NTNU

Norwegian University of
Science and Technology

A Matheuristic Approach For Planning Interrelated Voyages With Separation Requirements In Maritime Transportation

Solveig Godhavn Lunde

Industrial Economics and Technology Management
Submission date: June 2018
Supervisor: Kjetil Fagerholt, IØT
Co-supervisor: Frank Meisel, Kiel University
Jone R. Hansen, IØT

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

Preface

This master thesis is the final result of the work for my Master of Science degree at the Norwegian University of Science and Technology (NTNU). The master thesis was written during the spring of 2018, in the course Managerial Economics and Operations Research at the Department of Industrial Economics and Technology Management.

The purpose of the thesis has been to develop a heuristic method for the planning of interrelated voyages with separation requirements in roll-on roll-off shipping. I would like to thank my supervisors Kjetil Fagerholt, Frank Meisel, and Jone R. Hansen for their guidance, contributions and valuable feedback throughout the semester.

Trondheim, June 2018

Solveig Lunde

Abstract

Roll-on/Roll-off (RoRo) vessels represent the primary source for transporting vehicles and other types of rolling material over long distances. However, comparatively little operational research has been done on RoRo shipping, suggesting there is room for improvement. In this thesis, I focus on the single trade ship routing and scheduling problem (STSRSP) in RoRo-shipping. This problem considers which ports a voyage should visit along a trade route, which vessels should be deployed for a given voyages, as well as where and when these vessels should load and unload goods.

In the STSRSP, the objective is to minimize the total cost of all activities, while satisfying contractual requirements Hansen et al. (2018). Efforts to solve mathematical formulations of this problem with standard MIP solvers have shown that the run time is prohibitively large for direct use in operational level decisions. Hence, the purpose of this thesis have been to create a heuristic that can be used to reduce the time used to solve single instances of the STSRSP to within an acceptable accuracy.

In this thesis, I propose a new solution method to solve the STSRSP. The solution method uses a matheuristic approach that divides the problem into several parts, combining both mathematical models to determine routes and vessel assignment, as well as a heuristic for placing the contracts on voyages. Finally, a service level search is implemented to make sure the separation requirement is satisfied. Computational results show that the matheuristic has the potential to solve STSRSP instances within very short time limits.

Sammendrag

Roll-on/Roll-off shipping er den primære metoden for transport av kjøretøy og andre typer objekter som kan rulles på skip, over lange avstander. Det har imidlertid blitt gjort relativt lite forskning innen optimering av RoRo-shipping, noe som tyder på at det er rom for forbedring. I denne masteroppgaven fokuserer jeg på operasjonelle beslutninger knyttet til det å planlegge flere reiser over en enkelt handelsrute innen RoRo-shipping, samt en plan for hvilke kontrakter som skal betjenes og hvilke mengder som skal lastes og losses ved planlagte portbesøk.

Målet er å minimere den totale kostnaden for alle aktiviteter, samtidig som alle kontraktsbetingelser er oppfylt (Hansen et al., 2018). Arbeidet med å løse matematiske formuleringer av dette problemet med standard MIP-løsere (Mixed Integer Programming) har vist at kjøretiden er uforholdsmessig stor for direkte bruk i beslutninger på operasjonelt nivå. Derfor har formå let med denne avhandlingen vært aa skape en heuristisk som kan brukes til å redusere tiden som brukes til løse enkelt forekomster av problemet innenfor en akseptabel nøyaktighet.

I denne masteroppgaven foreslår jeg en ny løsningsmetode for løse problemet. Løsningsmetoden bruker en matheuristisk tilnærming som deler problemet i flere deler, kombinere begge matematiske modeller for å bestemme ruter og fartøysoppgave, samt en heuristisk for å plassere kontraktene på reiser. Til slutt, implementeres et servicenivå søk for sikre at separasjonskravet er tilfredsstilt. Beregningsresultater viser at matheuristikken har potensial til løse problem-instanser innen svært kort tid.

Table of Contents

Preface **1**

Abstract **i**

Sammendrag **i**

Table of Contents **vi**

List of Tables **ix**

List of Figures **xi**

1 Introduction **1**

1.1 Maritime transportation 1

1.2 Levels of planning 3

1.3 Roll-on roll-off shipping 3

1.4 The single trade ship routing and scheduling problem . . . 5
1.5 Thesis outline . . . 5

2 Problem Description **7**
2.1 General description . . . 7
2.2 Example of the STSRSP . . . 10

3 Literature Review **13**
3.1 RoRo-shipping . . . 13
3.2 Voyage separation requirements . . . 14
3.3 Transit time requirements . . . 15

4 Mathematical models **17**
4.1 Full voyage model . . . 17
4.1.1 Notation . . . 17
4.1.2 The model . . . 19

5 Matheuristic for the STSRSP **27**
5.1 Matheuristic overview . . . 28
5.2 Step 1 - 4: Obtaining routes and assigning vessels . . . 31
5.2.1 Step 1: Generating candidate routes . . . 31
5.2.2 Step 2: Route model . . . 31
5.2.3 Step 3: Assigning vessels to routes model . . . 33

5.2.4 Step 4: Determining the sequence of routes 35
5.3 Step 5: Placing contracts on voyages 35
5.3.1 Sorting contracts 36
5.3.2 Placing contract 39
5.3.3 Repair functions 41
5.4 Step 6: Solving the reduced model with fixed variables . . 42
5.4.1 Preproccessing 42
5.4.2 The reduced model 42
5.5 Step 7: Service level search 46

6 Computational Study **49**
6.1 Test instances 49
6.1.1 Creating instances 50
6.1.2 Naming of the instance 52
6.1.3 Service level requirement 53
6.2 Results 53
6.2.1 Without service level requirement 53
6.2.2 With service level requirement 55
6.3 Analysis 57

7 Concluding Remarks **63**

8 Future Research **65**

Bibliography **67**

A Voyage model **71**

A.1 Notation 71

A.2 The model 73

B Results - No service level **77**

B.1 Matheuristic 77

B.2 Model 79

C Results - With service level requirement **81**

C.1 Matheuristic 81

C.2 Model 83

D Results - No transit time requirement **85**

D.1 Heuristic 85

D.2 Model 86

List of Tables

2.1 Overview of the contracts to be serviced during the planning period 10

2.2 Overview of the vessels available to service the trade . . . 11

2.3 A solution to the example problem with quantities picked up 12

5.1 Example contracts - unsorted 37

5.2 Example contracts - sorted 38

6.1 Sets of test instances 52

6.2 Instance naming scheme 52

6.3 Number of instances solved and average gap (%) compared to the optimal solution, for the instance sets by running the matheuristic and model 54

6.4 Number of instances solved, and average gap (%) compared to the solution from running the model for 1000 seconds, for the instance sets by running the matheuristic and model 55

6.5 Number of instances solved, and average gap (%), for the instance sets by running the matheuristic and model 56

6.6 Number of instances solved, and average gap (%) compared to the solution from running the model for 1000 seconds, for the instance sets by running the matheuristic and model 56

B.1 The result from running the heuristic on the small instances with no service level requirement for 100 sec and for 1000 sec 77

B.2 The result from running the heuristic on the medium instances with no service level requirement for 100 sec and for 1000 sec 78

B.3 The result from running the heuristic on the large instances with no service level requirement for 100 sec and for 1000 sec 78

B.4 The result from running the model on the small instances with no service level requirement for 100 sec and for 1000 sec 79

B.5 The result from running the model on the medium instances with no service level requirement for 100 sec and for 1000 sec 79

B.6 The result from running the model on the large instances with no service level requirement for 100 sec and for 1000 sec 80

C.1 The result from running the heuristic on the small instances with service level requirement for 100 sec and for 1000 sec 81

C.2 The result from running the heuristic on the medium instances with service level requirement for 100 sec and for 1000 sec 82

C.3 The result from running the heuristic on the large instances with service level requirement for 100 sec and for 1000 sec 82

C.4 The result from running the model on the small instances with service level requirement for 100 sec and for 1000 sec 83

C.5 The result from running the model on the medium instances with service level requirement for 100 sec and for 1000 sec 83

C.6 The result from running the model on the large instances with service level requirement for 100 sec and for 1000 sec 84

D.1 The result from running the heuristic without transit time requirements on the large instances with no service level requirement for 100 sec and for 1000 sec 85

D.2 The result from running the model without transit time requirements on the large instances with no service level requirement for 100 sec and for 1000 sec 86

List of Figures

1.1 An example of a RoRo vessel. Source: Logistics (2016b) . 4

1.2 Example of a trade route from Asia to Europe. Hansen et al. (2018) 6

2.1 An example of the STSRSP for a trade from US to Japan with only three contracts 10

2.2 A solution to the example problem 11

5.1 Flow chart of the matheuristic 30

6.1 Flow chart 59

Chapter 1

Introduction

1.1 Maritime transportation

Maritime trade is an integral part of the globalized world economy. Over 80% of global trade by volume, and more than 70% of its total value is being transported on board cargo ships and handled at seaports worldwide. (UNCTAD, 2017) This makes the shipping industry the largest transport provider used for distribution of goods. In 2017 it is estimated that seaborne trade will increase by 2.8% with total volumes reaching 10.6 billion tons. The total volume is projected to keep increasing, with an estimated compound annual growth rate of 3.2% between 2017 and 2022. Volume is projected to grow across all segments, however it will grow fastest for containerized trade and major dry bulk commodities. (UNCTAD, 2017)

The global shipping industry is to a large degree affected by developments in the world economy. Political and economic turmoil can quickly affect this industry, through mechanisms as fuel price, regulations, decrease in international trade, ect. A current example is the import tariffs on metals and goods which the US has threatened to introduce, which may end up causing a trade war that can be prove to be harmful for the shipping industry. Hence, demand for seaborne trade may change rapidly. However, the supply side of the shipping industry has a long-time horizon, and is repre-

sented by the ships that carry the cargos. (UNCTAD, 2017). While demand can fluctuate rapidly, supply is rigid. It is simple to see why: ships have a relatively long life cycle, and from decision to finished product it takes a large amount of time to build a new cargo ship. Supply can be affected in the short run by scrapping vessels, but it is not very adaptable to current economic conditions. For the last five years, the growth of the world fleet has been diminishing. Even so, the supply of ship carrying capacity increases faster than the demand. This again leads to a continuation of global overcapacity in supply, which puts a downward pressure on freight rates and earnings. The current mismatch between supply and demand reduces the profitability in most shipping market segments. As an example, the container-shipping market reported a collective operating loss of $3.5 billions in 2016. (UNCTAD, 2017). For an industry with large investment costs and very high daily operational costs, it is still very important to search for new ways to improve profitability of operations.

Lawrence (1972) introduced the concept that maritime transportation can be divided into three different segments: industrial, tramp and liner shipping. Both Lawrence (1972) and Ronen (1983) gave detailed descriptions of each of these three segments. Industrial shipping companies own both the cargo and their ships they use to transport it, giving them control over a larger part of the supply chain, if not all of it. As an example, oil- and gas companies typically prefer having their own fleet for transport of their products from production centers to markets, and are therefore industrial shipping companies. Since these types of companies profits from production rather than transportation of goods, their objective is to minimize costs. A tramp shipping companys activities is partially serving spot cargoes and partially serving contracts. They are available at short notice, and typically follow available goods around the globe, and can therefore be compared to a taxi service. Examples of typical goods transported by tramp shipping companies are oil, coal, gas, minerals and grain. Liner shipping companies has a set of predefined routes and schedules, and transport goods along these routes. They offer predictability, allowing production companies to have long term contracts for transportation of their products. This can be compared to a bus service. For tramp and liner shipping the goal is to maximize profits, as they profit from the transportation of goods.

1.2 Levels of planning

The decisions made during planning in maritime transportation can be categorized into three different planning levels, depending on the magnitude of the time horizon as well as the complexities of the decisions. The different planning levels are strategic, tactical and operational level. (Christiansen et al., 2013).

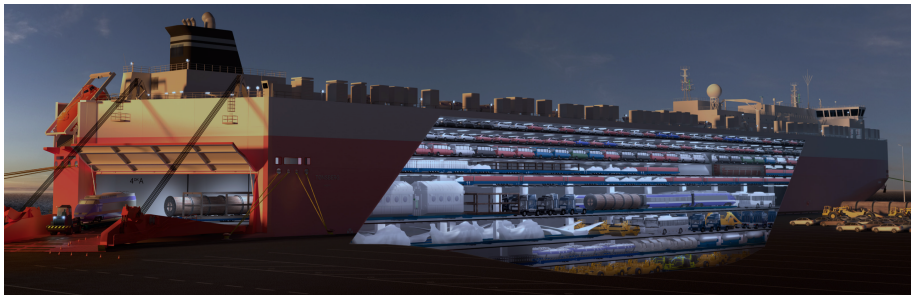
The strategic level is comprised of decisions with a long term planning horizon, typically years. Examples of strategic level decisions can be designing trade routes, route network design, ship design, and fleet composition. These decisions are difficult to change after they have been made. The tactical level is comprised of decisions with a lesser planning horizon, typically months and up to a year. Examples of tactical level decisions can be ship routing and scheduling, transshipment, and fleet deployment Christiansen et al. (2007). The operational level is comprised of decisions with a short term planning horizon, typically days or weeks. Examples of operational level decisions can be stowage plans and sailing speeds.

1.3 Roll-on roll-off shipping

Roll-on Roll-off (RoRo) vessels are a type of vessel in the general cargo ship category, that usually operate in the liner shipping mode of transportation. (UNCTAD, 2017). Deepsea RoRo shipping is a vessel specialization that were developed after the Second World War, and designed for transport of road haulage vehicles and private cars. At each port where these vessels operated, a ramp or link span was provided enabling vehicles to drive on and off the vessels (hence the name Roll-on Roll-off.) The obvious advantage of roll-on roll-off shipping is speed. Since the goods are wheeled they can be driven or hauled straight on to a vessel at one port, and then driven or hauled off at a different port within minutes of the vessel docking. In addition, this eliminated cargo handling and the dependency on cranes to load goods onto vessels for these types of goods. Modern RoRo vessels can transport a large number of wheeled vehicles, such as for example automobiles, trucks, semi-trailer trucks, trailers, railroad cars, heavy rolling

machinery. Additionally, RoRo vessels can handle complex and large cargos that are placed on trolleys or platform vehicles and driven or rolled on and off the vessels. Examples of these types of cargoes are turbines, yachts, and windmill blades. These vessels typically have built-in ramps that allow the cargo to efficiently be rolled on and off the vessel in any port dimensioned for the vessel and the ramp.

Figure 1.1 An example of a RoRo vessel. Source: Logistics (2016b)



There are different types of RoRo vessels, but the type most relevant for this thesis is the Pure Car/Truck Carriers (PCTC.) These vessels have a distinct box-like exterior, and the interior works in a similar manner as a parking garage (Logistics (2016a)). Inside the PCTC there are several decks with different storage capacities and capabilities. See as an example the vessel in Figure 1.1. Here the lower decks are built to transport rolling equipment and break bulk cargoes, while the middle decks is built to transport heavy and/or large cargoes and the upper decks are built to transport cars. PCTCs can typically be adapted on the go to accommodate differences in cargoes. They typically have hoistable decks which can allow for increasing vertical clearance. However, while lifting these decks accommodates higher cargos, it also reduces the total capacity.

RoRo shipping is an important segment in the shipping industry, with a global fleet of around 5000 vessels with a total capacity larger than 24 million deadweight tons (ISL, 2017). All though the volume of cargoes transported by the RoRo industry is small compared to the total volume of cargoes transported by the entire cargo ship industry, it is the preferred

choice when transporting wheeled goods and vehicles around the world.

1.4 The single trade ship routing and scheduling problem

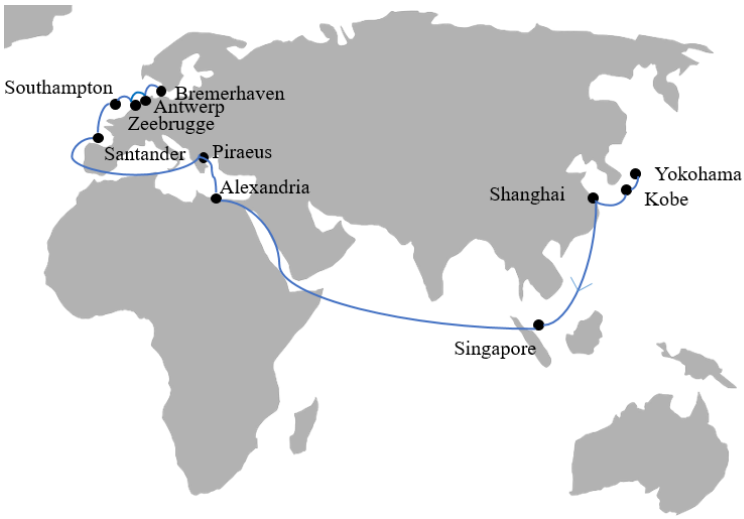
The STSRSP is the problem of planning the routes sailed and which vessels should be deployed for a set of interdependent voyages which are to be performed on a given *trade route*, as well as where and when these vessels should load and unload goods. A *trade route*, or trade, is a logistical network identified as a set of pathways and stoppages for commercial transport. An example of a trade route can be seen in Figure 1.2. A vessel that is deployed on the trade performs a *voyage*, i.e visits a sequence of stoppages, usually ports, along the trade. In the STSRSP these activities are constrained by a set of contracts, and the objective is to minimize the total cost of all activities and keeping them within a specific planning horizon, while satisfying contractual requirements (Hansen et al., 2018). One of the largest contributors to the total costs of performing voyages is port visits, so by using operations research to optimize route design, shipping companies can hopefully reduce their operational costs.

1.5 Thesis outline

The contributions in this thesis is a new solution method that has the potential to solve STSRSP instances within very short time limits. The solution method uses a matheuristic approach that divides the problem into several parts, combining both mathematical models to determine routes and vessel assignment, as well as a heuristic for placing the contracts on voyages. Finally, a service level search is implemented to make sure the separation requirement is satisfied.

Hansen et al. (2018) propose and compare two novel mixed integer programming models for the STSRSP, one called the *vessel model* and the other called *voyage model*. Both including new ways of modeling the con-

Figure 1.2 Example of a trade route from Asia to Europe. Hansen et al. (2018)



tracts' separation requirements. First, they showed that the voyage model performs significantly better than the vessel model. They then showed through a computational study on a set of realistic test instances that significant gains can be obtained compared to current planning practice. This thesis is an extension of the work done by Hansen et al. (2018), using the voyage model presented in their paper.

A detailed problem description is presented in Chapter 2. Chapter 3 introduces a literature review which describes relevant literature to the STSRSP.In Chapter 4, the mathematical formulation of the STSRSP is presented. A heuristic solution method is proposed in Chapter 5, and a computational study is presented in Chapter 6. Finally, concluding remarks and proposed future research are given in Chapters 7 and 8, respectively.

Chapter 2

Problem Description

In this chapter the single trade ship routing and scheduling problem (STSRSP) in RoRo-shipping is presented. The general description of the problem is given in Section 2.1. In Section 2.2, a small, illustrative example of the STSRSP is given.

2.1 General description

The STSRSP is the problem of planning the routes sailed and which vessels should be deployed for a set of interdependent voyages, as well as where and when these vessels should load and unload goods. These activities are constrained by a set of contracts, and the objective is to minimize the total cost of all activities and keeping them within a specific planning horizon, while satisfying contractual requirements. The planning horizon defines the overall time window for the problem. A typical length of the planning horizon is one or two months.

A *trade route*, or trade, is a logistical network identified as a set of pathways and stoppages for commercial transport. A vessel that is deployed on the trade performs a *voyage*, i.e visits a sequence of stoppages, usually ports, along the trade. A vessel sailing a voyage on a specific trade, does not

necessarily visit every port along that trade. Vessels have different properties, as for example stowage capacities, cargo handling equipment, sailing speeds and operating costs. These properties put restrictions on which ships can be deployed to specific voyages, as they limit what contracts the vessel can service. Not all vessels can perform loading and unloading operations at every port along the trade because of size restrictions at ports. Vessels can also vary the speed at which they sail, depending on what is required on a specific voyage. Vessels have different maximum sailing speeds and different fuel consumption, which is a function of sailing speed, all of which affects costs and which vessels can be used for which voyages.

In addition, a vessel has several decks for stowage, and capacities and cargo requirements for these decks are usually not the same. Their stowage capacities might vary in both area and volume, and the different decks might have different weight restriction limiting what can be stored there. The reason for these differences in deck properties, is that the goods that are being transported can be divided into different product types with different requirements. A product type is a class of products that are similar in some important attributes, such as size, weight, and functionality. Therefore, a deck might only be limited to store certain product types, while other decks can store all product types. For example, cars can be stored on decks which is meant for heavy construction equipment, but heavy construction equipment can usually not be stored on decks meant for cars.

Which ports to visit and which goods should be transported during a voyage are determined by a set of *contracts* between the shipping company and customers. These contracts define which port the goods should be loaded in and where it should be unloaded. For example, goods could be loaded in America and unload in Europe after sailing the Atlantic ocean. In addition, contracts state the demand for products, that is, which products to be transported at which quantities. Inventory capacities and production rates impose limits on the quantity of products that is available for transport at given times. A contract might therefore require pickups to be *fairly evenly spread* in time across the planning horizon. That is, the customer will expect that goods can be shipped out at regular intervals. One way of ensuring that the voyages are fairly evenly spread, is to use time windows. The time window defines the earliest and latest start up time for a voyage on a trade. For example, consider a contract that has to be serviced two times

over a planning period of 30 days. It is possible to say that the contract should be serviced once between day 1 and day 14, and once between day 15 and day 30. However, using this approach, one possible solution is for the the contract to be picked up at day 13 and day 16, which is not fairly evenly spread in time. A way to avoid this is to make the time windows smaller. For example, one can say that the contract should be serviced once between day 1 and day 5, and once between day 15 and day 20. However, tighter time windows reduce flexibility and might lead to poor fleet utilization. Another way of enforcing fairly evenly spread, is to use trade specific time separation limits. For example, a limit defining the maximum acceptable deviation in days from the desired spread, when servicing a contract with evenly spread requirement.

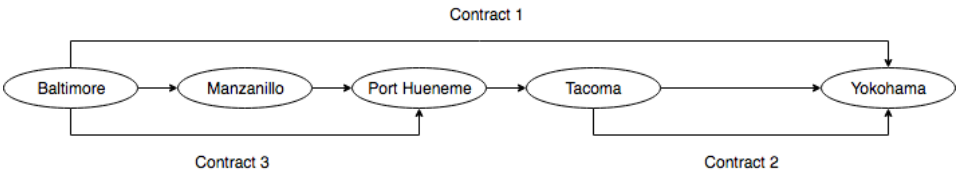
Some contracts have pickup frequency restrictions, that is, it may require that the total quantities to be transported be divided over several or all voyages during the planning horizon. These contracts specify a minimum, and maximum, number of pickups for that contract. A contract also define an upper and lower bound on quantities which have to be transported during the voyage if the contract is handled. These limits ensure that impractical pickups are omitted. For example, consider a contract that has a total demand of 1000 cars. Loading 10 cars on one voyage, and 990 cars on another voyage, fulfills the total demand. However, it is not very practical. A contract also defines any restrictions on transit time. The *transit time restrictions* dictate within what time interval a contract has to be fulfilled.

In summary, the objective of the STSRSP is to minimize the total costs of all transport activities while keeping the activities within a specific planning horizon and satisfying all contractual requirements. Important decisions to be made are which vessels to deploy for which voyages, and which routes these vessels should follow, which schedule and sailing speeds the vessels should keep, how to divide contracts among the different vessels, and how to divide contract demand over the different voyages.

2.2 Example of the STSRSP

In this section, a small, illustrative example of the STSRSP is presented, based on the example from Hansen et al. (2018). We consider a trade from US to Japan, with five ports and only three contracts, shown in Figure 2.1. The planning period of this example is set to one month, i.e. 30 days.

Figure 2.1 An example of the STSRSP for a trade from US to Japan with only three contracts



As in Hansen et al. (2018), contract 1 has a total demand of 1000 units of product type A, to be split into two to four partial cargoes. The cargoes must be loaded in Baltimore and unloaded in Yokohama. Contract 2 has a total demand of 1500 units of product type B, to be split into three or four partial cargoes. The cargoes must be loaded in Tacoma and unloaded in Yokohama. Contract 3 is an intra-regional contract, to be loaded in Baltimore and unloaded in Port Hueneme. The total demand of contract 3 is 300 units of product type A, to be split into one or two partial cargoes. In addition, contract 1 has a transit time requirement of 30 days. The other two contracts have no transit time requirements. An overview of the contracts can be found in Table 2.1. The maximum allowed deviation from the desired spread is 4 days.

Table 2.1: Overview of the contracts to be serviced during the planning period

	Total demand	Product type	Pickups		Ports		Transit time req.
			Min	Max	Load	Unload	
Contract 1	1000	A	2	4	Baltimore	Yokohama	30 days
Contract 2	1500	B	3	4	Tacoma	Yokohama	None
Contract 3	300	A	1	2	Baltimore	Port Hueneme	None

There are three vessels available to service the trade route. Vessel 1 has

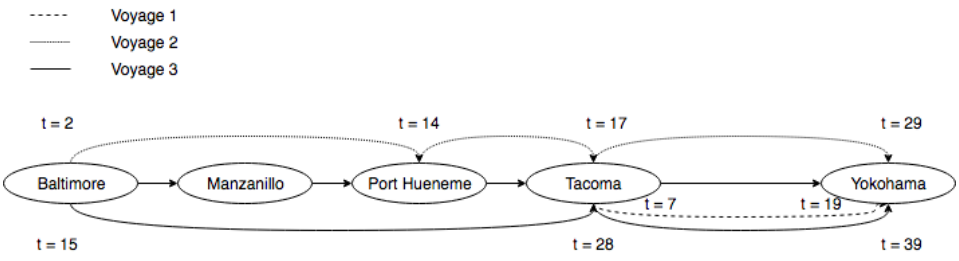
capacity to carry 700 units of product type A, and 350 units of product type B. Vessel 1 is available on day 2 in Baltimore. Vessel 2 has capacity to carry 500 and 600 units, of product type A and B, respectively, and is available on day 7 in Tacoma. Lastly, vessel 3 has capacity to carry 400 units of product type A, and 650 units of product type B. Vessel 3 is available in Baltimore in day 15 of the planning period. An overview of the vessels can be found in Table 2.2.

Table 2.2: Overview of the vessels available to service the trade

	Capacity		Available (day)	Origin port
	A	B		
Vessel 1	700	350	2	Baltimore
Vessel 2	600	500	7	Tacoma
Vessel 3	400	650	15	Baltimore

Figure 2.2 and Table 2.3, shows a solution to the example problem. As in Hansen et al. (2018), voyage 1, assigned to vessel 3, visits only two ports, starting in Tacoma on day 7 and ending in Yokohama on day 19. Voyage 2, assigned to vessel 1, starts in Baltimore on day 2, then visiting Port Hueneme, Tacoma and Yokohama on days 14, 17 and 29, respectively, while voyage 3, assigned to vessel 3, starts in Baltimore on day 15 and visits Tacoma on day 28, before ending up in Yokohama on day 39. In this solution, contract 1 is serviced twice, i.e. on days 2 and 15, contract 2 is serviced three times, i.e. on days 7, 17 and 28, while contract 3 is serviced once on day 2.

Figure 2.2 A solution to the example problem



From Table 2.3, we can see that the quantities picked up from each contract is within the given ranges.

Table 2.3: A solution to the example problem with quantities picked up

	Voyage 1		Voyage 2		Voyage 3	
	A	B	A	B	A	B
Contract 1			400		600	
Contract 2		650		350		500
Contract 3			300			
Total:		650	700	350	600	500

The desired spread for the partial cargoes of the contracts is given by T^{PH}/b_c, where T^{PH} denotes the length of the planning period (30 days), and b_c is the number of pickups for the contract, i.e. number of partial cargoes. That is, for contract 1, which is serviced twice, the desired spread is 15 days between the pickups. As noted above, contract 1 is picked up on days 2 and 15, which corresponds to a spread of 13 days between the pickups. This gives a deviation, from the desired spread, of two days. For contract 2, which is serviced three times, the desired spread is 10 days. The contract is picked up on days 7, 17, and 28, which corresponds to a spread of 10 days between the first and second pickup, and 11 days between the second and third. This gives a deviation, from the desired spread, of 1 day. Finally, since contract 3 is only picked up once, the pickup is evenly spread. This means that the total deviation from the desired spread for the planning period is 3 days. We can see that the services of each of the contracts are fairly evenly spread, and satisfy the separation requirement.

Chapter 3

Literature Review

In this chapter a literature review is presented. Section 3.1 presents a selection of relevant literature on RoRo-shipping. Section 3.2 and 3.3 presents relevant literature on voyage separation requirement and transit time requirement, respectively.

3.1 RoRo-shipping

Roll-on Roll-off (RoRo) shipping is a minor section of liner shipping, and is therefore not as well discussed in the literature as in other segments. In this section, a selection of Operations Research done in RoRo shipping the last decade is presented.

Sigurd et al. (2005a) present a network design problem for a set of routes between several ports, using fast RoRo vessels. Fagerholt et al. (2009) look into the fleet deployment problem in RoRo shipping, modeling voyages on a liner route with starting time windows. It is common to use a sequential approach when it comes to modeling speed in planning shipping routes. That is, the speed is first given, and then later optimized. Andersson et al. (2015) propose a new model for planning shipping routes. In their model the optimization of ship speed along the routes was integrated into

the model. This deployment and routing problem is studied using RoRo shipping.

Kang et al. (2012) and Jung et al. (2011) present a decision support system for a car carrier where the objective is to determine the number of cars each vessel should pick up on each route, at the same time minimizing cost. There is an extra cost added to the objective function for cars not serviced in one planning horizon. Lindstad et al. (2011) analyze carbon dioxide (CO2) emissions based on the RoRo segment in the world fleet, among others. Patricksson et al. (2015) propose a model of the maritime fleet renewal problem, with RoRo-shipping as their case study. This model also include emission regulations as a limitation.

Fischer et al. (2016) present different strategies to include robustness, as well as different strategies for handling disruption when assigning a fleet of vessels to predefined voyages while trying to minimize cost in RoRo-shipping.

3.2 Voyage separation requirements

Voyage separation requirements can be modeled as either hard or soft constraints. Hard constraints set conditions for variables that are required to be satisfied. Soft constraints have some variable values that are penalized in the objective function for not being satisfied.

One way of that the voyage separation has been modeled is by using time windows directly. Norstad et al. (2015) use data from the Norwegian shipping company Saga Forrest Carriers to model the voyage separation requirements both as hard constraints and as soft constraints. To enforce the separation requirements, they use a parameter that determines the minimum accepted time between two consecutive voyages on a trade route. Two different models are presented in the article; an a priori path generation method and an arc flow method, of which the path flow model performs best according to solution time. Both Bakkehaug et al. (2016) and Vilhelmsen et al. (2017) use the same data as Norstad et al. (2015). However, in Bakkehaug et al. (2016) the fleet has been expanded from 25 to 32 ships. In their

article, they propose an adaptive large neighborhood search heuristic for a ship routing and scheduling problem, where the voyage separation requirement is modeled as the minimum time elapsed between two consecutive sailings on a trade. As this model is based on a heuristic it is not an exact method. Vilhelmsen et al. (2017) base their method on a branch-and-price procedure and use a dynamic programming algorithm to generate columns. The voyage separation requirements are relaxed in the master problem. A time window branching scheme is used to enforce the separation requirements. This method is an exact method. Vilhelmsen et al. (2017) state, after comparing, that their model is significantly faster than Norstad et al. (2015)'s method.

For the aforementioned papers, the voyages are identical, i.e. visiting the same ports in the same order and sailing with the same speed. However, there is more planning flexibility in RoRo-shipping, and there can be differences between the voyages both to when to start each voyage as well as when and how often to visit each port along the trade. Hansen et al. (2018) claim that the STSRSP is a problem that is new to the research literature. However, in addition to the literature discussed above, the STSRSP has some similarities with other problems, where one needs to separate services for each customer in time. Such problems include the periodic vehicle routing problem (e.g. Campbell and Wilson (2014)) and the supply vessel planning problem (e.g. Kisialiou et al. (2018) and Borthen et al. (2017)), as well as the special liner shipping network design problem considered by Sigurd et al. (2005b).

3.3 Transit time requirements

In the STSRSP, we also consider that some contracts may impose transit time limits. To mimic real world problem in maritime transportation, it is important to implement transit time restrictions. Gelareh et al. (2010) present a hub and spoke network model where the demand between two ports is dependent on both transit time, as well as the price of shipping.

Both Álvarez (2012) and Wang and Meng (2011) deal with level of service by using transit time restrictions for transporting cargo. Álvarez

(2012) claims that cost minimization does not properly address the level of service. To account for this, he uses a linear function of the cargoes transit time through the liner shipping network. Wang and Meng (2011) also tries to explain the level of service by looking at the transit time of cargo that is transported.

Chapter 4

Mathematical models

4.1 Full voyage model

This mathematical method is based on the *voyage model* formulated in Hansen et al. (2018). This model gives a full description of the optimization problem. The model and notation can also be found in Appendix A.

4.1.1 Notation

Let $\mathcal{V}$ be the set of possible voyages during the planning horizon, and $\mathcal{N}^P$, indexed by i, be the set of ports in the network. C_i^V gives the cost of calling port i.Let $\mathcal{K}$ be the set of vessels indexed by k. Each vessel k has a starting position $o(k)$, an artificial ending position $d(k)$, and a graph $\mathcal{G}_k$ = ($\mathcal{N}_k$, $\mathcal{A}_k$) associated with it. All ports that can be visited by vessel k are included in $\mathcal{N}_k$, i.e. $\mathcal{N}_k^P$, including its starting position and artificial ending position, $\mathcal{N}_k \subseteq \mathcal{N}^P \cup \{o(k),\ d(k)\}$. The set of arcs $\mathcal{A}_k \subset \mathcal{N}_k \times \mathcal{N}_k$ define the feasible movements for vessel k.

Let $\mathcal{C}$ be the set of contracts indexed by c. $\mathcal{C}^E$ is the set of contracts with evenly spread requirements, while the set of contracts with transit time re-

quirements is given by $\mathcal{C}^T$. T_c^T gives the corresponding transit time of a contract c. Each contract has a given loading port $l(c)$ and an unloading port $u(c)$. $\mathcal{C}_i^L$ and $\mathcal{C}_i^U$ are the sets of cargoes that may be loaded at port i and unloaded at port i, respectively. Let $\mathcal{P}$ be the set of product types. Further, let $\mathcal{P}_p^S$ be the set of product types that can be stored in the same space as product type p. D_{cp} defines the total demand, in square meters, for product type p for contract c. $\underline{Q}_{cp}$ and $\overline{Q}_{cp}$ denotes the minimum and maximum quantity that has to be picked up when contract c is serviced, i.e. the quantity picked up by a vessel must be between [$\underline{Q}_{cp}$, $\overline{Q}_{cp}$]. Additionally, for the evenly spread requirements, the total number of pickups must be within the interval [$\underline{P}_c$, $\overline{P}_c$], i.e. the integer interval over the minimum and the maximum number of pickups for contract c.

C_k^C gives the daily charter rate for vessel k. To model the fuel consumption for each vessel, the piecewise linear approximation method proposed by Andersson et al. (2015) is used. Let $\mathcal{S}$ be the set of discrete speed alternatives indexed by s, ordered from low to high. The cost of sailing from a node i to a node j for vessel k using speed alternative s is denoted C_{ijks}^S, where as the sailing time is given by T_{ijks}^S. T_k^A defines the time vessel k becomes available at its origin. The handling time, i.e. the time used to load or unload one square meter of product type p on vessel k, is given by T_{kp}^H. Finally, the length of the planning period is given by T^{PH}.

Variables x_{ijv} define whether voyage v use the arc between nodes i and j or not. The binary variable y_{vk} defines whether vessel k sails voyage v or not, w_{ijvks} represents the weight of the speed alternative s for vessel k on the arc (i,j) on voyage v, and l_{ijvp} equals the load of product type p on the arc (i,j) on voyage v. The binary variable δ_{vc} defines whether voyage v serves contract c or not, and q_{vcp} represents the quantity of product type p picked up of contract c on voyage v. The time variable t_{iv} define the start of service at node i on voyage v.

4.1.2 The model

Objective function

$$\min z = \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}_k} \sum_{v\in\mathcal{V}} \sum_{s\in\mathcal{S}} C^{SC}_{ijks} w_{ijvks} + \sum_{(i,j)\in\mathcal{A}} \sum_{v\in\mathcal{V}} C^{V}_{i} x_{ijv} + \sum_{k\in\mathcal{K}} C^{C}_{k} t^{HW}_{k} \quad (4.1)$$

The objective function (4.1) is to minimize the total cost, which consists of the sailing and chartering costs associated with piloting and sailing, the costs associated with visiting ports, and the time charter costs for the handling and waiting time.

Network constraints

$$\sum_{k\in\mathcal{K}} \sum_{j\in\mathcal{N}^P_k \cup d(k)} x_{o(k)jv} = 1, \quad v \in \mathcal{V} \quad (4.2)$$

$$\sum_{i\in\mathcal{N}} x_{ijv} - \sum_{i\in\mathcal{N}} x_{jiv} = 0, \quad v \in \mathcal{V}, j \in \mathcal{N}^P \quad (4.3)$$

$$\sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{N}^P_k \cup o(k)} x_{id(k)v} = 1, \quad v \in \mathcal{V} \quad (4.4)$$

$$\sum_{j\in\mathcal{N}} \sum_{v\in\mathcal{V}} x_{o(k)jv} = 1, \quad k \in \mathcal{K} \quad (4.5)$$

$$\sum_{i\in\mathcal{N}} \sum_{v\in\mathcal{V}} x_{id(k)v} = 1, \quad k \in \mathcal{K} \quad (4.6)$$

Constraints (4.2) - (4.4) describe the network flow on a route for each voyage v. Constraints (4.5) and (4.6) state that each vessel must sail at most one time out from its origin and to its destination. These constraints are included to tighten the linear relaxation.

Vessel constraints

$$x_{ijv} = \sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}} w_{ijvks}, \quad (i,j) \in \mathcal{A}, v \in \mathcal{V} \tag{4.7}$$

$$\sum_{s \in \mathcal{S}} w_{ijvks} \leq y_{vk}, \quad k \in \mathcal{K}, (i,j) \in \mathcal{A}_k, v \in \mathcal{V} \tag{4.8}$$

$$\sum_{v \in \mathcal{V}} y_{vk} = 1, \quad k \in \mathcal{K} \tag{4.9}$$

$$\sum_{k \in \mathcal{K}} y_{vk} = 1, \quad v \in \mathcal{V} \tag{4.10}$$

Constraints (4.7) ensure that the weights of the speed alternatives add up to 1 if vessel k sails arc (i, j), or not. Constraints (4.8) ensure that a vessel k cannot sail the arc (i, j) on voyage v unless the vessel is assigned to the given voyage. Constraints (4.9) and (4.10) ensure that each vessel is assigned to a voyage and each voyage is sailed by one vessel, respectively.

Capacity constraints

$$0 \leq l_{ijvp} \leq \sum_{k \in \mathcal{K}} K^V_{kp} y_{vk} - \sum_{p' \in \mathcal{P}^S_p} l_{ijvp'}, \quad (i,j) \in \mathcal{A}, v \in \mathcal{V}, p \in \mathcal{P} \tag{4.11}$$

$$l_{ijvp} \leq M^C_p x_{ijv}, \quad (i,j) \in \mathcal{A}, v \in \mathcal{V}, p \in \mathcal{P} \tag{4.12}$$

$$\sum_{j \in \mathcal{N}} l_{jivp} + \sum_{c \in \mathcal{C}^L_i} q_{vcp} - \sum_{c \in \mathcal{C}^U_i} q_{vcp} = \sum_{j \in \mathcal{N}} l_{ijvp}, \quad i \in \mathcal{N}, v \in \mathcal{V}, p \in \mathcal{P} \tag{4.13}$$

$$\sum_{k \in \mathcal{K}} \sum_{j \in \mathcal{N}} l_{o(k)jvp} = 0, \quad v \in \mathcal{V}, p \in \mathcal{P} \tag{4.14}$$

Constraints (4.11) and constraints (4.12) ensure that the capacity limit of each product type on voyage v sailed by vessel k is respected. Constraints (4.13) ensure that the load on voyage v in node j equal the load in the previous node i adjusted for the quantity loaded and unloaded in node i. Constraints (4.14) define the initial load on voyage v.

Pickup constraints

$$\underline{P}_c \leq \sum_{v \in \mathcal{V}} \delta_{vc} \leq \overline{P}_c, \quad c \in \mathcal{C} \tag{4.15}$$

$$\delta_{vc} \leq \sum_{i \in \mathcal{N}} x_{il(c)v}, \quad v \in \mathcal{V}, c \in \mathcal{C} \tag{4.16}$$

$$\delta_{vc} \leq \sum_{i \in \mathcal{N}} x_{iu(c)v}, \quad v \in \mathcal{V}, c \in \mathcal{C} \tag{4.17}$$

Constraints (4.15) ensure that the number of pickups of an evenly spread contract is within the required interval. Constraints (4.16) and (4.17) ensure that a voyage v only service contract c if both the loading and unloading port for that contract is visited.

Loading constraints

$$\underline{Q}_{cp}\delta_{vc} \leq q_{vcp} \leq \overline{Q}_{cp}\delta_{vc}, \quad v \in \mathcal{V}, c \in \mathcal{C}, p \in \mathcal{P} \tag{4.18}$$

$$\sum_{v \in \mathcal{V}} q_{vcp} = D_{cp}, \quad v \in \mathcal{V}, p \in \mathcal{P} \tag{4.19}$$

Constraints (4.18) require that the quantity picked up from contract c is within a given interval. Constraints (4.19) ensure that the contracted demand is serviced.

Time constraints

$$t_{o(k)v} = T_k^A y_{vk}, \quad v \in \mathcal{V}, k \in \mathcal{K} \tag{4.20}$$

$$t_{iv} + \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}} T_{ijks}^S w_{ijvks} + T_{iv}^P x_{ijv} + \sum_{c\in\mathcal{C}_i^L\cup\mathcal{C}_i^U} \sum_{p\in\mathcal{P}} T_{kp}^H q_{vcp} \leq t_{jv}, \quad (i,j) \in \mathcal{A}, v \in \mathcal{V} \tag{4.21}$$

$$t_{l(c)v} + T_c^T + M_c^T(1-\delta_{vc}) \geq t_{u(c)v}, \quad v \in \mathcal{V}, c \in \mathcal{C} \tag{4.22}$$

$$t_{jv} - M_{jk}^S(1-x_{o(k)jv}) \leq T^{PH}, \quad j \in \mathcal{N}^P, v \in \mathcal{V}, k \in \mathcal{K} \tag{4.23}$$

$$t_k^{HW} \geq t_{d(k)v} - t_{o(k)v} - \sum_{(i,j)\in\mathcal{A}_k} T_{iv}^P x_{ijv} - \sum_{(i,j)\in\mathcal{A}_k} \sum_{s\in\mathcal{S}} T_{ijks}^S w_{ijvks} - M_k^L(1-y_{vk}), \quad v \in \mathcal{V}, k \in \mathcal{K} \tag{4.24}$$

$$\sum_{k\in\mathcal{K}} t_k^{HW} \geq \sum_{v\in\mathcal{V}} \sum_{i\in\mathcal{N}^P} \sum_{c\in\mathcal{C}_i^L\cup\mathcal{C}_i^U} \sum_{p\in\mathcal{P}} T_{kp}^H q_{vcp} \tag{4.25}$$

$$\sum_{i\in\mathcal{N}^P} \sum_{k\in\mathcal{K}} x_{o(k)i(v+1)} \leq \sum_{i\in\mathcal{N}^P} \sum_{k\in\mathcal{K}} x_{o(k)iv}, \quad v \in \mathcal{V}\backslash\{|\ \mathcal{V}\ |\}, c \in \mathcal{C} \tag{4.26}$$

Constraints (4.20) ensure that a a voyage v can start when the vessel k assigned to that voyage is available. Constraints (4.21) ensure that the time of starting service at a node j must be greater than or equal to the start of service at the previous node i, plus the sailing time between the nodes, the piloting time in node i, and the contract handling time. Constraints (4.22) ensure that the transit time restrictions are respected. Constraints (4.23) require that each voyage that is taken must visit the first port within the planning horizon. Constraints (4.24) sets the time each vessel uses on handling and waiting. Constraints (4.25) are used to tighten the formulation, and define a lower bound on the minimum time used to handle the contracts. Constraints (4.26) are symmetry breaking, and ensure that empty voyages are placed last in voyage ordering.

Variable constraints

$$x_{ijv} \in \{0,1\}, \quad (i,j) \in \mathcal{A}, v \in \mathcal{V} \tag{4.27}$$
$$\delta_{vc} \in \{0,1\}, \quad v \in \mathcal{V}, c \in \mathcal{C}^E \tag{4.28}$$
$$0 \leq w_{ijvks} \leq 1, \quad k \in \mathcal{K}, (i,j) \in \mathcal{A}_k, v \in \mathcal{V}, s \in \mathcal{S} \tag{4.29}$$

Constraints (4.27) and (4.28) put binary restrictions on the arc-flow and pickup variables, respectively. Constraints (4.29) require the speed variable w_{ijvks} to take values between 0 and 1.

Evenly spread constraints

In this section, additional notation for the evenly spread constraints are presented, as well as the evenly spread constraints.

Let the binary variable z_{vwc} define whether voyage w is the next voyage after voyage v, servicing contract c or not. If both voyage v and voyage w service contract c, i.e. z_{vwc} = 1, we will refer to the pair of voyages (v, w) as a spread pair. This means, that if a contract is picked up n times, there will exist (n - 1) spread pairs for contract c. Further, let ϕ_{nc} be 1 if contract c is picked up n times during the planning horizon, 0 otherwise. Variables s_c define the number of days contract c deviates from the evenly spread requirement. Finally, let L be the total maximum allowed deviation in days from the evenly spread requirement, i.e. the service level requirement.

$$\sum_{v\in\mathcal{V}}\sum_{w\in\mathcal{V}_v^S} z_{vwc} \geq \sum_{v\in\mathcal{V}} \delta_{vc} - 1, \quad c \in \mathcal{C}^E \tag{4.30}$$

$$\sum_{w\in\mathcal{V}_v^S} z_{vwc} \leq \delta_{vc}, \quad v \in \mathcal{V}, c \in \mathcal{C}^E \tag{4.31}$$

$$\sum_{w\in\mathcal{V}\setminus(\mathcal{V}_v^S\cup\{v\})} z_{vwc} \leq \delta_{vc}, \quad v \in \mathcal{V}, c \in \mathcal{C}^E \tag{4.32}$$

$$\sum_{n=\underline{P}_c}^{\overline{P}_c} n\theta_{nc} = \sum_{v\in\mathcal{V}} \delta_{vc}, \quad c \in \mathcal{C}^E \tag{4.33}$$

$$\sum_{n=\underline{P}_c}^{\overline{P}_c} \theta_{nc} = 1, \quad c \in \mathcal{C}^E \tag{4.34}$$

$$\sum_{n=\underline{P}_c}^{\overline{P}_c} \frac{T^{PH}\theta_{nc}}{n} - s_c - M_c^E(1 - z_{vwc}) \leq t_{l(c)w} - t_{l(c)v}, \quad v \in \mathcal{V}, w \in \mathcal{V}_v^S, c \in \mathcal{C}^E \tag{4.35}$$

$$\sum_{n=\underline{P}_c}^{\overline{P}_c} \frac{T^{PH}\theta_{nc}}{n} + s_c + M_c^E(1 - z_{vwc}) \leq t_{l(c)w} - t_{l(c)v}, \quad v \in \mathcal{V}, w \in \mathcal{V}_v^S, c \in \mathcal{C}^E \tag{4.36}$$

$$\sum_{c\in\mathcal{C}^E} s_c \leq L \tag{4.37}$$

Constraints (4.30) require that the sum of spread pairs for contract c is greater than or equal to the number of pickups minus 1. Constraints (4.31) and (4.32) require a voyage v to pick up contract c in order to be included in a spread pair, while also ensuring that a voyage v is present in at most two spread pairs for each contract. Constraints (4.33) and (4.34) ensure that θ_{nc} is 1 if contract c is picked up n times. Constraints (4.35) and (4.36) require that voyages v and w arrive at the loading port of contract c evenly spread . To correct for deviation from the desired spread, the spread slack variable s_c may take a positive value. Constraints (4.37) limits the sum of deviation for all evenly spread contracts.

Variable constraints

$$z_{vwc} \in \{0,1\}, \quad v \in \mathcal{V}, w \in \mathcal{V}_v^S, c \in \mathcal{C}^E \tag{4.38}$$

$$\theta_{nc} \in \{0,1\}, \quad c \in \mathcal{C}^E, n = \underline{P}_c..\overline{P}_c \tag{4.39}$$

$$s_c \geq 0, \quad c \in \mathcal{C}^E \tag{4.40}$$

Constraints (4.38) and (4.39) put binary restrictions on the spread pair and pickup-counter variables, respectively. Constraints (4.40) ensure that the spread deviations are non-negative.

Chapter 5

Matheuristic for the STSRSP

In this chapter the matheuristic proposed for solving the STSRSP is presented. Section 5.1 gives an overview of the matheuristic. Section 5.2 describes step 1 to 4 of the matheuristic, i.e. the process of determining the y_{vk}-variables from Chapter 4, while Section 5.3 describes step 5 of the matheuristic, i.e. determining the δ_{vc}-variables. Section 5.4 describes step 6 of the matheuristic, i.e. the fixing of variables, and the reduces model. Finally Section 5.5 presents step 7 of the matheuristic, i.e. the service level search.

The model presented in Section 4.1, aims to determine to assign vessels to voyages, determine which routes these vessels should sail at what speed, and which contracts to service so that all contract requirements regarding pick up frequency and quantities, transit times, and fairly evenly separation in time are satisfied at minimum cost. The model in 4.1 is an improved formulation of a previous model, which has shown that better formulations can decrease run time (Hansen et al., 2018). However, even with these improvements the run time is prohibitively large. With no evenly spread constraints, only 25 of 30 problem instances were solved to optimality within a maximum run time of 3 hours. With service level (evenly spread constraints active) the model solves 20 of 30 problem instances to optimality. In both cases, the model found solutions for nearly all problem instances (30 of 30 without evenly spread, 29 of 30 with.) The substantial run time of this mo-

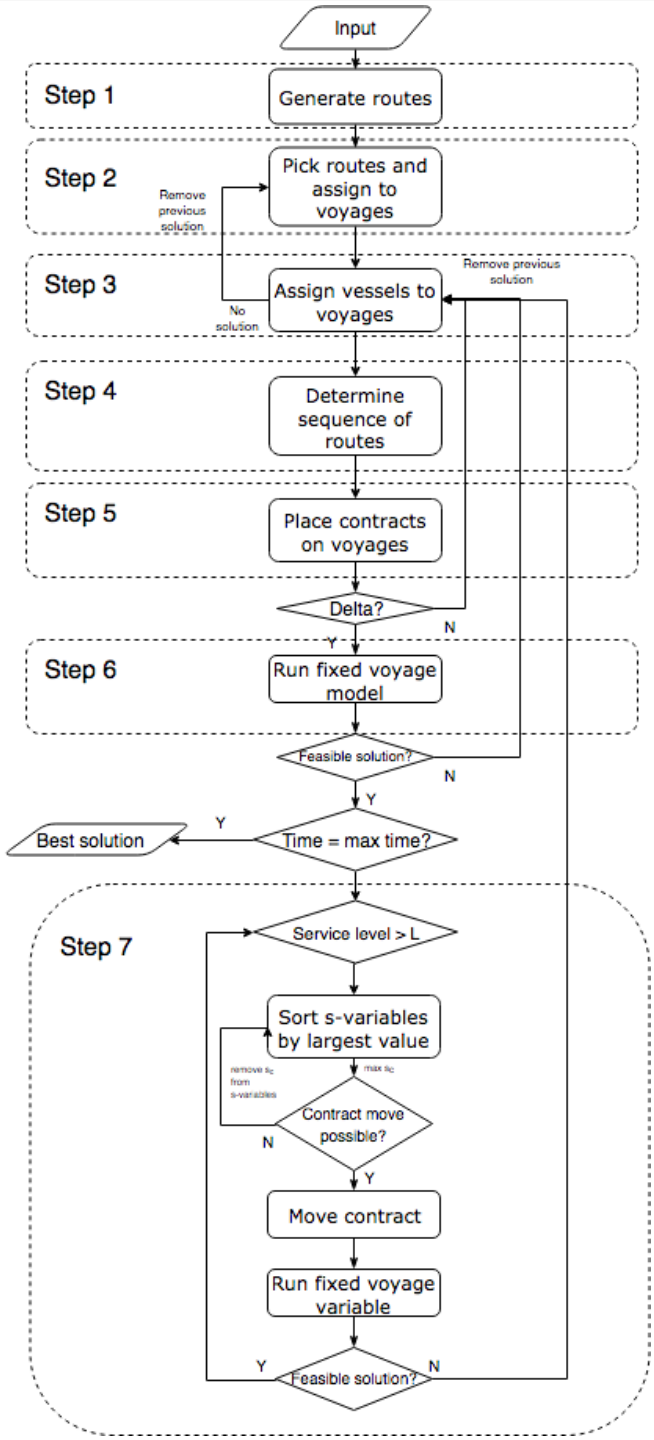
tivates the development of a matheuristic, which could allow the finding of approximate solutions that are good enough in practice, both with regards to solution quality and run time.

5.1 Matheuristic overview

The idea behind the matheuristic is to divide different parts of the model into separate steps in a sequential process. This way the matheuristic makes approximately good choices for each step and reduces the search space considerably on its way. Let us think of the search space as a tree, where each branch represents a solution. One way to think of the matheuristic is that it chooses the branch it considers the best solution, and evaluates only that branch. The matheuristic continues by iteratively evaluating the branches it considers are best, until it finds a solution that is good enough, hopefully without having to evaluate the entire solution tree. The matheuristic does this by at first generating all routes which can be a part of a feasible solution in step 1, and then combining them in plausibly good routing solutions by using a smaller optimization model in step 2. This routing solution is a set of routes, where the number of routes equal to the number of voyages. As there are high costs associated with calling ports, as well as extra sailing distance and time, determining which ports to call for each voyage is an essential decision for the RoRo-shipping companies. The model selects this set to minimize cost, and to ensure that contract pickup requirements can be fulfilled. In step 3 the matheuristic uses another smaller optimization model to find feasible vessel-voyage combinations for each voyage. In step 4 the matheuristic determines the sequence that the voyages will be performed in, to ensure that the vessels start in the optimal order in relation to the time they are available. In step 5 a greedy algorithm is used to decide which contracts should be picked up on which voyage, while ensuring quantity, capacity and transit time constraints. In step 6, the reduced model is run using the variables acquired in the earlier steps. If it cannot place contracts, or the model does not return a feasible solution, it returns to step 3. If step 3 does not yield a possible vessel combination. It returns to step 2. If it can place the contracts successfully, it fixes the variables in the original optimization problem corresponding to all choices above, and runs it. If it

returns a feasible solution, the objective value is compared with the current best objective value, and if it is an improvement, it replaces the current best solution. In step 7 the model checks the solution in relation to the evenly spread constraints, and if it is poorly spread the matheuristic iteratively tries to improve the solution by re-organizing or adding extra pickups for contracts with an evenly spread requirement. A flow chart for the matheuristic is presented in Figure 5.1.

Figure 5.1 Flow chart of the matheuristic



5.2 Step 1 - 4: Obtaining routes and assigning vessels

In this section, the process of obtaining, and assigning vessels, to voyages, i.e. steps 1 to 4, is described in detail. Section 5.2.1 describes step 1, the process of generating routes. The route model, from step 2, is given in Section 5.2.2. In Section 5.2.3, the assigning-vessels-to-voyages model, from step 3, is presented. Finally, Section 5.2.4 describes how the sequence of the routes are determined, i.e. step 4.

5.2.1 Step 1: Generating candidate routes

In step 1, all possible routes for the instance is generated. Routes which only has one port visit, only visits loading ports, or only visits unloading ports are removed from the set. The reason for this is that in any feasible problem solution, the model must unload what is loaded during a voyage. Therefore these routes cannot be part of a feasible solution, and can be removed to reduce run time. The route with no port visits are kept, as a feasible solution does not have to use all voyages

5.2.2 Step 2: Route model

The model presented in this section, constitutes step 2 in the matheuristic. The model takes the set of routes obtained in Section 5.2.1, and returns a combination of routes equal to the number of voyages that should be performed. Let us consider a combination of routes. For these routes to be feasible, all contractual requirements have to be fulfilled. For a given set of contractual requirements, there are usually many combinations of routes which can fulfill them. These solution will however not be equal in efficiency. If the contractual requirements for a problem can be fulfilled with fewer than the allowed voyages, and optimized routes on the voyages that are sailed, costs will in general be much lower. That is, efficient routing is one of the most important drivers of cost for this problem, which is a reason

that route generation and selection is the first step of this matheuristic. It also makes sense for this to be the first step considering the idea behind the matheuristic. If the first step was assigning contracts to voyages (for example), this step would also implicitly have to create routes, consider capacity, load quantity, evenly spread, and transit time restriction as well as vessel assignment. This would require a much more sophisticated algorithm and would be a sub problem closer in complexity to the original problem . By fixing the routes, the matheuristic can use fast and relatively simple algorithms to assign vessels, to optimize the contract assignment for that route, and to improve evenly spreading in separate steps. This allows for a faster search of the solution space for feasible, and possibly good enough solutions.

The objective of the route model is to minimize the distance sailed, while ensuring that every loading and unloading port specified in the contracts are visited at least as many times as the minimum pickup frequency.

Notation

Let $\mathcal{R}$ be the set of possible routes, from Section 5.2.1, that can be sailed on a given trade route, indexed by r. C_r defines the total distance of sailing route r. Let A_{cr} define whether route r visits both the loading and the unloading port of contract c, and A^T_{cr} define whether route r visits both the loading and unloading port within the transit time requirement of contract c. The sailing time between ports is calculated using the fastest vessel at the highest speed. Let $|\mathcal{V}|$ be the total number of voyages. Let integer variables x_r define how many, if any, times route r is sailed.

The model

$$\min \sum_{r \in \mathcal{R}} C_r x_r \tag{5.1}$$

$$\sum_{r \in \mathcal{R}} A_{cr} x_r \geq \underline{P}_c, \quad c \in \mathcal{C} \tag{5.2}$$

$$\sum_{r \in \mathcal{R}} A^T_{cr} x_r \geq \underline{P}_c, \quad c \in \mathcal{C}^T \tag{5.3}$$

$$\sum_{r \in \mathcal{R}} x_r = |\mathcal{V}| \tag{5.4}$$

$$x_r \geq 0, \text{and integer} \quad r \in \mathcal{R} \tag{5.5}$$

The objective function (5.1) is to minimize the total cost; the sum of the distance of sailing route r. Constraints (5.2) ensure that the routes visit the loading and unloading port of a contract at least as many times as the minimum pickup requirement. Constraints (5.3) ensure that the routes visit the loading and unloading port within the transit time requirement of a contract at least as many times as the minimum pickup requirement. Constraints (5.4) ensure that the sum of routes sailed is equal to the total number of voyages.

Removing previous solution from the set of possible routes

To ensure that the model provides new solutions every time the route model is solved, the previous route solution is removed from the set by adding a new constraint. Let $\mathcal{R}'$ be the set of predetermined routes from the route model.

$$\sum_{r \in \mathcal{R}'} x_r \leq |V| - 1 \tag{5.6}$$

5.2.3 Step 3: Assigning vessels to routes model

In this section, vessels are assigned to the predetermined routes from Section 5.2.2, i.e. step 3. This is an important part of the heuristic, since the product capacity of a vessel determines which contracts, as well as how many contracts, can be serviced on a voyage. The vessels also have different sailing costs, which affects the total cost.

New notation

Let $\mathcal{R}'$ be the set of predetermined routes from Section 5.2.2, and C_{rk} be the estimated cost of sailing route r with vessel k. Let binary variables x_{rk} define whether vessel k is assigned to route r, or not.

The model

$$\min \sum_{k \in \mathcal{K}} \sum_{r \in \mathcal{R}'} C_{rk} x_{rk} \tag{5.7}$$

$$\sum_{r \in \mathcal{R}'} x_{rk} = 1, \quad k \in \mathcal{K} \tag{5.8}$$

$$\sum_{k \in \mathcal{K}} x_{rk} = 1, \quad r \in \mathcal{R}' \tag{5.9}$$

$$\sum_{k \in \mathcal{K}} \sum_{r \in \mathcal{R}'} A_{cr} K^V_{kp} x_{rk} \geq D_{cp}, \quad p \in \mathcal{P}, c \in \mathcal{C} \tag{5.10}$$

$$\sum_{k \in \mathcal{K}} \sum_{r \in \mathcal{R}'} A^T_{cr} K^V_{kp} x_{rk} \geq D_{cp}, \quad p \in \mathcal{P}, c \in \mathcal{C}^T \tag{5.11}$$

$$x_{rk} \in \{0, 1\}, \quad r \in \mathcal{R}', k \in \mathcal{K} \tag{5.12}$$

The objective function (5.7) is to minimize the cost of sailing the predetermined routes. Constraints (5.8) ensure that a vessel takes exactly one route, while constraints (5.9) ensure that a route is assigned to only one vessel. Constraints (5.10) ensure that the vessels that service contract c has enough capacity to satisfy the demand for contract c. Constraints (5.11) ensure that the vessels that service contract c has enough capacity to satisfy the demand for contract c, as well as satisfy the transit time requirements.

Removing previous solution from the set

To ensure that the model provides new solutions every time the assigning-vessels-to-routes model is solved, the previous route-vessel solution is re-

moved from the set by adding a new constraint. Let X_{rk} be the set of route-vessel pairs obtained from running the assigning-vessels-to-routes model.

$$\sum_{r \in \mathcal{R}'} x_{rk} \leq |V| - 1, \quad k \in \mathcal{K} \tag{5.13}$$

5.2.4 Step 4: Determining the sequence of routes

The vessels can be available at the start of the planning horizon, or become available during the planning horizon, due to duties on other trades. A vessel cannot arrive at a port before a vessel on an earlier voyage, if both vessels service that port. Therefore, to reduce the waiting time for each vessel, the sequence of the routes is determined based on when the vessels assigned to the routes, are available. The vessel that is first available, along with the corresponding route, is assigned the first voyage, the second vessel that is available is assigned to the second voyage, and so on.

5.3 Step 5: Placing contracts on voyages

In this section, the process of deciding which voyages will service which contracts, i.e. fixing the δ_{vc}-variables, of step 5 of the matheuristic is described. Section 5.3.1 describes how the sequence of contracts to be placed on voyages is determined. How the contracts are placed on voyages, and the repair functions to fix load or transit time requirements, are described in Subsection 5.3.2 and 5.3.3, respectively.

A crucial part of the matheuristic is to place the contracts on the voyages obtained in Section 5.2. There are several requirements that has to be fulfilled for a voyage to be able to service a contract. First of all, the voyage has to visit both the loading and unloading port of that contract. Due to the inherent planning flexibility in RoRo-shipping, the voyages do not necessarily visit all the ports on the trade. Second, the vessel assigned to the voyage has to have enough capacity to service at least the minimum

required quantity to be picked up, of the product type specified in the contract. The vessels have different capacities, and this also restricts which, as well as how many, contracts that can be placed on the corresponding voyage. In addition, any transit time requirements of the contracts have to be fulfilled. If the δ_{vc}-variables violate any of these requirements, the problem will be infeasible.

5.3.1 Sorting contracts

Each contract is sorted lexicographically by four properties: product type, number of route options, transit time requirement, and quantity. This is done, so that the contracts that have the tightest restrictions, or have the fewest options in terms of which voyages can service that contract, are placed first.

Product type

All contracts are sorted by product types, where the product type that has the fewest storage options is placed first. E.g. if product type A can only be stored in space facilitated for product type A, and product type B can be stored both in space facilitated for product type B, and space facilitated for product type A. Then product type A will be placed before product type B.

Route options

The contracts are then sorted based on the number of routes that can service the contract, i.e. routes that visit both the loading port and the unloading port of the contract, minus the minimum number of times the contracts has to be picked up, i.e. the minimum number of routes that have to service the contract. So if a contract has three possible routes that can service it, and a minimum number of pickups equal to two, then the contract will have one route option.

Transit time requirement

Next, the contracts are sorted by transit time requirements, where the contracts with lowest transit time requirement are placed first. Contracts that do not have any transit time requirements, are placed last.

Quantity

Lastly, the contracts are sorted by quantity, i.e. the minimum quantity to be picked up every time, i.e. $\underline{Q_{cp}}$, where the contracts with the largest quantity are placed first.

Example

To illustrate how the contracts are sorted, a small example is provided. Here, product type A can only be stored on space facilitated for product type A, while product type B can be stored on space facilitated for product type B or product type A. A "-" in the transit time req.-column indicates that the contract has no transit time requirement. Table 5.1 shows the, unsorted, contracts with the corresponding four properties.

Table 5.1: Example contracts - unsorted

Contract	Product type	Route options	Transit time req.	$\underline{Q_{cp}}$
1	A	0	-	2100
2	B	0	10	1700
3	B	1	-	2400
4	A	1	13	3000
5	B	0	17	1400
6	B	1	-	2300

Contracts 1 and 4 both contain product type A, and therefore placed before contracts 2, 3, 5, and 6, which contain product type B. Next, contract

1 is placed before contract 4, and contracts 2 and 5 are placed before contracts 3 and 6, as contracts 2, 4, and 5 have one extra route option, while contracts 1, 3, and 6 have none, i.e. contracts 1, 3, and 6 have to be placed on all the voyages that visit both the unloading and unloading port of that contract. Since, both contracts 2 and 5 have transit time requirements, the contract with the lowest transit time, contract 2, will be placed first. Lastly, since neither contract 3 nor 6, have transit time requirement, the sequence is determined by the size of the load. Therefore, contract 3, which has a higher load, is placed before contract 6. Table 5.2 below shows the sorted contracts.

Table 5.2: Example contracts - sorted

Contract	Product type	Route options	Transit time req.	$\underline{Q_{cp}}$
1	A	0	-	2100
4	A	1	13	3000
2	B	0	10	1700
5	B	0	17	1400
3	B	1	-	2400
6	B	1	-	2300

5.3.2 Placing contract

This section describes how a contract is placed on a voyage, i.e. setting the δ-variables. The spread, or the number of pickups, of a contract is determined by the minimum number of pickups required. The spread defines how many voyages a contract should be placed on.

As mentioned above, when placing a contract on a voyage, there are several requirements to be met. Therefore, three different checks have been implemented to increase the possibility of getting feasible δ-variables: check port visits, check capacity, and check transit time requirement. In addition, the sorting of the contracts in Section 5.3.1, is used to place the contracts with fewest possible options, or most requirements, first, as these would be the most difficult contracts to place. If all these requirements are fulfilled, the contract can be placed on the corresponding voyage. A pseudo code for placing contracts is given in Algorithm 1.

Algorithm 1 Pseudo code for placing the contracts on voyages

```
Input: Contracts
Output: δ_vc
contracts ← sorted(contracts)              ▷ Sort contracts - see section 5.3.1
 while contracts not empty do
    c ← pop(contracts)
     placed = 0
     placed_bound ← minimum_pickup_frequency(contract)
     possible_voyages ← get_possible_voyages(voyages)          ▷ Returns
     voyages that satisfy port visit req. for contract c
     while placed < placed_bound and possible_voyages not empty do
        v ← get_voyage(possible_voyages)  ▷ Returns the voyage with most
         available space
         if check_capacity(v, c) and check_transit_requirement(v, c) then
            δ_vc = 1                                  ▷ Place contract c on voyage v
             placed += 1
        possible_voyages.remove(v)
    end
end
```

Check port visits

The voyage has to visit both the loading and the unloading port of the contract, to be able to service that contract. Which ports a voyage visits is determined in Section 5.2. These routes are used as a guide when placing contracts, to determines if a voyage can service a contract, or not. This check returns a set of all possible voyages that can service the contract in terms of visiting the loading and unloading ports.

Check capacity

A vessel has to have capacity to pick up at least the minimum quantity of product p for contract c, $\underline{Q}_{cp}$, if the number of pickups is larger than one, to be able to service the contract. If the contract is only picked up once, the vessel has to have capacity to pick up the total demand of that contract, D_{cp}. This quantity is referred to as the load of the contract. It is important to note that the load used in this chapter is just a minimum quantity to be picked up, and does not necessarily reflect the real load of the product, after solving the model. The capacity check is performed by checking if the vessel assigned to the voyage has enough capacity for the contract, as well as all the other contracts already placed on that voyage.

Check transit time requirements

The transit times requirements give the maximum time allowed from the load port to the unload port for the contract, i.e. the sum of the maximum sailing time, piloting, handling, and waiting times. The sailing time is calculated by using the highest speed alternative. The handling time is based on the load of the product for that contract.

5.3.3 Repair functions

The transit time requirement check in Section 5.3.2 is performed without any information about the contracts that have not already been placed. Therefore, an extra repair function, described below, is implemented to fix any violations of transit time requirements.

Repair transit time requirements

Since the handling time depends on the the amount of product being loaded or unloaded at the ports, the check in Subsection 5.3.2 is insufficient for some contracts, as it only takes into consideration the load of that particular contract. Therefore, an extra check is performed after all contracts are placed. For each contract with transit time requirement, the sailing time is calculated using the highest speed alternative. Then the handling time is calculated for each visited port from, and including, the load to the unload port, based on the loads of the contracts that are placed on that voyage. Lastly, the node time for each port visited is added to the total time. If the total time exceed the transit time requirement, the contract is added to a list of violating contracts.

For each violating contract in the list, one of the following procedures is done. The first procedure checks if the violating contract can be moved to another viable voyage that has a lower total time from the loading port to the unloading port. If there is no other viable voyage, or no viable voyage with lower total time, the second procedure checks if other contracts on the same voyage can be moved to another voyage. This is done in order to try too reduce the handling time on the voyage.

5.4 Step 6: Solving the reduced model with fixed variables

In this section, step 6 of the matheuristic is presented. Section 5.4.1 describes how the variables are fixed, while the reduced model, i.e. the mathematical model from Chapter 4 after fixing the variables, is presented in Section 5.4.2.

5.4.1 Preproccessing

The x-, z- and ϕ-variables are calculated based on the δ-variables from Section 5.3, and the y-variables from Section 5.2. The routes from Section 5.2 are just used as a guide for placing the contracts, and does not directly affect the x-variables. To calculate the x-variables, i.e. which ports are visited on a voyage, the δ-variables are used to find which contracts are placed on each voyage. Each contract has a specific load and unload port associated with it, and the voyage has to visit both ports of all the contracts that the voyage is servicing. The voyage does not visit any port that is not either a loading or unloading port of a contract assigned to that voyage.

From the δ-variables, the z- and ϕ-variables are easily obtained. For each contract the voyages servicing that contract, is added to a list. The list of voyages is then sorted, and counted. The z-variables define whether a voyage is the next voyage to service a contract after another voyage, and are based on the sequence of the voyages in the list. The ϕ-variables define how many times a contract is picked up during the planning period, and is based on the count of the list of voyages that service the contract.

5.4.2 The reduced model

When fixing the x-, z-, ϕ-, δ-, and y-variables, (4.2) - (4.5), (4.9), (4.10), (4.15) - (4.17), (4.27), (4.28), (4.30) - (4.34), (4.38), and (4.39) are redundant, and can be removed from the model. This leaves the LP-model below.

Objective function

$$\min z = \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}_k} \sum_{v\in\mathcal{V}} \sum_{s\in\mathcal{S}} C^{SC}_{ijks} w_{ijvks} + \sum_{(i,j)\in\mathcal{A}} \sum_{v\in\mathcal{V}} C^{V}_{i} X_{ijv} + \tag{5.14}$$

$$\sum_{k\in\mathcal{K}} C^{C}_{k} t^{HW}_{k} + \sigma \sum_{c\in\mathcal{C}^{\mathcal{E}}} s_c \tag{5.15}$$

Vessel constraints

$$X_{ijv} = \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}} w_{ijvks}, \quad (i,j)\in\mathcal{A}, v\in\mathcal{V} \tag{5.16}$$

$$\sum_{s\in\mathcal{S}} w_{ijvks} \leq Y_{vk}, \quad k\in\mathcal{K}, (i,j)\in\mathcal{A}_k, v\in\mathcal{V} \tag{5.17}$$

Capacity constraints

$$0 \leq l_{ijvp} \leq \sum_{k\in\mathcal{K}} K^{V}_{kp} Y_{vk} - \sum_{p'\in\mathcal{P}^{S}_{p}} l_{ijvp'}, \quad (i,j)\in\mathcal{A}, v\in\mathcal{V}, p\in\mathcal{P} \tag{5.18}$$

$$l_{ijvp} \leq M^{C}_{p} X_{ijv}, \quad (i,j)\in\mathcal{A}, v\in\mathcal{V}, p\in\mathcal{P} \tag{5.19}$$

$$\sum_{j\in\mathcal{N}} l_{jivp} + \sum_{c\in\mathcal{C}^{L}_{i}} q_{vcp} - \sum_{c\in\mathcal{C}^{U}_{i}} q_{vcp} = \sum_{j\in\mathcal{N}} l_{ijvp}, \quad i\in\mathcal{N}, v\in\mathcal{V}, p\in\mathcal{P} \tag{5.20}$$

$$\sum_{k\in\mathcal{K}} \sum_{j\in\mathcal{N}} l_{o(k)jvp} = 0, \quad v\in\mathcal{V}, p\in\mathcal{P} \tag{5.21}$$

Loading constraints

$$\underline{Q}_{cp}\Delta_{vc} \leq q_{vcp} \leq \overline{Q}_{cp}\Delta_{vc}, \quad v \in \mathcal{V}, c \in \mathcal{C}, p \in \mathcal{P} \tag{5.22}$$

$$\sum_{v \in \mathcal{V}} q_{vcp} \leq D_{cp}, \quad v \in \mathcal{V}, p \in \mathcal{P} \tag{5.23}$$

Time constraints

$$t_{o(k)v} = T_k^A Y_{vk}, \quad v \in \mathcal{V}, k \in \mathcal{K} \tag{5.24}$$

$$t_{iv} + \sum_{k \in \mathcal{K}} \sum_{s \in \mathcal{S}} T_{ijks}^S w_{ijvks} + T_{iv}^P X_{ijv} + \sum_{c \in \mathcal{C}_i^L \cup \mathcal{C}_i^U} \sum_{p \in \mathcal{P}} T_{kp}^H q_{vcp} \leq t_{jv}, \quad (i,j) \in \mathcal{A}, v \in \mathcal{V} \tag{5.25}$$

$$t_{l(c)v} + T_c^T + M_c^T(1 - \Delta_{vc}) \geq t_{u(c)v}, \quad v \in \mathcal{V}, c \in \mathcal{C} \tag{5.26}$$

$$t_{jv} - M_{jk}^S(1 - X_{o(k)jv}) \leq T^{PH}, \quad j \in \mathcal{N}^P, v \in \mathcal{V}, k \in \mathcal{K} \tag{5.27}$$

$$t_k^{HW} \geq t_{d(k)v} - t_{o(k)v} - \sum_{(i,j) \in \mathcal{A}_k} T_{iv}^P x_{ijv} - \sum_{(i,j) \in \mathcal{A}_k} \sum_{s \in \mathcal{S}} T_{ijks}^S w_{ijvks} - M_k^L(1 - Y_{vk}), \quad v \in \mathcal{V}, k \in \mathcal{K} \tag{5.28}$$

$$\sum_{k \in \mathcal{K}} t_k^{HW} \geq \sum_{v \in \mathcal{V}} \sum_{i \in \mathcal{N}^P} \sum_{c \in \mathcal{C}_i^L \cup \mathcal{C}_i^U} \sum_{p \in \mathcal{P}} T_{kp}^H q_{vcp} \tag{5.29}$$

Evenly spread constraints

$$\sum_{n=\underline{P}_c}^{\overline{P}_c} \frac{T^{PH}\theta_{nc}}{n} - s_c - M_c^E(1 - Z_{vwc}) \leq t_{l(c)w} - t_{l(c)v}, \quad v \in \mathcal{V}, w \in \mathcal{V}_v^S, c \in \mathcal{C}^E \tag{5.30}$$

$$\sum_{n=\underline{P}_c}^{\overline{P}_c} \frac{T^{PH}\theta_{nc}}{n} + s_c + M_c^E(1 - Z_{vwc}) \leq t_{l(c)w} - t_{l(c)v}, \quad v \in \mathcal{V}, w \in \mathcal{V}_v^S, c \in \mathcal{C}^E \tag{5.31}$$

$$\sum_{c \in \mathcal{C}^E} s_c \geq L \tag{5.32}$$

Variable constraints

$$0 \leq w_{ijvks} \leq 1, \quad k \in \mathcal{K}, (i,j) \in \mathcal{A}_k, v \in \mathcal{V}, s \in \mathcal{S} \tag{5.33}$$

$$s_c \geq 0, \quad c \in \mathcal{C}^E \tag{5.34}$$

The objective function (5.14) replaces (4.1), where a penalty function is added to reduce the total deviation in days from the evenly spread requirement for contracts. Constraints (5.16) and (5.17) corresponds to (4.6) and (4.8). Constraints (5.18) - (5.21) replaces (4.11) - (4.14). Constraints (5.22) - (5.29) replaces (4.18) - (4.25). Constraints (5.30) - (5.32) corresponds to (4.35) - (4.37). Constraint (5.32) replaces (4.37). Constraint (5.33) and (5.34) is equivalent to (4.29) and (4.37), respectively.

If the penalty, for deviation from the evenly spread requirement, is high enough the model will prioritize reducing the deviation, even if that means a higher cost. However, the service level requirement is defined by the service level threshold L, which describes how much deviation is allowed. Reducing the deviation further is not necessary.

Let us consider a small problem, where the service level threshold L is 15 days. Solution 1 has a total deviation of 15 days, i.e. $\sum_{c \in \mathcal{C}^E} s_c$ = 15, and

a total cost, without penalty, of 1000$. Solution 2 has a total deviation of 10 days, and a cost of 1200$. Since both the solutions, have an acceptable service level, the solution with the lowest cost would be preferable, i.e. solution 1. However, if the deviation penalty function σ is set to 100, then the total cost, with penalty, of solution 1 would be $1000 + \sigma * \sum_{c \in \mathcal{C}^E} s_c = 1000 * 100 * 15 = 2500$. The total cost of solution 2, would amount to $1200 + 100 * 10 = 2200$, which would be preferred over solution 1.

By replacing the constraint (5.31) with constraint (4.38), there is a possibility that a solution will have total deviation larger than the service level threshold. Therefore, a service level search, described in Section 5.5 is implemented, to correct the deviation.

5.5 Step 7: Service level search

This section describes the service level search performed in step 7. If the LP-model returns a solution, the total deviation in days for all contracts from the evenly spread requirement, i.e. $\sum_{c \in \mathcal{C}^E} s_c$, is calculated. If the total deviation is higher than the upper limit, L, the service level search is performed.

The service level search is performed by sorting the contracts by size of the deviation in descending order. The contract with the highest deviation is then chosen from the list, and if possible, the contract is added to another voyage, increasing the number of pickups of the contract by one. If adding an extra pickup to that contract, the contract with the second highest deviation is chosen, and is then tried to be placed on another voyage. This continues until a contract is placed on an extra voyage, or until there are more voyages to try. If a pickup is added to a contract the matheuristic returns to step 6, with the new δ_{vc}. Algorithm 2 shows the pseudo code for the service level search.

Algorithm 2 Pseudo code for the service level search

Input: s_c, voyages, delta
Output: δ_{vc}
while *solution and* $sum(s_c) \geq L$ **do**
    s $\leftarrow$ sorted s_c by descending order
    s_{max} = pop(s)
    if s_{max} *can be moved* **then**
        $\delta_{vc} \leftarrow$ move_contract(s_{max})
        s_c, solution $\leftarrow$ voyage_model(voyages, δ_{vc})
    end
end

Chapter 6

Computational Study

In this chapter, the results from running the heuristic described in Chapter 5 is presented. Section 6.1.1, describes the instances used for testing, while Section 6.2 presents the results. In Section 6.3, an analysis of the matheuristic is presented.

To better compare the matheuristic and the exact method, the mathematical model in Chapter 4 was implemented using the modeling language OPL, and solved with the optimization software Cplex. The matheuristic has been coded in Python, and PyCharm has been the integrated development environment. All computational experiments have been run on a MacBook Pro with Intel Core i5 processor and 16 GB RAM running High Sierra.

6.1 Test instances

To evaluate the performance and capabilities of the heuristic presented in Chapter 5, the same instances as in Hansen et al. (2018) have been used. Section 6.1.1 describes how the instances are created, while Section 6.1.2 describes the instance naming scheme.

6.1.1 Creating instances

The instances used in this thesis is based on a combination of real data and generated data. Each instance has been created using one of three different trade routes that are geographically similar to trade routes sailed by shipping companies. The difference is in the number of port visits made during voyages over these trade routes. The instances has been created using the following trade routes: Asia - Europe, Europe - US, US - Japan. These trade routes have five, ten, and fifteen ports, respectively. For each trade route, two different sets of instances are created. The first set consists of instances with 50 contracts which must be serviced during the planning horizon, while the second set consists of instances with 100 contracts. The total volume of goods to be transported during the planning horizon is based on real data from a shipping company.

For all instances 40% of the contracts comes with evenly spread requirements, 20% of the contracts comes with transit time requirements and 40% of contracts comes without any service requirements. Further, 90% of the contracts have loading and unloading ports randomly drawn from different regions (for example, loading in Europe and unloading in the US), while 10% of the contracts have loading and unloading ports randomly drawn from the same region (for example, loading and unloading in Europe.) All voyages must begin within time window of the planning horizon, which is set to 30 days for all instances.

The following contract data is generated randomly for all instances, and is drawn from uniform distributions. A random number is drawn, between 0.05 and 1 for ordinary contracts and 0.2, 1.5 for evenly spread contracts. These numbers are then normalized, meaning that this number is the contract demand to total demand ratio for each contract. Using these ratios, total demand is distributed among each of the contracts. Each contract has a service frequency requirement, which is drawn from the following integer interval: [1, minimum number of required voyages]. The minimum number of required voyages is calculated by finding the sum of all contractual demand over the entire planning horizon, and dividing it by the capacity of the largest vessel (rounding it up to the nearest integer.) That is, the amount of voyages necessary if all vessels had equal capacity as our largest vessel. For the contracts with transit time requirements, transit time limits are

chosen in the following interval: [minimum sailing time 1.2, maximum sailing time without waiting]. For a contract with loading port A and unloading port B, the minimum sailing time is the time it takes to sail directly from A to B at the highest possible speed, while the maximum sailing time without waiting is the time it takes to sail from A to B while visiting all ports between A and B at the lowest possible speed.

For contracts with evenly spread requirements, the minimum and maximum quantity bounds are created with by using equation 6.1 and 6.2, which constrains the minimum quantity picked up for a contract to be the contract demand divided by the pickup frequency for that contract, multiplied by a factor of 80% to allow some degree of freedom. Similar for the maximum quantity, except that it is multiplied by a factor of 120%, to create a quantity window.

$$Q_{cp} = 0.8D_{cp}/\bar{P}_c \tag{6.1}$$
$$\bar{Q}_{cp} = 1.2D_{cp}/P_c \tag{6.2}$$

For all other types of contracts, the minimum and maximum quantity bounds are created by using equation 6.3 and 6.4.

$$Q_{cp} = D_{cp}/|\mathcal{K}| \tag{6.3}$$
$$\bar{Q}_{cp} = D_{cp} \tag{6.4}$$

Loading and unloading ports are drawn in such a manner that their distribution over the contracts reflect the actual, typical demand in these ports. Port visit costs are uniformly drawn from the following interval: [25 000 USD, 40 000 USD]. Vessel characteristics, such as speed-dependent fuel consumption functions, stowage capacities, and estimations of time charter rates are provided by a shipping company, and each instance have a randomly selected subset of these vessels.

6.1.2 Naming of the instance

The instances are grouped into sets based on the size (that is, the number of ports) of the trade route, and how many contracts to be serviced during the planning period, see Table 6.1. For example, M-50 describes a set of five instances on the medium trade, i.e. the Asia - Europe trade, with 50 contracts

Table 6.1: Sets of test instances

Instance set	Trade route	Contracts
S-50	US - JAPAN	50
S-100	US - JAPAN	100
M-50	ASIA - EUROPE	50
M-100	ASIA - EUROPE	100
L-50	EUROPE - US	50
L-100	EUROPE - US	100

To differentiate between the instances within a group, an index N is assigned to each instance. An instance is identified by its group name, Size-Contracts (S-C), as well as the index (N), see Table 6.2. For example, M-50-0 is the first instance on the medium trade, with 50 contracts.

Table 6.2: Instance naming scheme

Instance name	S-C-N	
where		
	S - Size of the trade route	$\in [S, M, L]$
	C - Number of contracts	$\in [50, 100]$
	N - Index to separate instances in the same set	$\in [0, 9]$

6.1.3 Service level requirement

The service level requirement is defined by the service level threshold L. The service level used in this thesis corresponds to the medium service level in Hansen et al. (2018). The medium service level threshold is set as $L^M = L^H + (L^N - L^H)/3$, where L^N is the post calculated threshold with no service level requirement. L^H, is set by solving the STSRSP where the objective function is replaced by $L^H = \sum_{c \in \mathcal{C}} s_c$.

6.2 Results

In Section 6.2.1, the results from running the matheuristic and the model with no service level requirement is presented. In Section 6.2.2, the results from running the matheuristic and the model with service level requirement is presented.

6.2.1 Without service level requirement

In this section the heuristic is run with no service level requirement. The heuristic was run once for 100 seconds and once for 1000 seconds for all instances. A summary of the test results can be found in Table 6.3.. Each of the six sets of instances are represented by separate rows in the table, and each set contains 5 instances of the same type. The heuristic results are compared with the solution obtained by running the model with the same time limit as the heuristic. The table also shows the number of instances where a feasible solution was obtained, as well as the average gap (%) for those instances. Note that for some cases the gaps are missing. This is because the the instances in those cases could not be solved to optimality within a reasonable run time, and therefor no solution comparison could be made. Complete results from running the heuristic with no service level requirement, can be found in Appendix B.

Table 6.3: Number of instances solved and average gap (%) compared to the optimal solution, for the instance sets by running the matheuristic and model

Set of instances	Model				Matheuristic			
	100 sec	Gap (%)	1000 sec	Gap (%)	100 sec	Gap (%)	1000 sec	Gap (%)
S-50	5/5	0%	5/5	0%	5/5	0,20%	5/5	0,20%
S-100	5/5	0%	5/5	0%	5/5	0,26%	5/5	0,26%
M-50	5/5	0,97%	5/5	0%	5/5	1,65%	5/5	1,65%
M-100	5/5	9,72%	5/5	0%	5/5	1,20%	5/5	1,20%
L-50	4/5	NA	5/5	NA	0/5	NA	1/5	6,41%
L-100	2/5	NA	5/5	NA	1/5	0,48%	2/5	0,48%

For both the smallest and the medium instances, we can see that feasible solutions are found fairly quickly by the heuristic. There is no gain by running the heuristic for a longer period of time. We can also see that solutions are fairly close to optimal. However, for the largest instances the heuristic struggles to find feasible solutions within the time limit, finding feasible solutions to only 1 of the 10 instances within 100 seconds, and only 3 of the 10 within 1000 seconds. The average gap (%) for the instances that the heuristic manages to solve is within 0.23% for the smallest instances, 1.43% for the medium instances, and 3.45% for the large instances. It is important to note that the average gap for the largest instances is based only on three solutions.

We can also see that the model outperforms the heuristic in finding feasible solutions for many of the test cases with the same run time. The main reason for this will be discussed below. However, it is important to understand that although the model finds very good feasible solutions, sometimes even optimal, within short run time, they are not proven to be optimal. The model would have to run for a considerable longer time to prove that these are optimal solutions. And as the size of the instances grows, the model is struggling with finding even feasible non-optimal solutions within the short time limit. These instances are still considered small compared to the large real instances, so all though running the model for a short while can result in good solutions for small, unrealistic instances, it is not a practical solution.

Table 6.4: Number of instances solved, and average gap (%) compared to the solution from running the model for 1000 seconds, for the instance sets by running the matheuristic and model

Set of instances	Model				Matheuristic			
	100 sec	Gap (%)	1000 sec	Gap (%)	100 sec	Gap (%)	1000 sec	Gap (%)
S-50	5/5	0%	5/5	0%	5/5	0,20%	5/5	0,20%
S-100	5/5	0%	5/5	0%	5/5	0,26%	5/5	0,24%
M-50	5/5	0,97%	5/5	0%	5/5	1,65%	5/5	1,65%
M-100	5/5	9,72%	5/5	0%	5/5	1,20%	5/5	0,81%
L-50	4/5	39,82%	5/5	0%	0/5	NA	1/5	1,69%
L-100	2/5	23,94%	5/5	0%	1/5	1,57%	2/5	1,12%

In Table 6.4 we can see the same results, however here we calculate the gap using the solutions found by the model with time limit 1000 seconds. Hence we get gap values for most of the missing gap values in the Table 6.3, with the exception of were the heuristic could not find any solution. When we compare the model and the heuristic for the time limit of 100 seconds, we can see that the heuristic outperforms the model for larger instances when it is able to find a feasible solution.

6.2.2 With service level requirement

In this section the heuristic is run with service level requirement. The service level used corresponds to the medium service level in Hansen et al. (2018). The heuristic was run once for 100 seconds and once for 1000 seconds for all instances. A summary of the test results can be found in Table 6.3. Each of the six sets of instances are represented by separate rows in the table, and each set contains 5 instances of the same type. The heuristic results are compared with the solution obtained by running the model with the same time limit as the heuristic. The table also shows the number of instances where a feasible solution was obtained, as well as the average gap (%) for those instances. Complete results from running the heuristic with medium service level requirement, can be found in Appendix C.

Table 6.5: Number of instances solved, and average gap (%), for the instance sets by running the matheuristic and model

Set of instances	Model				Matheuristic			
	100 sec	Gap (%)	1000 sec	Gap (%)	100 sec	Gap (%)	1000 sec	Gap (%)
S-50	5/5	0%	5/5	0%	5/5	14,05%	5/5	102,17%
S-100	5/5	2,17%	5/5	2,17%	4/5	33,66%	5/5	116,08%
M-50	5/5	NA	5/5	5,34%	1/5	7,97%	5/5	105,55%
M-100	1/5	NA	3/5	NA	0/5	NA	2/5	NA
L-50	0/5	NA	2/5	NA	0/5	NA	0/5	NA
L-100	0/5	NA	3/5	NA	0/5	NA	2/5	NA

As shown in Table 6.5, the number of instances solved decreases, and the average gap (%) increases, when we add the service level requirement. For the smallest instances, the heuristic solves 9 of 10 instances within 100 seconds, and finds feasible solutions to all the instances when run for 1000 seconds. However, the average gap is significantly higher for the solutions found after only 100 seconds compared to the solutions found after 1000 seconds. For the medium instances the heuristic only finds feasible solutions to 1 of 10 instances within 100 seconds, while 7 of 10 is solved within 1000 seconds. For the largest instances, the heuristic is not able to solve any of the instances within 100 seconds, and only finds a feasible solution to 1 of 10 instances within 1000 seconds. We can also see from the table that the model struggles with finding any feasible solutions.

Table 6.6: Number of instances solved, and average gap (%) compared to the solution from running the model for 1000 seconds, for the instance sets by running the matheuristic and model

Set of instances	Model				Matheuristic			
	100 sec	Gap (%)	1000 sec	Gap (%)	100 sec	Gap (%)	1000 sec	Gap (%)
S-50	5/5	0%	5/5	0%	5/5	114,05%	5/5	2,16%
S-100	5/5	0%	5/5	0%	4/5	133,66%	5/5	13,39%
M-50	5/5	13,95%	5/5	0%	1/5	107,97%	5/5	2,04%
M-100	1/5	38,26%	3/5	0%	0/5	NA	2/5	-8,31%
L-50	0/5	NA	2/5	0%	0/5	NA	0/5	NA
L-100	0/5	NA	3/5	0%	0/5	NA	2/5	-3,76%

In Table 6.6 we can see the same results, however here we calculate the gap using the solutions found by the model with time limit 1000 seconds.

Hence we get gap values for most of the missing gap values in the Table 6.3, with the exception of were the heuristic couldnt find any solution. When we compare the model and the heuristic for the time limit of 100 seconds, we can see that the heuristic outperforms the model for larger instances with a large margin. In addition, note that for larger instances with time limit set to 1000 seconds, the heuristic get a negative gap. That is, it outperforms the model with the same time limit.

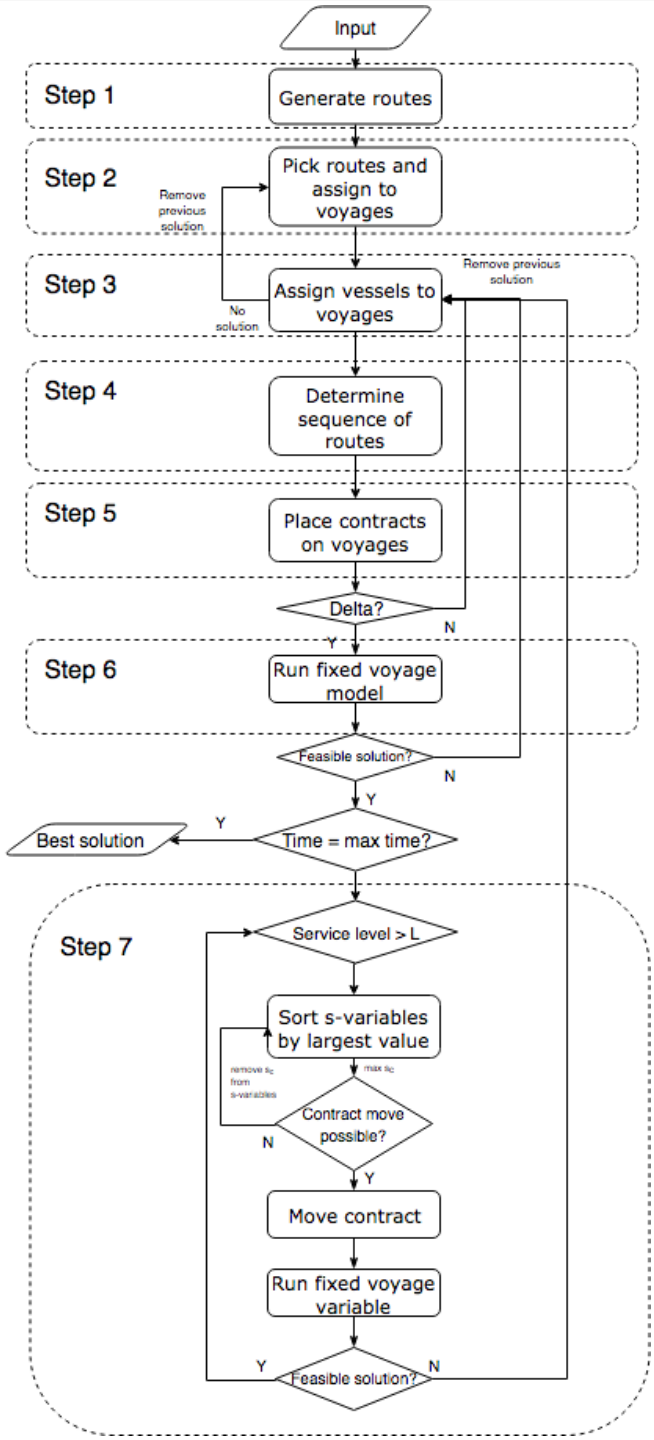
6.3 Analysis

This section contains an analysis of the results above, with focus on the quality of the solutions obtained with the heuristic. It is important to remember that the size of the instances used affects the apparent equivalency of the model and heuristic when run with the same time limits. For small instances it seems almost trivial to find a feasible (and sometimes optimal) solution by allowing the model to run for a limited time. However, as the size of the instances grows, the model is struggling with finding even feasible non-optimal solutions within the short time limit. These instances are still considered small compared to the large real instances, so all though running the model for a short while can result in good solutions for small, unrealistic instances, it is not a practical solution. We can also see from Table 6.4, that he heuristic outperforms the model as the instances grows for a time limit of 100 seconds. In addition, we can see from Table 6.6 that when the complexity is added by increasing the service level, the heuristic outperforms the model for large instances with 1000 seconds as the time limit.

As we saw in Chapter 5, this heuristic framework consists of several separate algorithms, which solves separate parts of the problem in an effort to do a quick search through the search space for good feasible solutions, without necessarily finding the optimal solution to the problem. This process is outlined in the flowchart in Figure 6.1. The result above indicates that there is some issues either with the entire heuristic, or possibly with part or parts of the heuristic. There are two main issues to consider. Firstly, the heuristic seems to struggle with finding solutions to larger instances. Secondly, the solution quality of the heuristics, all though reasonably good

for smaller instances, does not reach the necessary accuracy for the larger instances. We shall see that both of this issues stem from the same problem.

Figure 6.1 Flow chart



First we consider step 1 to step 3, in relation to both issues above. Step 1 will not be affected by an increase in the size of an instance, as this step simply generate possible routes, a process which is independent of both number of voyages and number of contracts. Step 2 to step 3 could possibly be the cause of the issues, as the models which selected route combinations and couples vessels with voyages could select a large number of infeasible routes and vessel combinations before finding a combination that could actually lead to a feasible solution. This would mean that the heuristic would use a lot of time generating and testing solutions that could never be feasible. However, this has been thoroughly tested for by allowing the heuristic run for very large time limits. If this was the root of the issue, one would expect to see a difference between solutions found with very large time limits and those found in Section 6.2. No such difference were found. Further more, Step 2 to step 4 were tested by locking the variables related to routes, voyages and vessels to the values found in those steps for selected runs, and running the model. All though the solver still used considerable time to solve to optimality, the model statistically found feasible solutions for routes and vessels combination more often then the heuristic, indicating that the issue lies elsewhere.

Another possibility is step 5, the algorithm for placing contract pickups on voyages. Let us consider this in relation to the first of the two issues above, the difficulty with finding solutions for large instances. Based on the results in Section 6.2 we could reasonably expect that as the size of the instances increases, the number of feasible solutions as a percentage of total number of solutions generated would decrease. In fact, this is exactly what has was found to be happening. With smaller instances, as much as 50-80% of suggested solutions were feasible, while that percentage decreased for larger instances, sometimes to lower than 10%. A natural hypothesis is that the contract placement algorithm is a function of the instances size related dimensions, and the one current used in this heuristic in this thesis does not properly take size variations into consideration. Two additional observations allow us to further understand were in the algorithm the issue lays. The logs from the heuristic shows that as the instance size increase and fewer feasible solutions are generated, the solutions that are infeasible are infeasible because they break capacity constraints. In addition, running the heuristic without transit time requirements in the model (a constraint closely related to loading and unloading constraints) allowed the heuristic

to find solutions for 8 out of 10 of the large test instances (results can be seen in Table D.1 and Table D.2 in appendix D.) Unlike a normal model run, this is not caused by the reduction in complexity from relaxing constraints, as the heuristic uses the exact same algorithms to find solutions. This all relates to the second of the two issues above as well, the solutions quality. The algorithm that places contracts on voyages is not as good as it should be, which possibly removes the good solutions from the search space. For example, if the algorithm has difficulty placing pickups, it might succeed only on routes and vessel combinations with more port visits than necessary.

Another possibility is that the error lays in step 7, the algorithm for improving solutions with respect to service level requirements. The logs saved from heuristic runs shows that the algorithm does find a number of improvements for solutions found with regards to service level requirements. However, it is difficult to say if anything about the performance of this algorithm when considering the issues with contracts placement and the effect this can have on the evenly spread constraints.

Chapter 7

Concluding Remarks

The STSRSP is the problem of planning the routes sailed and which vessels should be deployed for a set of interdependent voyages which are to be performed on a given *trade route*, as well as where and when these vessels should load and unload goods. A *trade route*, or trade, is a logistical network identified as a set of pathways and stoppages for commercial transport. In the STSRSP these activities are constrained by a set of contracts, and the objective is to minimize the total cost of all activities and keeping them within a specific planning horizon, while satisfying contractual requirements Hansen et al. (2018). Efforts to solve mathematical formulations of this problem with standard MIP solvers have shown that the run time is prohibitively large for direct use in operational level decisions. Hence, the purpose of this thesis have been to create a heuristic that can be used to reduce the time used to solve single instances of the STSRSP to within an acceptable accuracy.

The idea behind the heuristic is to divide different parts of the model into separate steps in a sequential process. This way the heuristic makes approximately good choices for each step and reduces the search space considerably on its way. The goal is of the heuristic is to iteratively build and evaluate solutions in a smart way, and hopefully generate solutions that are good enough without having to evaluate too much of the entire search space.

The solution method created in this thesis uses a matheuristic approach that divides the problem into several parts, combining both mathematical models to determine routes and vessel assignment, as well as a heuristic for placing the contracts on voyages. Finally, a service level search is implemented to make sure the separation requirement is satisfied. A heuristic framework was created in python to connect all the separate problems in such a way that solutions could be generated and tested iteratively. To study the performance of the heuristic, the best solutions it generated were compared to the best solutions from the commercial MIP solver at different time limits, as well as the optimal solutions. The results show that the heuristic framework can be used to solve problem instances within very short time limits. However, the results also indicate that the heuristic suffers from two main issues. Firstly, the heuristic seems to struggle with finding solutions to larger instances. Secondly, the solution quality of the heuristics, all though reasonably good for smaller instances, does not reach the necessary accuracy for the larger instances. An analysis of the heuristic were performed to see if the issue lay with the entire heuristic, or if it possibly lay with part or parts of the heuristic.

The results of the analysis is that the algorithm responsible for coupling contract pickups with port visits for specific routes is the source of both issues. However, the other results indicates that the heuristic framework is working well in its entirety, and that the results of the heuristic can most likely be improved by improving just this specific algorithm. In addition to improving this algorithm, the entire algorithm could be made to run much faster if it were rewritten from python to a compiled programming language like C++ or java. This would mean that the algorithm for coupling contract pickups with port visits could be more complex and/or that more possible solutions could be generated and tested over the same time limit.

Chapter 8

Future Research

The results from this thesis indicates that the general framework of the heuristic can successfully be used to solve problem instances within very short time limits. The result does however show some issues in the part of the heuristic that solves the problem of coupling contract pickups and voyages. Specifically, when transit time requirements are included for a proportion of the contracts for larger instances. Based on the results however, the conclusion of this thesis is that the general framework of the heuristic works, and that a good algorithm for contract and voyage coupling can be created. This suggest a natural next step for research into a heuristic solver for the STSRSP would be to create such an algorithm, and amend the heuristic. This could more easily be accomplished by using a faster programming language than the one used in this thesis (python), which would mean that the algorithm could be more complex and/or that more possible solutions could be generated and tested over the same time span.

Another next step can be to further extend the model while continuing to develop the heuristic to also be able to solve the extended models. Any model is a simplification of reality, and can usually be made more accurate by including more information. A current simplification of this model is that the vessels can only sail in one direction over the trade. For example, a vessel could not skip a port to travel to another port which geographically comes later during the voyage to unload goods and clear stowage space, and

then travel back to the skipped port. Disallowing this could cut off potentially better solutions from the search space, but allowing this does however make the run time explode. By adapting the route generation algorithm to allow the possibility of sailing both ways, and using domain knowledge to limit the possible routes (for example, a vessel would never travel back over a deep sea leg), these solutions could be included in the search space.

A major simplification in this model is that all data values used are static, and that the environment the voyages takes place in are static as well. Real-life optimization problems however, often contain uncertain data. As an example, the speed that a vessel can travel between two ports can be affected by the weather state in that area. Another example could be that mechanical errors in the engines of a vessel, or any other damage to the vessel, can affect port visit length or sailing times. One interesting application of this heuristic would be to see if it could be used to do robustness analysis within a reasonable run time, either by extending the framework to include functionality for robust or stochastic optimization.

Bibliography

Álvarez, J. F., 2012. Mathematical expressions for the transit time of merchandise through a liner shipping network. Journal of the Operational Research Society 63 (6), 709–714.

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

Bakkehaug, R., Rakke, J. G., Fagerholt, K., Laporte, G., 2016. An adaptive large neighborhood search heuristic for fleet deployment problems with voyage separation requirements. Transportation Research Part C: Emerging Technologies 70, 129–141.

Borthen, T., Loenechen, H., Wang, X., Fagerholt, K., Vidal, T., 2017. A genetic search-based heuristic for a fleet size and periodic routing problem with application to offshore supply planning. EURO Journal on Transportation and Logistics https://doi.org/10.1007/s13676-017-0111-x.

Campbell, A. M., Wilson, W. H., 2014. Forty years of periodic vehicle routing. Networks 63 (1), 2–15.

Christiansen, M., Fagerholt, K., Nygreen, B., Ronen, D., 2007. Maritime transportation. Handbooks in Operations Research and Management Science, (Eds. C. Barnhart and G. Laporte) 14, 189–284.

Christiansen, M., Fagerholt, K., Nygreen, B., Ronen, D., 2013. Ship routing

and scheduling in the new millennium. European Journal of Operational Research 228 (3), 467–483.

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

Fischer, A., Nokhart, H., Olsen, H., Fagerholt, K., Rakke, J. G., 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.

Gelareh, S., Nickel, S., Pisinger, D., 2010. Liner shipping hub network design in a competitive environment. Transportation Research Part E: Logistics and Transportation Review 46 (6), 991–1004.

Hansen, J. R., Fagerholt, K., Meisel, F., Rakke, J. G., 2018. Planning interrelated voyages with separation requirements in roll-on roll-off shipping. Working paper, Norwegian University of Science and Technology.

ISL, 2017. Shipping statistics and market review 2017 61 (9/10), 39–66.

Jung, J. U., Kang, M. H., Choi, H. R., Kim, H. S., Park, B. J., Park, C. H., 2011. Development of a genetic algorithm for the maritime transportation planning of car carriers. In: Dynamics in logistics. Springer, pp. 481–488.

Kang, M. H., Choi, H. R., Kim, H. S., Park, B. J., 2012. Development of a maritime transportation planning support system for car carriers based on genetic algorithm. Applied Intelligence 36 (3), 585–604.

Kisialiou, Y., Gribkovskaia, I., Laporte, G., 2018. The periodic supply vessel planning problem with flexible departure times and coupled vessels. Computers & Operations Research 94, 52–64.

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

Lindstad, H., Asbjørnslett, B. E., Strømman, A. H., 2011. Reductions in greenhouse gas emissions and cost by shipping at lower speeds. Energy Policy 39 (6), 3456–3464.

Logistics, W. W., 2016a. Roro the safer, smarter way. Accessed: 2016-10-17.
URL http://www.2wglobal.com/global-network/fleet/fleet-overview/interactive-vessel/

Logistics, W. W., 2016b. Take a look inside. Accessed: 2016-10-17.
URL http://www.2wglobal.com/news-and-insights/infographics/take-a-look-inside/

Norstad, I., Fagerholt, K., Hvattum, L. M., Arnulf, H. S., Bjørkli, A., 2015. Maritime fleet deployment with voyage separation requirements. Flexible Services and Manufacturing Journal 27 (2-3), 180–199.

Patricksson, Ø. S., Fagerholt, K., Rakke, J. G., 2015. The fleet renewal problem with regional emission limitations: Case study from roll-on/roll-off shipping. Transportation Research Part C: Emerging Technologies 56, 346–358.

Ronen, D., 1983. Cargo ships routing and scheduling: Survey of models and problems. European Journal of Operational Research 12 (2), 119–126.

Sigurd, M. M., Ulstein, N. L., Nygreen, B., Ryan, D. M., 2005a. Ship scheduling with recurring visits and visit separation requirements. In: Column generation. Springer, pp. 225–245.

Sigurd, M. M., Ulstein, N. L., Nygreen, B., Ryan, D. M., 2005b. Ship scheduling with recurring visits and visit separation requirements. Column Generation, (Eds. G. Desaulniers and J. Desrosiers and M.M. Solomon), 225–245.

UNCTAD, 2017. Review of maritime transport. United Nations Conference on Trade and Development, United Nations publication.

Vilhelmsen, C., Lusby, R. M., Larsen, J., 2017. Tramp ship routing and scheduling with voyage separation requirements. OR Spectrum 39 (4), 913–943.

Wang, S., Meng, Q., 2011. Schedule design and container routing in liner shipping. Transportation Research Record: Journal of the Transportation Research Board (2222), 25–33.

Appendix A

Voyage model

A.1 Notation

Sets

$\mathcal{V}$	Set of voyages
$\mathcal{V}_v^S$	Set of voyages succeeding voyage v
$\mathcal{K}$	Set of vessels
$\mathcal{P}$	Set of product types
$\mathcal{P}_p^S$	Set of product types that can be stored in the same space as product type p
$\mathcal{N}$	Set of ports
$\mathcal{C}$	Set of contracts
$\mathcal{C}^T$	Set of contracts with transit time requirements
$\mathcal{C}_i^L$	Set of cargoes that may be loaded at port i
$\mathcal{C}_i^U$	Set of cargoes that may be unloaded at port i
$\mathcal{C}^E$	Set of contracts with evenly spread requirements
$\mathcal{N}^P$	Set of ports in the network
$\mathcal{A}_k$	Set of arcs that define the feasible movements for vessel k
$\mathcal{A}$	Set of arcs that define movements

Parameters

C^{SC}_{ijks}	Sailing cost and chartering cost of the piloting and sailing time from node i to j with vessel k using speed alternative s
C^{V}_{i}	Cost of calling port i
C^{C}_{k}	The daily charter rate for vessel k
K^{V}_{kp}	The capacity for product p on vessel k
$\overline{P}_{c}$	Maximum number of pickups for contract c
$\underline{P}_{c}$	Minimum number of pickups for contract c
$\overline{Q}_{cp}$	Maximum quantity of product p to be picked up by a vessel for contract c
$\underline{Q}_{cp}$	Minimum quantity of product p to be picked up by a vessel for contract c
$\mathcal{D}_{cp}$	Demand for product type p for contract c in square meters
T^{A}_{k}	The time vessel k is available at its origin
T^{S}_{ijks}	The sailing time from node i to node j by vessel k using speed alternative s
T^{P}_{iv}	The time used on piloting at port i on voyage v
T^{H}_{kp}	The time used to handle (load or unload) one unit of product type p on vessel k
T^{T}_{c}	The corresponding transit time for contract c
T^{PH}	The length of the planning horizon

Variables

x_{ijv}	1 if voyage v is routed from node i to node j, 0 otherwise
y_{vk}	1 if vessel k sails voyage v, 0 otherwise
δ_{vc}	1 if voyage v serves contract c, 0 otherwise
l_{ijvp}	The load of product type p on voyage v on the arc (i, j)
w_{ijvks}	The weight of speed alternative s for vessel k on the arc (i, j) on voyage v
q_{vcp}	The quantity picked up of product type p from contract c on voyage v
t_{iv}	The start of service at node i on voyage v
t^{HW}_{k}	The total time used for handling and waiting by vessel k during the voyage

A.2 The model

$$\min \sum_{k\in\mathcal{K}} \sum_{(i,j)\in\mathcal{A}_k} \sum_{v\in\mathcal{V}} \sum_{s\in\mathcal{S}} C^{SC}_{ijks} w_{ijvks} + \sum_{(i,j)\in\mathcal{A}} \sum_{v\in\mathcal{V}} C^{V}_{i} x_{ijv} + \sum_{k\in\mathcal{K}} C^{C}_{k} t^{HW}_{k} \tag{A.1}$$

$$\sum_{k\in\mathcal{K}} \sum_{j\in\mathcal{N}^P_k \cup d(k)} x_{o(k)jv} = 1, \quad v \in \mathcal{V} \tag{A.2}$$

$$\sum_{i\in\mathcal{N}} x_{ijv} - \sum_{i\in\mathcal{N}} x_{jiv} = 0, \quad v \in \mathcal{V}, j \in \mathcal{N}^P \tag{A.3}$$

$$\sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{N}^P_k \cup o(k)} x_{id(k)v} = 1, \quad v \in \mathcal{V} \tag{A.4}$$

$$\sum_{j\in\mathcal{N}} \sum_{v\in\mathcal{V}} x_{o(k)jv} = 1, \quad k \in \mathcal{K} \tag{A.5}$$

$$\sum_{i\in\mathcal{N}} \sum_{v\in\mathcal{V}} x_{id(k)v} = 1, \quad k \in \mathcal{K} \tag{A.6}$$

$$x_{ijv} = \sum_{k\in\mathcal{K}} \sum_{s\in\mathcal{S}} w_{ijvks}, \quad (i,j) \in \mathcal{A}, v \in \mathcal{V} \tag{A.7}$$

$$\sum_{s\in\mathcal{S}} w_{ijvks} \leq y_{vk}, \quad k \in \mathcal{K}, (i,j) \in \mathcal{A}_k, v \in \mathcal{V} \tag{A.8}$$

$$\sum_{v\in\mathcal{V}} y_{vk} = 1, \quad k \in \mathcal{K} \tag{A.9}$$

$$\sum_{k\in\mathcal{K}} y_{vk} = 1, \quad v \in \mathcal{V} \tag{A.10}$$

$$0 \leq l_{ijvp} \leq \sum_{k\in\mathcal{K}} K^{V}_{kp} y_{vk} - \sum_{p'\in\mathcal{P}^S_p} l_{ijvp'}, \quad (i,j) \in \mathcal{A}, v \in \mathcal{V}, p \in \mathcal{P} \tag{A.11}$$

$$l_{ijvp} \leq M^{C}_{p} x_{ijv}, \quad (i,j) \in \mathcal{A}, v \in \mathcal{V}, p \in \mathcal{P} \tag{A.12}$$

$$\sum_{j\in\mathcal{N}} l_{jivp} + \sum_{c\in\mathcal{C}^L_i} q_{vcp} - \sum_{c\in\mathcal{C}^U_i} q_{vcp} = \sum_{j\in\mathcal{N}} l_{ijvp}, \quad i \in \mathcal{N}, v \in \mathcal{V}, p \in \mathcal{P} \tag{A.13}$$

$$\sum_{k\in\mathcal{K}}\sum_{j\in\mathcal{N}} l_{o(k)jvp} = 0, \quad v \in \mathcal{V}, p \in \mathcal{P} \tag{A.14}$$

$$\underline{P}_c \leq \sum_{v\in\mathcal{V}} \delta_{vc} \leq \overline{P}_c, \quad c \in \mathcal{C} \tag{A.15}$$

$$\delta_{vc} \leq \sum_{i\in\mathcal{N}} x_{il(c)v}, \quad v \in \mathcal{V}, c \in \mathcal{C} \tag{A.16}$$

$$\delta_{vc} \leq \sum_{i\in\mathcal{N}} x_{iu(c)v}, \quad v \in \mathcal{V}, c \in \mathcal{C} \tag{A.17}$$

$$\underline{Q}_{cp}\delta_{vc} \leq q_{vcp} \leq \overline{Q}_{cp}\delta_{vc}, \quad v \in \mathcal{V}, c \in \mathcal{C}, p \in \mathcal{P} \tag{A.18}$$

$$\sum_{v\in\mathcal{V}} q_{vcp} = D_{cp}, \quad v \in \mathcal{V}, p \in \mathcal{P} \tag{A.19}$$

$$t_{o(k)v} = T^A_k y_{vk}, \quad v \in \mathcal{V}, k \in \mathcal{K} \tag{A.20}$$

$$t_{iv} + \sum_{k\in\mathcal{K}}\sum_{s\in\mathcal{S}} T^S_{ijks} w_{ijvks} + T^P_{iv} x_{ijv} + \sum_{c\in\mathcal{C}^L_i\cup\mathcal{C}^U_i}\sum_{p\in\mathcal{P}} T^H_{kp} q_{vcp} \leq t_{jv}, \quad (i,j) \in \mathcal{A}, v \in \mathcal{V} \tag{A.21}$$

$$t_{l(c)v} + T^T_c + M^T_c(1-\delta_{vc}) \geq t_{u(c)v}, \quad v \in \mathcal{V}, c \in \mathcal{C} \tag{A.22}$$

$$t_{jv} - M^S_{jk}(1 - x_{o(k)jv}) \leq T^{PH}, \quad j \in \mathcal{N}^P, v \in \mathcal{V}, k \in \mathcal{K} \tag{A.23}$$

$$t^{HW}_k \geq t_{d(k)v} - t_{o(k)v} - \sum_{(i,j)\in\mathcal{A}_k} T^P_{iv} x_{ijv} - \sum_{(i,j)\in\mathcal{A}_k}\sum_{s\in\mathcal{S}} T^S_{ijks} w_{ijvks} - M^L_k(1-y_{vk}), \quad v \in \mathcal{V}, k \in \mathcal{K} \tag{A.24}$$

$$\sum_{k\in\mathcal{K}} t^{HW}_k \geq \sum_{v\in\mathcal{V}}\sum_{i\in\mathcal{N}^P}\sum_{c\in\mathcal{C}^L_i\cup\mathcal{C}^U_i}\sum_{p\in\mathcal{P}} T^H_{kp} q_{vcp} \tag{A.25}$$

$$\sum_{i\in\mathcal{N}^P}\sum_{k\in\mathcal{K}} x_{o(k)i(v+1)} \leq \sum_{i\in\mathcal{N}^P}\sum_{k\in\mathcal{K}} x_{o(k)iv}, \quad v \in \mathcal{V}\backslash\{|\,\mathcal{V}\,|\}, c \in \mathcal{C} \tag{A.26}$$

$$\sum_{v\in\mathcal{V}}\sum_{w\in\mathcal{V}^S_v} z_{vwc} \geq \sum_{v\in\mathcal{V}} \delta_{vc} - 1, \quad c \in \mathcal{C}^E \tag{A.27}$$

$$\sum_{w\in\mathcal{V}^S_v} z_{vwc} \leq \delta_{vc}, \quad v \in \mathcal{V}, c \in \mathcal{C}^E \tag{A.28}$$

$$\sum_{w \in \mathcal{V} \setminus (\mathcal{V}_v^S \cup \{v\})} z_{vwc} \leq \delta_{vc}, \quad v \in \mathcal{V}, c \in \mathcal{C}^E \tag{A.29}$$

$$\sum_{n=\underline{P}_c}^{\overline{P}_c} n\theta_{nc} = \sum_{v \in \mathcal{V}} \delta_{vc}, \quad c \in \mathcal{C}^E \tag{A.30}$$

$$\sum_{n=\underline{P}_c}^{\overline{P}_c} \theta_{nc} = 1, \quad c \in \mathcal{C}^E \tag{A.31}$$

$$\sum_{n=\underline{P}_c}^{\overline{P}_c} \frac{T^{PH}\theta_{nc}}{n} - s_c - M_c^E(1 - z_{vwc}) \leq t_{l(c)w} - t_{l(c)v}, \quad v \in \mathcal{V}, w \in \mathcal{V}_v^S, c \in \mathcal{C}^E \tag{A.32}$$

$$\sum_{n=\underline{P}_c}^{\overline{P}_c} \frac{T^{PH}\theta_{nc}}{n} + s_c + M_c^E(1 - z_{vwc}) \leq t_{l(c)w} - t_{l(c)v}, \quad v \in \mathcal{V}, w \in \mathcal{V}_v^S, c \in \mathcal{C}^E \tag{A.33}$$

$$\sum_{c \in \mathcal{C}^E} s_c \leq L \tag{A.34}$$

$$x_{ijv} \in \{0, 1\}, \quad (i, j) \in \mathcal{A}, v \in \mathcal{V} \tag{A.35}$$

$$\delta_{vc} \in \{0, 1\}, \quad v \in \mathcal{V}, c \in \mathcal{C}^E \tag{A.36}$$

$$0 \leq w_{ijvks} \leq 1, \quad k \in \mathcal{K}, (i, j) \in \mathcal{A}_k, v \in \mathcal{V}, s \in \mathcal{S} \tag{A.37}$$

Appendix B

Results - No service level

B.1 Matheuristic

Table B.1: The result from running the heuristic on the small instances with no service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Heuristic			
		100 sec	Gap (%)	1000 sec	Gap (%)
S-50-0	3890067	3917742	0,71%	3917742	0,71%
S-50-1	3933852	3934403	0,01%	3934403	0,01%
S-50-2	3494305	3496859	0,07%	3496859	0,07%
S-50-3	3620722	3622688	0,05%	3622688	0,05%
S-50-4	3823285	3827966	0,12%	3827966	0,12%
S-100-5	3765947	3770675	0,13%	3770675	0,13%
S-100-6	3912489	3925737	0,34%	3925737	0,34%
S-100-7	3521849	3524243	0,07%	3524243	0,07%
S-100-8	4089479	4121349	0,78%	4121349	0,78%
S-100-9	3725234	3725234	0,00%	3725234	0,00%

Table B.2: The result from running the heuristic on the medium instances with no service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Heuristic			
		100 sec	Gap (%)	1000 sec	Gap (%)
M-50-0	6277901	6584003	4,88%	6584003	4,88%
M-50-1	7004046	7083527	1,13%	7083527	1,13%
M-50-2	6387312	6402072	0,23%	6402072	0,23%
M-50-3	6615870	6745818	1,96%	6745818	1,96%
M-50-4	5993253	5995983	0,05%	5995983	0,05%
M-100-5	6599676	6607854	0,12%	6607854	0,12%
M-100-6	6482037	6503913	0,34%	6503913	0,34%
M-100-7	6182858	6183867	0,02%	6183867	0,02%
M-100-8	6397729	6623325	3,53%	6623325	3,53%
M-100-9	7799346	7955136	2,00%	7955136	2,00%

Table B.3: The result from running the heuristic on the large instances with no service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Heuristic			
		100 sec	Gap (%)	1000 sec	Gap (%)
L-50-0	6189820	NA	NA	6586424	6,41%
L-50-1	NA	NA	NA	NA	NA
L-50-2	NA	NA	NA	NA	NA
L-50-3	NA	NA	NA	NA	NA
L-50-4	6617845	NA	NA	NA	NA
L-100-5	NA	NA	NA	NA	NA
L-100-6	6773103	6805831	0,48%	6805831	0,48%
L-100-7	NA	NA	NA	NA	NA
L-100-8	NA	NA	NA	NA	NA
L-100-9	NA	9135644	NA	9050218	NA

B.2 Model

Table B.4: The result from running the model on the small instances with no service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Model			
		100 sec	Gap (%)	1000 sec	Gap (%)
S-50-0	3890067	3890067	0%	3890067	0%
S-50-1	3933852	3933852	0%	3933852	0%
S-50-2	3494305	3494305	0%	3494305	0%
S-50-3	3620722	3620722	0%	3620722	0%
S-50-4	3823285	3823285	0%	3823285	0%
S-100-5	3765947	3765947	0%	3765947	0%
S-100-6	3912489	3912489	0%	3912489	0%
S-100-7	3521849	3521849	0%	3521849	0%
S-100-8	4089479	4089479	0%	4089479	0%
S-100-9	3725234	3725234	0%	3725234	0%

Table B.5: The result from running the model on the medium instances with no service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Model			
		100 sec	Gap (%)	1000 sec	Gap (%)
M-50-0	6277901	6343397	1,04%	6277935	0%
M-50-1	7004046	7163756	2,28%	7004046	0%
M-50-2	6387312	6419496	0,50%	6387312	0%
M-50-3	6615870	6682565	1,01%	6615870	0%
M-50-4	5993253	5993253	0,00%	5993253	0%
M-100-5	6599676	6881244	4,27%	6599676	0%
M-100-6	6482037	6568302	1,33%	6482037	0%
M-100-7	6182858	7725021	24,94%	6182858	0%
M-100-8	6397729	7510763	17,40%	6397729	0%
M-100-9	7799346	7852010	0,68%	7802892	0,05%

Table B.6: The result from running the model on the large instances with no service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Model			
		100 sec	Gap (%)	1000 sec	Gap (%)
L-50-0	6189820	NA	NA	6477268	104,64%
L-50-1	NA	9434313	NA	7122511	NA
L-50-2	NA	10154274	NA	7821080	NA
L-50-3	NA	10103715	NA	7001703	NA
L-50-4	6617845	10160487	53,53%	6655337	0,06%
L-100-5	NA	8763046	NA	5926470	NA
L-100-6	6773103	6773746	0,00%	6773103	0,00%
L-100-7	NA	NA	NA	8277735	NA
L-100-8	NA	NA	NA	7870752	NA
L-100-9	NA	NA	NA	8899472	NA

Appendix C

Results - With service level requirement

C.1 Matheuristic

Table C.1: The result from running the heuristic on the small instances with service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Heuristic			
		100 sec	Gap (%)	1000 sec	Gap (%)
S-50-0	3975144	5118291	28,76%	3980082	0,12%
S-50-1	4004882	4071643	1,67%	4153293	3,71%
S-50-2	3567693	3651131	2,34%	3654165	2,42%
S-50-3	3682495	3916196	6,35%	3795716	3,07%
S-50-4	3830775	5023684	31,14%	3888291	1,50%
S-100-5	3810662	5138481	34,84%	3822061	0,30%
S-100-6	4130666	5443042	31,77%	4515144	9,31%
S-100-7	3528324	4545760	28,84%	4545760	28,84%
S-100-8	4106115	5662462	37,90%	4174666	1,67%
S-100-9	4072472	NA	NA	5713406	40,29%

Table C.2: The result from running the heuristic on the medium instances with service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Heuristic			
		100 sec	Gap (%)	1000 sec	Gap (%)
M-50-0	6329721	6833927	7,97%	6833927	7,97%
M-50-1	7076120	NA	NA	7267827	2,71%
M-50-2	6456452	NA	NA	6824265	5,70%
M-50-3	NA	NA	NA	7209662	NA
M-50-4	NA	NA	NA	6982359	NA
M-100-5	NA	NA	NA	7227241	NA
M-100-6	NA	NA	NA	NA	NA
M-100-7	NA	NA	NA	NA	NA
M-100-8	NA	NA	NA	NA	NA
M-100-9	NA	NA	NA	8165453	NA

Table C.3: The result from running the heuristic on the large instances with service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Heuristic			
		100 sec	Gap (%)	1000 sec	Gap (%)
L-50-0	NA	NA	NA	NA	NA
L-50-1	NA	NA	NA	NA	NA
L-50-2	NA	NA	NA	NA	NA
L-50-3	NA	NA	NA	NA	NA
L-50-4	NA	NA	NA	NA	NA
L-100-5	NA	NA	NA	NA	NA
L-100-6	NA	NA	NA	6880431	NA
L-100-7	NA	NA	NA	NA	NA
L-100-8	NA	NA	NA	NA	NA
L-100-9	NA	NA	NA	NA	NA

C.2 Model

Table C.4: The result from running the model on the small instances with service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Model			
		100 sec	Gap (%)	1000 sec	Gap (%)
S-50-0	3975144	3975144	0%	3975144	0%
S-50-1	4004882	4005087	0%	4005087	0%
S-50-2	3567693	3567693	0%	3567693	0%
S-50-3	3682495	3682495	0%	3682495	0%
S-50-4	3830775	3830775	0%	3830775	0%
S-100-5	3810662	3810662	0%	3810662	0%
S-100-6	4130666	4240829	2,67%	4130666	0%
S-100-7	3528324	3528324	0%	3528324	0%
S-100-8	4106115	4106115	0%	4106115	0%
S-100-9	4072472	4405495	8,18%	4072472	0%

Table C.5: The result from running the model on the medium instances with service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Model			
		100 sec	Gap (%)	1000 sec	Gap (%)
M-50-0	6329721	6904860	9,08%	6425753	0,02%
M-50-1	7076120	7835191	10,73%	7076123	0.0%
M-50-2	6456452	7764875	20,27%	6527922	NA%
M-50-3	NA	7379064	NA	7206523	
M-50-4	NA	9443102	NA	7252515	NA
M-100-5	NA	NA	NA	NA	NA
M-100-6	NA	NA	NA	NA	NA
M-100-7	NA	9874545	NA	7142268	NA
M-100-8	NA	NA	NA	9801953	NA
M-100-9	NA	NA	NA	8905512	NA

Table C.6: The result from running the model on the large instances with service level requirement for 100 sec and for 1000 sec

Instance	Optimal	Model			
		100 sec	Gap (%)	1000 sec	Gap (%)
L-50-0	NA	NA	NA	7226492	NA
L-50-1	NA	NA	NA	NA	NA
L-50-2	NA	NA	NA	NA	NA
L-50-3	NA	NA	NA	7499185	NA
L-50-4	NA	NA	NA	7129414	NA
L-100-5	NA	NA	NA	6337408	NA
L-100-6	NA	NA	NA	7149261	NA
L-100-7	NA	NA	NA	NA	NA
L-100-8	NA	NA	NA	8637864	NA
L-100-9	NA	NA	NA	NA	NA

Appendix D

Results - No transit time requirement

D.1 Heuristic

Table D.1: The result from running the heuristic without transit time requirements on the large instances with no service level requirement for 100 sec and for 1000 sec

Instance	Heuristic			
	100 sec	Gap (%)	1000 sec	Gap (%)
L-50-0	NA	NA	6303125	NA
L-50-1	7099372	NA	7026990	NA
L-50-2	7798099	NA	7743374	NA
L-50-3	NA	NA	6993036	NA
L-50-4	NA	NA	NA	NA
L-100-5	NA	NA	NA	NA
L-100-6	6802567	NA	6802681	NA
L-100-7	8062008	NA	8062435	NA
L-100-8	7941418	NA	7823697	NA
L-100-9	9141755	NA	9060027	NA

D.2 Model

Table D.2: The result from running the model without transit time requirements on the large instances with no service level requirement for 100 sec and for 1000 sec

Instance	Model			
	100 sec	Gap (%)	1000 sec	Gap (%)
L-50-0	6337477	NA	5982456	NA
L-50-1	9485506	NA	7006578	NA
L-50-2	10338526	NA	7748661	NA
L-50-3	11277473	NA	6918582	NA
L-50-4	10052114	NA	6657172	NA
L-100-5	7329700	NA	6008687	NA
L-100-6	6813369	NA	6771504	NA
L-100-7	10009302	NA	7944368	NA
L-100-8	7759654	NA	7758043	NA
L-100-9	11054900	NA	8656299	NA