASP Repair operator uses the procedure of the GRASP heuristic from Section 5.2.1 to repair the solution. The only difference between the GRASP heuristic and the GRASP Repair operator, is that the repair operator is used on a solution where some vehicles are placed in advance, i.e. not removed by a destroy operator. Therefore, the algorithm has been slightly modified in order to handle this exception. Instead of starting with the empty set in line 5 in Algorithm 2, the GRASP Repair starts with $\mathcal{X} = \mathcal{X}'$. This will limit the possible insertions in the RCL, as more squares are used in advance.

The inclusion of this operator is based on the fact that as the GRASP is able to provide feasible solutions for an empty deck, it should also be able to create feasible solutions from destroyed solutions. At least it is able to replicate the exact solution present before the destroy procedure, given unlimited time to repair. This is of course not desirable, as this gives the exact same shifting cost as before. Since the GRASP does not use any information from the problem other than inserting the most complicated vehicles with respect to the area or available squares first, there is no factor implying an improved stowage solution after the repair procedure. Despite this fact, the repaired solution may be better and the operator is therefore included. An upper limit on the time used by the operator is set by

t^{rep}. If no feasible solution is provided within this time frame, an empty stowage solution is returned.

Grouping Repair

The Grouping Repair operator tries to exploit the fact that a higher number of grouped vehicles is expected to reduce the shifting cost of a stowage plan. The repair procedure evaluates the squares according to the default search direction. For each unused square (i, j), the operator checks if the square is a neighbor square to any of the cargoes. A square is classified as a neighbor square for a given cargo if there is a vehicle from the same cargo placed in one of the squares $(i + S_c^L, j)$, $(i - S_c^L, j)$, $(i, j + S_c^W)$, or $(i, j - S_c^W)$. Then, if any of the removed vehicles belong to a cargo where the square is classified as a neighbor square, an insertion is attempted. If the square is a neighbor square for multiple cargoes, the vehicle with the highest area usage is tried inserting first. The insertion is successful if all squares needed to place the vehicle with its lower left corner in the neighbor square are free and usable. When all squares are evaluated, the procedure stops.

At this point, some of the removed vehicles are probably reinserted on the deck, reducing the number of unplaced vehicles. In order to place the remaining vehicles, the GRASP Repair is used. If the GRASP manages to create a feasible solution, the repair operator is expected to yield promising stowage plans. However, the insertions made by the Grouping Repair operator could complicate the GRASP Repair procedure and in some cases make it impossible to create a feasible stowage plan. The closer the gap between the total area use of all carried cargoes and the deck area capacity is, the fewer possible stowage plans exist. Thus, by forcing some of the removed vehicles to stay in certain neighbor squares, a feasible insertion of the remaining vehicles could simply not exist. Therefore, we assume the relative success rate of this operator compared to the other repair operators to be better on decks with less vehicles.

Placement Repair

This operator considers the placement cost of each vehicle when reinserting the vehicles. The Placement Repair operator is motivated by the expectation that vehicles placed farthest away from the entry/exit are less exposed to shifting. Therefore, the squares farthest away from the entry/exit are evaluated first and vehicles from the cargoes with the most number of sailing legs are desirable to place in these squares. A pseudocode for the Placement Repair operator is given in Algorithm 6

Algorithm 6 Pseudocode for the Placement Repair operator

procedure PlacementRepair($\mathcal{X}'$)
    $\mathcal{S} =$ Set of all squares (i, j) $\forall$ $i \in \mathcal{I}, j \in \mathcal{J}$
    Sort($\mathcal{S}$) by square cost (C^s_{ij}) in ascending order
    while $\mathcal{S} \neq \emptyset$ **do**
        $\mathcal{L} = \emptyset$
        $(i, j) = \mathcal{S}_{first}$, and remove $\mathcal{S}_{first}$ from $\mathcal{S}$
        for all $c \in \mathcal{C}^{unplaced}$ **do**
            if PossibleInsertion($(i, j), c$) = true **then**
                $\mathcal{L} = \mathcal{L} \cup \{c\}$
            end if
        end for
        if $\mathcal{L}$ $! = \emptyset$ **then**
            $c =$ Element in $\mathcal{L}$ with highest $P^U_c - P^L_c$
            $x_{ijc} = 1$, where (i, j) is set according to insertion corner
        end if
    end while
    if CheckFeasibility($\mathcal{X}'$) = true **then**
        return $\mathcal{X}'$
    else
        return $\{\}$
    end if
end procedure

The procedure starts by evaluating the square with the lowest square cost. No unusable and occupied squares are evaluated. In lines 7-11, the repair operator then checks if one of the corners of a vehicle from cargo c can be placed in square (i, j) for all cargoes. Normally, only the lower left corner of a vehicle is tried inserted in these situations. However, for the placement repair operator, all corners of the vehicle are tried. Given an example based on Figure 5.3 where the deck is empty, the usable square with the lowest square cost is placed at the lower right corner, with $C_{ij}^S = 4$. If there are multiple squares with the same square cost, one of these squares is selected randomly, as is the case in Figure 5.3. We choose the square at stern, and a vehicle sized 3×3 is desired to insert, according to the evaluation criteria in line 13. If only the lower left corner is considered, the operator could insert a vehicle poorly evaluated by the evaluation criteria and sized 3×1. Thus, all four corners of the vehicle are attempted. If multiple cargoes are evaluated equally, the largest cargo is selected. When the chosen vehicle is placed on the deck in line 14, the location of the lower left corner is identified and set accordingly. In the example, a possible insertion is to place the desired vehicle sized 3×3 with its lower right corner in square $(1, 12)$, which gives $x_{1,12-S_c^W+1,c} = x_{1,10,c} = 1$.

For each iteration, the $\mathcal{C}^{unplaced}$ is updated according to the remaining vehicles. If all vehicles in a carried cargo are placed on the deck, the cargo is removed from $\mathcal{C}^{unplaced}$. The loop continues until all unused squares are evaluated. Then, in line 17, the feasibility check ensures that all vehicles in every carried cargo are placed on the deck. If this is not the case, an empty stowage plan is returned. Unlike the GRASP and Grouping repair operators, this operator terminates after one construction, and no time limit is needed.

Random Repair

The Random Repair operator is the simplest of all the repair operators. All the unused squares on the deck are stored in a set. The operator then picks and removes a random square from the set and tries to insert an unplaced vehicle in this square. The vehicles are attempted inserted in descending order with respect to area usage. This reinsertion procedure is done until the set is empty and a feasibility check then is conducted. If feasible, the stowage plan is returned to the ALNS.

The performance of this removal heuristic is expected to be poor when the destruction degree is high, as unstructured vehicle placement gives low success rate. However, if only a small part of the solution is destroyed, the operator success rate increases. In addition, the operator is extremely fast due to its simple structure. The operator also adds diversification to the search, as for the Random Removal.

MIP Repair

The MIP Repair operator uses the stowage model from Section 4.2 to repair the destroyed solution. The variables x_{ijc} are fixed to one for the vehicles not removed by the destroy operator. In addition, the variables y_c are fixed according to the optional cargoes carried, selected by the GRASP. The remaining variables are then set to either one or zero by the solver and the repaired solution is returned to the ALNS. If no time limit is set, the solver provides an optimal solution, given the fixed vehicle placements. It is obvious that this repair operator is more time consuming than the proposed heuristic repair operators, but the time usage could be justified if it often provides promising solutions.

5.3.4 Adaptive Weight Adjustment

This section describes how the weights of the destroy and repair operators can be automatically adjusted using the scores they are given from earlier iterations. The content in this section is fully based on the adaptive weight adjustment section in Ropke and Pisinger (2006). Unless otherwise stated, the described procedure is exactly the same as given in Ropke and Pisinger (2006), and should be treated as a reproduction of their work.

The basic idea behind the adaptive weight adjustment is to keep track of a score for each repair and destroy operator, which measures how well the operators have performed recently. A high score corresponds to a successful operator. The entire search is divided into a number of segments. A segment is a number of iterations of the ALNS heuristic, denoted M in our case. The score of all operators is set to zero at the start of each segment. The score of an operator is increased by either σ_1, σ_2, or σ_3 in the situations shown in Table 5.1. The last possible outcome is that the new solution is equally good or worse than the current solution and is not accepted by simulated annealing. As none of the descriptions in Table 5.1 cover this outcome, a score of zero is given in our problem.

Table 5.1: Values of ALNS parameters.

Parameter	**Description**
σ_1	New global best solution is found
σ_2	The new solution is better than the current solution
σ_3	The new solution is accepted as a part of simulated annealing

In each iteration a destroy operator and a repair operator are applied. The scores for both operators are updated by the same amount, as we cannot tell whether it was the destroy or the repair operator that was the reason for the outcome. At the end of each segment, the new weights are calculated using the recorded scores. In the first segment, we weight all operators equally. At the end of each segment, the new weights are calculated by Equation 5.3.

$$w_i = w_i(1 - \eta) + \eta \frac{\pi_i}{\theta_i} \tag{5.3}$$

Here, π_i is the score of operator i obtained during the last segment, and θ_i is the number of times we have attempted to use operator i during the last segment. The reaction factor, η, controls how quickly the weight-adjustment algorithm reacts to changes in the effectiveness of the operators. If η is zero, then we do not use the scores at all and stick to the initial weights. If η is set to one, then we let the score obtained in the last segment decide the weight.

5.3.5 Acceptance Criteria

When deciding whether to keep the repaired stowage plan in favor of the current stowage plan, some sort of acceptance criteria must be set. If the repaired stowage plan has a lower shifting cost than the current stowage plan, it is accepted. This is quite natural, as the objective of the ALNS is to find the stowage plan with the lowest shifting cost. However, this greedy acceptance criterion alone may trap the heuristic in a local minimum. Thus, it seems reasonable to accept worse solutions occasionally (Ropke and Pisinger, 2006) in order to diversify the search to find better solutions. There exist several acceptance criteria that can be used to achieve this. Threshold Accepting (Dueck and Scheuer, 1990), Old Bachelor Acceptance (Hu et al., 1995), Great Deluge Algorithm (Dueck, 1993), and Simulated Annealing (Kirkpatrick et al., 1983) are all possible alternatives as an acceptance method. In Section 5.3, we mentioned some papers using ALNS on other problem types. Common for all these papers is that the acceptance criteria from simulated annealing are used. As this seems like the preferred acceptance criteria in combination with ALNS, this is also used in our implementation.

For simulated annealing, every improving solution is accepted. We also accept a solution $\mathcal{X}'$, given the best solution $\mathcal{X}_{best}$, with probability $e^{-(Z(\mathcal{X}')-Z(\mathcal{X}_{best}))/T}$. Here T is the temperature, $Z(\mathcal{X}')$ and $Z(\mathcal{X}_{best})$ are the shifting costs of the repaired solution and the best solution, respectively. The temperature starts out at T_{start} and is decreased every iteration using the expression $T = cT$, where $0 < c < 1$ is the cooling rate. As in Ropke and Pisinger (2006), we set the start temperature based on the initial solution. The start temperature is set such that a solution that is γ worse than the best solution is accepted with probability 0.5. As T_{start}

is set based on the initial solution and γ, γ is now the parameter that needs to be set, denoted *start temperature control parameter*.

5.4 Evaluating the Stowage Plans

This section introduces the shifting cost estimation procedure used in the ALNS part of 2DRSSP Heuristic and provides some implementation details. The shifting cost estimation is an essential part of the ALNS, as the shifting cost is used to compare stowage plans. Section 4.4 presented the shifting evaluation model, which is used to find the optimal entry and exit routes for all vehicles that minimize the shifting cost of a stowage plan. However, preliminary testing has shown that Xpress is not able to solve realistically sized instances within a reasonable time limit, as the time usage heavily depends on the grid size and number of vehicles. As the shifting cost is calculated for every promising stowage plan in the ALNS, the running time of the evaluation method becomes very important. Thus, we have chosen to estimate the shifting cost of a stowage plan by solving a one-to-all shortest path problem (SPP) for every cargo both for its loading and unloading port. This approach gives an upper bound on the shifting cost since it does not take the shifts made for other entering/exiting vehicles into account, as is illustrated in Figure 4.3. A comparison between the shifting evaluation model and the SPP approach for small instances is given in Section 7.1.3.

The one-to-all SPP is the problem of finding the shortest path between a start node and all other nodes in a graph, such that the sum of the cost of traversing the edges is minimized. In the 2DRSSP, this translates to finding the cheapest path between the entry/exit square, and a vehicle's location on the deck for all vehicles. Each square is a node, and the edge cost equals the cost of moving a given vehicle from a square to a neighbor square. The term neighbor square is earlier used when referring to a square where a neighboring vehicle is placed. Note that in this section, unlike the previous ones, a neighbor square corresponds to a square placed right above, below, to the left, or the right of the given square. The procedure is divided into two steps. First, we create the graphs, described in Section 5.4.1. Then, in Section 5.4.2, the solution procedure used to solve the one-to-all SPP in a graph is discussed. Here, the shifting cost of a stowage plan is estimated, using the routes from the SPP solutions.

5.4.1 Creating the Graphs

The first step of the SPP approach is to create the graphs. However, we are not concerned with finding the shortest path in distance, but rather the cheapest path, i.e. the exit route that gives the lowest contribution to the shifting cost. In the previous section, it is stated that we need to solve a one-to-all SPP for each cargo in both the loading and unloading port of the cargo. Thus, we need to create two times the number of carried cargoes graphs ($2|\mathcal{C}^{carried}|$), which may be confusing. The most apparent approach would be to solve one SPP for each port, where the cheapest path for each entering or exiting vehicle is calculated. The problem with this approach is that the edge costs between the nodes are dependent on the size of the vehicle traversing the edge. We explain this using a 2×2 sized vehicle and a 4×4 sized vehicle, both initially placed with their lower left corner in square $(1, 3)$, as illustrated in Figure 5.4. Then, we evaluate the edge cost associated with moving one square to the left for both vehicles. For the smallest vehicle, the edge cost is zero, as moving from square $(1, 3)$ to square $(1, 2)$ does not impose any shifts. However, the largest vehicle uses a square where another vehicle is placed. The edge cost of this move is six, given by the cost of shifting the blocking vehicle.

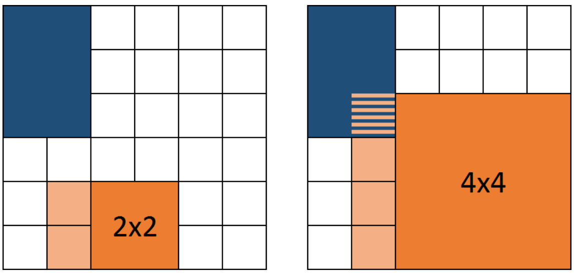


Figure 5.4: The cost of moving a vehicle from a square to a neighbor square varies with the size of the moving vehicle.

Since the edge costs vary with the size of the cargo, a graph must be created for every cargo, for their loading and unloading port. The procedure of creating the graphs is given in Algorithm 7.

The procedure takes a feasible stowage plan as input. Then, a graph is created for all carried cargoes, for their loading and unloading port. Here, V denotes the set of nodes (vertices), E the set of edges, and C the set of all edge costs. For a given square (i, j), cargo c and port p, the procedure checks if is feasible to place the lower left corner of a vehicle from cargo c in square (i, j) in line 5. In order for this to be feasible, it is sufficient that all squares (i, j) for $i = i..i+S_c^L, j = j..j+S_c^W$ are usable, i.e. not defined as unusable space. Thus, other vehicles may well occupy one of these squares as the squares can be used at the cost of shifting the vehicle placed there. If this feasibility evaluation succeeds, the square is added to the graph as a node. Then, in lines 7-8, all neighbor squares are evaluated. Usually, four neighbor squares exist for a square. However, fewer neighbor squares exist for the squares on the edge of the deck and in the corners. If it is possible to place

Algorithm 7 Creation procedure for the graphs

1: **procedure** MAKEGRAPHS($\mathcal{X}$)
2:     **for all** $c \in \mathcal{C}^{carried}, p \in \mathcal{P}_c^{LU}$ **do**
3:         Create a graph $G_{cp}(V, E, C)$
4:         **for all** $i \in \mathcal{I}, j \in \mathcal{J}$ **do**
5:             **if** FeasibleCornerPlacement($(i, j), c$) = true **then**
6:                 $V = V \cup \{(i, j)\}$
7:                 **for** $(i', j') = (i + 1, j)$ & $(i - 1, j)$ & $(i, j + 1)$ & $(i, j - 1)$ **do**
8:                     **if** FeasibleCornerPlacement($(i', j'), c$) = true **then**
9:                         $E = E \cup \{(i, j, i', j')\}$
10:                         Calculate $c_{i,j,i',j'}$, $C = C \cup \{c_{i,j,i',j'}\}$
11:                     **end if**
12:                 **end for**
13:             **end if**
14:         **end for**
15:     **end for**
16:     **return** all graphs $G_{cp}(V, E, C)$
17: **end procedure**

the corner of a vehicle from cargo c in a neighbor square, then there must exist an edge between these squares. The feasible edges are added to the set E.

In line 10, the cost of traversing an edge is calculated. This cost depends on the size of the vehicle in a given cargo that is to be moved, as illustrated in Figure 5.4. In addition, the cost is only accounted for if it is the first time the vehicle moves to a square occupied by another vehicle that is not loaded/unloaded at the same port. We see this in the example based on Figure 5.4, where the edge cost of moving the large vehicle from square $(1, 3)$ to $(1, 2)$ is six, denoted $c_{1,3,1,2} = 6$. Continuing the example, we now state that the a new move from $(1, 2)$ to $(1, 1)$ has the cost of zero, $c_{1,2,1,1} = 0$. As the move from $(1, 3)$ to $(1, 2)$ already has accounted for the cost of shifting the blocking vehicle, the second move induces no cost. To make sure this is accounted for in our solution method we consider the lower left corner of the moving vehicle, square (i, j), together with the other squares that the vehicle covers, squares $i = i..i + S_c^L, j = j..j + S_c^W$. The procedure then checks if any other vehicle uses these squares. If so, these vehicles are recorded in a list. Then, when making a move, e.g. to the left, a new list is created, recording the vehicles occupying the new squares. The two lists are then compared, and the cost of moving equals the area of the vehicles that are present in the new list, but not included in the first list. One final note regarding this procedure is that vehicles that are to be loaded or unloaded in the same port are never added to the lists.

This is based on the fact that there always exists a loading/unloading sequence, where none of the loading/unloading vehicles block the other loading/unloading vehicles, as discussed in Section 4.3.

Example

The procedure of creating the graphs is concerned with a considerable amount of aspects. In order to clarify the most important aspects, we give an example illustrating the graph creation procedure. Figure 5.5 gives the stowage plan of a small deck at a given port. The two orange vehicles from cargo 1 are to be loaded at this port, and placed in the illustrated squares. The other vehicles are loaded at a previous port and have to be shifted if the orange vehicles use a path going through them. Here, the selected exit square is marked with green and unusable squares marked with gray. We now create the graph belonging to cargo 1's loading port, based on this stowage plan, given in Figure 5.6.

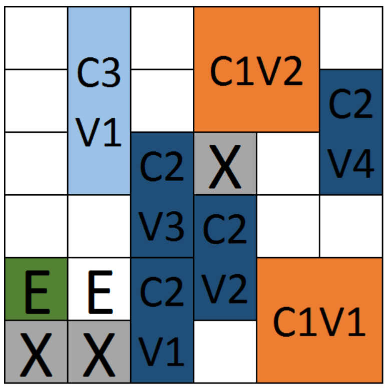


Figure 5.5: Stowage plan of a small fictive deck. C = cargo, V = vehicle, E = exit square, X = unusable space.

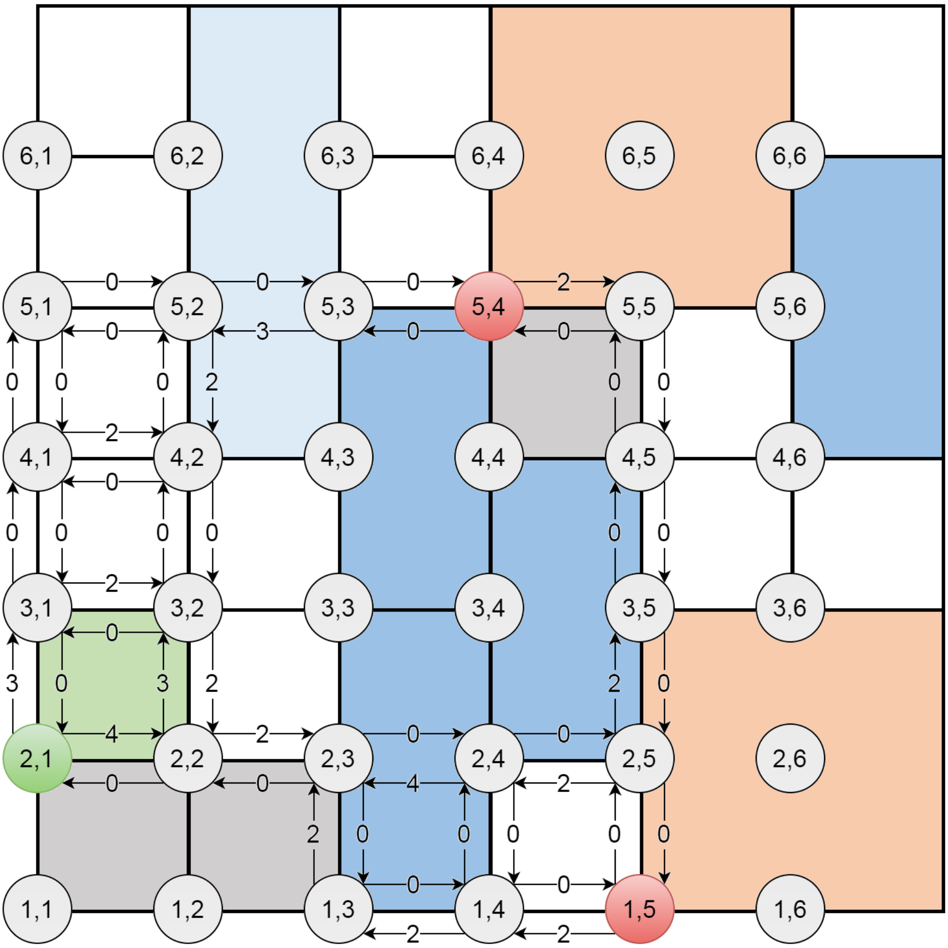


Figure 5.6: Graph for C1, at its loading port. The green node is the start node, and the red nodes are target nodes.

The most important aspects regarding the graph, illustrated by this example are:

- When a node (i, j) is visited, it corresponds to a vehicle placing its lower left corner in square (i, j)
- The vehicles enter the deck in square $(2, 1)$, which sets the start node $= (2, 1)$ marked with green. When in node $(2, 1)$, the vehicles use the squares $(2, 1)$, $(2, 2)$, $(3, 1)$, and $(3, 2)$, as all vehicles in cargo 1 have a size of 2×2.
- From the start square, the vehicles may move one square up or one square

to the right. The cost of moving one square up is 3, as it would induce a shift of the vehicle from C3. The cost of moving one square to the right is 4, as two of the vehicles from C2 are shifted.

- From the start square, it is not possible to move one square down to node $(1,1)$, as the squares $(1,1)$ and $(1,2)$ are unusable.
- The edge cost between $(3,1)$ and $(4,1)$ is zero, as the shifting cost of C3V1 already is accounted for in an earlier move to node $(3,1)$.
- It is not possible to move from node $(5,1)$ to $(6,1)$. If the lower left corner of the vehicle is placed in square $(6,1)$, the vehicle spans over the squares $(6,1)$, $(6,2)$, $(7,1)$, and $(7,2)$, where the two last squares do not exist.
- Node $(3,3)$ can never be visited, as square $(4,4)$ is unusable. A vehicle sized 2×2 would use this square if the lower left corner is placed in square $(3,3)$.
- The unreachable nodes are included in this example for illustrative purposes only, thus not added to the set V when creating a graph G.

5.4.2 Shifting Cost Calculations

In the previous section, two graphs were created for every carried cargo, one for the loading port and one for the unloading port. In this section, the solution procedure is given. When solving for a loading port, the start node is set to one of the entry/exit squares on the deck. Then, the shortest (cheapest) path to every node is found. The shortest path for each vehicle in the cargo is now given by the shortest path between the start node and the vehicle's target node. The target node is the node where a vehicle's lower left corner is to be placed, provided by the stowage plan.

When solving for an unloading port, the exact same procedure is used. Intuitively, it makes more sense to set the start node at a vehicle's location and find the shortest path to the exit square. The drawback of this procedure is that one needs to solve a one-to-one SPP for each vehicle in the unloading cargoes. A better approach is to set the exit square as the start node, and solve a one-to-all SPP for every unloading cargo. This is possible as the graph has the property of providing the same solution regardless of whether the exit square is the starting node and the vehicle's location the target node or vice versa. In other words, the shortest path from the exit to a vehicle's location is always the same as the shortest path from the vehicle's location to the exit, despite the graph being directed. Using the example from Figure 5.6, the optimal path from (2,1) to (1,5) and the optimal path from (1,5) to (2,1) both have a cost of 5. As is seen from the figure, the path from (2,1) to (1,5) has positive edge costs: $c_{2,1,3,1} = 3$ and $c_{5,4,5,5} = 2$. The path from (1,5) to (2,1) has positive edge costs: $c_{2,5,3,5} = 2$ and $c_{5,3,5,2} = 3$. As shown, the paths have positive edge costs between different nodes, but the total cost is always the same. This property enables us to also solve a one-to-all SPP for each cargo at its unloading port, despite the graph being directed.

The graphs created in the previous section are directed, non-negative graphs. Thus, the Dijkstra's algorithm is used to solve the one-to-all SPP for each graph. As this algorithm is widely known, we are only mentioning some key points of our implementation here. The *Dijkstra-NoDec* procedure given by Chen et al. (2007) is used to solve the shortest path problems. The graphs used to solve the problem are sparse, as each node in the graph has a maximum of four outgoing edges and the number of nodes in the graph usually exceeds 10,000 nodes for problems with realistic size. Thus, the *Dijkstra-NoDec* procedure is used as Chen et al. (2007) state that it runs faster than the other proposed procedures on average for sparse graphs. A detailed description of the procedure is given in Chen et al. (2007).

Based on the solutions, we can estimate the shifting cost of a stowage plan. For a given cargo in its loading port, the loading cost of each vehicle is given by the value of a vehicle's corresponding target node. The sum of all loading cost and unloading cost for all vehicles in all carried cargoes gives an estimated shifting cost. However, we can improve this estimate by post-processing the results. Continuing the example from Figure 5.6, the optimal routes for the two loading vehicles are calculated. The loading route for C1V1 from (2,1) to (1,5) costs 5. C1V2 has a loading route cost of 3, from (2,1) to (5,4). These results can be verified visually. A poor estimate of the shifting cost for this loading port is then given by 5+3=8. In order to get a better estimate, we use the fact that each vehicle is only shifted once and backtrack the routes, recording the shifted vehicles. For C1V1, the shifted vehicles are $S_{C1V1} = \{C2V4, C3V1\}$. For C1V2, only one vehicle is shifted, $S_{C1V2} = \{C3V1\}$. All unique shifted vehicles in this port are given by: $S_p^{all} = S_{C1V1} \cup S_{C1V2} = \{C2V4, C3V1\}$. A better shifting cost estimation is given by the cost of shifting all vehicles in S_p^{all}, with the shifting cost of 5.

As there often are several cargoes loaded or unloaded at each port, S_p^{all} contains all shifted vehicles in a port. In Section 5.3.2, some of the presented destroy operators used information from the shifting evaluation procedure, i.e. the route and vehicle shifting cost. The route costs are given directly from the distance to the target nodes. Vehicle shifting costs are derived from S_p^{all} for all ports. If a vehicle is included in the set S_p^{all} for n unique ports, the vehicle's shifting cost is given by n times its area. Thus, by post-processing the results, we get a better estimate on the shifting cost of a stowage plan, in addition to useful information used by the destroy operators.

6. Test Instances

In this chapter, we present a detailed description of the test instances used in the computational study. The instances are generated based on data provided by the case company, but no real voyages are replicated. Thus, for every instance, some of the data are real, while some of the information is generated based on real data. We believe that the resulting instances are good approximations of typical voyages, so that the results apply to real cases.

The created instances are defined by three different aspects: Deck information, cargo list information, and grid resolution. The deck information gives the layout of the deck, the size of the deck, where the entry/exit point is located, and where other unusable spaces are located, such as pillars. The cargo list contains information about the size and weight of each cargo, the number of mandatory and optional cargoes, in which ports the cargoes are to be loaded and unloaded, and revenue for bringing optional cargoes. Last, the grid resolution determines how accurate the size of the deck and the cargoes is in the model compared to their actual size. The instances are generated using a combination of Microsoft Excel and Matlab, and made into readable data for both Xpress, which is the MIP solver used, and the 2DRSSP Heuristic, created in Java.

6.1 Deck Construction

The deck layout is important when considering the stowage problem's complexity, and is the factor that sets the limit on how much cargo a deck can carry during a voyage and how large the different cargoes can be. Based on data provided by the case company, a typical real deck has a length greater than $100m$ and width greater than $20m$. In this computational study, both fictive and real decks are used in the construction of the instances. The characteristics of a deck layout include the outline of the deck, entry/exit points, and spots on the deck that are unusable.

Seven different decks are created to use in various testing scenarios, and they are presented in Table 6.1. Two fictive decks are used when comparing the MIP model and the heuristic, given in Figure 6.1. These decks are a lot smaller than real decks, as the MIP solver does not provide solutions within a reasonable amount of time for realistically sized decks, as is shown in Section 7.2.3. These deck layouts are created based on a scaled outline of a typical realistically sized deck.

For the 2DRSSP Heuristic parameter tuning, two new fictional decks are used, illustrated in Figure 6.2. These decks have a realistic size and the layouts are close to a real deck layout for a RoRo ship. At last, three decks in real size are created to use in the testing of the heuristic. The two decks in Figure 6.3 and Figure 6.5, are created using the actual outlines of real decks, provided by the case company. Deck 6 is created using a real deck outline, but the size of the deck is slightly scaled down, illustrated in Figure 6.4. This is done in order to have a greater variety of decks in the instances.

Table 6.1: Deck characteristics.

Deck #	**Deck type**	**Used in**	**Length** (m)	**Width** (m)	**Actual Area** (m^2)
1	Fictive	MIP/Heuristic	45	20	763
2	Fictive	MIP/Heuristic	20	10	160
3	Fictive	Parameter tuning	100	28	2581
4	Fictive	Parameter tuning	180	32	5396
5	Real	Heuristic	129.6	27.76	2923
6	Real	Heuristic	200	31.5	5634
7	Real	Heuristic	265	31.5	7437

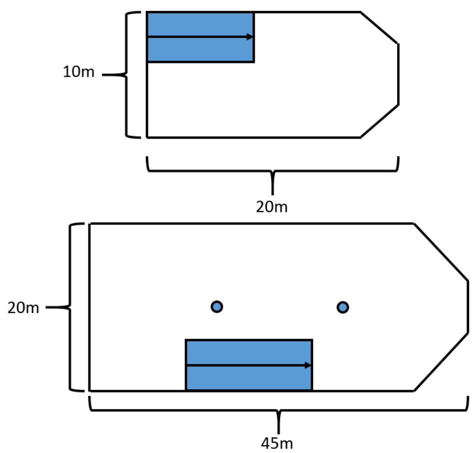


Figure 6.1: Layout of fictive deck 1 (bottom) and fictive deck 2 (top). Used in the MIP instances.

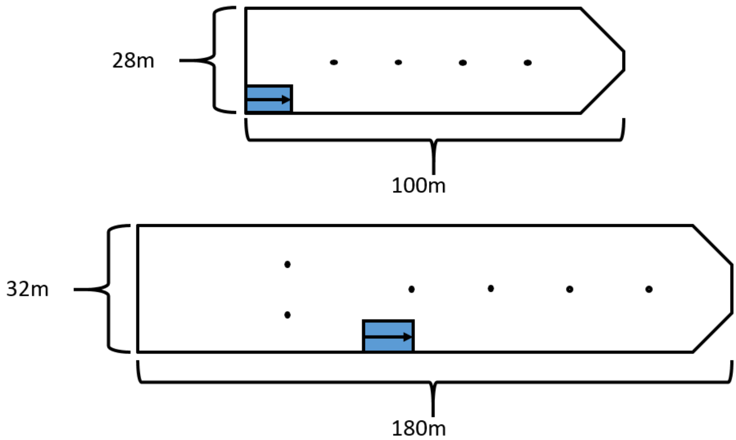


Figure 6.2: Layout of fictive deck 3 (top) and fictive deck 4 (bottom). Used for parameter tuning.

Figure 6.3: Layout of deck 5, 129.6 meters long and 27.76 meters wide.

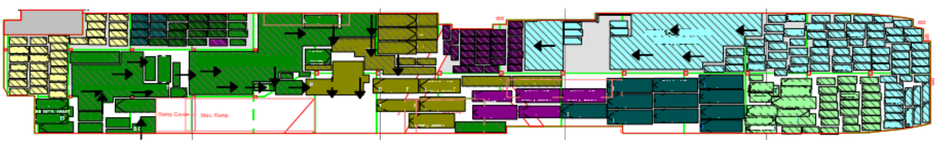


Figure 6.4: Layout of deck 6, 200 meters long and 31.5 meters wide.

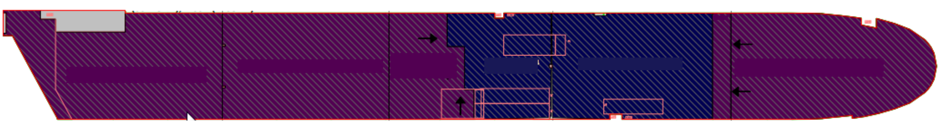


Figure 6.5: Layout of deck 7, 265 meters long and 31.5 meters wide.

6.2 Cargo Lists

The characteristics of a cargo list affect the running time of the solution methods and the complexity of the problem instances. These characteristics include the number of mandatory cargoes and optional cargoes, the number of vehicles in each cargo, and the number of port calls. In order to evaluate these factors, different cargo lists are generated with respect to the different decks. The vehicles included in a cargo list are randomly picked based on data from the case company, but in some cases, fictive vehicles are included. This is done in order to get a higher degree of variation in the cargo lists.

First, the number of mandatory and optional cargoes and the number of vehicles in each cargo are set. The total amount of vehicles in a cargo list is distinguished into three levels: Low, medium, and high filling degree. The *filling degree* is defined by the total approximated area usage of all vehicles in a cargo list, divided by the approximated area of a deck when the highest grid resolution is used. For cargo list 15, and deck 6, the approximated area usage of all vehicles is $6466m^2$, and the approximated area of the deck is $5630m^2$. Thus, the filling degree is given by $6466m^2/5630m^2 = 115\%$, as given in Table 6.2.

Low filling degree implies that the total number of vehicles in a cargo list covers approximately 70% of the deck. For medium, the total area of all the vehicles in the cargo list must exceed the deck's capacity for the lowest grid resolution, but not exceed it when using the highest grid resolution. Approximately 90% of the deck is filled for the medium case. For high filling degrees, the total area of all the vehicles exceeds the deck's capacity for the highest grid resolution.

The number of port calls is fixed for each voyage, and the carried cargoes must be loaded and unloaded at the given ports. The number of loading and unloading ports is derived directly from the cargo lists. Hence, there is no need to explicitly define the voyages. For all optional cargoes, a cargo revenue is set. The revenue of carrying a cargo is set to the total area of the cargo, multiplied with 1000. This linearly dependency is discussed in Section 5.1.

In Table 6.2, an overview of the different cargo lists is given. The cargo lists created are tied to a specific deck. There are two reasons for this. First, a cargo list created for the largest deck may not be feasible for the smaller decks, as the total area of mandatory cargoes in cargo lists for large decks may exceed the capacity of smaller decks. Second, we want to keep the filling degree percentage for each cargo list constant, as this ensures a fair comparison between the decks when analyzing the results.

The filling degree percentage for each cargo list is given in Table 6.2. A score over 100% means the total area of all the vehicles exceeds the deck's area. The cargo lists created for testing the MIP solver's capabilities all have a high degree of filling. This is done in order to ensure that the MIP solver has to evaluate which optional cargoes to carry, which is an important aspect of the stowage model. Similarly for the parameter tuning instances, the cargo lists have a filling degree exceeding 100%. Instances where one has to choose which optional cargoes to carry are considered more difficult to solve, therefore, we tune the parameters on these instances. The parameters set on these strict instances are assumed to work well for low and medium cases as well, as they are expected to be easier to solve.

Table 6.2: Cargo list characteristics.

Cargo list	**Deck**	**# of mand. cargo**	**# of opt. cargo**	**# of load ports**	**# of unload ports**	**Filling degree**	**Area of deck filled (%)**
1	1	3	4	2	2	High	116
2	1	5	5	3	2	High	103
3	2	3	4	2	2	High	124
4	2	3	4	3	3	High	124
5	2	5	3	2	2	High	103
6	3	11	14	4	3	High	117
7	3	7	20	3	3	High	124
8	4	13	14	3	3	High	122
9	4	8	15	2	2	High	121
10	5	10	8	4	4	Low	70
11	5	10	9	2	2	Medium	90
12	5	7	12	3	4	High	109
13	6	8	12	3	3	Low	70
14	6	10	9	3	3	Medium	90
15	6	8	12	3	3	High	115
16	7	8	10	3	3	Low	70
17	7	9	11	4	4	Medium	90
18	7	8	8	3	3	High	109

6.3 Grid Resolution

The grid resolution determines how accurate the actual area of the deck and the vehicles are represented. This decision greatly influences the computational time, as an increase in grid resolution gives an increase in the number of variables in the

MIP models. This means that when using a MIP solver to solve the models, it is not limited by the size of the deck, but the grid resolution. The grid resolution is represented using rows and columns, which forms a grid layout, such as the one in Figure 6.6, which is the grid layout for deck 1. From the figure, one can see that a higher grid resolution means better representation of the deck layout, and also a better representation of the actual size of the vehicles. This means that even though the MIP solver is not limited by a deck's size, a too low grid resolution would, for a large deck, give a bad resolution and provide insufficient results. For example, a 20×20 grid resolution on deck 6 from Table 6.1, would give a square $10m$ long and $1.575m$ wide, which certainly overestimates the vehicle's area usage too much. Therefore, appropriate grid resolutions are used for the different decks, as seen in Table 6.3. This implies that decks with realistic size need a higher grid resolution than for example the fictive decks 1 and 2 from Table 6.1.

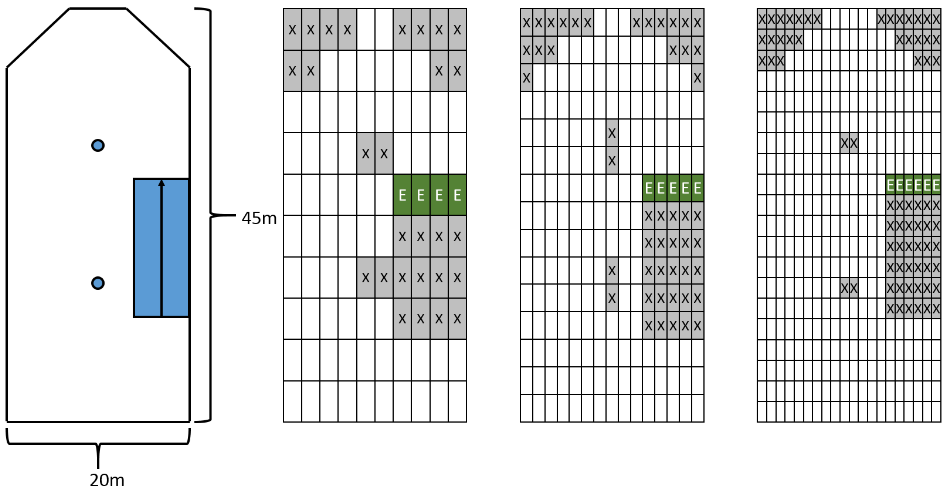


Figure 6.6: Grid layout of deck 1, for grid resolutions 10×10, 15×15, and 20×20.

In order to have a ratio between the number of rows and columns, we use the ratio between length and width in a car equivalent unit (CEU), which often is used to state a RoRo ship's capacity. The CEU is based on the size of a Toyota Corona, which has an approximate length-width ratio of 4:1.5. However, since this master thesis is a continuation on the project thesis by Hansen and Hukkelberg (2015), this ratio was not used on the test instances developed for the MIP model, since they were made during the project thesis. As it is desirable to evaluate the performance of the 2DRSSP Heuristic by comparing it with the MIP model results from Hansen et al. (2016), these same instances are used, having a ratio of 1:1.

Table 6.3: The different grid resolutions and which decks they are used on.

Grid resolution	Deck
10×10	1, 2
15×15	1, 2
20×20	1, 2
50×19	3, 4
100×38	5-7
200×75	3-7
300×113	5-7
500×188	5-7

For each vehicle, the grid resolution decides the number of length and width squares needed, based on the vehicle's length, width and the minimum clearance required between the cars. In this thesis, the minimum clearance is set to $0.15m$. In Table 6.4, the actual area for the cargo lists for deck 5-7 compared to the area when using a grid representation is shown. As can be seen, the area of a vehicle's square usage is always an overestimation of the actual area usage. A deck's area capacity is underestimated, but higher grid resolution gives better representation. If a small enough square size is used, e.g. $0.01m$ x $0.01m$, the representation of the area usage of each vehicle and the deck's area is close to perfect, but this results in a drastic increase in variables. In Appendix A.2, Tables A.4 and A.5, the difference in the area usage for the other cargo lists is shown.

Even though the overestimation most often reduces with higher grid resolution, exceptions can be found. An exception is found for cargo list 2 on deck 1, which can be seen in Table A.4, Appendix A. For test instance $MIP\_i15j15c2d1$, the area usage of the optional cargo is less than for instance $MIP\_i20j20c2d1$, even though the last instance has a higher grid resolution. The reason for this is the size of the squares, given by $3m$ x $1.333m$ (15×15) and $2.25m$ x $1m$ (20×20). Even though the area of a square for 20×20 is lower, there are some situations where the square size from 15×15 is the better choice. For a vehicle with length $2.9m$ and width $1.3m$, the 15×15 grid representation estimates the area usage in the best way. With a grid resolution of 15×15, only one square is needed to place the vehicle. For 20×20, an area of 2×2 squares in needed to place the vehicle, which gives a higher estimated area usage. However, on average, the highest grid resolution will represent the actual area the best. When doubling the rows and columns, i.e from 10×10 to 20×20, the highest resolution is strictly equally good or better. An area usage of 3×3 squares on a 10×10 grid gives a maximum area usage of 6×6 squares on a 20×20 grid, but possibly less.

Table 6.4: Area usage for deck 5-7, cargo list 10-18.

Cargo list	Actual representation Mand. Area (m^2)	Opt. Area (m^2)	Deck Area (m^2)	Grid res.	Grid representation Mand. Area (m^2)	Opt. Area (m^2)	Deck Area (m^2)
10	1226	598	2923	100×38	1631	738	2850
10	1226	598	2923	200×75	1483	708	2895
10	1226	598	2923	300×113	1406	674	2916
10	1226	598	2923	500×188	1372	671	2921
11	1666	702	2923	100×38	2079	987	2850
11	1666	702	2923	200×75	1950	853	2895
11	1666	702	2923	300×113	1890	822	2916
11	1666	702	2923	500×188	1841	789	2921
12	1300	1546	2923	100×38	1786	1990	2850
12	1300	1546	2923	200×75	1591	1846	2895
12	1300	1546	2923	300×113	1505	1748	2916
12	1300	1546	2923	500×188	1468	1724	2921
13	2352	1159	5634	100×38	3382	1522	5527
13	2352	1159	5634	200×75	2875	1414	5590
13	2352	1159	5634	300×113	2763	1356	5615
13	2352	1159	5634	500×188	2635	1311	5630
14	3218	1298	5634	100×38	4307	1893	5527
14	3218	1298	5634	200×75	3843	1603	5590
14	3218	1298	5634	300×113	3591	1532	5615
14	3218	1298	5634	500×188	3613	1458	5630
15	3010	2691	5634	100×38	4226	3657	5527
15	3010	2691	5634	200×75	3753	3360	5590
15	3010	2691	5634	300×113	3464	3250	5615
15	3010	2691	5634	500×188	3409	3057	5630
16	2349	2263	7437	100×38	3251	2922	7269
16	2349	2263	7437	200×75	2790	2705	7315
16	2349	2263	7437	300×113	2765	2625	7363
16	2349	2263	7437	500×188	2639	2534	7432
17	3435	2491	7437	100×75	4694	3394	7269
17	3435	2491	7437	200×75	4077	3022	7315
17	3435	2491	7437	300×113	4030	2875	7360
17	3435	2491	7437	500×188	3864	2814	7432
18	4582	2733	7437	100×38	6146	3469	7269
18	4582	2733	7437	200×75	5503	3201	7315
18	4582	2733	7437	300×113	5312	3017	7363
18	4582	2733	7437	500×188	5079	3019	7432

6.4 Generating the Test Instances

Test instances are generated by combining different deck layouts, grid resolutions, and cargo lists. The cargo lists presented in Table 6.2 are only combined with suitable decks from Table 6.1. Each instance varies with respect to the number of vehicles, number of mandatory and optional cargoes, number of ports, deck type, and grid resolution. For each instance, a data file is generated by the procedure found in Appendix B. The instances created are divided into three main groups, based on their testing purpose. These three groups are (1) MIP model testing and comparing the results with the 2DRSSP Heuristic, (2) parameter tuning, and (3) testing the 2DRSSP Heuristic on realistically sized problems.

The label for the test instances is $NAME\_iX_ijX_jcX_cdX_d$. Here, $NAME$ indicates what main group it belongs to, i.e. either MIP, $Param$, or $Heur$. X_i and X_j are the number of rows and columns which represents the grid resolution, X_c is the cargo list, and X_d is the deck type. A list of all the instances generated can be found in Appendix A.1, Tables A.1-A.3, and for each instance a percentage on how much of the actual area of the deck that can be used is given. As can be seen from the appendix, a total of 56 instances were created. 12 are used to test the MIP models and to compare the MIP solvers results with the 2DRSSP Heuristic. Eight instances are used in the parameter tuning, and 36 to test the heuristic for realistically sized problems.

7. Computational Study

In this chapter, a computational study of the 2DRSSP Stowage Model, Shifting Model and Heuristic is conducted. The goal is to examine different properties of the heuristic and the mathematical models, using the test instances made in Chapter 6. Important features such as how grid resolution affects the area utilization, the performance between the heuristic and the mathematical models and technical analysis of the heuristic are considered.

First, in Section 7.1, the parameters used in the heuristic are set, and the two shifting evaluation methods are compared. Section 7.2 presents the results obtained with the 2DRSSP Stowage Model, and compares the results with the heuristic. In addition, two methods to reduce the shifting cost when using the heuristic are evaluated and compared. This is followed by a technical analysis of the heuristic in Section 7.3. Here the performance of the operators and the computational time for different grid resolutions is evaluated. Last, in Section 7.4 we look at the results from an economic perspective with regards to revenue and the shifting cost.

The mathematical models have been implemented using the modeling language Mosel, and solved with the commercial optimization software Xpress-IVE Version 1.24.06 64 bit. The 2DRSSP Heuristic has been coded in Java, and Eclipse Version 4.5.1 has been the integrated development environment. The test instances were solved on a computer with Intel Core i7-3770 (3.40GHz) CPU and 16 GB RAM, running on Windows 7 Enterprise 64-bit Operating System.

7.1 Parameter Tuning

In this section, we aim to decide the values of the parameters used in the heuristic, both for the GRASP and the ALNS. Additionally, the importance of the parameters is discussed. First, the parameters for the GRASP heuristic are set, followed by the parameter tuning of the ALNS heuristic. The parameter tuning is important in order to ensure satisfying performance by the 2DRSSP Heuristic. The values are set using a combination of suggested values from relevant literature and the results from test instances. The eight instances used in the parameter tuning are given in Appendix A, Table A.2. Finally, the performance of the proposed shifting evaluation procedure is evaluated by a comparison with the shifting model.

7.1.1 Tuning the GRASP Parameters

In this section, the initial values for the GRASP parameters are set. The GRASP has five parameters that need to be tuned, and these are as follows: w_d, w_l, w_r, w_o, and δ. The first four are the weight parameters of the four search guiding parameters: Search direction (d), RCL length (l), random insertion (r), and insertion objective (o), respectively. Each of the search guiding parameters can take a set of different possible values, given in Table 7.1.

Table 7.1: Possible values each of the search guiding parameters can take in the GRASP.

Parameter	**Possible values**
d	{1, 2, 3, 4, 5, 6, 7, 8}
l	{1, 2, 3, 4, .. , 19, 20}
r	{0, 0.05, 0.1, 0.15, .. , 0.95, 1}
o	{1, 2}

The weight parameters w_d, w_l, w_r, and w_o give the probability of selecting each of these possible values for the corresponding search guiding parameter. This is explained using the search direction parameter. Search direction, d, has to be set to one of the values $\{1, 2, 3, 4, 5, 6, 7, 8\}$ in every iteration of the GRASP, defining the search direction. The probabilities of setting d to each of these values are given by the corresponding weight parameter w_d. If $w_d = \{0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125\}$, all search directions have an equal probability of being chosen. However, as the search progresses, the weight parameter w_d is updated based on the obtained results and the weight adjustment parameter δ. After a number of iterations, the weight parameter might be $w_d = \{0.2, 0.2, 0.2, 0.2, 0.05, 0.05, 0.05, 0.05\}$. Now, the probability of choosing search directions 1-4 is higher, because they worked best in the previous iterations. On the other hand, search directions 5-8 have a lower probability of being chosen. In this example, equal initial weights were given. However, by deciding the initial weights based on the parameters performance in the test instances, the GRASP will have a higher chance of succeeding in early stages when applied to practical instances, due to a better start basis. Thus, the remaining of this section aims to decide the initial value of the weight parameters.

A total of 6720 combinations exists when combining the different possible values for each of the search guiding parameters. A combination is denoted $C_n = \{d, l, r, o\}$, where e.g. $C_1 = \{1, 1, 0, 1\}$ refers to a combination using search direction one, RCL length = 1, random insertion = 0, and objective one. In order to decide the initial weighting of the parameters, the GRASP is run 100 times for each of the 6720 combinations, for all the eight test instances. For every combination and instance, the total number of times the GRASP managed to find a feasible solution is recorded. A combination's success rate is given by the number of feasible solutions divided by 100. The following paragraphs present the results for each of the search guiding parameters.

Search direction

In Figure 7.1, the results for all eight search directions are given for each of the eight instances. For search direction one, the average success rate of all combinations where search direction one is used is reported, $C_n = \{1, l, r, o\}$ for all values of l, r, o. As can be seen from the figure, the two best search directions are 1 and 7, which are the search directions starting in the bottom left and bottom right corner, placing vehicles in the stern first, illustrated in Figure 5.1. Which of these two search directions that actually performs the best, seems to depend on the placement of the exit ramp. Despite the actual placement of the ramp, we conclude that the preferred search direction is to start in a corner in the ship's stern, filling the stern first.

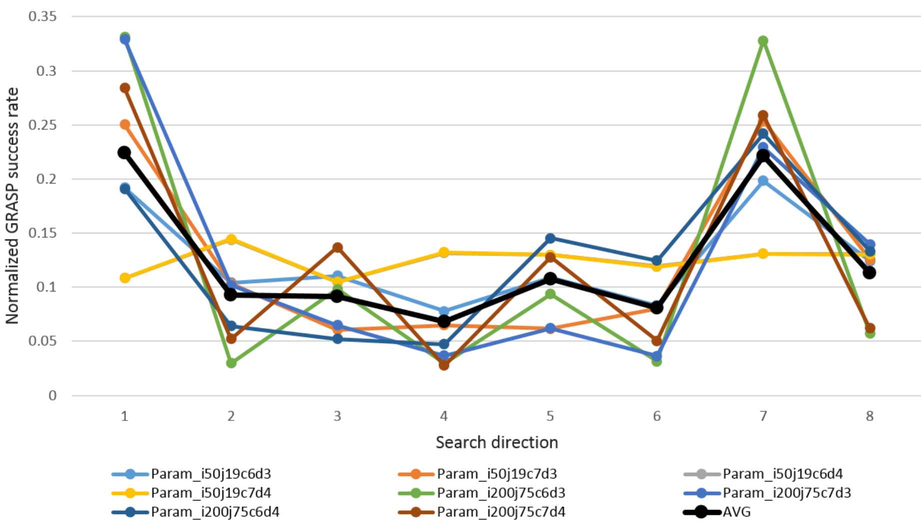


Figure 7.1: The normalized GRASP success rate for each search direction in every instance. The average normalized success rate for all search directions is given by the black line.

Restricted candidate list length

The performance of the RCL length search guiding parameter is given in Figure 7.2. It is evident that increasing the length of the RCL decreases the probability of obtaining a feasible stowage plan. However, it is important to allow some list length in order to explore different possible stowage solutions, as discussed in Section 5.2.1. Thus, an upper limit of five is set on the RCL length, as a higher list length shows poor performance. Possible values of d are now $\{1, 2, 3, 4, 5\}$, reducing the size of the corresponding weight parameter w_l to 5. The initial values of the weight parameter w_l are set accordingly to the values of the average results from the figure.

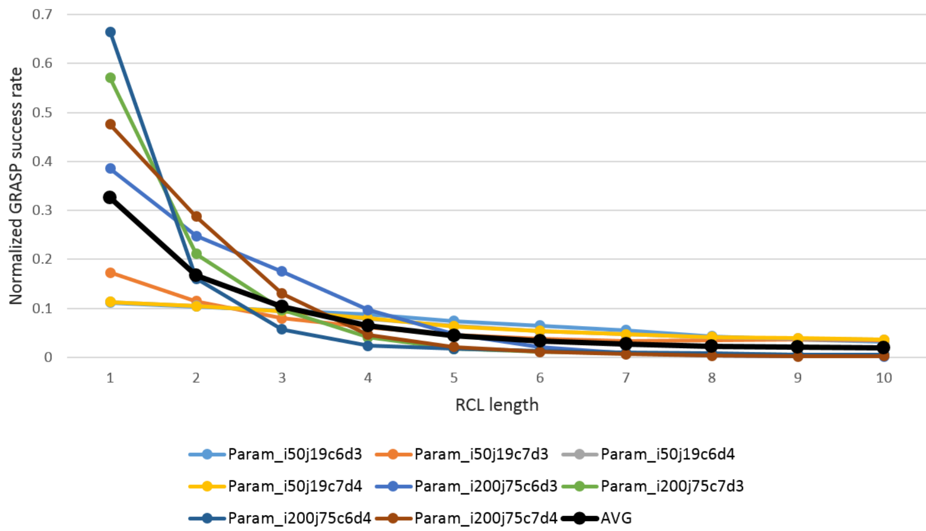


Figure 7.2: The normalized GRASP score for the RCL length from 1 to 10 for every instance. The average normalized success rate for the RCL length from 1 to 10 is given by the black line.

Random insertion

The results for the random insertion search guiding parameter, r, show a similar trend as the RCL length. From Figure 7.3 we see a downward trend as the randomness increases. Naturally, the success rate increases with lower random values, but it is still included in order to diversify the search. We set the maximum value of the random insertion parameter r to 0.5, and the initial weights of the w_r parameter are set based the average results.

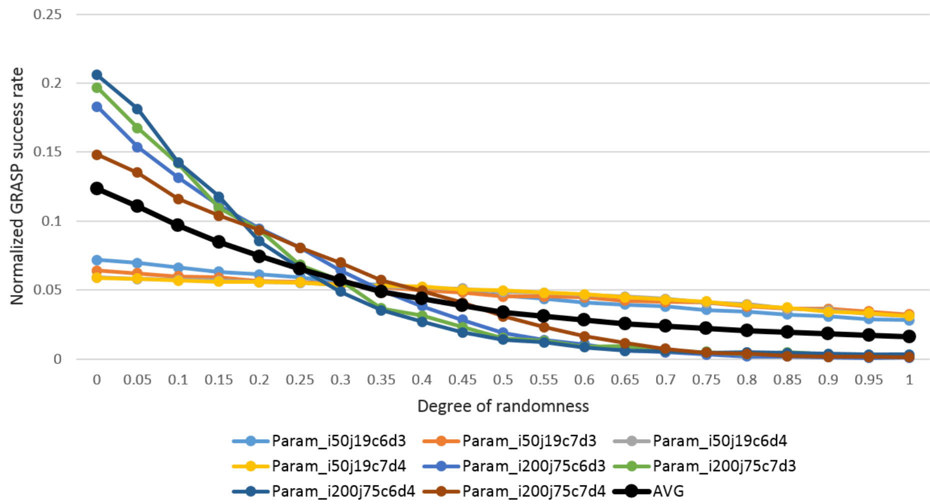


Figure 7.3: The normalized GRASP success rate for every degree of randomness in every instance. The average normalized success rate for every degree of randomness is given by the black line.

Insertion objective

The fourth search guiding parameter is the insertion objective, o, which decides how the insertion of vehicles should be done. Either should the largest vehicles be inserted first, or the vehicles with the least available squares left. The results from the parameter test show no clear difference between the two insertion objectives. Thus, equal initial weights are set to the insertion objective parameter w_o.

Summary

Based on the results, initial weights for w_d, w_l, w_r and w_o are set according to the values in Table 7.2. As can be seen from the table, the interval lengths for both w_l and w_r are updated, as the results from the parameter tuning indicated a very low GRASP success rate for high values of these two parameters. The weight adjustment parameter δ is set to a value of 0.005, based on preliminary testing. Note that fixing the values of the search guiding parameters d, l, r, and o according to the best performing values would limit the capabilities of the GRASP. Based on the results, the best values are given by the combinations $C_1 = \{1, 1, 0, 1\}$ and $C_2 = \{1, 1, 0, 2\}$, which generate the same stowage plan in each iteration, as no randomness is induced. Despite that these combinations provide feasible solutions for the most instances on average, the inclusion of other combinations is necessary in order to provide a feasible stowage plan for the other instances.

Table 7.2: Values of GRASP parameters.

Parameter	**Description**	**Value**
w_d	Initial search direction weights	$\{0.2, 0.1, 0.1, 0.1, 0.1, 0.1, 0.2, 0.1\}$
w_l	Initial RCL length weights	$\{0.46, 0.24, 0.15, 0.09, 0.06\}$
w_r	Initial random insertion weights	$\{0.16, 0.14, 0.12, 0.11, 0.1, 0.09,$ $0.07, 0.06, 0.06, 0.05, 0.04\}$
w_o	Initial objective insertion weights	$\{0.5, 0.5\}$
δ	Weight adjustment	0.005

7.1.2 Tuning the ALNS Parameters

There is a considerable number of parameters connected to the ALNS, which all need to be set. Testing all possible combinations as was done for the GRASP, is not possible within a reasonable time. Thus, the strategy used to find suitable values for these parameters is based on the approach used by Ropke and Pisinger (2006). First, the parameters are assigned values while developing the heuristic using a trial-and-error method. Then, different values of a given parameter are evaluated, while the rest of the parameters are kept fixed. The heuristic is run five times for each of the values that are to be evaluated, for each of the test instances, given in Table A.2. The value with the best average result is chosen as the parameter value and updated continuously. A new parameter is then tested, repeating the same process until all parameters have been tuned.

The following parameters belong to the ALNS and need to be set: $\xi, \sigma_1, \sigma_2, \sigma_3, \eta, p, g, M, w^{dest}, w^{rep}, t^{rep}, \gamma, c$. The score parameters σ_1, σ_2, σ_3, the reaction factor η and the cooling factor c are set according to the values used in Ropke and Pisinger (2006) and Gharehgozli et al. (2014), as they are the common choice in many papers and seem to be a suitable fit with the 2DRSSP. The initial destroy and repair operator weights w^{dest} and w^{rep} are all set to 1. Preliminary testing has shown that the weight update procedure is capable of adjusting these weights according to the performance of the operators. In addition, the success rate of each operator vary with the instance at hand, and all operators are therefore given equal weighting.

The maximum time, t^{rep}, the repair operators are given to create a feasible stowage plan from the destroyed solution is set to two seconds. As the Grouping Repair operator is unable to provide a feasible stowage plan given unlimited time in some situations, it is necessary to limit the search time. If a feasible solution is not provided within two seconds, a new iteration will start immediately, and a new destroy and repair procedure begins. Thus, as preliminary testing has shown that two seconds often is enough to provide a feasible stowage plan if possible, t^{rep} is set to two seconds. Five parameters remain to be set, ξ, p, g, M, and γ. These parameters are assumed to have a higher impact on the performance of the ALNS and are therefore set based on test results.

Degree of destruction

First, the destroy percentage parameter, ξ, is evaluated, while the rest of the parameters are kept fixed. Ropke and Pisinger (2006) point out that both a too low and a too high value is undesirable. However, from the average normalized score presented in Table 7.3, the best value seems to be $\xi = 1$. Although this may seem contradictional, an important difference is that in our implementation the destroy percentage is an upper limit to how much of the solution can be destroyed in each iteration. In Ropke and Pisinger (2006), the destroy percentage is a fixed value which stays the same for each iteration. From the results, it seems like the option to destroy a large part of the solution is preferable for the 2DRSSP, even though only 50% is destroyed on average.

Table 7.3: Scores for different values for the destory percentage parameter ξ. Lower score is better.

Value	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
Avg. normalized score (%)	11.41	11.82	9.89	8.80	9.71	9.54	11.58	10.20	9.92	7.13

Evaluation frequency

The grouping parameter, g, and the placement parameter, p, are tuned together as they serve the same purpose. The values influence how often a repaired stowage plan is evaluated. In Table 7.4, the average normalized score for different values to the two parameters is presented. The parameters can take discrete values from 0 to 0.5, with a step size of 0.05. From the results in the table, there is no clear indication on which value that gives the best performance, but the value 0.3 has a slightly better score than the other values, and g and p is set to 0.3.

Table 7.4: Scores for different values for the grouping and placement parameter g and p. Lower score is better.

Value	0	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
Avg. normalized score (%)	14.12	7.89	8.73	10.47	9.12	7.78	7.61	8.70	8.29	8.31	8.97

Segment size

In Table 7.5 the average normalized score from the parameter tuning of the segment size, M, is shown. The size of a segment sets the number of iterations between each time the destroy and repair operator weights are updated. Five different values have been tested, and the results show no clear indication on which value performed the best. The segment size 150 has a slightly better score than the other results, so M is set to that value.

Table 7.5: Scores for different values for the segment size M. Lower score is better.

Value	25	50	100	150	200
Avg. normalized score (%)	21.32	23.77	18.44	17.17	19.30

Start temperature

Finally, the start temperature control parameter, γ, is set, used to set the start temperature. The start temperature control parameter γ is tested for different values ranging from 0 to 2.5%, and the average normalized score is given in Table 7.6. As the results show, the value 0.5% gives the best result, so γ is set to this value. An important note is that if γ is set to 0, only better solutions are accepted, which means that the acceptance criterion from simulated annealing is unused. The results from Table 7.6 clearly show that the ALNS performs significantly worse when only accepting better solutions. This shows the importance of including an acceptance criterion, as the ALNS seems to be able to get stuck in a local optimum.

Table 7.6: Scores for different values for the start temperature control parameter γ. Lower score is better.

Value	0	0.005	0.01	0.015	0.02	0.025
Avg. normalized score (%)	21.04	12.42	17.74	17.39	15.86	15.55

Summary

A summary of the different values used in the ALNS when running the test instances is shown in Table 7.7. As is seen from the table, only four out of the five repair operators are given a weight. The MIP Repair operator is not included in the computational study. Preliminary testing has shown that even for a very low destruction degree, Xpress is unable to provide feasible solutions on practical problem instances within a reasonable amount of time. However, if an improved mathematical model is developed, the inclusion of this operator should be reconsidered, as the operator is expected to outperform all of the other repair operators.

Table 7.7: Values of ALNS parameters.

Parameter	**Description**	**Value**
ξ	Maximum destruction degree	1
σ_1	Score when new global best solution is found	33
σ_2	Score when new solution is better than the current solution	13
σ_3	Score when new solution is accepted as a part of simulated annealing	9
η	The reaction factor used in calculating operator weights	0.1
p	Placement evaluation factor	0.3
g	Grouping evaluation factor	0.3
M	The size of a segment, in number of iterations	150
w^{dest}	Initial destroy operator weights	{1, 1, 1, 1, 1, 1}
w^{rep}	Initial repair operator weights	1, 1, 1, 1
t^{rep}	Maximum run time of a repair operator, in seconds	2
γ	Start temperature control parameter	0.005
c	The cooling factor in simulated annealing	0.99975

7.1.3 Comparing the Shifting Evaluation Methods

Hansen and Hukkelberg (2015) conclude that solving the shifting evaluation model presented in Section 4.4 with Xpress, does not provide feasible solutions within reasonable time for large instances. Thus, in Section 5.4, we proposed a shifting cost estimation procedure solving multiple SPP for each stowage plan, referred to as the SPP approach. The SPP approach is capable of solving large instances within reasonable time, but we need to evaluate the quality of the shifting cost estimation. Thus, in this section, we aim to determine the quality of the SPP approach by comparing the shifting cost estimations from the SPP approach with the optimal solution from the shifting evaluation model.

Here, the quality of the SPP approach is defined by its ability to evaluate the quality of different stowage plans in the same way as the exact shifting evaluation model. This means that we are interested to see if the SPP approach follows the same trend as the shifting evaluation model, with respect to the shifting cost evaluated. That the SPP approach provide the exact same results as the model is not as important. However, if the SPP approach shows close to the same results as obtained by the shifting evaluation model, this would strengthen the quality of the approach. The MIP-instances from Table A.1 are used when comparing the methods, which are solvable within a reasonable amount of time by Xpress. The stowage plans used in this comparison are the resulting 2DRSSP Stowage Model version SP's solutions of the instances.

Figure 7.4 presents the results from the shifting evaluation method comparison. Here, the best bound and best solution are reported, in addition to the estimated shifting cost from the SPP approach, for each instance. A maximum run time of 7200 seconds is set. The results show a 0.96 positive correlation between the SPP approach and the best solution from the shifting evaluation model. For 61.5% of the instances, the methods report the same shifting costs. In a few instances, the shifting costs obtained are, in fact, better with the SPP approach than the shifting evaluation model. This may occur if an optimal solution is not found within the time limit. However, the optimal shifting cost from the shifting evaluation model would at least be as good as the estimated shifting cost from the SPP approach. As the SPP approach gives strictly equal or worse than the best bound provided, the results are consistent.

The results indicate that the SPP approach provides a satisfying upper bound on the shifting cost for the instances tested, as they follow the same trend as the optimal solution. In addition, the shifting costs estimated by the two methods are almost identical for all instances. Based on these results, we believe that the upper bound on the shifting cost provided by the SPP approach also gives a reasonable estimation for realistically sized problems as well. Thus, in the reminding part of the computational study, the SPP approach is used to evaluate the stowage plans, unless otherwise stated.

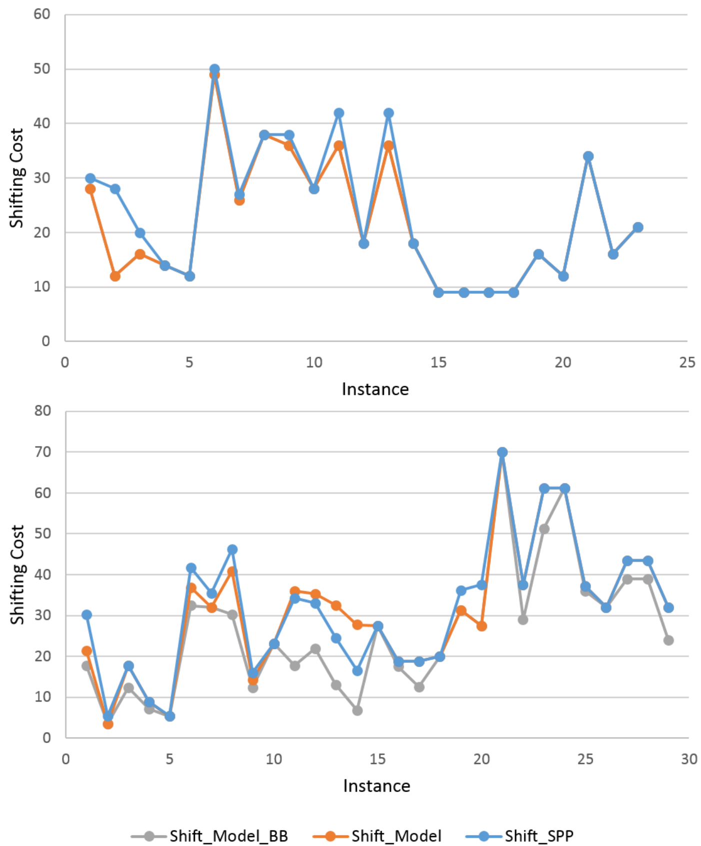


Figure 7.4: Comparison of the shifting cost between the shifting evaluation model and the SPP approach. Instances with grid resolution 10×10 is given on top, and 15×15-20×20 on the graph at bottom. *Shift_Model_BB* gives the best bound for the MIP model, *Shift_Model* is the shifting cost found by the MIP model, and *Shift_SPP* is the shifting cost obtained by the SPP approach.

7.2 Evaluating the Stowage Methods

Different methods to solve the 2DRSSP are presented in this section. First, the five model versions of the 2DRSSP Stowage Model are tested using Xpress, and the results are analyzed and discussed. Then the 2DRSSP Heuristic is tested, where two different solution methods for use in phase two of the heuristic are compared and analyzed. At last, we compare the performance of the stowage model and the heuristic.

7.2.1 2DRSSP Stowage Model Results

In this section, the performance of the five model versions of the 2DRSSP Stowage Model is evaluated. We seek to determine whether the concepts grouping and placement reduce the shifting cost of the stowage plans when considered. The MIP solver's capabilities with respect to problem sizes are then evaluated. The presented results are based on the findings of Hansen et al. (2016).

Each of the 12 instances from Table A.1 is solved using each of the five versions of the MIP model: N, P, H, S, and SP, given in Table 4.1. N is the basic model, only maximizing the revenue generated from optional cargoes. Model version P additionally influences the vehicles placement using the square costs. Model version H enforces a weak form of compactness to each cargo, placing the vehicles together. Model version S rewards grouping of vehicles. Finally, model version SP penalizes placement and rewards grouping.

A maximum run time of 7200 seconds is set. If optimality is not proven within the time limit, the best solution is reported together with the absolute gap. If the absolute gap between the best bound and the best solution is less than 0.01% the search is terminated. The value D, which increase the objective value if vehicles from the same cargo are grouped together, is set to 0.001, and the square costs, C^S_{ij}, are set to one thousandth of the minimum number of squares to reach an exit for each square (i, j), see Section 4.2. Table 7.8 shows the average results for all instances obtained within the time limit.

The revenue of bringing an extra cargo always exceeds the cost of where to place or/and group the vehicles since the extra terms in the objective functions (4.2)-(4.4) have a minor contribution to the objective value. The model versions N, P, S, and SP would generate the same optional revenue in their optimal solutions, but the vehicles' placement could differ. The H-version is a bit different, as constraints (4.13)-(4.14) reduce the solution space. This model could give an optimal solution which generates less revenue than the optimal solution for the other four

Table 7.8: Average results for all test instances for the different model versions.

Model version	**Gap (%)**	**Time (s)**	**Optional cargo #**	**Revenue optional**	**Area used (%)**	**Shifting cost**
N	59.72	3600	1.33	17.67	78.53	2.95
P	24.03	3075	1.58	20.33	81.23	2.55
H[1]	35.34	3212	1.42	18.08	65.32	4.85
S	17.38	3432	1.58	20.50	81.75	2.65
SP	0.01	1778	1.58	20.50	81.75	2.44

[1]Two instances did not provide a feasible solution

models, or even give infeasible solutions, as is the result for two of the instances. The infeasible solutions may indicate that constraints (4.13)-(4.14) are too strict, excluding possible good stowage solutions.

Without evaluating the shifting cost of the solutions, there are some interesting findings regarding the performance of the different model versions. Model versions S and SP provide the best average revenue generated within the time limit. The stowage plans are not necessarily identical, but they do at least carry the same set of optional cargoes for every instance. The average gap for model version SP is 0.01, which implies that the optimal set of optional cargoes is carried for every instance. Based on the average gap and the solution time, we conclude that version SP performs best on the given test instances.

For each model version and instance, the shifting costs for the resulting stowage plans are calculated using the 2DRSSP Shifting Evaluation Model. We here present the average of the weighted shifting cost for each instance, i.e. the shifting cost divided by the grid resolution, ensuring a fair comparison. When evaluating the result of the shifting cost evaluation, it is important to consider the number of optional cargoes carried. As the revenues generated using the different model versions vary, the number of vehicles on the deck differs. With more vehicles on the deck, the number of shifts is expected to be higher. The computational results from the stowage model show that the SP version of the model achieved the highest optional revenue on average. This implies that the resulting stowage plans from model version SP carry the most vehicles. Despite this, the stowage plans from model version SP actually give the best results with regards to the average shifting cost. On average, model version SP gives the stowage solutions with the lowest shifting cost, lowest computational time, and carries the most optional cargoes.

From the results, it is reasonable to conclude that both grouping and placement modifications are preferable to incorporate in a RoRo stowage model. The maximum grid resolution used in this testing is 20×20, which is tiny compared to the required grid resolution for practical instances. The run times reported in Table 7.8 give the average run time for instances with grid resolutions from 10×10 to 20×20. As the average run times for these instances are close to an hour, it is evident that for realistically sized grid resolutions, a heuristic approach is needed in order to obtain feasible solutions.

7.2.2 Comparing Two Heuristic Methods

The 2DRSSP Heuristic solves the problem in two phases, by initially creating a feasible stowage plan using a GRASP and improving the stowage plan with respect to shifting cost by an ALNS in phase two, as explained in Chapter 5. Another possible solution method is to replace the ALNS with the GRASP in phase two. Instead of using ALNS to improve the initial solution, destroying the whole stowage plan and rebuilding it from scratch using the GRASP is a valid and possibly better solution strategy. The solution methods are referred to as ALNS and GRASP, respectively. In this section, we aim to conclude which of these solution methods that provides the better stowage plans.

The realistically sized instances in Table A.3 are used to compare the two solution methods. For both methods, the first phase procedure is identical, which uses GRASP to provide a feasible initial stowage plan. The initial stowage plans found in this phase are then used in both methods, ensuring a fair starting point. For the GRASP method, the new stowage plan is accepted if it has a lower shifting cost than the best known so far. For the ALNS, the parameters set in Section 7.1.2 are used.

Each instance is solved three times, and the average shifting cost improvement from the initial shifting cost is reported together with the best shifting cost improvement. A maximum run time of 7200 seconds is set. If a stowage plan with zero shifting cost is found within the time limit, the search is stopped as an optimal stowage plan is found. For the first phase, a maximum run time of 900 seconds is set, which has shown to be sufficient to evaluate the optional sets. The time is split equally between each optional set evaluated by the GRASP, using the binary search procedure. In the second phase, the run time is given by 7200 seconds minus the time used in phase one, which may be 900 seconds or less.

The results, given in Table 7.9, show that both methods have no problem solving instances of realistic size, in contrary to the MIP solver. However, there is a great difference in the shifting cost improvement from the initial stowage plan for the two methods. The results show that the ALNS improves the shifting cost the most on average, for all instances. The best stowage plan generated by the ALNS is also

Table 7.9: Shifting cost improvement in percentage of the initial shifting cost for ALNS and GRASP. A score of 100% indicates a shifting cost of zero is reached.

	ALNS		GRASP	
Instance	**Avg. (%)**	**Best (%)**	**Avg. (%)**	**Best (%)**
$Heur\_i100j38c10d5$	87.90	94.07	26.84	27.67
$Heur\_i200j75c10d5$	90.63	97.06	29.48	33.25
$Heur\_i300j113c10d5$	89.96	93.76	48.90	49.71
$Heur\_i500j188c10d5$	80.92	84.40	38.92	43.19
$Heur\_i100j38c13d6$	99.11	99.70	32.38	35.21
$Heur\_i200j75c13d6$	99.78	100	36.78	38.32
$Heur\_i300j113c13d6$	100	100	38.56	46.93
$Heur\_i500j188c13d6$	99.81	100	17.50	18.70
$Heur\_i100j38c16d7$	95.99	100	54.98	87.98
$Heur\_i200j75c16d7$	100	100	44.25	45.59
$Heur\_i300j113c16d7$	97.90	100	47.90	48.89
$Heur\_i500j188c16d7$	99.52	100	52.17	56.56
$Heur\_i100j38c11d5$	53.55	59.16	17.85	20.02
$Heur\_i200j75c11d5$	36.46	42.86	19.45	27.10
$Heur\_i300j113c11d5$	36.15	47.50	19.18	47.50
$Heur\_i500j188c11d5$	43.33	52.34	37.31	43.87
$Heur\_i100j38c14d6$	69.67	75.60	28.98	32.02
$Heur\_i200j75c14d6$	43.06	50.35	34.23	40.20
$Heur\_i300j113c14d6$	52.66	57.13	41.63	43.92
$Heur\_i500j188c14d6$	42.30	52.28	25.11	31.24
$Heur\_i100j38c17d7$	70.09	75.18	54.79	61.35
$Heur\_i200j75c17d7$	43.93	49.93	20.95	37.18
$Heur\_i300j113c17d7$	30.75	35.37	6.25	8.35
$Heur\_i500j188c17d7$	32.21	36.38	21.89	28.01
$Heur\_i100j38c12d5$	71.28	75.49	44.93	47.46
$Heur\_i200j75c12d5$	33.33	48.92	24.02	38.03
$Heur\_i300j113c12d5$	40.94	56.64	30.10	38.85
$Heur\_i500j188c12d5$	39.74	44.58	28.76	39.28
$Heur\_i100j38c15d6$	68.98	71.86	53.92	60.50
$Heur\_i200j75c15d6$	44.15	60.90	14.70	34.37
$Heur\_i300j113c15d6$	48.82	51.42	16.14	29.78
$Heur\_i500j188c15d6$	20.66	29.09	2.02	3.66
$Heur\_i100j38c18d7$	58.77	74.74	23.14	37.13
$Heur\_i200j75c18d7$	56.60	74.12	31.58	48.34
$Heur\_i300j113c18d7$	18.96	21.24	0	0
$Heur\_i500j188c18d7$	41.67	58.49	37.02	43.34
Avg. score	62.21	68.63	30.63	38.15

strictly better than the GRASP's best stowage plan for all instances. An example of the comparison between how the shifting cost changes during one run between the ALNS and the GRASP is given in Figure 7.5. We use this figure to identify the reason for the superior performance of the ALNS.

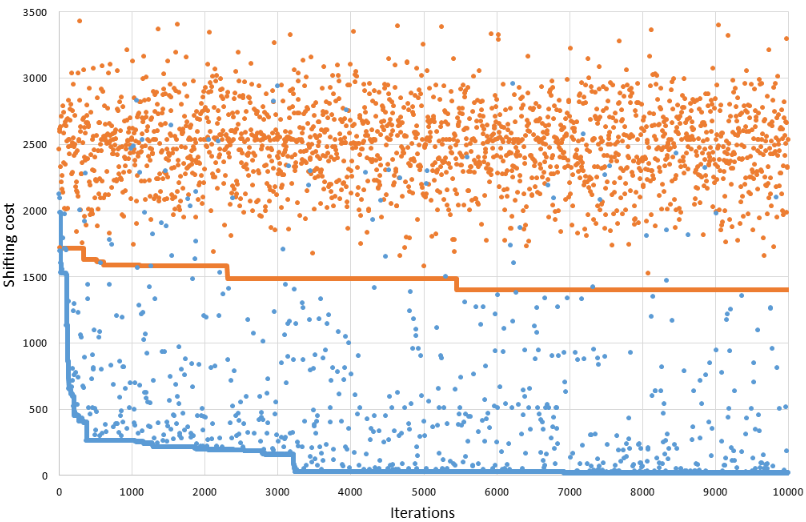


Figure 7.5: Comparing the shifting cost improvement for test instance $Her\_i100j38c13d6$. Blue represents the improvement provided by ALNS. Orange is the improvement by GRASP. For each of the two methods the value for every fifth iteration is plotted.

As seen in Figure 7.5, the ALNS improves the stowage plan from an initial shifting cost of 1988 to 18. As the ALNS uses knowledge obtained from previous iterations to build better stowage plans, it quickly improves the solution. In the beginning of the search, the ALNS often finds improved solutions because a larger number of stowage plans with a shifting cost lower than the shifting cost of the initial stowage plan exist. As the search progresses, fewer stowage plans give better solutions than the current best solution, and it becomes harder to find new and improved solutions. GRASP, on the other hand, struggles to improve the solution, and it only reduces the shifting cost from 1988 to 1384. The reason for this is that each iteration is independent of each other. As the GRASP destroys the whole stowage plan in each iteration, no knowledge from the previous iterations is used. The ALNS, however, uses different destroy operators to remove parts of the solution that contribute the most to the shifting cost, and then use repair operators to build a new stowage plan. In addition to using problem-specific information, the ALNS always starts by destroying the current solution. The probability of finding a new global best solution is assumed to be higher if the stowage plan, which is destroyed and repaired, is an already improved solution. Thus, we discard the GRASP method, and the ALNS procedure is used in the second phase of the 2DRSSP Heuristic in the remaining of this computational study.

7.2.3 Comparing the Stowage Model and the Heuristic

In this section, we compare the 2DRSSP Stowage Model and the 2DRSSP Heuristic, in order to evaluate the performance of the heuristic. First, we seek to determine whether the heuristic is able to provide stowage plans, carrying the same optimal sets of cargoes as the solutions found by the stowage model. The MIP-instances from Table A.1 are used when comparing the methods, which are solvable within a reasonable time by Xpress. Based on the findings in Section 7.2.1, the SP model version of the stowage model is used when comparing with the heuristic, as the SP model is able to find the optimal set of cargoes to carry in all of the instances. For the 2DRSSP Heuristic, every instance is run ten times, and a maximum running time of 120 seconds is set. The average results from the ten runs are presented in Table 7.10.

Table 7.10: Optimal solutions from the 2DRSSP Stowage Model version SP. The average result from the 2DRSSP Heuristic over 10 runs.

	Model version SP		**2DRSSP Heuristic**	
Instance	**Revenue**	**Time (s)**	**Revenue**	**Time (s)**
$MIP\_i10j10c1d1$	10	2.4	10	< 0.1
$MIP\_i10j10c2d1$	0	6.6	0	< 0.1
$MIP\_i10j10c3d2$	0	1.3	0	< 0.1
$MIP\_i10j10c4d2$	0	1.0	0	< 0.1
$MIP\_i10j10c5d2$	15	3.2	10	< 0.1
$MIP\_i15j15c1d1$	14	56.9	13	< 0.1
$MIP\_i15j15c2d1$	42	985.3	42	< 0.1
$MIP\_i20j20c1d1$	36	7199.7	36	< 0.1
$MIP\_i20j20c2d1$	54	7214.9	54	< 0.1
$MIP\_i20j20c3d2$	24	672.3	22	< 0.1
$MIP\_i20j20c4d2$	24	4326.3	22	< 0.1
$MIP\_i20j20c5d2$	27	869.9	25.8[1]	< 0.1

[1]4 out of 10 runs found a revenue of 27. The remaining 6 reported a revenue of 25.

The most important finding from these results is that for eight of the instances, the 2DRSSP Heuristic is able to create stowage plans generating the same revenue as the optimal solutions from the stowage model, in a fraction of the time. In the remaining four of the instances, a feasible solution generating less revenue is found, but the gaps are promising. Even though the results presented in Table 7.10 accurately describe the performance of the 2DRSSP Heuristic, they are a bit unfair. The binary search algorithm used in the heuristic, described in Section 5.1, is recommended to be used when there is a linear dependency between the area of a cargo and the revenue of carrying it. However, the MIP-instances used here do not necessarily have this trait, as they are obtained from Hansen and Hukkelberg (2015). The instances where less revenue is generated by the 2DRSSP Heuristic, actually carry cargoes with equal or higher area usage despite generating less revenue. It can be shown that using another search strategy would result in stowage plans carrying the same set of optional cargoes for both the stowage model and the heuristic, at the cost of higher computational time. However, since all other test instances used in this computational study have this dependency between area and revenue, we do not test the heuristics performance with other search algorithms. This result emphasizes the importance of choosing an appropriate search algorithm, when selecting which optional cargoes to carry.

For the instances where the same set of optional cargoes are carried, the shifting cost from each method is compared. Table 7.11 gives the results from the shifting cost comparison of the methods. Here, the shifting cost is calculated for all stowage plans using the SPP approach presented in Section 5.4. Comparing the shifting cost from the stowage model solutions with the best shifting costs obtained with the heuristic, the heuristic outperforms the stowage model. The heuristic creates stowage plans with lower shifting cost in all instances. Despite the fact that the stowage model is solved exactly, the results indicate that implicitly accounting for the shifting cost when determining a stowage plan, does not give an optimal stowage plan. The optimal stowage plan given by the model is only optimal for the given objective function. As this objective function does not explicit evaluate the shifting cost, the optimal stowage plan with respect to both revenue and shifting cost cannot be proven by this model.

Based on the presented findings, we conclude that the 2DRSSP Heuristic finds near optimal solutions within acceptable time on small instances. It cannot be expected that the optimal set of optional cargoes is carried in every solution, but the average gap is promising. In addition, the 2DRSSP Heuristic seems to create stowage plans with better shifting cost than the exact method. This shows that the placement and grouping concepts in the 2DRSSP Stowage Model alone are not enough to ensure optimal stowage plans.

Table 7.11: Shifting cost for the optimal stowage plans provided by the two stowage methods, for test instances in Table A.1

	Model version SP		2DRSSP Heuristic				
Instance	**SC**	**Time (s)**	**TF**	**TE**	**TO**	**Avg. SC**	**Best SC**
$MIP\_i10j10c1d1$	12	2.4	< 0.1	< 0.1	57.9	0.8	0
$MIP\_i10j10c2d1$	28	6.6	< 0.1	< 0.1	30.3	0.4	0
$MIP\_i10j10c3d2$	18	1.3	< 0.1	0.4	120	9.0	9
$MIP\_i10j10c4d2$	9	1.0	< 0.1	< 0.1	1.6	0	0
$MIP\_i15j15c2d1$	52	985.3	< 0.1	1.0	86.2	9.2	0
$MIP\_i20j20c1d1$	66	7199.7	< 0.1	2.4	93.2	12.4	0
$MIP\_i20j20c2d1$	110	7214.9	< 0.1	0.8	95.9	9.1	0
$MIP\_i20j20c5d2$	128	869.9	< 0.1	4.1	120	97.3	80

SC = Shifting cost. TF = Time to first solution with equal revenue.

TE = Time used to find a stowage plan with equally good shifting cost.

TO = Time elapsed until a stowage plan with zero SC is found.

7.3 Testing the 2DRSSP Heuristic

As seen in Section 7.2.2, the 2DRSSP Heuristic had no problem solving realistically sized problems. To see how the heuristic manages to solve such problems, a technical analysis of the heuristic is provided in this section, using the result in Section 7.2.2. First, the performance of the destroy and repair operators is evaluated. Then, the scalability of the heuristic is commented upon, and the computational performance is tested.

7.3.1 Operators

In the ALNS phase of the 2DRSSP Heuristic, a set of destroy and repair operators is used to find improved stowage plan solutions. The operators were presented in detail in Section 5.3.1, and consist of six destroy operators, i.e. Random, Shift, Port, Area, Grouping and Route Removal, and four repair operators, i.e. GRASP, Grouping, Placement and Random Repair. As we have developed the majority of these operators specifically to use with the 2DRSSP, the performance of each operator is of great interest. The performance of the operators is given by how many times each of the destroy and repair operators are used when we get an improved stowage plan. First, the destroy operators are tested, followed by the repair operators. Finally, the weight adjustment is commented upon.

The results from the testing are given in Tables 7.12 and 7.13. Here, the average score represents how often they were used to provide a better stowage plan, compared to the other operators. The tables also show the average score for each operator for cargo lists with respectively low, medium and high degree of filling of the deck. The instances from Table A.3 are used to evaluate the performance. It should be noted that the scores do not consider how much the operators manage to reduce the shifting cost in one single iteration. This is because it is easier to reduce the shifting cost with a large amount early in the iteration process than later on, as seen in Figure 7.5. As the shifting cost gets closer to its optimal value, fewer stowage plans providing a better shifting cost than the current best exist. However, it is still important to have operators that manage to find these stowage plans, thus, the scores are set based on the number of improving stowage plans found.

Destroy operators

In table 7.12 an indication on the operators' performance is given. The overall score, which does not differentiate between the filling degrees of the deck, shows that it is the Area Removal operator that gives the best score. The operator is followed by the Port Removal and the Shift Removal operators, respectively. As expected, the Random Removal has a poor performance, but the Grouping Removal has an even worse performance, despite using problem specific knowledge. A reason for this may be that removing all the vehicles with no neighbors on a deck can give a stowage plan with many holes. Repairing this could be hard to accomplish, especially if the remaining area of the deck is occupied with vehicles. The possible insertion placements are then greatly limited, making it harder to create a feasible stowage plan with an improved shifting cost. We see this clearly when looking at the performance of the destroy operators for different filling degrees of the deck. The Grouping Removal works better for decks with a low degree of filling. As there are more empty spaces on the deck, it is easier for the repair operator to insert the removed vehicles back onto the deck.

Table 7.12: The average score between the destroy operators for how often they were used to provide a lower shifting cost than the current shifting cost. The overall score is shown together with the scores for cargo lists with low, medium and high filling degree.

	Destroy operators (%)					
Filling degree	**Random**	**Shift**	**Port**	**Area**	**Grouping**	**Route**
Low	5.28	27.59	21.57	26.53	7.08	11.95
Medium	8.20	5.26	24.18	54.87	0.07	7.42
High	5.67	4.19	34.77	48.68	0.39	6.29
Overall	6.38	12.35	26.84	43.36	2.51	8.55

For decks with a medium and high degree of filling, the same trend in the destroy operators' performance is shown. Both have the Area Removal as the top performing operator, followed by Port removal. The reason for this similarity is most likely due to the filling degree of the deck, which is high for both of them in most of the cases. The area of the total amount of cargoes in cargo lists with medium filling degree surpasses the deck's area for lower grid resolutions, meaning it would actually have a high filling degree on decks when low grid resolutions are used, see Table 6.4.

On the other hand, the results for a low degree of filling have a somewhat different distribution of the performance of the destroy operators. Here it is actually the Shift Removal that has the highest performance. With fewer vehicles on the deck, it exists more available area for the repair operator to insert the vehicles that are removed by the destroy operator.

Repair operators

From Table 7.13, it is clear that the GRASP Repair operator works best, followed by respectively the Grouping, Placement and Random Repair operators. As expected, the Random Repair had the worst overall performance for all the different filling degrees. The result on the GRASP's performance shows a score of 46.58%, which means that close to every second time the shifting cost is improved, the GRASP Repair operator is used. For the medium and high filling degrees, we have the same trend in the performance of the repair operators, with the GRASP as the best repair operator. This is likely because it builds the stowage plan in a smart way using a given search direction, as explained in Section 5.2.1. In contrary to the method tested in Section 7.2.2, here, only parts of the solution are rebuild. This shows that the GRASP Repair operator builds promising new solutions, if it is given a destroyed solution.

Table 7.13: The average score between the repair operators for how often they were used to provide a lower shifting cost than the current shifting cost. The overall score is shown together with the scores for cargo lists with low, medium and high filling degree.

	Repair operators (%)			
Filling degree	**GRASP**	**Grouping**	**Placement**	**Random**
Low	18.68	49.68	25.85	5.79
Medium	57.37	25.88	13.44	3.30
High	63.70	17.41	16.79	2.10
Overall	46.58	30.99	18.70	3.37

For instances with low degree of filling both Grouping and Placement perform better than GRASP. As for the Shift Removal operator, the Grouping and Placement Repair operators have a higher probability of being successful when there is more free space on the deck. For the Grouping Repair operator, this seems logical, as the operator first place as many of the removed vehicles next to vehicles from the same cargo. The remaining vehicles are then placed on the remaining empty space by the GRASP Repair operator, and the probability of this being successful increases with less vehicles on the deck.

Adaptive weight adjustment

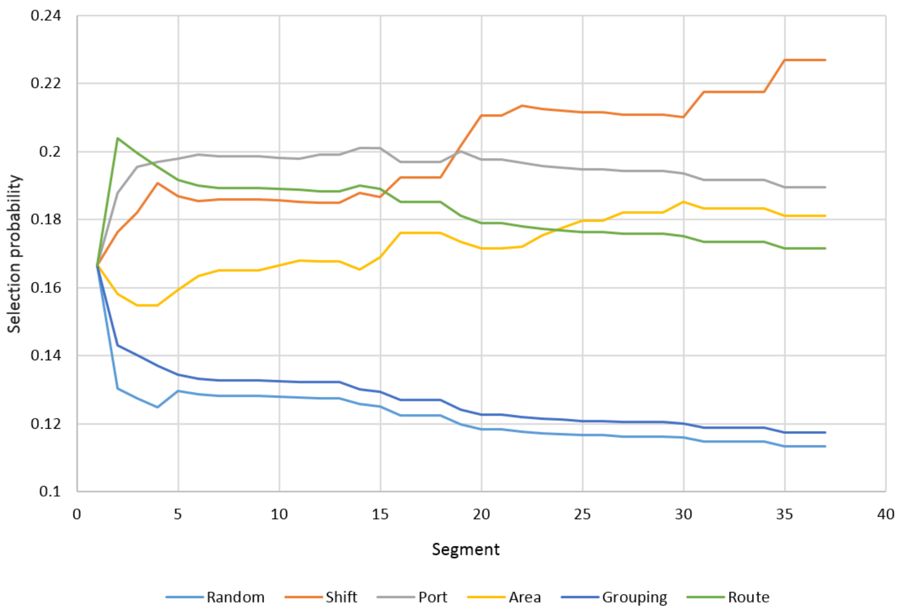


Figure 7.6: How the probability of selecting the different destroy operators changes during each segment for one run with test instance $Heur\_i200j75c13d6$.

In addition to the performance of the operators, the probability of selecting an operator changes during a run because of the weighting, described in Section 5.3.4. From Figure 7.6, we see how the probability of selecting different destroy operators changes with each segment during one run. As this is a deck with a low degree of filling, the probability of choosing Shift Removal increases. The ability to remove the vehicles that influence the shifting cost is crucial for reaching the lowest shifting cost possible, and this is exactly what the Shift Removal does, i.e. it removes the vehicles with the highest shifting cost. We also see a tendency that operators combining problem specific and random elements also have an increase in the selection probability, e.g. Port and Area Removal.

Summary

This section has shown that the Area Removal and the GRASP Repair operator are best overall, and that the performance of the different operators is dependent on the filling degree of the deck. For low filling degrees, defined as maximum 70% deck area usage, we see that operators that are more problem specific, e.g. Shift Removal, Grouping Repair, and Placement Repair, have a better performance. For medium to high filling degrees, operators that have a combination of problem specific and some random element perform better, e.g. Area Removal and GRASP Repair.

7.3.2 Scalability

In this section, we seek to determine which factors that affect the computational time of the 2DRSSP Heuristic. Higher grid resolution is desirable in order to reduce the area overestimation of the vehicles, and to improve the representation of the deck. However, increasing grid resolution comes at the cost of higher computational time. To evaluate how large grid resolutions the 2DRSSP Heuristic manages to solve within a reasonable amount of time, the computing times of the different parts of the 2DRSSP are tested. First, the GRASP phase of the 2DRSSP Heuristic is evaluated. Then, for phase two, the different parts of the ALNS are considered. The grid resolutions evaluated is 100×38, 200×75, 300×113, and 500×18.

GRASP

As can be seen from Table 7.14, the GRASP has a high number of iterations per second. An iteration consists of updating the weight parameters, selection of the search guiding parameters, attempt to create a feasible stowage plan, and a feasibility check. For the lowest grid resolution, 1169 iterations per second are achieved. A steady decline in the number of iterations per second is observed as the grid resolution increases. Even though the number of iterations for the highest grid resolution is considerably lower, it is still fast enough to obtain feasible initial solutions.

Table 7.14: Number of iterations per second for different grid resolutions for the GRASP heuristic in phase one.

Grid resolution	**Iterations per second**
100×38	1169
200×75	265
300×113	93
500×188	19

ALNS, SPP approach

The SPP approach used to evaluate the shifting cost of a stowage plan have the highest effect on the run time of all parts of the ALNS. As seen from Tables 7.15 and 7.16, the computational time of the SPP approach rapidly increases with increasing grid resolution. In addition to the grid resolution, the number of cargoes and vehicles carried seems to affect the run time of the SPP approach. As an SPP is solved for every cargo at its loading and unloading port, the overall time used by the SPP is dependent on the number of cargoes. This effect has a higher impact for higher grid resolutions, as the time used to solve an SPP increases with increasing grid resolution. In addition, the total number of vehicles on the deck have an impact on the computational time, as the time used to calculate each edge cost is highly dependent on the number of vehicles on the deck.

Table 7.15: Average stowage evaluation run time, number of cargoes and vehicles, for cargo lists 10-18, for grid resolutions 100×38 and 200×75.

Grid resolution	**100×38**			**200×75**		
Deck & cargo list	**Time (s)**	**Cargoes**	**Vehicles**	**Time (s)**	**Cargoes**	**Vehicles**
*c*10*d*5	0.25	18	95	0.96	18	95
*c*11*d*5	0.13	15	91	0.64	16.3	95.3
*c*12*d*5	0.17	11.7	91	0.87	13	106
*c*13*d*6	0.38	20	179	1.61	20	179
*c*14*d*6	0.31	15.3	179.3	1.44	16.7	175.7
*c*15*d*6	0.30	11	230	1.67	13.3	257.3
*c*16*d*7	0.29	18	153	1.26	18	153
*c*17*d*7	0.37	17	166	1.73	18	193.3
*c*18*d*7	0.18	10	165.7	0.98	11.7	176.7

Table 7.16: Average stowage evaluation run time, number of cargoes and vehicles, for cargo lists 10-18, for grid resolutions 300×113 and 500×188.

Grid resolution	**300×113**			**500×188**		
Deck & cargo list	**Time (s)**	**Cargoes**	**Vehicles**	**Time (s)**	**Cargoes**	**Vehicles**
$c10d5$	2.67	18	95	10.38	18	95
$c11d5$	1.79	16.7	94.7	7.47	17	100
$c12d5$	2.53	14	131	10.26	14	127
$c13d6$	4.02	20	179	13.91	20	179
$c14d6$	4.15	19	203	14.48	19	203
$c15d6$	4.39	14.3	253.7	16.86	15	269
$c16d7$	3.19	18	153	11.16	18	153
$c17d7$	4.55	19	206	16.67	20	209
$c18d7$	2.48	12	186	9.05	20	172.3

As seen from Table 7.16, $c17d7$ and $c18d7$ carry the same number of cargoes for grid resolution 500×188, but $c18d7$ has the higher run time, as it carries more vehicles. On the other hand, $c16d7$ carries both fewer vehicles and cargoes than $c18d7$, despite having a higher run time. This shows that in addition to grid resolution and the number of vehicles and cargoes, even more factors affect the solution time, such as the size of the vehicles. However, based on the results in Tables 7.15 and 7.15, it is reasonable to assume that grid resolution has the highest impact on the run time.

ALNS

One iteration of the ALNS procedure can roughly be split into three parts: Destroy, repair and evaluate. The evaluation procedure was shown to be time-consuming in the previous paragraphs. Compared to the time used by the SPP approach, the time use of the destroy operators is negligible. The time used by the repair operators in each iteration varies greatly. The GRASP and Grouping Repair operators use a maximum running time of $t^{rep} = 2s$. However, if a feasible stowage solution is found within this time limit, less time is used. The Placement and Random Repair only attempt one stowage plan creation in each iteration, thus significantly less time is used. The number of iteration per second for the ALNS for each of the instances is given in Figure 7.7.

As expected, we see from the figure that the number of iterations steadily declines as the grid resolution becomes higher and reaches an average value of 0.42 iterations per second for the highest grid resolution, i.e. 500×188. However, some interesting results exist for instances having a low filling degree on the deck. The first three instances, $Heur\_i100j38c10d5$, $Heur\_i100j38c13d6$, and $Heur\_i100j38c16d7$ have a higher iterations per second than the other instances with the same grid reso-

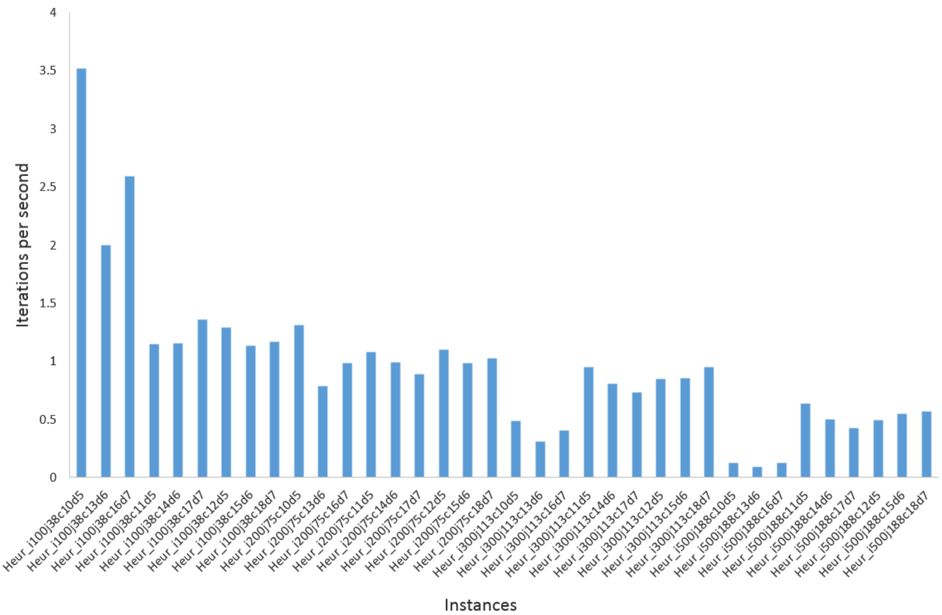


Figure 7.7: Average number of iterations per second for every instances tested in the ALNS part of the 2DRSSP Heuristic.

lution. These instances have a low filling degree, and the time used by the repair operators is assumed to be less as it is easier to repair the destroyed solutions with free space on the deck. In addition, the number of cargoes carried seems to affect the iteration time as well. Instances $Heur\_i100j38c13d6$ carries 20 cargoes, while $Heur\_i100j38c16d7$ carries 18 cargoes, and this may be the reason why $Heur\_i100j38c16d7$ has a higher amount of iterations per second, even though this instance has a larger deck and can carry more vehicles with the same filling degree as the smaller decks.

Another interesting observation is that for instances with a low filling degree and high grid resolutions, i.e. 300×113 and 500×188, the number of iterations per second is lower than instances with a medium and high filling degree. This is most likely due to the fact that the stowage evaluation is conducted more often, as more promising solutions are found during the search.

Summary

The results provided in this section clearly show that the bottleneck in the 2DRSSP Heuristic is the SPP approach. The grid resolution seems to be the factor which affects the run time of the SPP approach the most, while the number of cargoes and vehicles contributes less. As it is shown that the shifting evaluation procedure in the ALNS becomes very time-consuming for the highest grid resolution tested, 500×188 is set as the upper limit on grid resolution in the remaining part of this computational study.

7.4 Economic Analysis

This section looks at the economic aspect of the 2DRSSP, using the test instances found in Table A.3. Our modeling approach implicates that higher grid resolution increases the representation of the vehicles' area usage. With less overestimation, more optional cargoes can be carried, increasing the revenue of the voyage. Thus, in Section 7.4.1, we discuss the grid resolution's impact on revenue generated for the instances used. In Section 7.4.2, we show how the number of port calls on a voyage impacts the shifting cost. Finally, in Section 7.4.3, we examine whether it is a good policy to always bring an extra cargo if possible.

7.4.1 Effect of Grid Resolution

In Table 6.4, it is shown how different grid resolutions affect the representation of the actual area of the cargoes and the decks. Low grid resolutions give a higher overestimation of the vehicles' area usage than high grid resolutions do. For the decks, the approximated area is an underestimation of the actual area available. It is therefore of interest to see how great the difference is in approximated area usage and actual area usage of a deck for different grid resolutions. *Approximated area usage* refers to the approximated deck area divided by the approximate area usage of all vehicles, given a grid resolution. *Actual area usage* shows how much of the actual area of a deck that is filled with cargoes using their actual area.

The results are presented in Table 7.17, and they show the average area usage for the three different decks for low, medium and high filling degree, respectively.A clear trend for all the filling degrees shows that increasing the grid resolution decreases the gap between the approximated area usage and the actual area usage. This is logical, as increasing the grid resolution would give a better representation of the area of the deck and the cargoes. An example of a stowage plan solution, for instance $Heur\_i200j75c16d7$, can be seen in Figure 7.8, and it shows the approximated area of the cargoes carried covers 75.1% of the approximated deck area.

Table 7.17: Approximated area usage and actual area usage for different filling degrees and grid resolutions.

Filling degree	**Grid res.**	**Approximated area usage (%)**	**Actual area usage (%)**
Low	100×38	85.87	66.81
	200×75	75.97	66.81
	300×113	72.71	66.81
	500×188	69.92	66.81
Medium	100×38	93.92	73.89
	200×75	92.06	81.58
	300×113	90.54	83.90
	500×188	88.89	84.80
High	100×38	93.92	72.92
	200×75	92.31	80.33
	300×113	91.45	84.11
	500×188	89.81	86.03

Another interesting observation is that the actual area increases as the grid resolution increases, which is the opposite of the trend for the approximated area. This indicates that increasing the grid resolution and giving a lower overestimation of the vehicles would make more of the deck's area available. Thus, it becomes possible, when using the GRASP in phase one, to build an initial stowage plan that better utilizes the deck's area so more optional cargoes may be carried. An exception is found in the instances with a low filling degree, where the actual area stays the same, 66.81%, for every grid resolution. This is because we manage to place all the available cargoes in the cargo list on the deck for every grid resolution used on the test instances. Therefore, an increase in the grid resolution gives no impact on the revenue generated. However, we see how the approximated area usage reduces from 85.87% to 69.92% because of less overestimation of the vehicles.

Increasing the number of optional cargoes carried along the voyage would increase the revenue. In Table 7.18 the results for how the revenue changes for deck 6, with medium and high filling degree when the grid resolution is changed, are shown. The results from low filling degree are not considered, since all the cargoes from the cargo lists is carried for every grid resolution. From an economic perspective, we observe that the revenue increases when we increase the grid resolution. However, the amount of the increase in revenue declines as the grid resolution gets higher. This indicates that we earn less extra revenue each time the grid resolution is

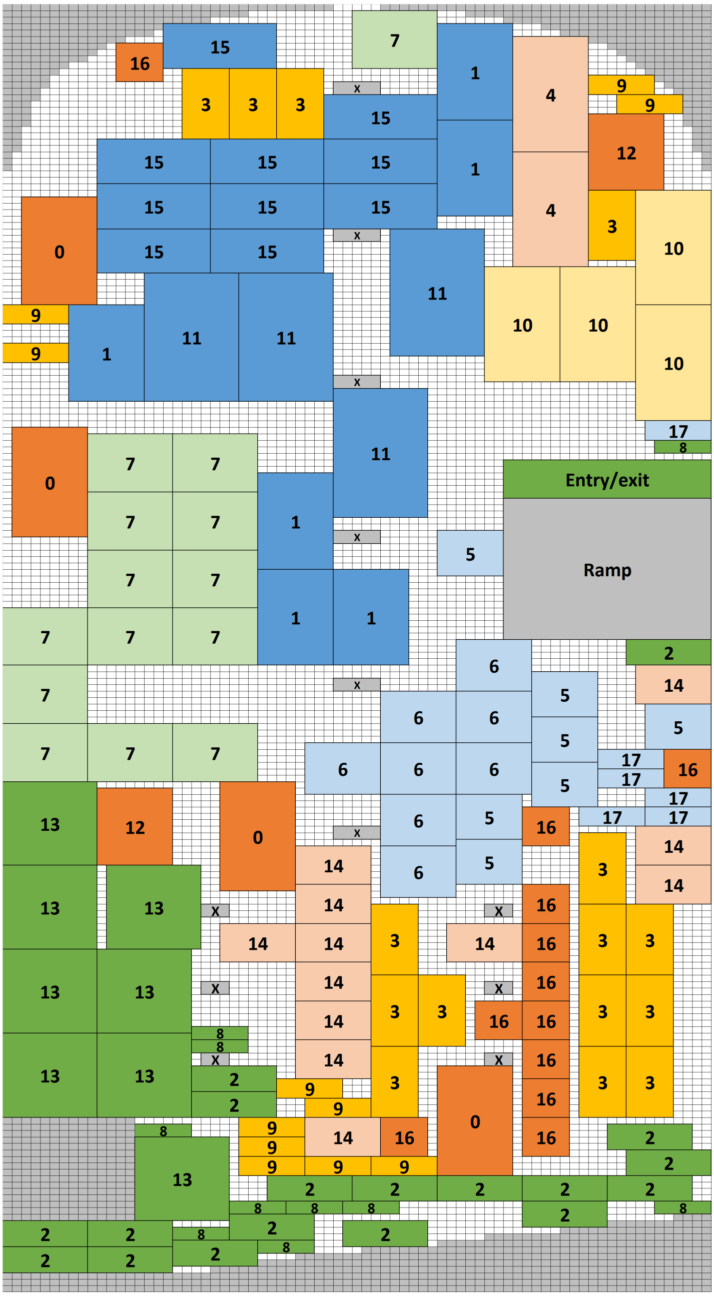


Figure 7.8: Example of a stowage plan solution for instance $Heur\_i200j75c16d7$, where the cargoes cover 75.1% of the approximated deck.

increased because we get closer to best obtainable revenue to earn on a voyage. The best obtainable revenue is the revenue we get when the actual area of the cargoes utilizes the actual area of the deck in the best possible way, i.e. the optimal stowage solution with infinite grid resolution. However, a trade-off exists due to computational limitations, and one needs to consider how high grid resolution to use in order to get a higher revenue against the computational time. For example, using a grid resolution of 500×188 in the medium filling degree case is unnecessary since the exact same revenue is obtained with a resolution of 300×113. For high filling degree, the revenue increases 33.77% when the grid resolution is changed from 100×38 to 200×75, but when changing it from 300×113 to 500×188 the revenue increases less, with only 7.78%. Similar trends were found for Deck 5 and 7.

Table 7.18: Average revenue improvement and weighted shifting cost for deck 6, approximated by different grid resolutions.

Filling degree	**Grid res.**	**Avg. revenue**	**Revenue improv. (%)**	**Avg. weighted shifting cost**	**Actual area usage (%)**
Medium	100×38	653897	-	125.79	73.62
	200×75	1128681	42.07	291.02	82.45
	300×113	1297661	13.02	217.60	85.82
	500×188	1297661	0	213.03	85.82
High	100×38	805644	-	186.93	73.33
	200×75	1216516	33.77	174.42	81.31
	300×113	1454280	16.35	178.15	84.36
	500×188	1576920	7.78	246.54	88.17

Table 7.18 also shows how the shifting cost changes for different grid resolutions. The shifting cost is adjusted so that it is possible to compare it for different grid resolutions. As we can see from the results, the weighted shifting cost does not give any clear trend, and it is difficult to say if increasing the grid resolution has any effects. However, as seen from Table 7.9, the filling degree of a deck affects the shifting cost, and it is reasonable to assume that when more optional cargoes are carried due to increased grid resolution, the shifting cost increases as well.

7.4.2 Ports Impact on the Shifting Cost

The number of port calls during a voyage is assumed to have a complicating effect on the problem since a large number of ports makes it difficult to place the cargoes such that the shifting cost becomes zero. In Table 7.19, we see how the average weighted shifting cost increases with an increasing number of ports for cargo lists with medium and high filling degree. For example, increasing the number of ports from four to eight increases the average weighted shifting cost from 235.7 to 431.0. This is because a higher number of ports would increase the number of possible combinations of where each cargo is to be loaded and unloaded. For example, a voyage with two loading and two unloading ports gives four possible combinations the cargoes can be loaded and unloaded, i.e. L1U3, L2U3, L1U4, and L2U4, where Ln and Un are loading and unloading port numbers, respectively. Increasing the number of ports to eight, with four ports for loading and four for unloading, would raise the number of possible combinations to 16. This complicates the procedure of placing vehicles from different cargoes on the deck in a smart way to avoid shifts.

Table 7.19: Average weighted shifting cost for different numbers of ports.

# of ports	**Avg. weighted shifting cost**
4	235.7
6	245.8
7	374.1
8	431.0

The reason why the cargo lists with a low filling degree are not included, is that they often tend to give a low shifting cost close to zero because of their low filling degree. For example, when using the stowage plan solution given in Figure 7.8, we see in Figure 7.9 how the heuristic has placed the cargoes with respect to their loading and unloading port in order to avoid shifting costs when visiting six ports, i.e. three unloading and three loading ports.

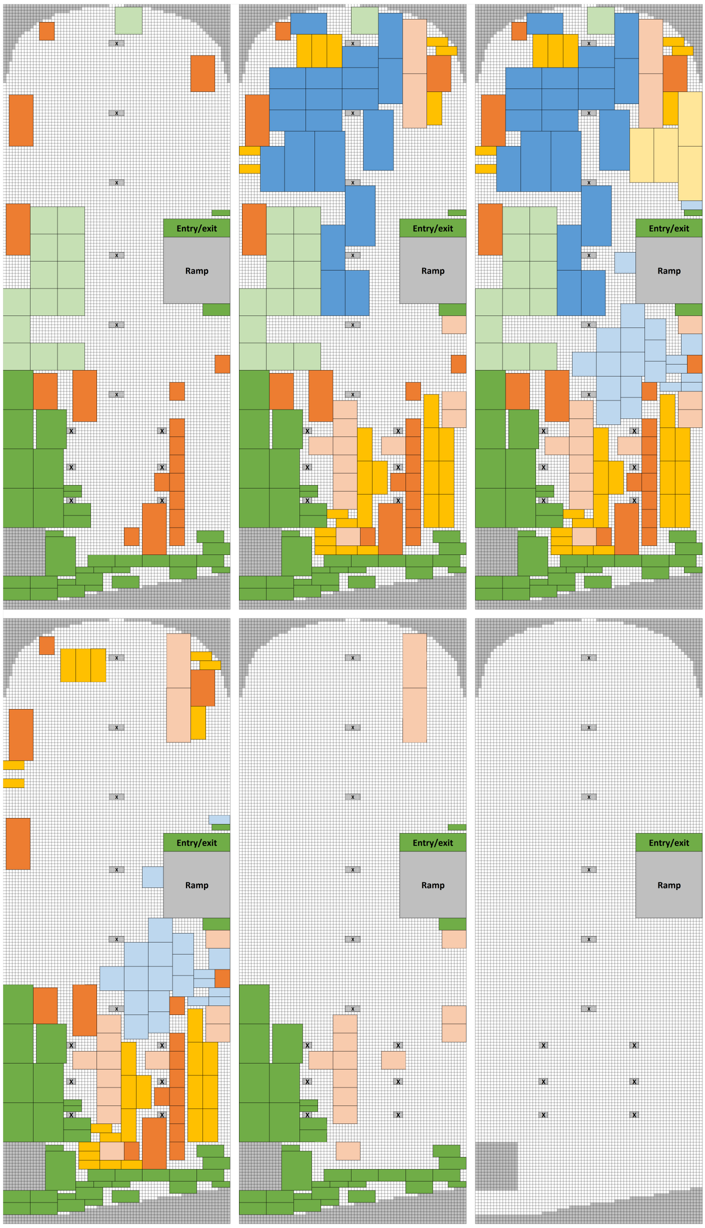


Figure 7.9: Example of a loading and unloading sequence for instance *Heur_i200j75c16d7*. From top left, loading ports 1-3. From bottom left, unloading ports 4-6. Cargoes with similar color are to be loaded and unloaded in the same ports.

7.4.3 Trade-off Between Revenue and the Shifting Cost

In Chapter 3, we stated that in the 2DRSSP, one always carries an additional cargo if there is enough free space on the deck, no matter how much the shifting cost increases. However, it might still be interesting to see how the shifting cost differs if a slightly worse set of optional cargoes is carried instead. There might exist stowage plans carrying cargoes generating a slightly lower revenue, but where the shifting cost varies greatly. It is expected that stowage plans with lower revenue generated from optional cargoes will have a lower shifting cost. This is based on the linear dependency between the area usage of a cargo, and the revenue from carrying it. Table 7.9 in Section 7.2.2, shows that lower filling degree of the deck gives a higher probability to find a shifting cost of 0. Because of this, it is conceivable that it may be appropriate for a company to consider stowage plans with slightly less revenue in order to get a stowage plan with easier loading and unloading process, i.e. low shifting cost. Therefore, in this section, we look at the four best stowage plans created in phase one of the 2DRSSP Heuristic, and analyze the difference in the shifting costs.

The instances with a high filling degree, i.e. cargo lists 12, 15, and 18 are used in this analysis, as it is impossible to carry all optional cargoes in these instances. For low and medium filling degree, it could be possible to carry all available cargoes for all/some grid resolutions. Now, the 2DRSSP has to select which optional cargoes to carry in order to maximize the generated revenue. When a set is selected, we additionally evaluate three optional sets with slightly lower revenue from the list consisting of optional cargo sets, sorted by revenue. Each instance is run three times, and the average score is reported. The results from two of the instances are presented in Figure 7.10 and 7.11, and shows the change in shifting cost with respect to change in revenue for grid resolution 100x38 and 200x75, on deck 5.

From the figures we see that the shifting cost varies between the different stowage plans even if the change in revenue is small. However, the results do not indicate that reducing the revenue slightly leads to a great reduction in the shifting cost. From Figure 7.10 and 7.11, this seems to be fully uncorrelated, but it may be possible that further analysis can reveal a weak positive trend, if more than four optional sets were evaluated. Despite this result, we believe that it would be of interest for a company to consider carrying a set of optional cargoes that generates slighly less revenue. It might be, in some circumstances, that a small reduction in revenue may lead to a great reduction in the shifting cost. From Figure 7.11, we observe that selecting the second best optional set proposed by the 2DRSSP Heuristic, reduces the shifting cost by approximately 50%, while the revenue only reduces 0.3%. In these cases it might be advantageous for a company to evaluate the trade-off between revenue and the shifting cost.

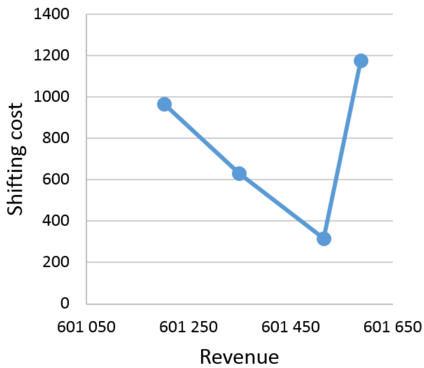


Figure 7.10: Shifting cost versus revenue for the four best stowage plan for instance $Heur\_i100j38c12d5$.

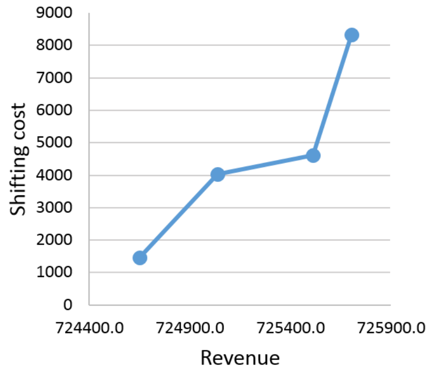


Figure 7.11: Shifting cost versus revenue for the four best stowage plan for instance $Heur\_i200j75c12d5$.

Finally, it should be noted that in this thesis the revenue is only based on the size of the area to the cargoes that are carried on the voyage, and therefore the values are completely fictive. Of that reason, we recommend that further research should be done within this field, as using real prices would provide an even better evaluation criteria.

8. Concluding Remarks

The RoRo ship stowage problem (RSSP) is an essential part of the operational decisions for RoRo operators in order to maintain their competitive position in the rolling cargo transportation market. The problem must be solved for every voyage, ensuring that the selected cargoes are possible to carry. In this thesis, we have presented the two-dimensional RoRo ship stowage problem (2DRSSP), which is an important sub-problem to the RSSP. Decisions such as which cargoes to carry and how to stow the vehicles carried during the voyage must be made.

A MIP model was developed, providing a new description of the stowage problem for one deck, where the 2DRSSP is a special case of the RSSP, solved for one deck. The model is richer than previous attempts, considering components such as unusable space and loading/unloading routes for each vehicle. In addition, a model describing the shifting evaluation problem was made. A commercial MIP solver was used to assess the difficulty of solving instances with the proposed models. The main conclusion was that small instances are solvable, but for realistically sized instances the need for heuristic methods is evident.

Thus, the 2-phase 2DRSSP Heuristic was designed using a greedy randomized adaptive search procedure (GRASP) in the first phase to determine which optional cargoes to carry. In the second phase, an adaptive large neighborhood search (ALNS) was used to improve the initial stowage plan provided by the GRASP, with regards to shifting cost. Several new destroy and repair operators, using problem-specific information, were built in order to improve the performance of the ALNS.

A shifting cost estimation approach was developed, solving multiple shortest path problems for each stowage plan using Dijkstra's algorithm. Compared to the exact method solved by a commercial solver, this approach was significantly faster and gave near optimal estimations in all tested instances.

The computational study showed that for small instances the heuristic provides near optimal solutions, and is also capable of solving realistically sized instances. The problem specific operators proved especially valuable for instances with fewer vehicles on the deck. Tests showed that increasing the representation accuracy of the cargoes resulted in an improved area utilization, and more optional cargoes were carried. This resulted in increased revenue, at the cost of higher computational time. It was also shown that carrying a set of optional cargoes generating slightly less revenue, would in some cases significantly reduce the shifting cost.

The results showed that placing vehicles from the same cargo together often improved the quality of a stowage plan, known as grouping. Another concept that proved to improve the quality of a stowage plan is placement, i.e. placing cargoes that are on the ship for the most number of sailing legs farther away from the entry/exit than cargoes with shorter time on the vessel. Although these concepts were not explicitly considered by the heuristic when placing vehicles on the deck, the resulting stowage plans showed that the heuristic grouped vehicles that were to be loaded and unloaded in the same ports together. Creating plans that are easy for the planners to follow is an important aspect. Even though the stowage solutions may not be directly implementable in every situation, we believe this heuristic approach provides results that could assist the planners when creating better stowage plans for a deck.

9. Future Research

For further research, we find the natural extension to multiple decks interesting, as a stowage plan must be made for the whole ship for every voyage. For decks where breakbulk units are placed, the need for decision tools is prominent. However, for decks filled with ordinary vehicles, stowage plans may be derived by the planners themselves. A decision support system combining the ability to manually assign cargoes to decks and use information from the 2DRSSP Heuristic to place the remaining cargoes would be an exciting area for further research. The problem becomes even more interesting if the stowage decisions are included in fleet deployment decisions.

The proposed 2DRSSP Heuristic could also be further developed. New repair and destroy operators could be added, using more of the information from the shifting evaluation results. For example, how often each square is traversed by a vehicle in the loading/unloading procedure could be used to update the square costs, making the most often traversed squares more expensive to place vehicles in. The search algorithm used to select the set of cargoes to carry should be further investigated. It would also be interesting to develop other heuristics, such as a genetic algorithm, which has proved to work well on the two-dimensional knapsack problem, as discussed in the literature review. Realistically sized problem instances with correctly priced cargoes and shifting cost should be used to evaluate the trade-off between revenue generated from optional cargoes and the shifting cost associated with a stowage plan. A more exact shifting cost could be derived from the expected time needed to move a vehicle.

For the modeling approach, an exact model considering both shifting cost and revenue generated simultaneously could reveal other factors that influence the shifting cost of a stowage plan, in addition to grouping and placement. Several elements can be added to the presented models, the most important ones being multiple decks and stability constraints. Uncertainty could be modeled, as rush orders may occur, generating a lot of extra revenue if uncertain cargoes are carried. Other possible extensions are: Flexible cargo quantities, as done by Øvstebø et al. (2011b), combined loading and unloading ports, non-rectangular vehicle representation, realistically loading/unloading evaluation that considers the turning radius of the vehicles, and allowing rotated vehicles placed on the deck.

References

Alvarez-Valdes, R., Parreño, F., and Tamarit, J. M. (2005). A grasp algorithm for constrained two-dimensional non-guillotine cutting problems. *Journal of the Operational Research Society*, 56(4):414–425.

Alvarez-Valdés, R., Parreño, F., and Tamarit, J. M. (2008). Reactive grasp for the strip-packing problem. *Computers & Operations Research*, 35(4):1065–1083.

Ambrosino, D., Sciomachen, A., and Tanfani, E. (2004). Stowing a containership: the master bay plan problem. *Transportation Research Part A: Policy and Practice*, 38(2):81–99.

Andersson, H., Fagerholt, K., and Hobbesland, K. (2015). Integrated maritime fleet deployment and speed optimization: Case study from roro shipping. *Computers & Operations Research*, 55:233–240.

Aslidis, A. H. (1989). *Combinatorial algorithms for stacking problems*. PhD thesis, Massachusetts Institute of Technology.

Avriel, M. and Penn, M. (1993). Exact and approximate solutions of the container ship stowage problem. *Computers & Industrial Engineering*, 25(1):271–274.

Avriel, M., Penn, M., and Shpirer, N. (2000). Container ship stowage problem: complexity and connection to the coloring of circle graphs. *Discrete Applied Mathematics*, 103(1):271–279.

Avriel, M., Penn, M., Shpirer, N., and Witteboon, S. (1998). Stowage planning for container ships to reduce the number of shifts. *Annals of Operations Research*, 76:55–71.

Chandra, S., Fagerholt, K., and Christiansen, M. (2015). Maritime fleet deployment in ro-ro shipping under inventory constraints. *Procedia-Social and Behavioral Sciences*, 189:362–375.

Chen, M., Chowdhury, R. A., Ramachandran, V., Roche, D. L., and Tong, L. (2007). *Priority queues and dijkstra's algorithm*. Computer Science Department, University of Texas at Austin.

Christiansen, M., Fagerholt, K., Nygreen, B., and Ronen, D. (2007). Maritime transportation. *Handbooks in operations research and management science*, 14:189–284.

Crainic, T. G. and Laporte, G. (1997). Planning models for freight transportation. *European Journal of Operational Research*, 97(3):409–438.

DanishShipFinance (2016). Roro vessels, segements. `http://www.shipfinance.dk/en/SHIPPING-RESEARCH/Ro-Ro-Skibe/Segmenter`. Accessed: 2016-04-21.

Dubrovsky, O., Levitin, G., and Penn, M. (2002). A genetic algorithm with a compact solution encoding for the container ship stowage problem. *Journal of Heuristics*, 8(6):585–599.

Dueck, G. (1993). New optimization heuristics: the great deluge algorithm and the record-to-record travel. *Journal of Computational physics*, 104(1):86–92.

Dueck, G. and Scheuer, T. (1990). Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing. *Journal of computational physics*, 90(1):161–175.

Dyckhoff, H. (1990). A typology of cutting and packing problems. *European Journal of Operational Research*, 44(2):145–159.

Fagerholt, K. and Christiansen, M. (2000a). A combined ship scheduling and allocation problem. *Journal of the Operational Research Society*, 51(7):834–842.

Fagerholt, K. and Christiansen, M. (2000b). A travelling salesman problem with allocation, time window and precedence constraints—an application to ship scheduling. *International Transactions in Operational Research*, 7(3):231–244.

Fagerholt, K., Johnsen, T. A., and Lindstad, H. (2009). Fleet deployment in liner shipping: a case study. *Maritime Policy & Management*, 36(5):397–409.

Feo, T. A. and Resende, M. G. (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations research letters*, 8(2):67–71.

Fischer, A., Nokhart, H., Olsen, H., Fagerholt, K., Rakke, J. G., and Stålhane, M. (2016). Robust planning and disruption management in roll-on roll-off liner shipping. *Transportation Research Part E: Logistics and Transportation Review*, 91:51–67.

Gendreau, M. and Potvin, J.-Y. (2010). *Handbook of metaheuristics*, volume 2. Springer.

Gharehgozli, A. H., Laporte, G., Yu, Y., and de Koster, R. (2014). Scheduling twin yard cranes in a container block. *Transportation Science*, 49(3):686–705.

Gonçalves, J. F. (2007). A hybrid genetic algorithm-heuristic for a two-dimensional orthogonal packing problem. *European Journal of Operational Research*, 183(3):1212–1229.

Hadjiconstantinou, E. and Christofides, N. (1995). An exact algorithm for general, orthogonal, two-dimensional knapsack problems. *European Journal of Operational Research*, 83(1):39–56.

Hansen, J. and Hukkelberg, I. (2015). 2d-packing with an application to stowage in roll-on roll-off liner shipping. Project Thesis TIØ4500.

Hansen, J. R., Hukkelberg, I., Fagerholt, K., Stålhane, M., and Rakke, J. G. (2016). 2d-packing with an application to stowage in roll-on roll-off liner shipping. In *7th International Conference on Computational Logistics (ICCL'16)*. Forthcoming, see Appendix B.

Hopper, E. and Turton, B. C. (2001). An empirical investigation of meta-heuristic and heuristic algorithms for a 2d packing problem. *European Journal of Operational Research*, 128(1):34–57.

Hu, T. C., Kahng, A. B., and Tsao, C.-W. A. (1995). Old bachelor acceptance: A new class of non-monotone threshold accepting methods. *ORSA Journal on Computing*, 7(4):417–425.

Hvattum, L. M., Fagerholt, K., and Armentano, V. A. (2009). Tank allocation problems in maritime bulk shipping. *Computers & Operations Research*, 36(11):3051–3060.

IMO (2016). International maritime organization. `https://business.un.org/en/entities/13`. Accessed: 2016-05-23.

Kang, J.-G. and Kim, Y.-D. (2002). Stowage planning in maritime container transportation. *Journal of the Operational Research Society*, 53(4):415–426.

Kirkpatrick, S., Vecchi, M. P., et al. (1983). Optimization by simmulated annealing. *science*, 220(4598):671–680.

Kreuzer, E., Schlegel, V., and Stache, F. (2007). Multibody simulation tool for the calculation of lashing loads on roro ships. *Multibody System Dynamics*, 18(1):73–80.

Lawrence, S. A. (1972). *International sea transport: the years ahead.* Lexington Books.

Leung, S. C., Zhang, D., Zhou, C., and Wu, T. (2012). A hybrid simulated annealing metaheuristic algorithm for the two-dimensional knapsack packing problem. *Computers & Operations Research*, 39(1):64–73.

Lindstad, H., Asbjørnslett, B. E., and Strømman, A. H. (2012). The importance of economies of scale for reductions in greenhouse gas emissions from shipping. *Energy policy*, 46:386–398.

MAIB (2016). Report on the investigation into the listing, flooding and grounding of hoegh osaka. https://assets.publishing.service.gov.uk/media/56e9a7afe5274a14d9000000/MAIBInvReport6_2016.pdf. Marine Accident Investigation Branch. Accident Report Accessed: 2016-04-21.

MarineTraffic (2016). http://www.marinetraffic.com/en/ais/details/ships/shipid:2672751/mmsi:563248000/imo:9185463/vessel:HOEGH_OSAKA. Accessed: 2016-05-12.

Martin, G. L., Randhawa, S. U., and McDowell, E. D. (1988). Computerized container-ship load planning: A methodology and evaluation. *Computers & Industrial Engineering*, 14(4):429–440.

Medialon (2014). Color line cruise ships. https://www.medialon.com/cruise-ships/color-line-cruise-ships/. Accessed: 2016-05-12.

Pacino, D., Delgado, A., Jensen, R. M., and Bebbington, T. (2011). Fast generation of near-optimal plans for eco-efficient stowage of large container vessels. In *Computational Logistics*, volume 6971, pages 286–301.

Pantuso, G., Fagerholt, K., and Wallace, S. W. (2015). Uncertainty in fleet renewal: a case from maritime transportation. *Transportation Science (forthcoming)*.

Parreño, F., Alvarez-Valdés, R., Oliveira, J., and Tamarit, J. M. (2010). A hybrid grasp/vnd algorithm for two-and three-dimensional bin packing. *Annals of Operations Research*, 179(1):203–220.

Pisinger, D. and Ropke, S. (2007). A general heuristic for vehicle routing problems. *Computers & operations research*, 34(8):2403–2435.

Port Technology (2014). Shipping liner insight #5: Cosco. https://www.porttechnology.org/news/shipping_liner_insight_5_cosco. Accessed: 2016-05-12.

Prais, M. and Ribeiro, C. C. (2000). Parameter variation in grasp procedures. *Investigación Operativa*, 9(1):1–20.

Resende, M. G. and Ribeiro, C. C. (2014). Grasp: greedy randomized adaptive search procedures. In *Search methodologies*, pages 287–312. Springer.

Ribeiro, G. M. and Laporte, G. (2012). An adaptive large neighborhood search heuristic for the cumulative capacitated vehicle routing problem. *Computers & Operations Research*, 39(3):728–735.

Ropke, S. and Pisinger, D. (2006). An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, 40(4):455–472.

Seixas, M. P., Mendes, A. B., Pereira Barretto, M. R., Da Cunha, C. B., Brinati, M. A., Cruz, R. E., Wu, Y., and Wilson, P. A. (2016). A heuristic approach to stowing general cargo into platform supply vessels. *Journal of the Operational Research Society*, 67(1):148–158.

Shaw, P. (1997). A new local search algorithm providing high quality solutions to vehicle routing problems. *APES Group, Dept of Computer Science, University of Strathclyde, Glasgow, Scotland, UK.*

Ship World Brokerage (2014). `http://www.shipworldbrokerage.com/Pages/CommercialVessels.aspx`. Accessed: 2016-05-12.

UNCTAD (2015). Review of maritime transport. United Nations Conference on Trade and Development. United Nations publication.

Wallenius Wilhelmsen Logistics (2014). `http://www.2wglobal.com/globalassets/folders/mark-v-folder-lowres-pdf.pdf`. Accessed: 2016-06-03.

Wallenius Wilhelmsen Logistics (2016a). `http://www.2wglobal.com/globalassets/services/ocean-terminals-4-compressed-50.jpg`. Accessed: 2016-06-08.

Wallenius Wilhelmsen Logistics (2016b). `http://www.2wglobal.com/globalassets/route-maps/asia-to-europe-route-map.gif`. Accessed: 2016-05-30.

Wallenius Wilhelmsen Logistics (2016c). `http://www.2wglobal.com/global-network/fleet/fleet-overview/`. Accessed: 2016-05-12.

Wäscher, G., Haußner, H., and Schumann, H. (2007). An improved typology of cutting and packing problems. *European Journal of Operational Research*, 183(3):1109–1130.

Wilson, I. and Roach, P. (2000). Container stowage planning: a methodology for generating computerised solutions. *Journal of the Operational Research Society*, 51(11):1248–1255.

Øvstebø, B. O., Hvattum, L. M., and Fagerholt, K. (2011a). Optimization of stowage plans for roro ships. *Computers & Operations Research*, 38(10):1425–1434.

Øvstebø, B. O., Hvattum, L. M., and Fagerholt, K. (2011b). Routing and scheduling of roro ships with stowage constraints. *Transportation Research Part C: Emerging Technologies*, 19(6):1225–1242.

Appendix

A. Overview of Test Instances and Area Usage

A.1 Test Instances

A complete overview of all the test instances used in the computational study. The approximated available deck area represents the percentage of how much of the deck's actual area is given in an instance. All the instances can be found as readable files in Appendix B.

Table A.1: Test instances used to test the MIP-models.

Test instance	**Deck #**	**Cargo list #**	**Grid resolution**	**Approx. available deck area**
$MIP\_i10j10c1d1$	1	1	10×10	80 %
$MIP\_i15j15c1d1$	1	1	15×15	88 %
$MIP\_i20j20c1d1$	1	1	20×20	96 %
$MIP\_i10j10c2d1$	1	2	10×10	80 %
$MIP\_i15j15c2d1$	1	2	15×15	88 %
$MIP\_i20j20c2d1$	1	2	20×20	96 %
$MIP\_i10j10c3d2$	2	3	10×10	90 %
$MIP\_i20j20c3d2$	2	3	20×20	95 %
$MIP\_i10j10c4d2$	2	4	10×10	90 %
$MIP\_i20j20c4d2$	2	4	20×20	95 %
$MIP\_i10j10c5d2$	2	5	10×10	90 %
$MIP\_i20j20c5d2$	2	5	20×20	95 %

Table A.2: Test instances used for the parameter tuning.

Test instance	**Deck #**	**Cargo list #**	**Grid resolution**	**Area available of total area**
$Param\_i50j19c6d3$	3	6	50×19	96.4 %
$Param\_i200j75c6d3$	3	6	200×75	98.3 %
$Param\_i50j19c7d3$	3	7	50×19	96.4 %
$Param\_i200j7576d3$	3	7	200×75	98.3 %
$Param\_i50j19c8d4$	4	8	50×19	97.4 %
$Param\_i200j75c8d4$	4	8	200×75	99.4 %
$Param\_i50j19c9d4$	4	9	50×19	97.4 %
$Param\_i200j75c9d4$	4	9	200×75	99.4 %

Table A.3: Test instances used to test the 2DRSSP Heuristic for realistically sized problems.

Test instance	**Deck #**	**Cargo list #**	**Grid resolution**	**Area available of total area**
$Heur\_i100j38c10d5$	5	18	100×38	97.5 %
$Heur\_i200j75c10d5$	5	18	200×75	99.0 %
$Heur\_i300j113c10d5$	5	18	300×113	99.7 %
$Heur\_i500j188c10d5$	5	18	500×188	99.9 %
$Heur\_i100j38c11d5$	5	19	100×38	97.5 %
$Heur\_i200j75c11d5$	5	19	200×75	99.0 %
$Heur\_i300j113c11d5$	5	19	300×113	99.7 %
$Heur\_i500j188c11d5$	5	19	500×188	99.9 %
$Heur\_i100j38c12d5$	5	19	100×38	97.5 %
$Heur\_i200j75c12d5$	5	19	200×75	99.0 %
$Heur\_i300j113c12d5$	5	19	300×113	99.5 %
$Heur\_i500j188c12d5$	5	19	500×188	99.9 %
$Heur\_i100j38c13d6$	6	20	100×38	98.1 %
$Heur\_i200j75c13d6$	6	20	200×75	99.2 %
$Heur\_i300j113c13d6$	6	20	300×113	99.6 %
$Heur\_i500j188c13d6$	6	20	500×188	99.9 %
$Heur\_i100j38c14d6$	6	19	100×38	98.1 %
$Heur\_i200j75c14d6$	6	19	200×75	99.2 %
$Heur\_i300j113c14d6$	6	19	300×113	99.6 %
$Heur\_i500j188c14d6$	6	19	500×188	99.9 %
$Heur\_i100j38c15d6$	6	20	100×38	98.1 %
$Heur\_i200j75c15d6$	6	20	200×75	99.2 %
$Heur\_i300j113c15d6$	6	20	300×113	99.6 %
$Heur\_i500j188c15d6$	6	20	500×188	99.9 %
$Heur\_i100j38c16d7$	7	18	100×38	97.7 %
$Heur\_i200j75c16d7$	7	18	200×75	98.4 %
$Heur\_i300j113c16d7$	7	18	300×113	99.0 %
$Heur\_i500j188c16d7$	7	18	500×188	99.9 %
$Heur\_i100j38c17d7$	7	20	100×38	97.7 %
$Heur\_i200j75c17d7$	7	20	200×75	98.4 %
$Heur\_i300j113c17d7$	7	20	300×113	99.0 %
$Heur\_i500j188c17d7$	7	20	500×188	99.9 %
$Heur\_i100j38c18d7$	7	16	100×38	97.7 %
$Heur\_i200j75c18d7$	7	16	200×75	98.4 %
$Heur\_i300j113c18d7$	7	16	300×113	99.0 %
$Heur\_i500j188c18d7$	7	16	500×188	99.9 %

A.2 Approximated Area Usage

Tables showing the approximated area usage compared to the actual area usage for the different cargo lists and decks.

Table A.4: Approximated and actual area usage for deck 1 and 2, cargo list 1-5.

	Actual representation				**Grid representation**		
Cargo list	**Mand. Area** (m^2)	**Opt. Area** (m^2)	**Deck Area** (m^2)	**Grid res.**	**Mand. Area** (m^2)	**Opt. Area** (m^2)	**Deck Area** (m^2)
1	224	306	763	10×10	504	792	612
1	224	306	763	15×15	384	512	672
1	224	306	763	20×20	342	504	729
2	239	215	763	10×10	585	594	612
2	239	215	763	15×15	440	324	672
2	239	215	763	20×20	383	367	729
3	84	76	160	10×10	126	112	144
3	84	76	160	20×20	98	96	156
4	84	76	160	10×10	126	112	144
4	84	76	160	20×20	98	96	156
5	72	58	160	10×10	106	84	144
5	72	58	160	20×20	91	69	156

Table A.5: Approximated and actual area usage for deck 3 and 4, cargo list 6-9.

	Actual representation				**Grid representation**		
Cargo list	**Mand. Area** (m^2)	**Opt. Area** (m^2)	**Deck Area** (m^2)	**Grid res.**	**Mand. Area** (m^2)	**Opt. Area** (m^2)	**Deck Area** (m^2)
6	1573	978	2581	50×19	2216	1541	2488
6	1573	978	2581	200×75	1808	1172	2538
7	1440	1224	2581	50×19	2234	1801	2488
7	1440	1224	2581	200×75	1718	1435	2538
8	2966	2558	5396	50×19	4735	3777	5257
8	2966	2558	5396	200×75	3554	3007	5365
9	2971	2415	5396	50×19	4729	3959	5257
9	2971	2415	5396	200×75	3539	2969	5365

B. Attachment

The attached ZIP file contains:

1. 2D-Packing with an Application to Stowage in Roll-on Roll-off Liner Shipping, paper by Hansen et al. (2016).
2. Code
   (a) Java files
   (b) Mosel files
   (c) Matlab file
3. Readme files
   (a) Java
   (b) Test instance generation
4. Test instances
5. Cargo lists in Excel