NTNU
Norwegian University of
Science and Technology

Development of a Simulation-Optimisation Method for Solving a Maritime Fleet Size and Mix Problem

An Analytical Model Enhancement Approach

Hans Tobias Slette

Marine Technology
Submission date: February 2018
Supervisor: Bjørn Egil Asbjørnslett, IMT

Norwegian University of Science and Technology
Department of Marine Technology

Dedicated to mom and dad for always believing in me.

Preface

This thesis constitutes the complete workload of the course TMR4930 - Marine Technology, Master's Thesis, at the Norwegian University of Science and Technology (NTNU). The thesis is written during the fall of 2017, in collaboration with SINTEF Ocean.

The objective of the thesis is to construct a simulation-based optimisation algorithm which solves a maritime fleet size and mix problem based on the objective of the Arctic Offshore Logistics project at SINTEF Ocean. The connected simulation model is developed by SINTEF Ocean, as part of the project. The reader is expected to possess basic knowledge in the fields of both optimisation and simulation, as basic concepts are not explained.

I would like to thank my supervisor Professor Bjørn Egil Asbjørnslett at the Department of Marine Technology, NTNU, for guidance and support. I would also like to thank SINTEF Ocean for allowing me to use the simulation model. A special thanks is directed to my co-supervisor Inge Norstad, at SINTEF Ocean, for facilitation and advice.

Trondheim, February 12, 2018

Hans Tobias Slette

Summary

This report describes the development of a simulation-based optimisation algorithm for solving a maritime fleet size and mix problem. The problem in question is a platform supply problem, identifying minimum cost fleets for reliably serving a number of offshore platforms from an onshore base. The simulation model for evaluating the different fleet compositions was provided by SINTEF Ocean through the ArcticLog-project, and it was treated as a black box.

A literature review was performed, covering solution methods for optimisation problems utilising simulation evaluation; (i) Mathematical Programming based methods, (ii) Direct Search methods and (iii) Simheuristics. The suitability of the different concepts was examined, with emphasis on flexibility and intuitiveness. Concept (iii) was selected; establishing an approximate analytic model for quick solution evaluations, and a combination of a Nested Partitions algorithm and a Genetic Algorithm directing the search process. Optimal Computing Budget Allocation, Intensification, and Variance Reduction Techniques were added to improve solution method efficiency.

The solution method was tested for several different cases, with varying mission sizes, the lists of possible vessel concepts, the number of platforms, and the characteristics of both the base and platforms. The results were promising; good solutions were found quickly, and the computation time increased linearly with respect to all relevant parameters. Also, the method was found to be flexible and intuitive. Thus, it can easily be applied to a range of problem variations, serving as a practical decision support tool.

Global convergence for the solution method was not proven, and no optimistic bound was established. Therefore, addressing these issues is recommended for further work. In addition, a more comprehensive assessment should be conducted on the effect of the different choices made in the implementation, and the usefulness of the solution method compared to other methods.

Sammendrag

Denne rapporten beskriver utviklingen av en simuleringsbasert optimeringsalgoritme som skal løse problemet med å finne sammensetningen til marine flåter. Det aktuelle problemet gjelder forsyning av oljeplattformer, hvor målet er å finne den flåten som, til lavest mulig kost, kan forsynet et antall plattformer fra en base på land. Simuleringsmodellen som brukes for å evaluere de ulike flåtesammensetningene er utviklet av SINTEF Ocean, og den ble behandlet som en "svart boks".

En litteraturstudie ble gjennomført. Den dekket løsningsmetoder for optimeringsproblemer som bruker simulering som evalueringsmetode; (i) Metoder basert på Matematisk Programmering, (ii) Metoder basert på direkte søk, og (iii) Simheuristikker. De ulike konseptene ble vurdert etter hvor godt de egnet seg til problemet, med fokus på hvor fleksible og intuitive de var. Konsept (iii) ble valgt. En tilnærmet analytisk modell ble laget for raske evalueringer av løsninger, og en kombinasjon av en Nestede Partisjoner-algoritme og en Genetisk Algoritme ble brukt til å dirigere søket. Optimal Computing Budget Allocation, intensivering og teknikker for varinse-reduksjon ble lagt til for å forbedre effektiviteten til løsningsmetoden.

Løsningsmetoden ble testet for flere ulike problem-variasjoner, med varierende oppdragsstørrelser, lister med mulige fartøyskonsepter, antall plattformer, og karakteristikker hos både basen og plattformene. Resultatene var lovende: Gode løsninger ble raskt funnet, og regnetiden økte lineært i forhold til alle relevante parametre. I tillegg viste metoden seg å være både fleksibel og intuitiv. Metoden kan derfor enkelt benyttes i en rekke variasjoner av problemet, og fungere som et praktisk beslutningsstøtteverktøy.

Global konvergens ble ikke bevist for løsningsmetoden. Noen optimistisk grense ble heller ikke etablert. Det anbefales å se nærmere på disse utfordringene, som videre arbeid. I tillegg bør det gjennomføres en mer omfattende vurdering av effekten av de ulike valgene som ble tatt under implementeringen, og nytten ved metoden i forhold til andre metoder.

Table of Contents

List of Figures

List of Tables

Abbreviations

AFO	=	Attraction Force Optimisation
AHP	=	Analytic Hierarchy Process
AHTS	=	Anchor Handling Tug Supply vessel
AME	=	Analytic model enhancement
ASO	=	Alternate simulation-optimisation
ATM	=	Asynchronous Transfer Mode
BAS	=	Backtracking Adaptive Search
COMPASS	=	Convergent Optimisation via Most-Promising-Area Stochastic Search
CR1S	=	Common realisations for each solution
DES	=	Discrete event simulation
DH	=	Discrete heuristics
DOE	=	Design of experiments
DOvS	=	Discrete optimisation via simulation
DPO	=	Discrete Parametric Optimisation
DR1S	=	Different realisations for each solution
GA	=	Genetic algorithm
IP	=	Integer Programming
ISC	=	Industrial Strength Compass
JIT	=	Just-In-Time
LAST	=	Learning automata search technique
LNG	=	Liquid Natural Gas
LP	=	Linear Programming
MFSMP	=	Maritime fleet size and mix problem
MFSP	=	Maritime fleet size problem
MIP	=	Mixed Integer (linear) Programming
MMH	=	Memory-based meta-heuristics

MP	=	Mathematical Programming
MSR	=	Modified Stochastic Ruler
NP	=	Nested partitions
OCBA	=	Optimal Computing Budget Allocation
OSI	=	Optimisation with Simulation-based iterations
OSR	=	Oil Spill Response
PAX	=	Passenger
PRS	=	Pure Random Search
PSO	=	Particle swarm optimisation
PSV	=	Platform supply vessel
R-BEES	=	Randomised Balanced Explorative and Exploitative Search
R-BEESE	=	Randomised Balanced Explorative and Exploitative Search with Estimation
SA	=	Simulated annealing
SAS	=	Stochastic adaptive search
SBO	=	Simulation-based optimisation
S-O	=	Simulation-Optimisation
SP	=	Stochastic Program
TS	=	Tabu Search
TSP	=	Travelling Salesman Problem
VRP	=	Vehicle Routing Problem
VRT	=	Variance reduction techniques

Chapter 1

Introduction

In this chapter an explanation is given as to what the purpose of this report is, and why it is worth pursuing. First, the background and motivation is given. Then a presentation of the state of the art methodologies follows, before the thesis objective is presented. Finally, there is a short description of the report structure.

Background and Motivation

Ever since the computer's entry into scientific research, engineers and scientists have been able to analyse the behaviour of systems using simulation models. The computers make it possible to generate great quantities of pseudorandom numbers, which is necessary to precisely imitate the randomness of real systems. Three features make simulation very useful both in business and science: (i) The real-world response of concepts and solutions may be closely estimated at a fraction of the cost of an actual experiment. (ii) Adequately precise simulation models may be constructed in a relatively short time. (iii) Experiments covering several years in the real world may be performed in a matter of seconds on a computer.

Before the era of simulation-based optimisation, simulation was not considered an optimisation technique, despite its ability to quickly and precisely evaluate possible problem solutions. At the time, a common approach was to simulate a number of predetermined system configurations and optimise in the way of choosing the configuration that appeared to provide the best performance (Koenigsberg and Lam, 1976). This approach is still used, especially on small, simple systems. For large, complex systems, this could be a tedious and inconvenient method.

The explosive increase in computational power has led to computers being able to simulate ever more complex systems (Lucas et al., 2015). The last decades have seen the computational power of computers reaching a level where most practical problems may be simulated to an adequate level of detail and precision, in a reasonable amount of time (Sanchez and Wan, 2011). Following this trend, over the past decades there has been a comprehensive development in the field of solution methods for optimisation problems utilising simulation for solution evaluation. There are both "old" methods, initially intended for mathematical programming, which have been modified to function with simulation models, and new methods, developed especially to work in combination with simulation models Gosavi (2015).

Optimisation may provide great benefits in terms of increased performance and savings of resources, but in order for this to happen the results have to be accepted and applied by the decision makers. Two aspects are important in this regard; the results must be sufficiently certain, and the decision maker must feel confident in the validity of the results (Larson et al., 1991). For complex, stochastic problems traditional methods fall short in both respects (Fagerholt et al., 2010). Stochastic aspects are not treated properly, and the Mathematical Programs (MP) quickly become so intricate that even the developer may have a hard time explaining the outputs. Decision makers are not willing to implement big changes based on the results of a black box. Simulation-Optimisation(S-O) has the potential to "solve" both these problems. In addition to accurately imitating the system of interest, simulation provides the possibility to verify the reasonableness of each solution; this process may even be supported by visualisations and graphs. If the associated optimisation algorithm is constructed so that the search process is intuitive, and easy to understand for the decision maker, the solution method may be widely applied even by companies with no expertise on the field.

An example of problems which are extremely complex to solve using analytic modelling is the maritime transportation problem of the Arctic Offshore Logistics(ArcticLog) project, of

SINTEF Ocean (Eskandari and Mahmoodi, 2016). The primary objective was to find safe, cost-effective and environmental friendly solutions for Arctic logistics which make it possible to realise field development in the high north. The project was started in 2015 as a collaboration between SINTEF Ocean, Statoil ASA, NTNU, Vard Design AS and Troms Offshore Management AS. A result of the project was the development of a simulation model, which allows for the evaluation of different logistics systems with respect to uncertain and changing conditions, such as weather and carriage needs. Building a search method on top of this simulation model, will meet the objective of the project, and if made intuitive it will also fulfil the potential of S-O (Chica and A. Juan PPrez, 2017).

The oil & gas industry is renowned for large investments and a high cost level. The problem of providing transportation of cargo and people to and from the offshore installations, is no exception. The difference between an acceptable and a good logistics solution could be in the order of millions of dollars a year. One concern is the operational costs, another is the costs entailing an unplanned production shut-down as a result of inadequate reliability.

State of the art

Today, there are three prominent strategies for solving optimisation problems which must, or should, incorporate simulation evaluation for adequate solution verification. The first is based on the combination of simulation and MP. The search process is conducted by a MP, and the simulation model is employed only to retrieve the true performance of the selected solutions. This methodology presupposes that a sufficiently precise MP may be constructed for the current problem. Examples of this solution method in the maritime sector is given by Fagerholt et al. (2010), and Halvorsen-Weare and Fagerholt (2011). A drawback of this approach is that for complex problems, the MP may only solve a simplified version of the problem, yielding uncertainty of the actual goodness of the solution with regards to the real problem. Also, this method does not coincide with the intuitive nature of the simulation model.

Another methodology is that of employing discrete heuristics with solution evaluation through simulation models. For small problems this may mean using ranking and selection, of which there are several examples of marine application; Larson (1988), and Larson et al. (1991). However, for problems with larger solution spaces where an exhaustive is impractical, more sophisticated metaheuristics may be used (Gosavi, 2015), (Fu, 2015). The search of these methods is directed and verified by simulation evaluations. That is, sampling and comparison of the performance of the different solutions is the basis for the development of the search process. Although the solution process may be intuitive and easy to follow, this is not certain, and the methods give no guarantee for finding the optimal solution, or establishing an optimistic bound, within reasonable time. In addition, because all knowledge of the solution space originates from simulation samples, many samples are needed which in turn takes a lot of time. However, this methodology is well suited for problems where the user knows little about the solution space in advance.

The third approach is to develop an analytic model which is used as an alternative to the simulation model for a portion of the evaluations. A metaheuristic is employed to perform the search, and the evaluations alter between the analytic model and the simulation model. This methodology presupposes that the user has some knowledge of the solution space in advance, making the analytic model able to return evaluations which direct the search in a way congruent with that of the simulation model. Examples of theory on this approach is provided by Juan et al. (2015), and Figueira and Almada-Lobo (2014). While this method also does not guarantee optimality within reasonable time, it may save significant computation time compared to the pure simulation evaluation method.

Thesis objective

The thesis objective is to construct a solution method for solving the main objective of the Arctic Offshore Logistics project; to find safe, cost-effective and environmental friendly solutions for Arctic logistics which make it possible to realise field development in the high north. The solution method is to be based on S-O and will utilise the ArcticLog simulation model. The solution method shall find good solutions within reasonable time. Also, the solution method must be intuitive and flexible, in order to be practical to use as a decision support tool. The hope is that the introduction of S-O methods greatly will benefit the industry, and that the findings of this report is a small contribution in that direction.

Report structure

In **chapter 2**, the problem at hand is thoroughly described, and classified. In **chapter 3**, relevant literature is studied to review and evaluate different solution methods to the previously described problem. Through **Chapter 4** the big decisions on the structure and components of the algorithm are made, then in **Chapter 5** the details are set and the algorithm is implemented. The preparation of the experiments are described in **Chapter 6**, followed by a presentation of the results. Outputs and results are discussed in **Chapter 7**, before the conclusion is presented in **Chapter 8**. **Chapter 9** is dedicated to recommendations for further work.

Chapter 2

Problem Definition

In this chapter a detailed description is given of the Arctic Logistics problem. In Section 2.1 the system will be defined, and all relevant system components will be presented. In Section 2.2 the ArcticLog simulation model is presented; the interface, and how it is to be treated in this report. Finally, in Section 2.3 the problem is classified, facilitating for the construction of an effective solution method.

2.1 System Description

In this section, the specific transportation system in question will be described in detail. That is, the problem of Arctic logistics, covering transportation of cargo and people to and from the offshore installations. The system is divided into three parts; (i) perpetual entities, (ii) temporary entities and (iii) the environment. The perpetual entities stay within the system throughout the period of the analysis. The temporary entities are generated and terminated within the course of the analysis. Finally, the environment is the backdrop for the events in the system, ultimately having an effect on every aspect of the operations and behaviour of the entities.

2.1.1 Perpetual Entities

The base is an onshore location at which cargo enters the system, and vessels may load this cargo for transportation to the platforms. The characteristics of the base may be detailed to any level but restricting it to what is necessary for the evaluation of the system, the included parameters are as shown in Table 2.1. In the simulation model the base is set to Hammerfest in Norway, yielding the presented coordinates. The port capacity determines how many ships which can load at the same time, affecting the build-up of a queue. Opening hours for the helicopters indicates when they are allowed to take off, while, for the ships, it indicates legal hours for starting the loading. The location is used to calculate the distance to the platforms, and the loading rates determines how long the ships must stay in port. The loading time of pax is assumed to be negligible. The base is open only at certain hours for the different operations, as described in Table 2.1. The layout of the input file may be seen in Appendix A.1.

Table 2.1: Parameters defining the base in the simulation model, with the actual values used.

Parameters	**Value**	**Notation**
Port capacity	1	Ship
Open heli (take off)	0700 - 1900	Hours
Open ships(load start)	0900 - 1400	Hours
Location	70.7, 23.7	deg N,E
Bulk load rate	120	tons/hour
Decl load rate	100	m^2/hour

The platforms are the consumers of the cargo transported by the vessels. Every platform has a unique set of characteristics, with the principal differences being the location and the cargo demands. Other defining sizes are the loading rates, the type of platform, the age of the platform, the function of the platform, and so on. The relevant characteristics included in the simulation model are presented in Table 2.2. "Add turn time" is additional time spent at a platform, beyond the time needed to unload cargo.

Table 2.2: Parameters defining an offshore installation in the simulation model, with example values. The layout of the input file defining all platforms may be seen in Appendix A.2.

Parameters	**Value**	**Notation**
Location	72, 23	deg N,E
Bulk load rate	100	tons/hour
Deck load rate	50	m^2/hour
Add turn time	1	Hour

The vessels transport cargo and people to and from the platforms. When running the simulation, the fleet size and mix is predefined by a list of vessel concepts and a vector which indicates the number of vessels of each type in the fleet, see Appendix A.2. The characteristics of a vessel, which is of interest in the simulation model, is restricted to those displayed in Table 2.3. Three different sized lists of vessel concepts, including all of these parameters, may be seen in Appendix A.1. When constructing a candidate fleet, the current list, of either 10, 21 or 42 vessel concepts, restricts the options. The vessels are divided into two categories, P and H. Type P must follow a predefined round tour route visiting every platform. Type H travels directly from the base to the platform of choice and then back to the base. The routes are illustrated in Figure 2.1, where P types must follow the blue line, and H types follow the green lines. Dayrate and fuel consumptions, are used to compute the costs of each vessel. Speed and capacities are used to determine the behaviour of the vessel, and thus how much it can transport in a given time period. The significant wave heights decide whether the vessel may operate or not.

Table 2.3: Parameters defining a vessel concept in the simulation model, with example values.

Parameters	**Value**	**Notation**
Vessel category	P	-
Dayrate	25 000	$
Fuel cons norm	10	tons/day
Fuel cons stdby	5	tons/day
Fuel capacity	2000	tons
Speed	15	knots
Bulk capacity	1200	m^3
Deck capacity	600	m^2
Pax capacity	30	people
Max Hs (cargo/pax)	4/2	m

2.1.2 Temporary Entities

For this transportation system, the temporary entities naturally include what is being transported. For simplicity the cargo is divided into three groups; bulk, deck and pax. It is assumed that cargo spaces may not be used for more than one type of cargo, and that all cargo of the same type may utilise all that cargo space. Another temporary entity of special interest is fuel for the vessels, because of its limiting effect on the operations of a vessel. While the user may determine the average weekly demand for the different types of cargo, there is substantial stochasticity and variation in the demands for each week, see Appendix D. The layout of the input file may be seen in Appendix A.1.

Bulk cargo is cargo which is filled into tanks, usually under the deck of a platform supply vessel(PSV). This may be oils for greasing equipment, solutions needed for extraction, or other fluids or dry cargo consumed on the platform. The weekly demands may be in the range of 100-10 000m^3, depending on the number, sizes and types of platforms.

Deck cargo is cargo which is placed on the deck of the PSV. Usually equipment, materials, big objects, containers, or pallets. All kinds of things, from spare parts to food. The weekly demands may be in the range of 100-10 000m^2. Deck cargo cannot be stacked.

People are transported when there is a change of crew, or visitors, inspectors, or specialists are needed. A person has to be carried both ways, one person one way is a pax. The weekly demands may be in the range of 10-500pax.

2.1.3 Environment

Everything not already mentioned, which affects the behaviour of the system, is part of the environment, presented in Table 2.4. The weather greatly affects the operations, likewise the waves and phenomena such as fog. These are also among the most important stochastic elements of the system, in addition to the cargoes. In the simulation, historical data are used to provide realistic conditions, and the number of fog days per year may be set independently by the user, if desired. Geography also plays a significant role, determining where ships may sail. Finally, external market factors have an effect on the system, the oil price being one that is included in the model. The layout of the input file may be seen in

Appendix A.1. The duration of the simulation may be set, as "Number of days". For the rest of the report, "Number of days" is set to 100. The number of realisations per solution is also set by the user and will vary in this report.

Table 2.4: Parameters defining the environment in the simulation model, with example values.

Parameters	Value	Notation
Start day	1	days
Start month	5	month
Number of days	100	days
Fog days	0	days
Number of simulations	100	simulations
Sea margin	10	%
Weather data	"filename"	historical

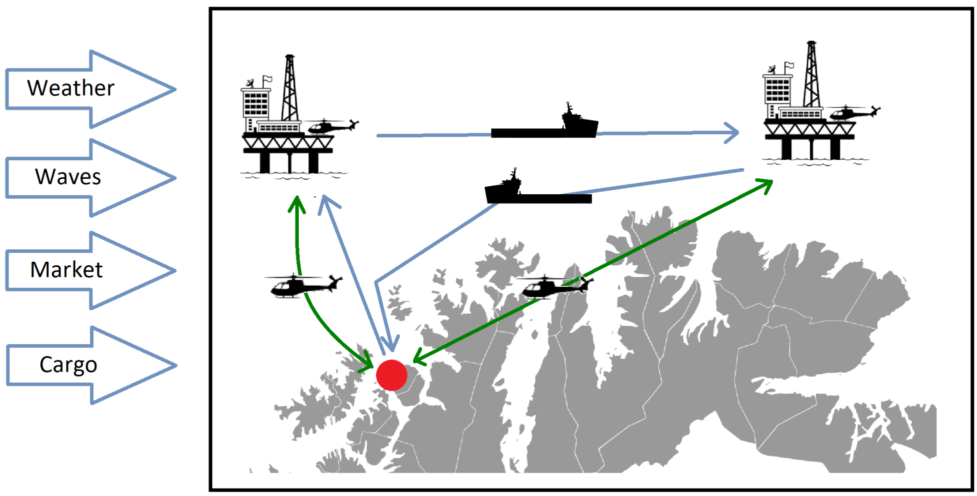


Figure 2.1: Conceptual illustration of the transportation system. The perpetual entities are contained inside the system, while the operations are affected by external factors and temporary entities.

2.2 Simulation Interface

The ArcticLog simulation model being treated as a black box means that it is fed with information, and then a result is returned. How the simulation model works, and any attempts at changing it, is beyond the scope of this thesis. The inputs are; (i) DataSet1.xlsx, previously referred to as "the input file", containing all information about the system, and (ii) ConceptsFile.xlsx, containing fleet concepts. The simulation model also has a third input file, determining the weather, but it is irrelevant as an input since it will stay unchanged during this report.

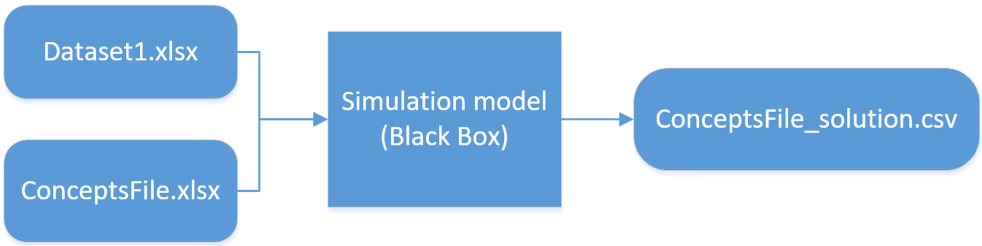


Figure 2.2: The inputs and output of the simulation model. DataSet1.xlsx may be seen in Appendix A.1. ConceptsFile.xlsx may be seen in Appendix A.2. ConceptsFile_solution.csv may be seen in Appendix B.4.

The simulation model may evaluate any number of fleets at the same time, exposing them to the same set of realisations. Loading the input files, running the simulation, and saving the outputs must be done manually. The performance measures covered in the evaluation, that is the values returned by the simulation model, are as follows:

- **Total cost** = Ship cost + Helicopter cost.
- **Ship cost** = Charter and voyage costs for all ships.
- **Helicopter cost** = Charter and flight costs for all helicopters.
- **Deck cargo lifted** = M^2 of deck cargo lifted per week, on average.
- **Deck cargo waiting** = M^2 of deck cargo that had to wait 48 hours or more.
- **Bulk cargo lifted** = M^3 of bulk cargo lifted per week, on average.

- **Bulk cargo waiting** = M^3 of bulk cargo that had to wait 48 hours or more.
- **Pax lifted** = Number of persons lifted per week, on average.
- **Pax waiting** = Number of persons that had to wait 48 hours or more.

2.3 Classification

The system presented in this chapter facilitates for the evaluation of a great variety of system configurations and changes. The change of any set of parameters presented in Section 2.1 may be evaluated with respect to the outputs presented in Section 2.2. Based on this functionality, the main objective of the ArcticLog project will be defined as follows, in this thesis: To find the cheapest transport solution which performs above some threshold with regard to each of the three cargo types. This means that for any given system configuration and possible vessel concepts, the fleet composition is to be found which accumulates the least costs during the period of interest, and transports a satisfactory amount of cargo, a traditional Maritime Fleet Size and Mix Problem (MFSMP). Since this is a stochastic problem, the threshold performance is given by a probabilistic measure, as presented in Equation 2.2. This also applies to the cost, as presented in Equation 2.1.

The problem may be expressed as follows, freely adopted from Juan et al. (2015):

$$Min\ f(s) = E[C(s)] \tag{2.1}$$

Subject to:

$$P(q_i(s) \geq l_i) \geq k_i \qquad \forall i \in I \tag{2.2}$$

$$s \in \mathbb{S} \tag{2.3}$$

Where:

- $\mathbb{S}$ represents a discrete, non-negative space of possible solutions s to the optimisation problem, which may contain upper boundaries.
- **C(s)** represents a stochastic cost function of indefinable structure.
- **E[C(s)]** represents a probabilistic measure of interest associated with the cost function - f.ex. the 95th percentile value.
- **Equation 2.2** states that the probability that the service quality $q_i(s)$ reaches a given threshold l_i must be above a user-defined value k_i
- **I** is the set of cargo types; bulk, deck and pax.

Optimisation problems may be classified into different groups based on various characteristics. The group to which the problem belongs decides what types of algorithms that are best suited to solve it. Some useful ways to sort the problems are: Continuous vs. discrete, stochastic vs. deterministic, parametric vs. control and linear vs. non-linear.

The problem to be solved by the solution method developed in this thesis is that of determining the number of vessels of each possible concept that is to be included in the optimal fleet. This means that all aspects of the system components, as given in the previous section is given for the problem. Consequently, the decision variables are constrained to positive integers, because the number of vessels may not be negative, nor fractional. Furthermore, the problem is parametric as opposed to being one of control, because the system is static while certain parameters are adjusted for optimising for an objective (Gosavi, 2015). Stochastic elements, most notably regarding weather conditions and cargo, makes the problem stochastic. The final classification of importance is that of linearity. Elements such as congestion of vessels at the base, and the limited, stochastic amount of cargo arriving at each time step, makes the problem non-linear.

Objective function

Normally, when solving optimisation problems, there is a well-defined function describing the relation between the fitness of a solution and the values of its decision variables. This objective function is of great help when searching for the optimal solution. If it is impossible, for some reason, to establish an analytic representation of the objective function for the problem in question, this eliminates the possibility of using common MP methods. Also, in the case of non-linear, combinatorial problems, there is generally no guarantee of finding the optimal solution without sampling the entire solution space. The true fitness of a solution may only be established through simulation, and stochasticity means that several different realisations must be simulated to establish a certainty about the performance.

Solution space

The solution space holds all possible solutions to the problem. Initially, the solution space goes from 0 to positive infinity, for every vessel concept. However, realistically, it is easy to restrict the solution space to finite maximum values. Even having only the slightest knowledge of the problem, one will probably know what a suitable order of magnitude for the maximum value is. For the problem discussed in this report a maximum value for each vessel concept of 100 would be plenty. For most of the cases described here, even 10 is sufficient. The number of dimensions in the solution space corresponds to the number of different vessel concepts, which, in this report, is set to either 10, 21 or 42, as described earlier (Appendix A.1). This results in a solution space of at least 11^{10}, and maximum 101^{42} solutions, which makes this a large solution space (Gosavi, 2015).

To conclude: A solution method is to be constructed for solving a discrete, stochastic, parametric and non-linear optimisation problem with no analytic expression for the objective function and a large solution space, with evaluation through the ArcticLog simulation model.

Chapter 3

Literature Review

This chapter serves as the first step of the solution method construction process. The purpose of the chapter is to review relevant literature to establish an understanding of which methods that are normally used, and what the cutting edge of research is, when solving a problem of the type presented in Chapter 2. In the construction of the solution method, these reviews will serve both as guidance when implementing established methods, and as inspiration where creativity is necessary. The literature review is divided into three parts covering different approaches for optimising problems with simulation evaluation; MP based methods(Section 3.1), Direct search methods (Section 3.2), and Simheuristics(Section 3.3). The literature study is, to a large extent, based on the extensive reviews and work of Hoff et al. (2010), Pantuso et al. (2014), Figueira and Almada-Lobo (2014), Gosavi (2015) and Amaran et al. (2017). The chapter is concluded with some final remarks.

3.1 Mathematical Programming Based Methods

In this section, some examples of the application and evaluation of MP based methods are presented. These methods combine mathematical programming and simulation in some way, mostly either to quickly search the solution space for promising areas or to select solutions which are then simply accepted or discarded by the simulation model. Most of the examples are from the maritime sector.

Bausch et al. (1998) Generates a deployment schedule covering 2-3 weeks, with hourly precision, within a couple of minutes. For bulk shipping. Demands and supplies are forecast. A decision support tool for the dispatcher. "The system must be user friendly". "The system must operate on a personal computer". Uses simulation to find all possible schedules for each ship, and then runs an optimiser to select the optimal, minimum cost, solution. Making sure that each ship is assigned to at most one schedule, and that all loads are transported. Applying this technique to a selection of possible vessels to charter, and extending the time horizon, would make it possible to apply to strategic MFSMPs.

Vis et al. (2005) Developed a Mixed Integer Problem (MIP) to minimise the fleet of vehicles needed for transporting transhipment containers from one buffer area in port, to another. Every container is given a transportation start time window. To validate the solutions of the MIP, they are simulated under various conditions. They found that the MIP gave results close to those of the simulations.

Imai and IV (2001) presents an analytic and a simulation model for determining the number of refrigerated containers a company should own, and how many to lease, in order to minimise costs. In the analytic model they assumed a constant cargo demand, for both a balanced and an unbalanced trade. In the case of the company in question, a constant demand is unlikely, therefore a simulation model was proposed. After running simulations for five different demand profiles and five different own fleet sizes, the resulting graphs were compared with the conclusion of what fleet size that incurred the lowest costs, given each demand pattern. Adding the knowledge of the probability of each demand pattern, the expected cost of each fleet size was calculated.

Halvorsen-Weare and Fagerholt (2011) solves the supply vessel planning problem by a combination of integer programming (IP) and simulation. Possible schedules are constructed, for periods of a week, by the mathematical program, then they are tested for robustness in different weather conditions in a simulation model. Using several approaches to reward robustness, they achieved a predicted cost saving of 3% compared to the previous solution method.

Fagerholt et al. (2010) combined optimisation and simulation to develop a methodology for solving strategic planning problems in tramp and industrial shipping. The idea is that the shortcomings of optimisation, with regard to stochasticity, and those of simulation, with regard to short term decisions such as routing and scheduling, will be overcome by combining the methods. The effect of strategic decisions is tested through a rolling horizon Monte Carlo simulation, where a solver makes the optimal routing decisions for the current horizon. The methodology is divided into four steps; (i) determining a set of strategic decisions, (ii) creating a set of scenarios, (iii) evaluating each decision in each scenario through simulation and (iv) analysing the results. A short-term scheduling optimisation is performed for each horizon in the simulation. A feedback loop is placed between (i) and (iv), leading to investigations of alternative strategic decisions if necessary.

Alvarez et al. (2011) Present a solution method for robust fleet sizing and deployment in bulk shipping. They developed a MIP model to solve the deterministic problem, and then the performance of the solutions was explored by "simulated realisations of the uncertain parameters."

Matta (2008) Investigates different mathematical programming approaches to represent discrete event systems. This includes both performance evaluation and optimisation models. Testing the approximate linear program shows that, for relatively simple cases, it is an efficient way of finding promising areas in the solution space.

Bertsimas and Sim (2003) Developed MIPs for robust combinatorial optimisation and network flow. The goal was to propose robust approaches of polynomial, deterministic problems, which also were polynomial. The robustness of the final solutions was evaluated by simulating the "distribution of the objective by subjecting the cost components to random perturbations."

3.2 Direct Search Methods

In this section a selection of heuristics, mostly metaheuristics and Stochastic Adaptive Search (SAS) methods, are presented. Only a few of these are presented in relation to maritime applications, but all are relevant given the simulation model and the final characterisation of the problem, given in Section 2.3. The books *Simulation-Based Optimization* by Gosavi (2015), *Handbook of Simulation Optimization* by Fu (2015), and *Metaheuristics* by Siarry (2016) have been of great guidance in studying the search methods of this section.

Ranking and Selection

Both the method developed by Larson et al. (1991) and the later improvement Richetta and Larson (1997) used a computer simulation model for evaluation of the fleet size of tugboats and barges for the transportation of refuse from New York City to Fresh Kills landfill on Staten Island. The outcomes of each simulation run were plotted on several graphs. These graphs were compared and assessed to determine the fleet size which responded best to a set of scenarios. Thus performing a simplified ranking and selection approach, with a *a priori* selection of interesting solutions.

Shyshou et al. (2010) used a simulation-based approach to test the effect of different future spot-rates on the number of long-term anchor handling tug supply vessel (AHTS) hires. Their goal was to develop a decision support tool, showing the consequences of having a certain number of long-term AHTS hires given different scenarios for future spot rates. They did not aim at finding the optimal fleet size given an uncertain future, rather they performed a ranking and selection to find the cost-optimal fleet size for each of three different future scenario spot rates; below average, average and above average. This was possible because the solution space was so small, only on the order of 10.

Tabu Search

Cousineau-Ouimet (2002) uses Tabu Search to solve an inventory routing problem.The demand is assumed to be deterministic. A general approach is described, and thereafter a test is performed, proving the utility of the method in a simple case. In Dengiz and Alabas (2000), increased productivity of a manufacturing system is the goal of employing tabu search to a JIT-system, by finding the optimal number of kanbans.

Genetic Algorithm

Aydın (2014) employs a genetic algorithm to solve an inventory routing problem with a homogeneous fleet of vehicles and stochastic demand at each customer. A chromosome is set to be a string containing all visits done by all vehicles over the set of time periods, including how much is delivered at each visit. An objective function including travel costs, back-order costs and the inventory holding costs is used to calculate a fitness value. A bigger objective function gives a lower fitness value. Good chromosomes have a higher probability of being selected as parents, and then a two-point crossover with different lengths is used to construct children. When mutating, two different chromosome elements, within the same chromosome switch places.

Syberfeldt et al. (2008) and Syberfeldt et al. (2015) are two examples of combining an evolutionary algorithm with discrete-event simulation models to solve network problems. The first describes optimisation of transport solutions for The Swedish Postal Services implementing a hybrid algorithm combining evolution strategies and genetic algorithms. The second presents a study of optimising waste collection from households in Sweden. The problem is approached as a travelling salesman problem (TSP) and is solved using evolutionary algorithms with a performance focused repair function, and a non-destructive cross-over operator to ensure only valid solutions.

Particle Swarm Optimisation

Introduced by Kennedy and Eberhart (1995) the method was initially constructed for the optimisation of continuous non-linear functions. Marinakis et al. (2013) gives an example of using particle swarm optimisation to solve a vehicle routing problem with stochastic demand. They claim that instances with a large number of costumers can't be solved to optimality within a reasonable amount of time, therefore an approximation technique may be used, and the Particle Swarm Optimisation (PSO) was chosen.

Learning Automata Search Technique

Introduces by Thathachar and Sastry (1987), Learning Automata Search Technique(LAST) is based on the ideas of pattern recognition, learning correct decision rules. In optimising the network throughput of an asynchronous transfer mode (ATM) network, Atlasis et al. (1998) uses a learning automata adaptive routing algorithm. The dynamic environment of the ATM network requires the use of an adaptive algorithm.

Simulated Annealing

Utilising the connection between statistical mechanics and combinatorial optimisation Kirkpatrick et al. (1983) develops a framework for the optimisation of large and complex systems, called Simulated Annealing (SA). Kokubugata and Kawashima (2008) proposes an algorithm using Simulated Annealing for solving a set of different routing problems in city logistics. This includes the Vehicle routing problem, the capacitated arc routing problem and the general routing problem with nodes, edges and arcs.

Backtracking Adaptive Search

Kristinsdottir et al. (2002) developed Backtracking Adaptive Search(BAS) as a means of better understanding the effect of accepting certain non-improving movements in a search for the global optimum. As opposed to SA, the acceptance threshold for worse solutions is dependent only on the difference in objective function value. Tarantilis et al. (2002) presents the development of a similar method; a stochastic search meta-heuristic algorithm, called the backtracking adaptive threshold accepting algorithm. This is intended to solve large instances of the Vehicle Routing Problem. This technique does not only allow some moves into worse solutions, to avoid being trapped prematurely in a local optimum, but also backtracking the threshold for accepting such moves. The local search method is a 2-opt.

Modified Stochastic Ruler

Introducing the Stochastic Ruler, Yan and Mukai (1992), propose the first non-heuristic for identifying the global optimum of discrete stochastic problems. The solution estimate is proven to converge to the global optimum. Alrefaei and Andradóttir (2001) proposes a solution algorithm to any type of stochastic system, for which the objective function is found by measurement or estimation through simulation. It is named the Modified Stochastic Ruler method, benefiting from three changes to the original stochastic ruler method; (i) A fixed number of observations per iteration, (ii) an approved approach for estimating the optimal solution, and (iii) a less restrictive transition procedure.

Nested Partitions

Shi and Ólafsson (2000) present the Nested Partitions (NP) method. They prove a convergence to global optimality with probability one, within finite time. The method combines both local search and global search in partitioning the solution space and concentrating the search on the most promising area. The theory and applications of NP is further presented in *Nested partitions method, theory and applications* by Shi (2009).

3.3 Simheuristics

In recent literature there are methods where extra emphasis is put on the relation between the simulation and the optimisation. Efforts are made to study, and exploit, the benefits of smart experiment design in order to spend only the time that is necessary on simulations of different realisations. Ólafsson (2006) described the use of metaheuristics in simulation optimisation, using GA, TS and NP to present how metaheuristics could be used in S-O. Emphasis was put on the gap between optimal solutions of exact models, and their practical counterparts. Juan et al. (2015) continued the research by presenting "a general methodology for extending metaheuristics through simulation to solve stochastic combinatorial optimisation problems". The paper builds on an extensive review of the field, extending previous work by introducing a new type of methaheuristics, which they call simheuristics. The method consists of two main features, as presented by the Juan et al. (2015):

1. "It promotes a closer integration between optimisation and simulation. In particular, the evaluation of solutions is performed not only by simulation, but also problem-specific analytic expressions. Hence, it mixes simulation and ad hoc approximations, although generic metamodels are avoided - while the simple nature of these models is appealing for optimisation purposes they do not accurately represent the real underlying system."

2. "The feedback of simulation can be used not only to evaluate solutions, but also to refine the analytic part, so that the latter is able to generate and/or evaluate more realistic solutions."

The curse of dimensionality is one of the problems sought to solve by Sanchez and Wan (2011), when discussing the design of experiments. Focusing on simulation studies the theory of Design of Experiments (DOE) is introduced. Examples are given on how it may be used to increase insight and information gain from an experiment. The article is written in the light of the interest in new, faster computers, and to the extent that they are predicted to alter science. The article authors aim at convincing the reader that DOE is at least as important, and that without DOE even the best computers would be hopeless at solving the simplest problems.

Hong and Nelson (2006) present an optimisation-via-simulation algorithm, called COMPASS, for use when the performance measure is estimated via stochastic, discrete-event simulation, and the decision variables are integer ordered. COMPASS is based on random search, as many other discrete optimisation via simulation algorithms, but it has a unique neighbourhood structure. The neighbourhood is defined as the most promising area, given by the area which is closer to the best solution than any other simulated solution. Candidate solutions are uniformly generated from the promising area, and the number of replications of each is chosen according to a simulation-allocation rule. An improved version of the metaheuristic is presented by (Hong et al., 2010), making it more efficient for high-dimensional problems. A further improved version, Industrial Strength Compass (ISC), was presented by Xu et al. (2010) it's based on a three-phase framework; global search, local search and clean-up. The authors claim that it: "offers correctness guarantees while also being competitive with the features provided by commercial products", and that it is one of the first to do so.

Pinho et al. (2012) propose, and evaluate, an optimisation method for discrete-event simulation models based on genetic algorithms which exhibits more efficiency in relation to computational time when compared to software packages that were on the market in 2011. The background of the proposition is the slow pace level of these commercial software packages when manipulating more than one input variable. The proposed method proved, through the selected performance tests, to be much faster than the commercial software SimRunner® in the majority of the cases.

Buchholz (2009) presents a brief overview of optimisation approaches for stochastic discrete event simulation. In particular it shows, by the combination of different methods, how to compose hybrid algorithms that fairly efficiently and reliably optimise medium sized models. The main message is combining algorithms to have both an exploration phase and an exploitation phase, in order to quickly find a good solution.

Andradóttir and Prudius (2009) discuss what features S-O methods include to be efficient numerically when applied to discrete problems of little known structure. The focus is on the balance between exploration and exploitation in the search. Two methods, called the R-BEES and R-BEESE, are presented. Made to solve, respectively, deterministic, and stochastic problems. The methods, the authors conclude, have the desired properties, the R-BEESE also including an estimation phase.

Eskandari and Mahmoodi (2016) S-O to compare the effect of fixed scheduling vs. demand-based scheduling for upstream offshore supply. An approximate analytic model was optimised in OptQuest® to find the optimal system configuration for a five different platform service levels. Then 50 simulations were run for each of the service levels, for each of the scheduling strategies, using these configurations. The Pareto fronts of the two strategies were compared, showing that demand-based scheduling could lead to great savings.

Introducing a nature-inspired heuristic called Attraction Force Optimisation (AFO). In the algorithm, (Ilaria et al., 2016), initially, place the particles at the vertexes of the solution space. The most fit particle in each iteration thereafter acts as a base for the attraction of the other particles. How far the non-base particles move in an iteration depends on their distance from the base, and the difference in fitness. The method is tested on a real industrial case, and shows promising results when compared to other, traditional optimisation techniques that are implemented in MATLAB® toolboxes.

A taxonomy providing a complete overview of the spectrum of S-O methods, is presented by Figueira and Almada-Lobo (2014). A four-dimensional spectrum, with each dimension divided into four sections categorises all possible S-O methods. The paper both serves as a contribution to establish a common taxonomy in the field, and as a guide for understanding the effect of the different elements of the solution methods. In addition, as pointed out by the authors, it shows what kinds of methods that has not yet been investigated.

Chica and A. Juan PPrez (2017) argues that simheuristics should be a first resort method for solving stochastic optimisation problems with large solution spaces. The benefits and limitations of simheuristics are discussed, and the authors present guidelines for designing a simheuristic. The paper is concluded with highlighting the "white-box" paradigm of simheuristics; "being understandable and enhancing the decision makers' participation."

3.4 Concluding the Literature Survey

This literature review barely scratches the surface of methodologies for the combination of simulation and optimisation. One reason for this is of course the vast amount of literature available. Another reason is the inconsistent taxonomy in the field, as remarked by Figueira and Almada-Lobo (2014), making the search process difficult. Still, the review covers a broad spectrum of relevant solution methods, indicating the forest of available methods. To figure out which method that is the best for any single problem may be an intractable task and is highly dependent on the implementation. Nevertheless, review will serve as a solid basis of guidance and inspiration for the construction of a solution method for the problem of Chapter 2.

Chapter 4

Establishing the Main Structure of the Solution Method

In this chapter a base concept for the algorithm will be established. This means that the main constituents of the algorithm and their order of action will be decided. These choices are based on the classification of the problem made in Section 2.3, and the methods described in Chapter 3. The chapter will start with the choice of concept in Section 4.1, before determining the main constituents in Section 4.2. Finally, a summary of the structure is presented in Section 4.3.

4.1 Choice of Concept

From the review of Chapter 3, the choice of a main concept for the current solution method is to be made from the three following concepts:

- Mathematical programming and simulation.
- Metaheuristic with simulation evaluations.
- Metaheuristic with both analytic and simulation evaluation.

In order to determine which concept to use, they are reviewed with respect to the method requirements, and the necessary main considerations which follow from the problem formulation of Chapter 2. However, as elegantly put by Amaran et al. (2017): "The sheer diversity of these algorithms also makes it somewhat difficult to assert which one is better than another in general, and also makes it hard to compare between algorithms or their implementations." This means that, although the decisions are justified, there is no guarantee they are perfect.

4.1.1 Method Requirements

As described in Section 1, the solution method is to be able to find good solutions within a reasonable time, be intuitive and flexible. In order to achieve this, the meaning of these requirements needs to be defined in relation to the features of the solution method.

Good solutions are defined by the user, comparing the quality of a solution to a set of requirements and expectations stated by the decision maker. In the current problem, three strict requirements are set concerning the carriage performance of a solution. If a solution performs below the threshold, the solution is bad, that is, not accepted to be returned to the user. On the other hand, if a solution performs above the threshold, it may be considered a good solution if the cost of the solution is low enough.

For problems where the discovery of the optimal solution may be proven, or that of an optimistic boundary, defining good solutions in terms of a maximum deviation is possible (Lundgren et al., 2012). When it is impossible to establish such a measure of deviation, the goodness of a solution may be based on the relative cost compared to other solutions discovered by the solution method, or it may be compared to solutions based on other solution methods such as expert judgement. In addition, the stagnation of a search process often is indicative of the quality of the current best solution.

Reasonable time is a magnitude which correlates to the problem type in question. In this case, tackling a strategic problem of which the solution will be in effect for several years, reasonable time may be in the order of a day. Even though this seems like a long time, it only allows for investigating a very small fraction of the solution space presented in Section 2.3.

Consequently, this necessitates a solution method which has strong convergence properties, and quickly establishes areas of interest. Such properties are usually the realm of MPs, but in the case of complex, combinatorial, stochastic problems, the computational time before reaching the first legal solution may be extensive. Metaheuristics are well known for quickly finding legal solutions, though not necessarily optimal, or even good.

Intuitive is defined by the Oxford dictionary as: 'Using or based on what one feels to be true even without conscious reasoning'. Preferably, every action in the decision process of the solution method should have a clear function. Two main "features" to avoid are; (i) elements of which the workings may not be readily explained, but when included in the solution method, it performs better, and (ii) a set of complex functions forming a sort of "black box". This element clearly eliminates the possibility of employing an MP in the solution process of the current problem. Naturally inspired metaheuristics exercise the opposite characteristics, they are inherently intuitive in their workings, but their convergence properties may be completely indecipherable (Gosavi, 2015). One such example is the Genetic Algorithm (GA) (Syberfeldt et al., 2008). Also, the "white-box" properties of the simheuristics

Flexible solution methods have the ability to solve a variety of problem types, with no or little change necessary. Most MPs are specialised, coping only with a very restricted set of problems, or even only one single problem (Fagerholt et al., 2010). Metaheuristics, on the other hand, may solve a great variety of problems and problem types. For example, GA and Simulated Annealing (SA) (Kirkpatrick et al., 1983) are conceptually only limited to the format of the input and the output. Virtually any problem, where every solution can be represented as a vector, can be solved by a metaheuristic. Whether the problem is solved to optimality is a completely different matter.

4.1.2 Main Considerations

(Having established a working definition of a good solution, the next step is to identify the necessary features of a solution method in order for it to find good solutions. Therefore, in this section there is a discussion on how to overcome the four main difficulties of the problem; dimensionality, discreticity, stochasticity, and non-linearity.

Discreticity

A discrete solution space means that the different solutions have to be sampled and compared, rather than a using a gradient, or other efficient methods applicable to continuous problems (Gosavi, 2015). Several methods developed for continuous problems, such as the gradient method, may be used for discrete problems, but the quality of the resulting solutions depend on the "density" of the solution space, and the nature of the objective function. For simple problems, these methods may yield acceptable performance, but for complex problems they entail several issues. For example, in the case of high dimensional solution spaces with small ranges, if the optimal "relaxed" solution is described by many fractional decision variables, there is no guarantee that rounding of the variables yields the optimal integer solution. In fact, there may be too many neighbours to even rank and select, and all of them may be illegal anyway (Sanchez and Wan, 2011). To solve these kinds of problems, it is recommended to use metaheuristics and Stochastic Adaptive Search methods (Gosavi, 2015).

Dimensionality

Dimensionality, or rather the curse of dimensionality, is an expression coined by Richard E. Bellman referring to the extremely rapid increase in the volume of the solution space when the number of dimensions increases. In the case of a range of 0 to 10 in each dimension: Going from Vessel data 1 (Appendix A.3) to Vessel data 2 (Appendix A.4), increasing the number of dimensions from 10 to 21 results in a volume increase from $11^{10} = 2.59 \cdot 10^{10}$ to $11^{21} = 7.40 \cdot 10^{21}$. That is, a 110% increase in the number of dimensions gives a $2.85 \cdot 10^{13}$% increase in the number of solutions in the solution space.

Solution methods with a computational time depending on the size of the solution space are not viable for large, high dimensional problems. Thus, algorithms sampling a proportion of the solutions, or in some way sorting or evaluating the solutions on any level, are too volatile to function for problems with a variable number of dimensions. For a solution method to work with such problems, the computational time has to be independent, or at least a polynomial function of the size of the solution space (Sanchez and Wan, 2011). COMPASS (Hong et al., 2010) and PSO (Kennedy and Eberhart, 1995) are examples of metaheuristics where the number of actions in the search process, to a great degree of precision, may be controlled independently of the number of solutions in the solution space. Other metaheuristics, such as NP (Shi, 2009) and AFO (Ilaria et al., 2016) are examples of a slow, polynomial increase of actions as the number of dimensions increases.

The remaining difficulty of the polynomial time solution methods is that of still identifying the good areas and solutions of these multi-dimensional solution spaces, although only sampling a tiny fraction. COMPASS, PSO, NP and AFO comprise a selection of different approaches, the suitability of which depend on other aspects of the problem in question.

As a consequence of the immense solution spaces of the current problem, the first fundamental error of Discrete Optimisation via Simulation (DOvS) problems, according to Fu (2015), may be presented:

1. *The optimal solution is never simulated.*

Stochasticity

Continuing, on the topic of stochasticity, with the second and third fundamental error of DOvS problems, according to Fu (2015):

2. *The best solution that was simulated is not selected.*
3. *We do not have a good estimate of the objective function value of the solution we do select.*

The statements above highlight the two main problems of stochastic problems. Preferably, the final solution method should contradict them by establishing certain estimates of the quality of the different solutions, and subsequently selecting the best solution. One of the reasons this is easier said than done is that the number of simulation realisations needed for establishing a certain probabilistic measure for a solution may be very high, in the order of 1000. In addition, evaluating a high number of realisations for each solution is of value only if the behaviour of the simulation model is close enough to the behaviour of the real system. One example of the difficulties in relation to the latter is that of the sea and weather conditions - the simulation scenarios are based on interpretations of historical data. This may very well give a realistic and close to true prediction, especially for strategic problems, but it will never be perfect in predicting the future. To quote George E.P. box: "All models are wrong, but some are useful".

One common feature of solution methods for discrete stochastic optimisation is to vary the number of visits to different solutions based on how promising the solution is. The modified stochastic ruler method (Alrefaei and Andradóttir, 2001) and NP (Shi, 2009) are two examples of methods which revisit only the most promising solutions, while bad solutions may never be visited. In the case of the general implementation of NP, the solution with the most visits is returned as the best. Another, common approach is to tackle stochastic problems as though they were deterministic, by identifying the number of realisations needed to get a certain estimate of the quality if the solution, and then simply running that many realisations for every candidate solution.

Non-linearity

The final aspect of the problem to be discussed is that of non-linearity, and especially when there is no closed form mathematical representation of the relation between the values of the decision variables and the performance of the solution. As an example; in the case of the problem in question, double the performance and twice the cost should not be assumed when doubling the number of vessels in the fleet of each type. Also, the observed effects of a change give no information about the effect of any other changes in the system. To get information about a solution, it must be sampled.

There are three main approaches for directing the search process, for these problems; (i) approximate response surface generation based on observations (Gosavi, 2015), (ii) analytic approximation of the problem (Juan et al., 2015), and (iii) pseudo-random selection based on relations between observed solutions (Kennedy and Eberhart, 1995). The first may be useful for mapping the major contours of the response in the solution space, but it is, naturally, guaranteed to fail at precise predictions. Constructing an approximate analytic function is an interesting and popular approach (Fagerholt et al., 2010), (Buchholz, 2009), (Sanchez and Wan, 2011), which has the same obvious weakness when it comes to precise predictions. Finally, the third method is used in established methods such as PSO

(Marinakis et al., 2013), LAST (Thathachar and Sastry, 1987), SA (Kirkpatrick et al., 1983) and MSR (Yan and Mukai, 1992).

4.1.3 Convergence Properties

Finally, before deciding on how to design the solution method, the desired convergence properties must be presented. The following definition is borrowed from Buchholz (2009):

1. If the algorithms runs infinitely long, then the probability of finding a point $\mathbf{x}$ such that $|\mu^* - f(\mathbf{x})| < \epsilon$ should approach zero for any $\epsilon > 0$. This is the intuitive definition of so called almost sure convergence (Andradóttir, 2006).

2. The algorithm should quickly find points with a small response.

3. If the algorithm determines point $\mathbf{x}$ as the point with the smallest response, then a confidence interval for $|\mu^* - f(\mathbf{x})|$ should be computable, if the feasible set of parameters is finite.

Furthermore, according to Buchholz (2009), to get the behaviour of points 1 and 2, the solution method must contain two phases; an exploration phase and an exploitation phase. This means that the algorithm first identifies the promising areas of the solution space before exploiting these areas to find local optima.

4.1.4 Concept Assembly

From the discussion in this section it is possible to make some decisions on the structure of the solution method. No attempt is made to state that the following choices are the best possible, or that the resulting structure is optimal, it is only believed that the resulting structure is well suited for solving the problem presented in Chapter 2.

The preferred approach for tackling the challenge of discontinuity is that of constructing an approximate analytic function. The reason for this is that this is an intuitive and flexible method, allowing for any level of detail in the implementation, and additions may be made when desired. Inspired by Fagerholt et al. (2010) and Juan et al. (2015), a feedback will be integrated from simulation results to the analytic function. Regarding stochasticity, an adaptive method will be attempted, simulating more realisations for promising solutions.

Now, the remaining problem, taking away the stochasticity and replacing it with an analytic function, is a discrete, parametric optimisation (DPO) problem. A selection of established metaheuristics for solving DPO problems are presented in Section 3.2. Two algorithms are to be chosen; one for the exploration phase, and one for the exploitation phase. This is done, respectively, in subsections 4.2.1 and 4.2.2.

4.2 Choice of Solution Method Constituents

Following the choice of concept in the previous section, the topic of this section is to decide on the concrete constituents of the solution method. That is, which metaheuristics to employ for the exploration phase and the exploitation phase, how the approximate analytic function is to be set up, and possible additions for enhanced performance.

4.2.1 Global Search

The most important function of the global search metaheuristic in this solution method is to quickly identify promising areas in a discrete solution space. As pointed out earlier, a key trait is a polynomial increase in computation time with an increasing number of dimensions. SAS methods are able to deliver on these requirements, in addition to guaranteeing global convergence (Gosavi, 2015).

Of the SAS methods, the following constitutes a popular selection: LAST, SA, BAS, MSR, NP and COMPASS. The performance of LAST, SA, BAS and MSR are dependent on a set of assumptions regarding the solution space, which make them less desirable than NP and COMPASS for this problem. For example, LAST needs a minimum and maximum for possible solution values, the closer to the true values the better. These values should be changed with a substantial change in the problem formulation. Another example is for SA; a scheme for the temperature change and neighbourhood structure is necessary. These problem specific assumptions, reduces the flexibility of the methods, and the intuitiveness of their search processes as compared to NP, AFO and COMPASS.

NP and COMPASS are conceptually similar in that they partition the solution space, without the need for any particular *a priori* knowledge of the problem or the solution values. Both quickly identifies solutions with small responses and explores widely. Their search processes are flexible and intuitive. With respect to the solution time, the NP is superior to COMPASS because it is easy to predict and control the increase in solution time as a function of the number of dimensions and their ranges. Also, NP is a more general approach, allowing for greater customisation of the implementation as opposed to COMPASS, in which basically only the number of samples may be changed. Furthermore, the computation of distances across the different dimensions in the solution space, as necessary in COMPASS, may cause for unfortunate comparisons leading to the wrong decisions (Shi, 2009), (Hong et al., 2010).

Review of the Nested Partitions algorithm

Introduced by Shi and Ólafsson (2000), NP seeks to step-wise reduce the search area until a good solution is reached. Then the search expands, and the procedure is repeated. In the long run, the solution which the algorithm visits the most times is assumed to be the optimal solution. In what way the search area is reduced is dependent on the problem in question and the gut feeling of the user. An extra feature added to the algorithm is the ability to take a step back. This means that if the algorithm, during any iteration, sees a better solution outside the reduced search area, it will restart by expanding the search area to include the whole solution space. This guarantees the optimal solution is found as the number of iterations goes to infinity, but it may be noted that very little information is stored from one iteration to another.

According to Shi (2009) there are several possible modifications of the algorithm, but the one presented is attempted to be general. The algorithm starts out by dividing the entire solution space into a set of subareas. For example, it may divide each dimension into two parts, making eight subareas out of a three-dimensional solution space. For each subarea, a random number of samples are done. The best performing sample solution for each subarea is stored. The algorithm then proceeds to reduce the search area to the subarea with the best performing sample solution. This subarea is then, further divided into subareas. Again, a random number of samples are done in each subarea. Now, in addition to the subareas, also a random number of samples are done in the surrounding parts of the solution space. If the best performing sample is found in any of the subareas, the algorithm will divide the search space once more. On the other hand, if the best performing sample is found outside the reduced search area, the algorithm will return to search the whole solution space. When the algorithm has reached the point where it is no longer possible to divide the search space any more, that is; the search space includes only one solution, this solution gets its visit count increased by one. The algorithm then returns to search the whole solution space. When the algorithm is done it returns the solution with the most visits as the best solution.

Generic Nested Partitions pseudo code

Let *x* be a solution.

Let *xa* be the best solution found in a subarea.

Let *xK* be the best *xa* found in a given iteration.

Let *xbest* be the best solution found overall.

Let *Mmax* be the maximum number of iterations.

Let *f(x)* be the objective function value of a solution.

Let *fa* be the objective function value of *xa*.

Let *fK* be the objective function value of *xK*.

Let *fbest* be the objective function value of the best found solution.

Let *S* be the entire solution space.

Let *S0* be the set of all singletons.

Let *F(m)* be the promising area, the reduced search space, of iteration m.

Let I be the number of subdivisions done in an iteration.

Let *Y(i)* be subarea i of *F(m)*.

Let K be the number of subareas to draw samples from.

Let *V(x)* denote the number of visits to the solution x.

Let T be an integer.

Algorithm 1 Nested Partitions

```
F(m) ← S
for x=1:S0 do
    V(x) ← 0
Select an initial solution x
xbest ← x
for m=1:Mmax do
    if F(m) ∈ S0 then
        I = 1
        Y(I) = F(m)
    else
        I ← Some user defined function or value
        Partition F(m) into I subareas; Y(1) through Y(I)
    if F(m)==S then
        K = I
    else
        Y(I+1) = S \ F(m)
        K = I + 1
    for k=1:K do
        L ← Uniform(1,2,...,T)
        for l=1:L do
            Select a random solution x in subarea k
            f(x) ← evaluate x
            if f(x) < fa then
                xa = x
        if fa < fK then
            xK = xa
    Set the subarea, which contains xK, as Ystar
    if F(m)==S then
        F(m+1) = Ystar
    else
        if Ystar ∈ F(m) then
            F(m+1) = Ystar
        else
            F(m+1) = S
    if F(m+1) ∈ S0 then
        V(F(m+1)) ← V(F(m+1)) + 1
    if V(F(m+1)) > V(xbest) then
        xbest ← F(m+1)
return xbest,fbest
```

4.2.2 Local Search

In addition to being subject to the same considerations as for the global search metaheuristic, the choice of a local search metaheuristic is dependent on efficient communication with the selected NP. The main concern is that the local algorithm is able to continue the search based on a set of promising solutions provided by the NP. Population based metaheuristics are an easy way of ensuring this feature; the set of promising solutions from the NP can be translated to the population of the algorithm. Three such algorithms include GA, PSO and AFO.

PSO and AFO are partly based on the assumption that there exists a well-defined area of global optimum, which the population is seeking to find through a collective effort. However, for the current problem this assumption is not made. The solution method must function adequately irrespective of the content of the provided list of vessel concepts. A clustering of the best solutions, justifying the assumptions of PSO and AFO may not be ensured.

In GA, on the other hand, every particle may commit to an individual search for a local optimum. This feature coincides better with the mission of exploiting the information provided by the NP search. The locations of the area of the best solutions are assumed found by the NP, the local algorithm is "simply" supposed to pin point the best solution in each area. Comparing these, the best solution of the solution space may be found. This task description harmonises the best with the GA. Also, determining the behaviour of the particles in the PSO and AFO, in such a way as to ensure proper convergence, is a substantially more complex task than for the GA. Finally, another reason to choose GA is the superior flexibility of the method, allowing for the addition or subtraction of elements such as mutation, selection, reproduction and elitism.

Review of the Genetic Algorithm

Genetic algorithms are inspired by Darwin's evolution theory, which is based on the term "survival of the fittest". This is about favouring reproduction of the fittest individuals, meaning that the fittest individuals are more likely to produce offspring. The algorithm is a very popular metaheuristic, and has been widely applied in the industry with a lot of success (Syberfeldt et al., 2008), (Aydın, 2014), (Syberfeldt et al., 2015). The statement can be amplified by looking at the amount of available literature.

The algorithm starts by selecting an initial population of solutions, also called particles. Thereafter, the particles are exposed to the "evolution" of a number of generations. For each generation the population is altered in several different ways; through mutation of the particles, "killing" the least fit, reproduction from the most fit. In addition, there are other possible elements of alteration such as elitism, performing a quick local search for each particle.

Selection and reproduction are the key features of the algorithm. "Children" are generated by combining traits from a set of "parents". The parents are chosen from the surviving particles of the selection process. The children restore the population size. The mutation usually means a small random alteration of the solution.

Generic Genetic Algorithm pseudo code

Let N be the number of decision variables.
Let P be the population size.
Let $x(p,:)=(x(p,1),x(p,2),\ldots,x(p,N))$ be solution p.
Let $A(i)$ be the set of values that decision variable i can take on.
Let $x(1{:}P,:)$ be the population.
Let G be the number of generations.
Let $f(x(p,:))$ be the objective function value of $x(p,:)$.
Let m be the number of solutions that survives the selection process.

Algorithm 2 Genetic Algorithm

```
for p=1:P do
    for i=1:N do
        x(p,i) ← Uniform A(i)
    f(x(p,:)) ← Evaluate x(p,:)
Sort x. x(1,:)=best, x(2,:)=2nd best, ..., x(P,:)=worst.
for g=1:G do
    x = x(1 : m,:)
    for p=(m+1):P do
        x(p,:) ← Child of a set of solutions from x(1 : m,:)
    for p=1:P do
        x(p,:) ← Randomly selected neighbour of x(p,:)
    Sort x. x(1,:)=best, x(2,:)=2nd best, ..., x(P,:)=worst.
return xbest
```

4.2.3 Analytic Approximation

As mentioned decided earlier, the constructed solution method employs two types of solution evaluations. One is simulation, which gives the true solution performance, but it is stochastic in every simulation thus only returning a statistical measure, and it is time consuming. The other is an analytic function, returning deterministic measures of the performance, and doing so very quickly. However, a drawback of the analytic function is the inherent lack of precision entailing a deterministic measure of a stochastic variable, and the possibly substantial deviation of the evaluations from the true values.

Because the simulation model is treated as a black box, it is sufficing to say that; evaluation is simply performed by inputting the desired data into the model and running it for the desired number of realisations. Thereafter the result is analysed. The analytic model, on the other hand, is constructed as a set of problem type-specific functions, approximating the performance of the solutions.

Cost evaluation

The analytic function is built to serve as the measure of solution quality for the NP and GA search processes. Because the algorithms rank solutions based only on one value, the restrictions of Equation 2.2 need to be included in the cost. This is done by setting the cost of illegal solutions, that is solutions which break at least one restriction, equal to infinity. Now, the functions necessary to calculate an approximation of the cost of a solution are established. The set of functions needed to approximate the service quality of a solution follow later.

$$SolutionQuality = \begin{cases} Cost & \text{if } \textit{legal solution} \\ Infinity & \text{if } \textit{illegal solution} \end{cases} \tag{4.1}$$

The main cost drivers are the dayrates of the vessels and the fuel. In addition, the cost of helicopters are based on a price pr. flight hour. This leads to the following equation:

$$Cost = Dayrates + FuelCost + FlightCost \tag{4.2}$$

Dayrates is simply the sum of the dayrates of each vessel in the solution multiplied by the number of days in the period under consideration. In Equation 4.3, AL is the set of different vessel concepts, $NoV(j)$ is the number of vessels of type j in the solution, $DV(j)$ is the dayrate of vessel type j in dollars, and T is the duration of the period, in days.

$$Dayrates = \sum_{j=1}^{AL} (NoV(j) \cdot DV(j)) \cdot T \tag{4.3}$$

The fuel cost is somewhat more complicated to approximate properly, because it depends on the ratio of sailing time, standby time and idle time for each vessel. In turn, this depends on the inter vessel departure time. In Equation 4.4 N_T is the total number of round trips the PSVs perform during the period. AC is the average consumption of the PSVs when sailing in tons pr day, D is the distance of a round trip in nautical miles, P_f is the price of fuel in dollars pr ton, and AS_P is the average sailing speed of the PSVs in knots.

$$FuelCost = \frac{N_T \cdot D}{24h \cdot AS_P} \cdot AC \cdot P_f \tag{4.4}$$

Explaining the N_T in Equation 4.5, it is approximated as the minimum of to values. The first is the maximum number of PSV departures allowed from the base, given the minimum inter vessel departure time, IVD. The second is the number of trips the PSVs are able to make during the duration of the simulation. P is the set of all vessel concepts which are categorised as PSVs.

$$Trips = min(\frac{T}{IVD}\ ,\ \frac{T \cdot AS_P \cdot \sum_{j=1}^{P} NoV(j)}{D}) \tag{4.5}$$

Flight cost is the product of flight hours, H, and the price pr flight hour, P_H. The latter is simply a constant, set to \$2000, the former is a function of the available workload, the number of helicopters and their interaction with the base and platforms. In order to have a relatively simple expression, the result, Equation 4.6 is partly empirically based. MP(MinPax) is the system input value that is the basis for the pax demand in the simulation, as described in Section 2.1. AFD is the average flight distance in nautical miles, which is twice the average distance from the base to the platforms. AC_p is the average pax capacity of the helicopters, AS_H is the average speed of the helicopters in knots, NoV is the total number of helicopters. H is the set of vessels categorised as helicopters.

$$FlightCost = P_h \cdot min(\frac{MP \cdot D \cdot AFD}{7 \cdot AC_p \cdot AS \cdot 0.77}\ ,\ \frac{2 \cdot \sum_{j=1}^{H} NoV(j) \cdot T}{24h} \cdot (\frac{AFD}{AS_H} + 2)) \tag{4.6}$$

The "min" expression of Equation 4.6 approximates the number of flight hours by choosing the lowest of the two alternatives. The first calculation is based on the number of helicopter trips needed to transport all the demand. The 7 is because the demand is set on a pr week basis in the input file, see Appendix A.1. The 0.77 is the assumed utilisation of the helicopters. The second part is based on the maximum number of trips the fleet of helicopters manage to perform in the given time period. The first 2 is due to the assumption that a helicopter may perform two trips a day, and the second 2 is the turnaround time in hours.

Performance evaluation

The performance of a solution is divided into three measures; (i) delivered bulk cargo, (ii) delivered deck cargo and (iii) delivered pax. The prediction of bulk and deck performance is straight forward; the average capacity of a vessel multiplied with the number of trips performed by the fleet in a week, This is shown for bulk cargo in Equation 4.7, and deck cargo in Equation 4.8. 168 is the number of hours in a week, $C_B(j)$ is the bulk capacity of vessel type j in m^3 and $C_D(j)$ is the deck capacity of vessel type j in m^2.

$$Bulk = \frac{168h \cdot N_T}{T} \cdot \frac{\sum_{j=1}^{AL}(NoV(j) \cdot C_B)}{\sum_{j=1}^{AL} NoV(j)} \tag{4.7}$$

$$Deck = \frac{168h \cdot T \cdot \sum_{j=1}^{AL} NoV(j) \cdot C_D}{PSVs \cdot Duration} \tag{4.8}$$

Approximating the pax performance is done somewhat differently because both helicopters and ships may transport pax. The total pax transportation capacity is thus the sum of the *ShipPax* and the *HeliPax*. *ShipPax* is calculated in the same manner as for bulk and deck, with $C_P(j)$ being the pax capacity of vessel type j. The number of flight hours is used as the basis for computing the *HeliPax*. The first part of the sum in Equation 4.9 is the *ShipPax* contribution, the second is that of *HeliPax*. H_F is the flight hours as calculated in Equation 4.6. AS_H is the average speed of the helicopters.

$$Pax = \frac{168h \cdot T \cdot \sum_{j=1}^{P}(NoV(j) \cdot C_P)}{\sum_{j=1}^{P} NoV(j) \cdot T} + \frac{168h \cdot C_P \cdot H_F \cdot AS_H}{AFD \cdot T} \tag{4.9}$$

4.2.4 Additions

As decided in Subsection 4.1.4, a feedback from the simulation results to the analytic function, from now on called adjustment, is to be integrated in the solution method. Also, other additions are considered to improve efficiency; Intensification (Juan et al., 2015), Optimal Computing Budget Allocation (Chen, 2011), Variance-reduction techniques (Figueira and Almada-Lobo, 2014) and Implicit enumeration (Lundgren et al., 2012). For additions to be implemented in the solution method, it must be certain that the benefits it yields are greater than the disadvantages with respect to computational time and complexity in implementation.

Adjustment

Adding the feature of adjusting the analytic function is a simple solution for improving the predictions. Comparing the predicted performance to the results of the simulations, gives a factor which in the next run is used to adjust the prediction in the way shown in Equation 4.10. Factor Bulk, FB, and its equals FD and FP are vectors, where a new element is added to the end after every simulation run. $Prod(FB)$ means the product of all the elements in FB.

$$Bulk = Prod(FB) \cdot \frac{168h \cdot N_T}{T} \cdot \frac{\sum_{j=1}^{AL}(NoV(j) \cdot C_B)}{\sum_{j=1}^{AL} NoV(j)} \tag{4.10}$$

Only the performance predictions are adjusted, not the costs. This is because the adjustment does not change the order of solutions, that is, it affects all solutions equally. When considering cost, it is only of interest how the solutions stand in relation to each other. On the other hand, when considering performance, there are absolute thresholds to reach. Therefore, which solutions that are legal depends on the adjustment.

Intensification

The idea is that a long memory saves information about the visited solutions. This can then be used to decide on a set of variables to fix to certain values in the future search. For example, fixing a variable to the most common value, or to the most common value of the 10 best solutions. The motivation for this is that it is more likely that the variable should take this value in an optimal solution. The effect is a significant reduction in the size of the solution space (Andradóttir and Prudius, 2009). Also, the opposite, diversification, could be applied to ensure exploration of less visited areas.

Optimal Computing Budget Allocation

This is a whole field in itself, as presented by Chen (2011). "The ultimate goal is to minimise the total simulation budget while achieving a desired optimality level, or to maximise the probability of finding the best design using a fixed computing budget"(Chen, 2011). The reason for this interest is, of course, the same as for the use of analytic solution evaluation; in simulation "(...) obtaining the function value, at even one point, can be time consuming. As a consequence, one must make an economic choice of the number of points to be sampled." (Gosavi, 2015)

There are two main drivers of the computational burden; the number of solutions which are simulated, and the number of realisations simulated for each solution. Especially in the case of computationally expensive simulations, minimising the number of simulations is of great importance. The first step is to make sure that only solutions of special interest are simulated. That is, solutions of which certainty about their performance is of great value. The second step is to only have as many realisations as are necessary. Initially, it is only of interest to use the simulations to adjust the predictions of the analytic function, rather than to find the best solutions. When it is believed that the analytic function is well adjusted, the number of realisations may be increased in order to increase certainty about the simulation results. A simple version of Optimal Computing Budget Allocation (OCBA) will be implemented in the solution method.

Variance-Reduction Techniques

"VRTs are methods that aim at reducing the variance of results for a given number of replications. This means that for the same confidence level, the application of Variance-Reduction Techniques (VRT)s may allow reducing the number of replications and hence, the computational effort."Figueira and Almada-Lobo (2014). The most common method, common random numbers (Chen, 2013), uses common scenarios for evaluation of the solutions in a set. The set may be of any size, from the closest neighbours of a solution to the entire solution space. Common random numbers will be implemented in the current solution method by simulating sets of solutions at a time, with the same realisations. In this way the variance is reduced, increasing the quality of the adjustment factors.

Another benefit from simulating a set of solutions at a time is the increased stability in the resulting adjustment factors. "Sampling more than one solutions increases the chance to converge to a better local optimal solution" (Hong and Nelson, 2006). Furthermore, due to the manual process of running the simulations, simulating sets at a time reduces the total time spent on manual actions.

Implicit Enumeration

Implicit enumeration is a technique for constraining the solution space based on the findings of legal solutions. The idea is: "(...) to use different tests to conclude that some solutions can not lead to better solutions than the best available." Lundgren et al. (2012). An illustrative example would be the case of only one vessel concept in the current problem, in which case a legal solution of four ships would probably be sufficient to disregard all higher solutions because of their higher cost. In the case of 10, 21 or 42 vessel concepts, the necessary tests would be difficult to design. Therefore, this feature is not implemented in the solution method.

4.3 Summary of Base Concept

The complete base concept is shown in Figure 4.1. In chapter 5 an algorithm of this design will be implemented in MATLAB and tested. First, some inputs are sent to NP, then NP returns a set of solution vectors in a matrix, then they are sorted and the best, unique solutions are forwarded to the GA. Then the GA is ran returning a set of solutions in a matrix, which in turn are sorted and the best, unique are forwarded to be simulated. If a termination criterion is met, the algorithm stops and the solutions are returned to the user, otherwise adjustments are made to the analytic function and the process repeats from NP. One complete round, from one run of the NP to the next run of the NP, is called a cycle.

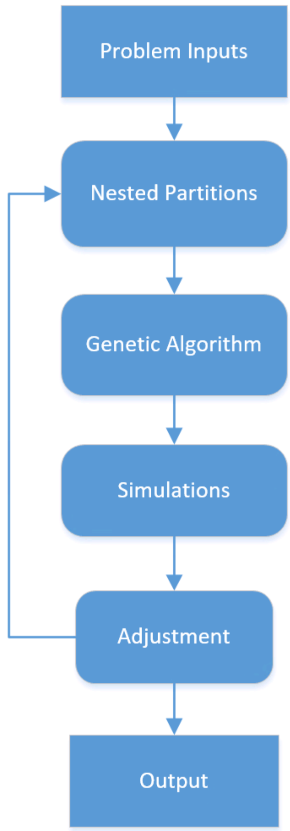


Figure 4.1: A high level flow chart of the Base Concept.

Chapter 5

Completing the Solution Method

The constituents of the solution method, and their order, were decided in Chapter 4. Now, the interaction between the different parts, and their implementation, is to be established. Choosing the method is only half the job, to make an algorithm work as intended it has to be implemented properly.

5.1 Implementing the Metaheuristics

The implementation process starts with the main constituents, the NP and the GA. The key aspect for this section is to make sure that they are customised to perform well on their respective tasks. In addition, emphasis is put on a flexible and intuitive structure. The main mission of the NP is to perform an adequate exploration, and the GA to perform an adequate exploitation. An important aspect of the relation between the two phases, is presented in Figure 5.1. Too early switch between the phases leads to a sub-optimal stagnation, too late leads to a longer computation time. Therefore, the tactic of the current solution method is rather to be too late than too early.

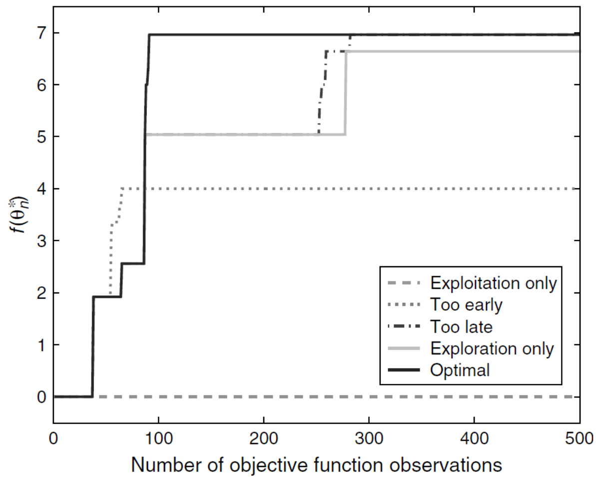


Figure 5.1: Identification of a Proper Switch Point from Exploration to Exploitation, in a maximisation problem. The illustration is borrowed from Andradóttir and Prudius (2009).

5.1.1 Nested Partitions

The decisions to be made for the implementation of the NP are; (i) the partitioning, (ii) sampling, (iii) backtracking, (iv) storage, (v) rounds and (vi) output. Terminating the subsection is a pseudo code of the final NP.

Partitioning

This is the main feature of NP, but there are many different ways to partition the solution space. First, polynomial time increase with respect to an increase in the number of dimensions must be ensured. This means a polynomial increase in the number of evaluations. One way to achieve this is by dividing along one dimension at a time. "To alleviate this problem, we suggest a coordinate sampling scheme which randomly chooses solutions that differ in only one coordinate from the current sample-best solution. Both empirical and analytic evidence shows that this is a much more efficient sampling scheme, especially when the dimension of the problem is high" Hong et al. (2010). The increase in the number of evaluation, with the addition of an extra dimension would then be constant, given a constant range for every dimension.

Sampling

Remembering that the purpose of the NP is to explore the solution space, only choosing one sample from each part at every partitioning seems like a good idea. This facilitates a great variation in the convergence from one round to another. Sampling a great number of solutions at each partitioning, the algorithm would be more likely to choose the same convergence path every time. For the same reason, uniform sampling is chosen.

Backtracking

To further enhance the exploration of the metaheuristic, backtracking is included, with two samples. This means that if the two parts of the promising, currently being investigated, are not covering the whole solution space, two samples are drawn from outside the promising area. These extra samples are uniformly chosen. If one of these has the best solution, the promising area is reset to equal the complete solution space.

Memory

From each round of the NP, the best solution visited during that round is saved. This differs from the usual NP implementation but makes more sense for this problem because the approximate analytic function is deterministic. The reason for not saving every solution which is visited is illustrated by the comparison of the running times of the NP in Table 5.1.

Table 5.1: Comparing the running times of the NP with memory enabled and disabled, as a function of the number of rounds and samples.

	Memory enabled		**Memory disabled**	
No. Rounds	**Running time [s]**	**No. Samples**	**Running time [s]**	**No. Samples**
1	0.0415	285	0.0315	285
2	0.05	558	0.0459	558
3	0.0626	828	0.0465	840
5	0.076	1380	0.0544	1389
10	0.149	2811	0.0708	2784
20	0.379	5541	0.1038	5597
40	1.156	11136	0.1633	11226
80	4.265	22389	0.2879	22335
160	20.631	44481	0.4937	44622
320	91.357	89205	0.9297	89172
32000	-	-	91.8608	8923770

Rounds

The number of rounds decides how long the NP is to run, and thus, how much of the solution space it explores. Naturally, the higher the number the better, but restrictions regarding reasonable time sets a practical limit. Seeing the computational times of Table 5.1, a value of 1000 should yield a good compromise, with a running time of about three seconds.

Output

1000 rounds mean that 1000 solutions are stored from running the NP. These are sorted, based on cost, and then the unique solutions are copied to another list. From running the NP, it is evident that the quality of the solutions vary greatly, and only the best are worth forwarding to the GA. Choosing the 100 best means a great diversity is maintained, and at the same time not wasting too much time on hopeless solutions or areas.

NP Summary

Based on the previous decisions, a description of the complete NP is provided. Initially, the promising area is the complete solution space. A dimension is chosen at random, and the solution space is divided into two equal parts, each covering half of the range of the chosen dimension. A random solution is drawn from each part. The space containing the best solution is labelled as the promising area. Now, a new dimension is chosen, and the process is repeated within the promising area of the previous iteration. If the promising area does not cover the entirety of the randomly selected dimension, two random samples are chosen from outside the promising area, in addition to the two others. If any of the two samples from outside give the best solution value, the promising area is reset to equal the entire range of the dimension in question. When the promising area has become so small that it only contains a single solution, the the best visited solution of the round is stored in memory. Then, the promising area is reset to cover the complete solution space, and the process is repeated 1000 times. After 1000 rounds, identical solutions are removed so that all solutions are unique. The 100 best of these solutions are forwarded to the GA. If there are fewer than 100, all are forwarded.

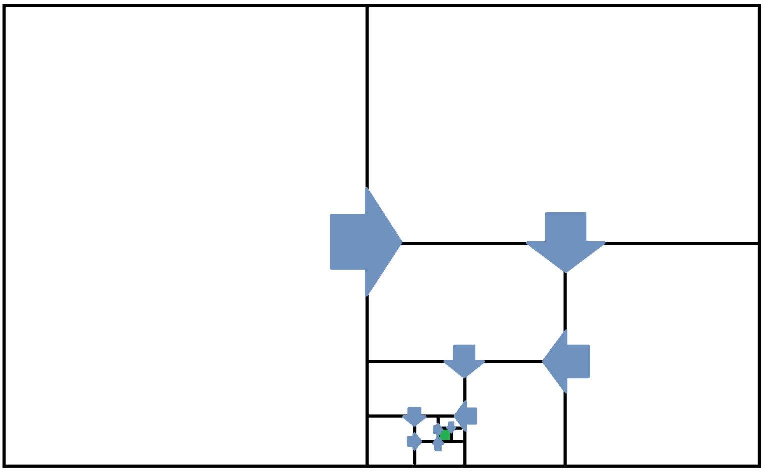


Figure 5.2: 2D conceptual illustration of the search process of the customised NP algorithm. Partitions are made in one dimension at a time.

Customised Nested Partitions pseudo code.

Let *x=x(1),x(2),..,x(N)* be a solution.

Let *f(x) be the objective value of solution x.*

Let *R* be the number of iterations of the algorithm.

Let *BestNPVal(1:R)* be the best solution value of each iteration.

Let *N* be the number of decision variables in a solution.

Let *ARS(d)* be the complete set of allowed values of variable d.

Let $ARS(d)_L$ and $ARS(d)_U$ be the lower and upper halves of the set *ARS(d).*

Let *RS(d)* be the promising area of variable d.

Let $RS(d)_L$ and $RS(d)_U$ be the lower and upper halves of the set *RS(d).*

Let *DP(d,1* and *DP(d,2* be the values of the decision parameter to be tested, from each side of the promising area.

Let *DP(d,3* and *DP(d,4* be the values of the decision parameter to be tested, from outside the promising area.

Let *A* be the number of different values the decision parameter takes on in the current partitioning.

Algorithm 3 Customised Nested Partitions

```
for r = 1 : R do
    BestNPVal(1:R) ← ∞
    for i = 1 : N do
        if ARS(i)_L == ARS(i)_U then
            cont(i) ← 0
        else
            cont(i) ← 1
    contS ← 1
    RS ← ARS
    while contS==1 do
        d ← Uniform N
        if cont(d)==1 then
            Limits(d) ← [RS(d)_L, RS(d)_U]
            DP(d,1) ← Uniform RS(d)_L
            DP(d,2) ← Uniform RS(d)_U
            for k = 1 : N do
                x(k) ← Uniform RS(k)
            if RS(d)==ARS(d) then
                A=2
            else
                A=4
                DP(d,3) ← Uniform (ARS(d) \ RS(d))
                DP(d,4) ← Uniform (ARS(d) \ RS(d))
            BestDP ← ∞
            for a = 1 : A do
                x(d) ← DP(d,a)
                f(x) ← Evaluate x
                if f(x)<BestDP then
                    PromisingArea ← Origin of DP(d,a)
                    BestDP ← f(x)
                    if f(x)<BestNPVal(r) then
                        BestNPSol(r) ← x
                        BestNPVal(r) ← f(x)
            RS(d) ← PromisingArea
            if RS is singleton then
                cont(d) ← 0
            if sum(cont)==0 then
                contS ← 0
return BestNPSol(:)
```

5.1.2 Genetic Algorithm

Implementing the GA, choices have to be made regarding the structure of the algorithm. This means whether to include elements such as selection, crossover, mutation, and elitism, and how to do so. Also, the number of particles, and the number of generations must be set. Not forgetting the intensification.

Particles

The GA starts with importing the list of the best unique solutions from the NP. These solutions are the first generation of the population and are stored as the best solutions of the GA, initially.

Intensification

If a decision variable has the same value for all the solutions transferred from the NP, then this decision variable is fixed to having that value for the complete GA. Also, limits are set for every variable, with a minimum value and a maximum value equal to the respective extremes of the transferred solutions. This is done to increase the exploitation.

Mutation

At the beginning of every generation, the current population is mutated randomly. This means that, every generation and for each particle a random decision variable is chosen, and its value either goes up or down by one. The values always stay within the limits set through intensification. The reason for only changing one variable, and only by one unit, is to maximise the exploitation.

Selection & Reproduction

After the mutated solutions have been evaluated, they are sorted based on cost, and the most expensive 75% solutions are removed. The reason the fraction is set so high as 75% is to increase the exploitation. When reproducing, new solutions are built by putting together the first half of one surviving solution, and the second part of another. Having only two parent solutions, combined in a regular manner, reduces the variation, and thus increases the exploitation.

Generations

The same consideration is made for the choice of the number of generations here, as for the number of rounds in the NP, but the reason is quite different. Here an increased number of generations ensures an increased exploitation of the promising areas. Therefore, as many generations as possible is preferred. As seen in Figure 5.3, 200 generations is a reasonable compromise.

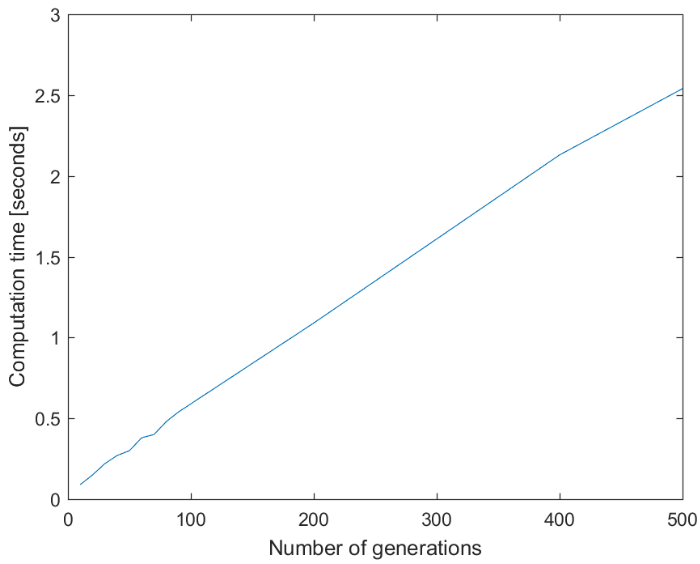


Figure 5.3: GA computation time as a function of the number of generations.

Output

The output of the GA is the list of the best 100 solutions. This list is sorted, to only contain unique solutions. Then only a couple of the best solutions are forwarded to be simulated. Exactly how many is determined in Section 5.2.2.

GA Summary

Based on the inputs from the NP, the limits for intensification are established. The current population is saved as the 100 best solutions. For each generation the particles are mutated, then the best are selected for survival, and finally the remaining population is restored through reproduction. In the mutation stage, for each particle a random decision variable is chosen, and its value is either increased or reduced by a unit. The choice is made randomly, but if the value of the variable is already at the intensification limit, the change is set in the opposite direction. After the mutation the particle is evaluated analytically, and if it is better than any of the 100 best solutions, it is stored.

After every particle of the population has been mutated and evaluated, the worst 75 in the population are removed. Then 75 new particles are constructed from randomly combining the first half of one surviving solution, and the second half of another. These new particles are then evaluated, and those good enough are stored as one of the 100 best. The complete process is repeated for 200 generations.

Customised Genetic algorithm pseudo code

Algorithm 4 Customised Genetic Optimisation

```
x(1 : M,:) ← Input
for j = 1 : N do
    A(j,:) ← [min(x(:, j)), max(x(:, j))]
BestGA ← x(1 : M,:)
for g = 1 : G do
    for i = 1 : M do
        j = Uniform(N \ A(j,1)  = A(j,2))
        x(i, j) ← Uniform[+1,−1], within A(j,:)
        if f(x(i,:)) < f(BestGA(end,:)) then
            BestGA(end,:) ← x(i,:)
            Sort BestGA(1 : M,:) based on f(x)
    Sort x(1 : M,:) based on f(x(i,:))
    x ← x(1 : (M/4),:)
    for i=(M/4 + 1) : M do
        x(i, 1 : (N/2)) ← x(Uniform((M/4) : M), 1 : (N/2))
        x(i, (N/2 + 1) : N) ← x(Uniform((M/4) : M), (N/2 + 1) : N)
        if f(x(i,:)) < f(BestGA(end,:)) then
            BestGA(end,:) ← x(i,:)
            Sort BestGA(1 : M,:) based on f(x)
return BestGA
```

Let *x(i,j)* be the value of decision variable j of solution i.
Let N be the number of decision variables in a solution.
Let $A(j,1)$ and $A(j,2)$ be, respectively, the lower and upper intensification limit of decision variable j.
Let *x(1:P,1:N)* be a population of P particles.
Let M be the population size.
Let G be the number of generations.
Let *f(x(i,:))* be the objective value of *x(i,:)*.

5.2 Implementing Additions

In addition to the metaheuristics, some other important aspects of the implementation must be decided. The adjustment of the analytic function, the size of the simulation sets, the OCBA, and the termination criteria of the solution method.

5.2.1 Adjustment

The purpose of the adjustment is to improve the predictions of the analytic approximation. That is, to make the predictions better match the results of the simulation model. As presented in Equation 4.10, a practical method for adjustment is to add a factor to the equation of each cargo type. There are two choices for what the factor is to represent; (i) the relation between actual delivery and predicted delivery, or (ii) the relation between the actual performance and the restriction threshold. The second is the most intuitive, because when it gets close to one the solutions match the restrictions well. For the first alternative, it is easier to relate to Equation 4.10, but when it approaches one, it is not guaranteed that the solutions approach the threshold of the restrictions.

The value of the adjustment must be chosen in such a way as to ensure a steady convergence towards 1. An easy way to do this is to set it as the 50^{th} percentile of the set. This is more stable than the average, because it may be influenced by extreme values in either of the ends. Thus, adjustment is as presented in Equation 5.1.

$$FB(i) = P_{50}(BulkPerformance) \qquad \forall i \in I \tag{5.1}$$

BulkPerformance is a vector with length equal to the number of simulations in the set. Each element is defined as follows, in line with the constraints from Equation 2.2. I is the set of cycles in the solution method.

$$BulkPerformance(j) = \frac{P_{1-T\_r(1)}(FracBulkLifted(j))}{T\_p(1)} \qquad \forall j \in J \tag{5.2}$$

FracBulkLifted(j) is a vector with a length equal to the number of realisations which the solutions were simulated for. J is the set of solutions in the set. T_r(1) is the desired level of certainty of the solution performance. That is, T_r(1)=0.95 means that a solution has to be 95% certain to perform above the threshold to be accepted. T_p(1) is the threshold. T_p(1)=0.9 means that the solution has to transport at least 90% of the cargo to be accepted.

$$FracBulkLifted(j,m) = \frac{(BulkLifted(j,m))}{WDB(m)} \qquad \forall m \in M \tag{5.3}$$

M is the set of realisations which were simulated. WDB(m) is the average weekly demand of bulk in scenario m. The same calculations are performed for all cargo types.

5.2.2 Simulation Evaluation

Firstly, because of the deviation of the simulated weekly demands of the cargoes, compared to the input value decided by the user, see Appendix D, an extra "zero-concept" will be simulated in every set in addition to the actual solutions. This "zero-concept" has no vessels, and thus the cargo piles up at the base. The reason for this approach is that the simulation model does not return an average weekly demand, but it does return a measure of waiting cargo which can be used to estimate the average weekly demand, by using Equation 5.4. *BulkWaiting* is given in days · m^3. The equation assumes that the graph takes the form of a triangle, Appendix B.6, thus it finds the height of a triangle when the area is given. The 7 is to make it the weekly demand, rather than the daily.

$$WDB(m) = \frac{BulkWaiting(m) \cdot 2}{T^2} \cdot 7 \qquad \forall m \in M \tag{5.4}$$

The next step is to decide the number of solutions in each simulation set. In this regard, two things are important; (i) the set must be big enough to ensure a stable adjustment, and (ii) the set has to cover the variance in the cost-ordered list from prediction to simulation. For the former, the convergence of the adjustment factor is studied as the number of solutions in the set increases. For the latter, the cost-ordered lists from the analytic approximation and the simulations are compared to see how big the variance is.

(i) For this convergence study, Table 5.2, one specific case is studied, and the results are assumed to be valid for all other cases. This is of course not perfect, but other trials seem to give the same results. The tested case is; B=3000, D=2000, P=280, Vessel data 2, and 100 realisations, and the factors are from the first cycle.

Table 5.2: Changes in adjustment factors for the 25 cheapest solutions. Retrieved from the first cycle when B=3000, D=2000 and P=280.

No. Simulations	**[FB, FD, FP]**
1	[1.0008, 0.6573, 0.2891]
2	[1.0056, 0.6582, 0.2979]
3	[1.0008, 0.6591, 0.3068]
4	[1.0008, 0.6705, 0.3134]
5	[1.0008, 0.6591. 0.3200]
6	[1.0008, 0.6705, 0.3134]
7	[1.0008, 0.6818, 0.3200]
8	[1.0015, 0.6832, 0.3134]
9	[1.0008, 0.6845, 0.3200]
10	[1.0015, 0.7991, 0.3195]
11	[1.0008, 0.9137, 0.3200]
12	[1.0008, 0.9169, 0.3200]
13	[1.0008, 0.9202, 0.3200]
14	[1.0015, 0.9326, 0.3233]
15	[1.0021, 0.9202, 0.3265]
25	[1.0083, 0.9451, 0.4088]

Clearly, and as expected, the adjustment factors do not converge, they increase. This is because every extra solution added to the set is more expensive, and probably better performing, than all those that are already in the set. Thus, the bigger the set, the less information the resulting adjustment factors actually give about the relation between the analytic function and the simulation. In addition, as few simulations as possible is a key aspect of OCBA. This means that a compromise is needed, as small as possible, but big enough so that an extraordinarily bad prediction does not shift the value. That is, the set must provide a stable convergence of the adjustment. Figure 5.4 shows that the increase in time is linearly dependent on the number of solutions in the set, which means that a set size of 5 solutions may work well in this respect.

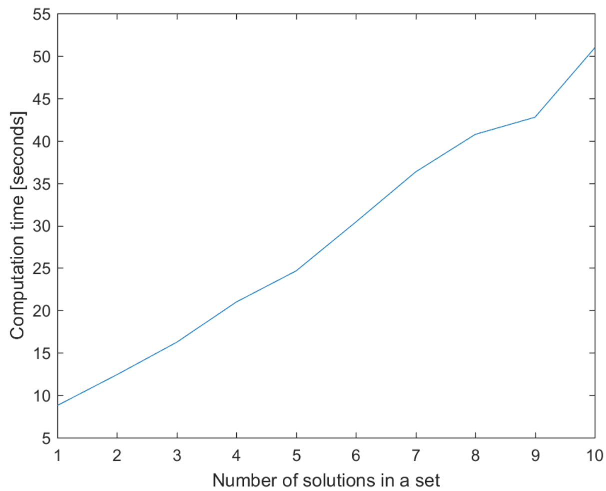


Figure 5.4: Simulation time for a set, as a function of the number of solutions in the set. 100 realisations for every solution.

(ii) The cost ordered variance is tested by comparing a cost ordered list of predictions for the analytic function, with the resulting list of the simulated costs. The greatest number of positions a solution has moved, indicates the variance that can be expected. The same 25 solutions from Table 5.2 are used in this test, in Table 5.3. The results of Table 5.3 indicates that a deviation of 5 positions can be expected, but in general it seems that the cost-ordered lists match very well. Taking this observation into account, the simulation set size is increased to 6 solutions. Experiencing bigger deviations is assumed to be unlikely.

Table 5.3: Mismatch between predicted cost and actual cost. The "identity" of the solutions are in parenthesis, making it possible to see how they "move" in the sorted list from predicted cost to simulated cost.

Predicted cost (ID)	Simulated cost (ID)	Position change
1.232e+07 (1)	6.265e+06 (1)	0
1.282e+07 (2)	6.767e+06 (2)	0
1.302e+07 (3)	6.965e+06 (3)	0
1.424e+07 (4)	8.154e+06 (4)	0
1.431e+07 (5)	8.220e+06 (5)	0
1.434e+07 (6)	8.470e+06 (6)	0
1.451e+07 (7)	8.626e+06 (7)	0
1.484e+07 (8)	8.962e+06 (8)	0
1.501e+07 (9)	9.127e+06 (9)	0
1.504e+07 (10)	9.166e+06 (10)	0
1.521e+07 (11)	9.324e+06 (11)	0
1.551e+07 (12)	9.625e+06 (12)	0
1.571e+07 (13)	9.826e+06 (13)	0
1.591e+07 (14)	1.002e+07 (14)	0
1.594e+07 (15)	1.007e+07 (15)	0
1.602e+07 (16)	1.014e+07 (16)	0
1.621e+07 (17)	1.035e+07 (17)	0
1.628e+07 (18)	1.042e+07 (18)	0
1.635e+07 (19)	1.051e+07 (20)	-1
1.638e+07 (20)	1.051e+07 (21)	-1
1.638e+07 (21)	1.057e+07 (22)	-1
1.644e+07 (22)	1.058e+07 (23)	-1
1.646e+07 (23)	1.064e+07 (24)	-1
1.652e+07 (24)	1.065e+07 (19)	+5
1.653e+07 (25)	1.078e+07 (25)	0

5.2.3 Optimal Computing Budget Allocation

In this section, a decision will be made as to how the OCBA of the solution method is to be implemented. The previous section decided the number of solutions to be simulated in each set, and the number of sets being simulated is a user decision with respect to the termination criteria of the next section. Therefore, the final aspect regarding OCBA is to decide the number of realisations in each simulation set. There are two allocation tactics which seem to be the most obvious; (i) gradually increasing the number when approaching good solutions, or (ii) having a small number in an initial phase, before increasing to a high number in the final phase. The first may yield the best savings, but is harder to implement, because of the need for some function adjusting the number of realisations from one simulation set to the next. The second may also yield good savings, but it necessitates knowledge about the quality of the solutions beforehand in order to know exactly when to increase the number of realisations.

In this thesis an implementation of the second suggestion will be attempted. The necessary preliminary knowledge will be provided by running the algorithm to termination with a low number of realisations. After termination, the algorithm will be ran again with a high number of realisations, starting with the adjustment factors of the best known solution from the previous run, establishing a more certain confidence interval for the solutions. If the solutions are still valid, they are returned as the output of the algorithm to the user.

What exactly a low number and a high number means can be defined as follows. The low number must be as small as possible, but big enough to return realisations which, in a satisfactory way, reflects the extreme values of the distributions of interest. The high number must be as small as possible, but big enough to, in a satisfactory way, completely reflect the distributions. The distributions of interest are, in this case, the weekly supply of bulk cargo, deck cargo and persons brought to the base. The definitions are illustrated in Figure 5.5, which is one of three distributions from one of three experiments to find the values. The two other distributions, and the distributions of the two other experiments may be seen in Appendix D.

By visual inspection, it is not obvious what the low number should be. The first criteria is that it should be significantly lower than the high number, therefore, the process is simplified by choosing that number first. The high number seems to be in the area of plot 10 - which translates to somewhere around 256 realisations. Without further discussion the choice falls on 250. Now, the low number may be chosen from the range of the first 9 plots. Seeing a stable increase in the range for every distribution, from plot 1 through plot 6, the decision falls on 75. A round number, close to the 76 of the 6^{th} plot. Only rarely, even for a much higher number of realisations, does values outside those of plot 6 occur. Also, 75 is still a significantly lower number than 250, yielding time savings of 70%.

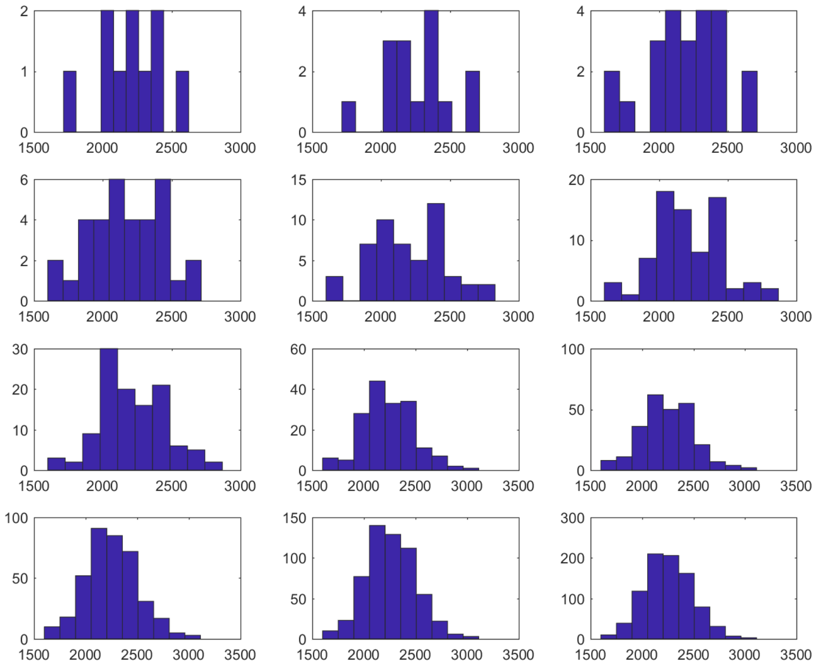


Figure 5.5: Distribution of realisations of bulk supply. The user input is Bulk=2000. x-axis=tons/week, y-axis=realisations. The total number of realisations in each subplot is, from upper left: 10, 15, 23, 34, 51, 76, 114, 171, 256, 384, 577, 865.

5.2.4 Termination Criteria

The termination criteria (TC) defines when the solution method terminates, when one, or a set, of the criteria are met. For this solution method four criteria will be established. If any one of these are met, the solution method terminates. The first is a maximum total duration. This being a strategic problem, 1 hour is suitable. This criterion will be called **TC1** in the remainder of the report. The three other termination criteria are defined by the development of the search process.

First, a two-cycle start-up phase is ran, with a low number of realisations. This is because

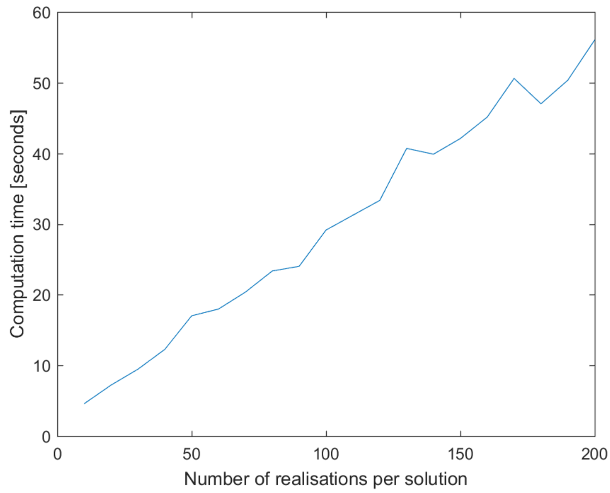


Figure 5.6: Simulation time for a set, as a function of the number of realisations. 6 solutions in the set.

the predictions of the analytic function may be way off initially and needs to be adjusted. Thereafter, the process checks the termination criteria every cycle, still with the low number of realisations. If a TC is met, the final phase is initiated, meaning a high number of realisations when simulating, and the best solutions are ran. If any of them prove to be legal, the search process is terminated. This termination process is illustrated in Figure 5.7.

TC2

No change in the set of solutions to be simulated from one turn to the next. This means that the algorithm has stagnated and is unable to find alternative solutions it believes to be better. Three scenarios may give this situation: The solutions are at the absolute lower end of the solution space, or the opposite, that the solutions are at the absolute upper limit of the solution space. For very special cases the third situation is an option - the adjustment is so small that, even though the solution set is in the middle of the solution space, the complete solution set of the previous turn is kept through the next. In the case of the latter, the situation is transitory because the solutions will yield factors less than 1 for every turn, eventually introducing new solutions. An example of TC2 is illustrated in Figure 5.8

TC3

Given that a legal solution is already found, the algorithm fails in improving the best solution during the next two turns. Hopefully, this happens because the optimal solution is found, but it may also be a different explanation; TC3 may happen when, after a good solution is found, the adjustment increases the cost of the next sets drastically, and it takes several cycles before the adjustment is back down. An example of TC3, though not very realistic, is illustrated in Figure 5.9.

TC4

If a legal set and an illegal set has solutions in common. This is a TC because finding a whole set of only illegal solutions indicates that there are very few legal solutions in the area. Thus, finding a legal solution in the vicinity of an illegal set could mean that the legal solution is close to optimal. An example of TC4 may be seen in Figure 5.10.

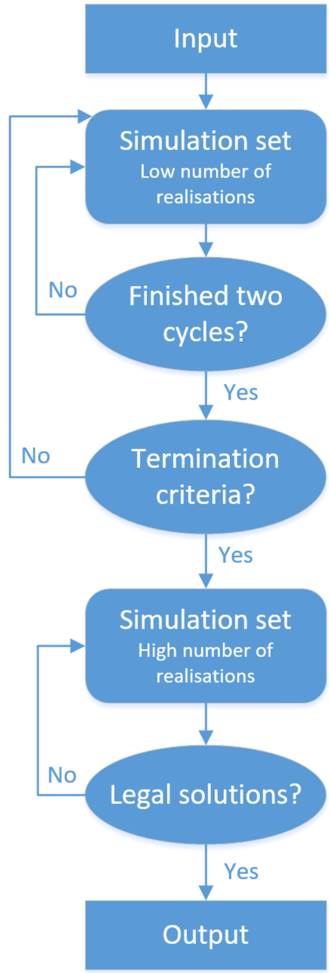


Figure 5.7: A flow chart of the complete termination process. First two initial cycles must be finished, then the termination criteria must be met, and finally at least one solution must be legal.

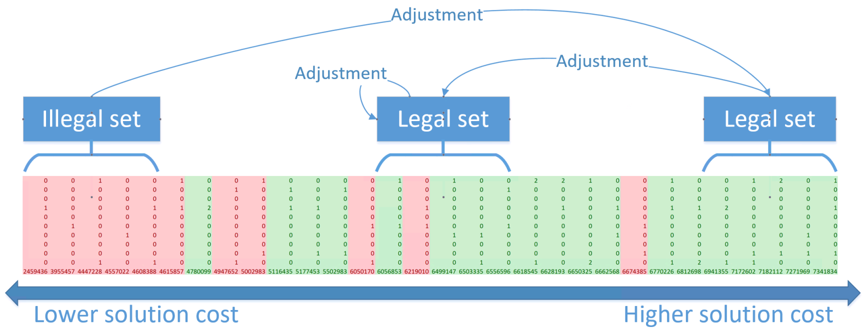


Figure 5.8: Termination Criteria 2. The simulated set does not change from one cycle to the next.

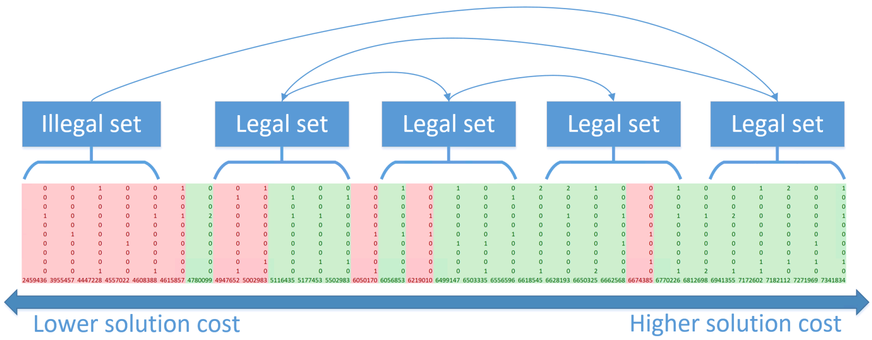


Figure 5.9: Termination Criteria 3. The best solution does not improve for the next two cycles.

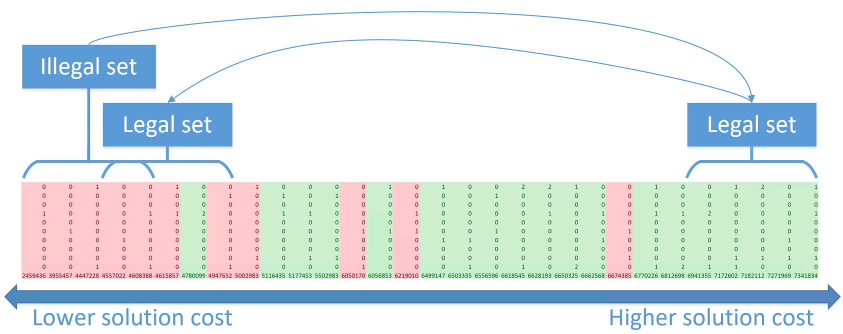


Figure 5.10: Termination Criteria 4. A legal and an illegal set is found which have at least one solution in common.

5.3 Summary of Solution Method

The complete algorithm, as defined so far, is presented as a flow chart in Figure 5.11. The full, detailed implementation may be seen in Appendix C, where the MATLAB code is displayed. In the block "Input from user" the case is defined by inputting the following, see Appendix A.1:

- Bulk, deck and pax demand pr week
- Vessel concept list
- Characteristics of platform
- Characteristics of base

The output the user gets back is a table of solutions on the form given by Table 5.4. In the example solution displayed, the threshold reliability is 95%, and the threshold performance is 90%. Also, the cost is defined as the 95th percentile value. As can be seen, the P_5 values are all above 90%, thus the displayed solution is legal. This is also a requirement for a solution to be returned to the user. The solution uses one vessel of type 5, and one of type 9.

Table 5.4: On what form legal solutions are returned to the user.

Solution	P_{95}(Cost)	P_5(BulkPerf)	P_5(DeckPerf)	P_5(PaxPerf)
[0 0 0 0 1 0 0 0 1 0]	$6.265e+06	90.07%	91.34%	90.61%

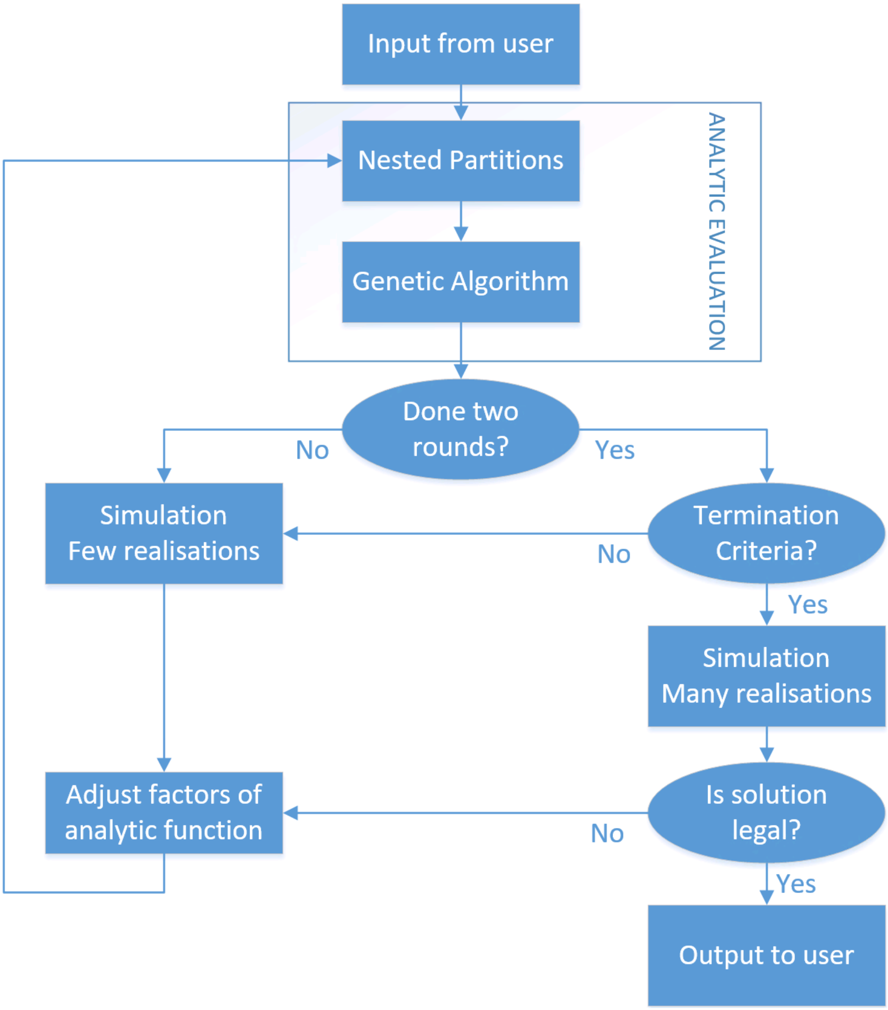


Figure 5.11: A high level flow chart of the complete solution method.

Chapter 6

Testing and Results

In this chapter, the solution method presented in Section 5.3 is tested to see how well it meets the criteria of finding good solutions within reasonable time, and being flexible and intuitive. The chapter starts, in Section 6.1, by designing the experiments necessary to test for the criteria, and by explaining the execution process as it is performed by the user. Thereafter, the tests are performed in Section 6.2, presenting all relevant results.

6.1 Preparations

Before performing the tests, certain preparations must be made. The tests have to be properly designed, in order to ensure that they yield the desired information about the solution method. Furthermore, the interaction between the solution method and the simulation model has to be established.

6.1.1 Experiment Design

For the testing to be of any value, it is important that the results actually give valuable information in relation to the purpose and criteria of the development of the solution method. At the same time, performing experiments is a tedious process and therefore smart design, reducing the number of tests necessary, may save a lot of time. By running the experiments, the following three questions are to be answered:

1. Does the solution method find good solutions within reasonable time?
2. How does computation time for the solution method increase with increasing problem size and vessel concepts list?
3. Is the solution method flexible?

For the tests to answer these questions, first the various terms must be defined.

Good solutions may be defined in several ways: (i) Those that can be proven to be optimal. (ii) Those which are as good or better than the solutions found by traditional methods. (iii) Those that are within some limit from an optimistic bound. Attempting (i) and (iii) is not realistic, therefore the choice falls on (ii). This means that the results of the solution method are to be compared to those of expert judgement. The search process of the expert judgement approach is completely independent from the results of the solution algorithm. Also, the goodness of a solution may be defended with qualitative arguments regarding the specific problem.

Reasonable time is defined as having a linear relation between the computation time, and a change in any single input parameter. This is tested both by comparing the solution times for the complete solution method, and the computation times for the constituents, for the different cases. In addition, there is an upper time limit, set by TC1 to 1 hour.

Flexibility is measured by the amount of work needed to modify the solution method for a different case. If only seconds or a few minutes are needed, the solution method is flexible. If several hours or more is needed, it is not flexible. The solution method should be flexible for all cases which can be evaluated by the simulation model. In order to test this, the cases have to reflect the range of problems that are possible to test with this set up. This means; different weekly transport requirements, different platforms, different characteristics in the base, and different lists of vessel concepts.

Following the previous discussion, the solution method is to be tested for 9 different cases, presented in Table 6.1. The explanations of the H's and P's are presented in Table 6.2, and a visualisation may be seen in Appendix B.2. For every case, the search process is logged, including complete solution time, the time necessary to prepare for the case, the number of cycles, the tested solutions, and the related adjustment factors.

Table 6.1: Definitions of the cases which are used in the experiments. All other parameters are equal for all of the cases. The platforms are listed in order of visit.

Case	Vessel data	Restrictions	Base	Platforms
1	VD1	[2000 1200 70]	H1	[P2 P4]
2	VD1	[4000 2400 140]	H2	[P1 P3 P5 P6]
3	VD1	[8000 4800 280]	H3	[P1 P2 P3 P4 P5 P6 P7]
4	VD2	[2000 1200 70]	H1	[P2 P4]
5	VD2	[4000 2400 140]	H2	[P1 P3 P5 P6]
6	VD2	[8000 4800 280]	H3	[P1 P2 P3 P4 P5 P6 P7]
7	VD3	[2000 1200 70]	H1	[P2 P4]
8	VD3	[4000 2400 140]	H2	[P1 P3 P5 P6]
9	VD3	[8000 4800 280]	H3	[P1 P2 P3 P4 P5 P6 P7]

Table 6.2: Coordinate locations and specifications of the onshore home-bases, H, and offshore platforms, P. Bulk loading rate(BLR), deck loading rate(DLR), inter vessel departure time(IVD).

Name	Position (N,E) [deg]	BLR [ton/h]	DLR [m^2/h]	IVD [h]
H1	70.7, 23.7	120	100	72
H2	70.7, 23.7	240	200	36
H3	70.7, 23.7	480	400	6
P1	73, 20	100	50	-
P2	74, 27	100	50	-
P3	75, 32	100	50	-
P4	76, 38	100	50	-
P5	73, 37	100	50	-
P6	73, 30	100	50	-
P7	72, 23	100	50	-

To finalise the case definitions, the threshold reliability and performance have to be set for each cargo type, as described in Section 2.3. In addition, the probabilistic measure of interest of the cost function has to be determined. Because there are no special reasons for choosing any specific value for each of them, the following choice was made: Threshold reliability for all cargo types; T_r(1)= T_r(2)= T_r(3)=95%. Threshold performance for all cargo types; T_p(1)=T_p(2)=T_p(3)=90%. Measure of the cost function; the 95^{th} percentile.

This means that, for a solution to be legal, the 5^{th} percentile value of the weekly average of delivered cargo must be at least 0.9 times the demand, for all cargo types.

6.1.2 Process

Before performing the tests, the solution method must be connected to the simulation model. The solution method is implemented in MATLAB and the simulation model is a runnable .jar-file which reads input from two excel-files and writes to a .csv-file. Opening the simulation model and loading the input-files is done manually. Also, saving the simulation results to the .csv file is done manually. This means that the simulation evaluation of a set, that is, once every cycle, has to be initiated by the user. Therefore, the solution method is divided into three MATLAB-scripts, with the connections seen in Figure 6.1. The input-files may be seen in Appendix A.1, the output-file may be seen in Appendix B, and the MATLAB scripts may be seen in Appendix C.

6.2 Execution

First the solution method is tested for the given cases, then the expert judgement approach is tested for comparison. Finally, the computation times of the solution method constituents are investigated.

6.2.1 Solution Method

The solution process of solving case 1 with the constructed solution method is displayed in its log in Table 6.3. Three cycles were performed. In each cycle the six solutions that were evaluated through simulation are presented. The "Vessels" column describes the fleet composition, the two following columns gives the 95% confidence values for cost and performance. "Legal?" says if the solution meets the requirements or not, and "Adjustment" presents the last adjustment factor included when finding the solutions of the set. For example, the first solution of the first cycle only included one vessel of type 6, had a cost of $

$3.89 \cdot 10^6$ and performed 0.94, 0.77 and 0.95. Because P_5(D) is less than 0.90, the solution is not legal. The adjustments used to find this set of solutions was FB=1, FD=1 and FP=1.

Table 6.3: A log of the solution process for Case 1.

Cycle nr.	Solutions			Legal?	Adjustment
	Vessels	**P_{95}(Cost)**	**P_5(B), P_5(D), P_5(P)**		
1	C_6=1	3.89e+06	0.94, 0.77, 0.95	No	FB(0)=1
	C_7=1	4.38e+06	0.78, 0.95, 0.95	No	
	C_4=1 C_{10}=1	4.63e+06	0.89, 0.92, 0.91	No	FD(0)=1
	C_4=1 C_6=1	6.22e+06	0.94, 0.95, 0.78	No	
	C_4=1 C_9=1	6.99e+06	0.94, 0.95, 0.95	Yes	FP(0)=1
	C_4=1 C_7=1	6.67e+06	0.94, 0.95, 0.80	No	
2	C_6=1	3.90e+06	0.93, 0.74, 0.95	No	FB(1)=1.040
	C_7=1	4.38e+06	0.79, 0.93, 0.95	No	
	C_4=1 C_{10}=1	4.63e+06	0.91, 0.92, 0.92	Yes	FD(1)=1.051
	C_4=1 C_6=1	6.22e+06	0.92, 0.92, 0.76	No	
	C_4=1 C_9=1	6.99e+06	0.92, 0.92, 0.95	Yes	FP(1)=1.034
	C_4=1 C_7=1	6.71e+06	0.92, 0.92, 0.85	No	
3	C_6=1	3.90e+06	0.92, 0.74, 0.95	No	FB=FB(0)
	C_7=1	4.38e+06	0.78, 0.91, 0.95	No	
	C_4=1 C_{10}=1	4.63e+06	0.91, 0.89, 0.89	No	FD=FD(0)
	C_4=1 C_6=1	6.22e+06	0.92, 0.91, 0.77	No	
	C_4=1 C_9=1	6.99e+06	0.91, 0.91, 0.95	Yes	FP=FP(0)
	C_4=1 C_7=1	6.70e+06	0.92, 0.91, 0.88	No	

Case 1 returned the same solution sets in cycle 1 and cycle 2, thus meeting TC2. The third cycle is performed to confirm the legality of the best found solution, having 250 realisations as opposed to the 75 of the previous cycles. Legality was confirmed, and the solution was returned to the user; a fleet composition of C_4=1 and C_9=1, costing \$$6.99 \cdot 10^6$.

While the logs of all the other cases are presented in Appendix E, a summary is shown in Table 6.4.

Table 6.4: Main results of test-runs of the solution method on cases 1 through 9. The complete log of each search process may be found in the Appendix, as indicated.

Case [-]	Preparation time [min:sec]	TC [-]	Search time [min:sec]	Process log [-]	Lowest cost [$]
1	1:40	2	3:15	Appendix E.1	6.99e+06
2	1:26	4	4:15	Appendix E.2	9.20e+06
3	0:50	2	16:16	Appendix E.3	2.07e+07
4	1:52	2 & 3	5:33	Appendix E.4	2.45e+06
5	1:20	4	6:55	Appendix E.5	5.65e+06
6	0:52	3	29:07	Appendix E.6	1.73e+07
7	1:30	2 & 4	3:34	Appendix E.7	2.45e+06
8	1:27	2 & 4	10:13	Appendix E.8	7.18e+06
9	0:43	3	20:50	Appendix E.9	1.70e+07

6.2.2 Expert Judgement

An attempt was made to solve Case 1 using expert judgement. The approach was simply to make educated guesses for good solutions and simulate them to get feedback on the choices. The solutions were chosen in sets of three, and the simulations included 75 realisations. The best solution was verified by running a simulation with 250 realisations. The result was a best solution of C_4=1, C_7=1 and C_{10}=1, costing $\$8.78 \cdot 10^6$, found and verified after 19 minutes and 30 seconds. Attempts at solving other cases seemed to give similar results; somewhat worse solutions and a quick increase in complexity and resulting increase in solution time.

6.2.3 Duration Dependency

In this section there will be a short presentation of the time consumption of the various parts of the solution method. Two main consumers, the search process and the evaluation, are presented in the following. These are chosen because their duration is dependent on factors which may vary from one problem to another. Summation of the presented durations will not add up to the durations of the complete processes as presented in the previous section. This is due to three factors: (i) The durations of the necessary manual actions are not included, (ii) the duration of other necessary code is not included, and (iii) the duration of read and write processes between different files is not included.

Search Process

NP and GA are tested for computation time dependency on the number of possible vessel concepts, and the amount of weekly cargo demand.

Table 6.5: Search time as function of problem size and number of vessel concepts. The durations are those of the final cycle of each case, as found when solving the cases in the previous subsection.

Case number [-]	Number of concepts [-]	NP duration [seconds]	GA duration [seconds]
1	10	0.26	1.41
2		0.52	1.71
3		0.52	1.47
4	21	1.25	0.99
5		1.25	0.79
6		1.23	0.64
7	42	1.97	0.55
8		2.85	0.58
9		3.08	0.53

Clearly, there is some variation in the computation times of both the NP and the GA. For the NP the computation time increases for both increased cargo demand, and even more so for an increased number of vessel concepts. The GA, on the other hand, has the exact opposite behaviour.

Evaluation Process

The computation time of the analytic function is a function of the number of vessels concepts in the current Vessel Design list. This is because the only elements that vary in duration are for-loops going through each concept in the list, Appendix . In addition, the linearity of the analytic function is evident by the computation times in Table 6.5. In 1.50 seconds, 1 000 000 analytic evaluations of 10-concept solutions are performed.

The variation in time of the simulation model as a function of the number of realisations or the number of solutions in the set may be seen in Section 5.2. Moreover, the simulation time does naturally not depend on the size of the current Vessel Data list. Computation time, as a function of problem size, is presented in Table 6.6.

Table 6.6: Computation times of simulations of the final sets of Case1, 2 and 3. 75 realisations.

Case number	Simulation times [seconds]
1	23.37
2	27.23
3	53.96

The simulation model using discrete event simulation (DES), an increase in time, following increased weekly demand, is expected. This comes as a result of a larger fleet performing more actions, which in turn take more time to compute. To compare the times with the analytic function; the times from Table 6.6 equals 13-29 simulation evaluations in 1.50seconds.

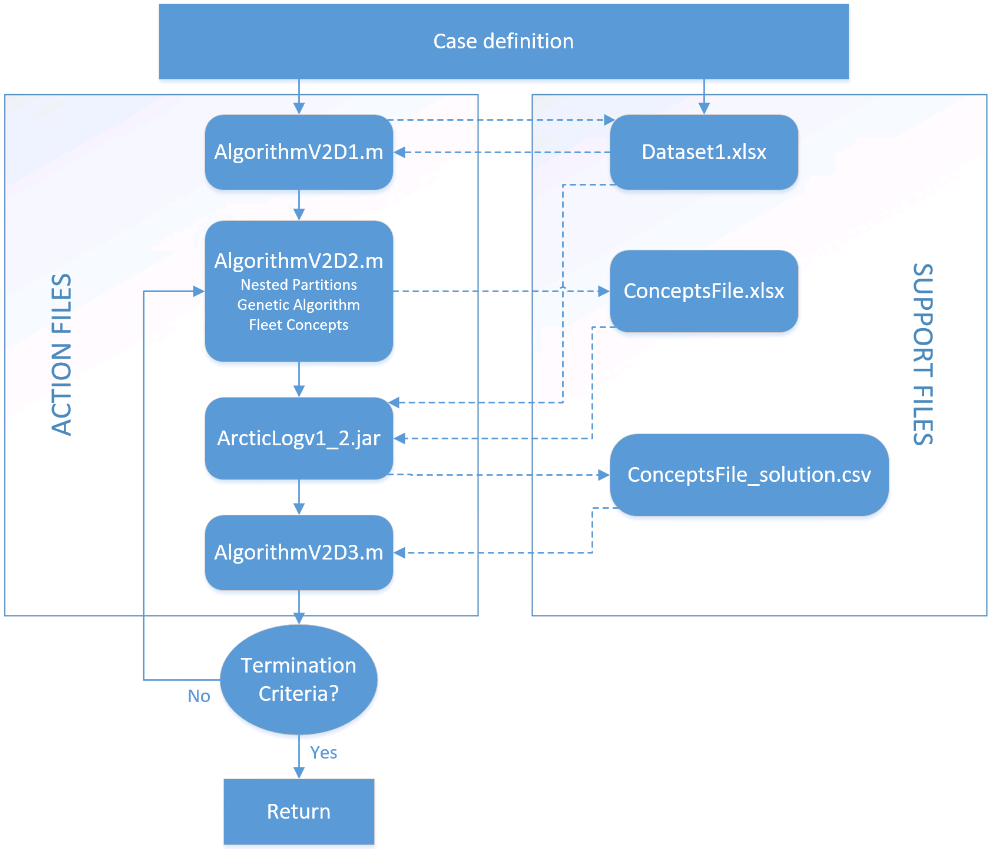


Figure 6.1: Algorithm execution flow chart. The user inputs data to both AlgorithmV2D1.m and DataSet1.xlsx. Then the action files are ran in the order displayed. See Appendix A.1 for details on "Dataset1.xlsx" and "ConceptsFile.xlsx". See Appendix C for details on the .m-files. See Appendix B for details on "ConceptsFile_solution.csv" and the "Return".

Chapter 7

Discussion

In this chapter, the performance and suitability of the solution method is discussed. The discussion will be based on both the results of the tests performed in Chapter 6, and the insight given by some relevant literature. First, in Section 7.1 the quality of the solutions found by the solution method will be discussed. Section 7.2 delivers a thorough assessment of the method suitability.

7.1 Solution Quality

Not having proven optimality, or even established an optimistic bound, the discussion will be based on logical assumptions and observations. The goal is to determine whether or not a statement about the goodness of the solutions may be supported at all, and if so; how good they are. In this case the goodness of a solution is, of course, measured by the cost of the solution, but it is important to note that for the costs to bear any information, the values have to be of sufficient certainty. As discussed in Section 5.2, the key point is that the order of the cost list of the analytic function gets as close to the true one, as possible. This means that, after making the solutions that are assumed to be illegal infinitely expensive, if the order of the list matches the true order, the rest is up to the search of the metaheuristics. The complete process is discussed in the following.

7.1.1 Assumptions

The first assumption made in the construction of the solution method is that the decision maker prefers to obtain a good solution within reasonable time, rather than finding the optimal solution within indefinite time. This is reasonable for practical problems, as confirmed by Juan et al. (2015). Although optimality is sought after, a decision maker can't wait for the first legal solution forever.

The second assumption is that there exists, for every problem case, a true sorted list based on some measure of the costs of all the solutions, in this case the 95th percentile value. This means that if every solution is simulated enough times, the sorted list would reach some steady state order, placing every solution in its true position. The assumption is based on the law of large numbers, which is believed to hold because every stochastic parameter is based on some set of probabilities. The behaviour of the entities in the system, and their interaction, on the other hand, is not certain to reach a steady state.

The third assumption is that only a finite number of realisations of each solution is necessary to establish the true order of the list. This is the first step towards a manageable solution time. The idea is that the distributions of the different stochastic elements may be observed to converge as the number of realisations increases. At some point the distributions are certain to the degree that more realisations give no extra information. This is only done for the weekly distributions; where clear normal distributions were seen. Maybe 250 realisations were too few, but at the same time the decision was based on a balance between computation time and accuracy. Also, only testing the distributions of the weekly demands greatly reduces the certainty of the solution quality. The effects of other stochastic elements such as weather and interaction between the vessels, may not reach a steady state at the same number of realisations, if ever.

The fourth assumption is that it is possible to say anything about the order of the list without performing any simulations. That is, the problem has a structure which can be exploited to predict, to some degree of certainty, the quality of solutions in relation to each other. The more accurate such analytic models may be in predicting the quality of the solutions, the fewer simulations are necessary. Therefore, putting some effort into the analytic model may well be worth it time-wise. This is clearly a reasonable assumption for this problem, but how accurate the analytic model actually is, is an important and difficult question. The number of necessary simulations is based on the variability of the analytic list, compared to the list based on 250-realisation simulations. This was the reason for choosing a 6-solutions set, but the choice was made after investigating a list of only 25 solutions.

As seen in the tests of Chapter 6, the analytic models were often far off in their initial predictions. Meaning that there were some important features of the simulation model which it did not consider. This issue was addressed by incorporating an adjustment of the transport performances. The fifth assumption is that a common adjustment of the restriction specific performance of every solution leads to the best solution. The cost is not adjusted because the exact value is not of importance in the analytic model, only the relations between the solutions. The validity of the assumption is hard to prove, but intuitively it is obvious that it is helpful in directing the search. Also, the implementation of the adjustment, leading the search to the "intersection" between legal and illegal solutions seems logical, but certainty about the validity of the approach is not established.

The sixth and final assumption is that the metaheuristics are able to identify the best solution in the deterministic solution space of the analytic function. The metaheuristic being globally convergent (Hong and Nelson, 2006), the optimal solution will be found, given enough time. However, even for the finite solution space in the current problem, optimality may not be proven in reasonable time. Therefore, the assumption does not hold, but assuming the identification of close to optimal solutions seems reasonable given that the metaheuristics are well designed. The convergence of the NP in Figure 7.1, may suggest that close to optimal solutions are found within reasonable time.

The solution method reaching termination criteria 2, 3 or 4 for every case indicates that the solutions are good. TC2 and TC3 requires stagnation of the search process. This is not, in itself, an evidence for high solution quality, but the fleet compositions of the solutions, and the cost reductions from the first cycle to the last, are strong indicators of that. For the solutions which were terminated by TC4, the assumption of the variability of the predicted list from the true one, is crucial. However, TC4 is much more certain than the assumption, meaning that TC4 may hold even if the variability is twice as large as assumed.

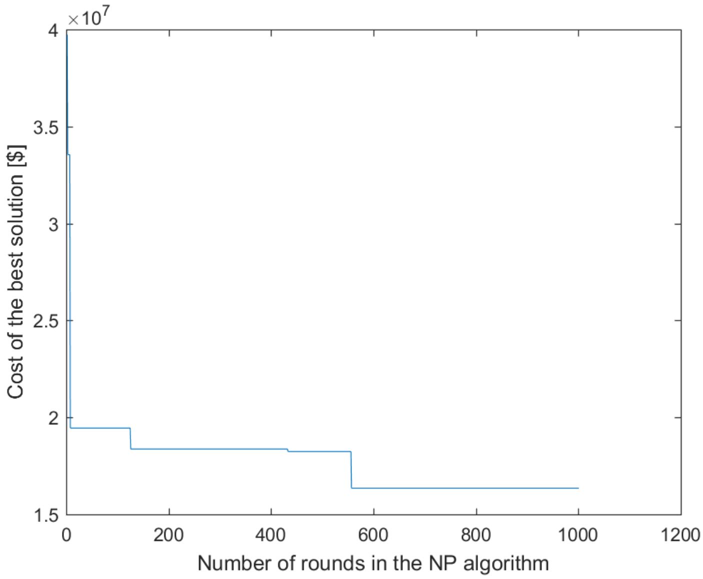


Figure 7.1: The convergence of the value of the best solution found by the NP algorithm. The test case is the last cycle of Case 9. 1000 rounds equated to 3.7seconds.

7.1.2 Observations

In order to evaluate the goodness of the solutions with respect to observations, some basis for comparison must be established. The "Expert judgement" approach attempted in Chapter 6, and the behaviour of the solution method when solving the cases, is discussed in the following.

Expert judgement means choosing solutions which are believed to be good, by some expert, and then simulate them with a high number of realisations, to get a proper evaluation. An expert interprets the results, before either simulating new solutions, or terminating the search. Case 1 was solved with this approach. The result was not convincing; after spending four times the time, an inferior best solution was found. However, it is worth to mention that no formal or structured approach was used, only the intuition and gut feeling of the author. Thus, the result may not prove anything, but a comparison of the methods makes it clear that the constructed solution method simply may be described as an automation of the traditional method, which would lead to the conclusion that the solution method is faster, if not also better. Relating this to the quality of the solutions of the solution method, strictly speaking proving optimality is not necessary if the solutions are better than those of the current method.

During the tests, some peculiar behaviour was observed, as results of the assumptions being imperfect. The first is related to the third assumption. In Case 8, during a 250-realisation simulation, a solution was found to be illegal, then, in the next 250-realisation simulation, it was found to be legal. This proves that the assumption does not hold for 250-realisations, a higher number is needed.

The second peculiarity is related to the sixth assumption. Although using the same factors in Cycle 9 and Cycle 12 of Case 6, the sets are not identical. This means that the metaheuristic does not manage to find the best solutions when searching the solution space, if it did; identical adjustment factors would yield identical sets. The metaheuristics may still be very good, but they are not perfect. A similar observation was made for Cycle 7 and Cycle 10 in Case 9, where it resulted in finding a new best solution in the 250-realisation simulation.

7.2 Method Suitability

Usually, for metaheuristics solving large-scale problems, the most certain measure of method suitability is how quickly it identifies solutions of a certain quality, compared to other methods (Gosavi, 2015). In particular, methods are compared to the performance of Pure Random Search (PRS). However, when PRS was tested on Case 9, it failed to identify any legal solutions. While this indicates that the constructed algorithm is, at least somewhat, suited for the problem, the poor performance of the PRS was expected. This is because the legal solutions constitute a very small part of the solution space; too few vessels and the capacity is too low, and too many vessels and a queue builds up reducing the capacity. Therefore, the probability of finding a legal solution through PRS is very small. This problem of a low concentration of legal solutions is solved in the NP by exploiting the trend in the solution space. For all the cases in question a lower number of vessels are needed, therefore when the algorithm compares two equal solutions, the one with the lower decision variable value is preferred. This trend suits the NP algorithm very well, because it means that most of the partitions are decisive, in that they are most likely not backtracked. Thus, the NP is very efficient.

In the following, the rest of the structure and constituents of the solution method is evaluated using the taxonomy and categorisation scheme developed by Figueira and Almada-Lobo (2014). They categorise methodologies based on four dimensions; (i) Hierarchical structure, (ii) Simulation purpose, (iii) Search scheme and (iv) Search method. Each dimension is divided into four levels, which in total covers the complete range of the dimension. The two illustrations Figure 7.2 and Figure 7.3 show the division of the dimensions, and how different methods fall into the two two-dimensional tables.

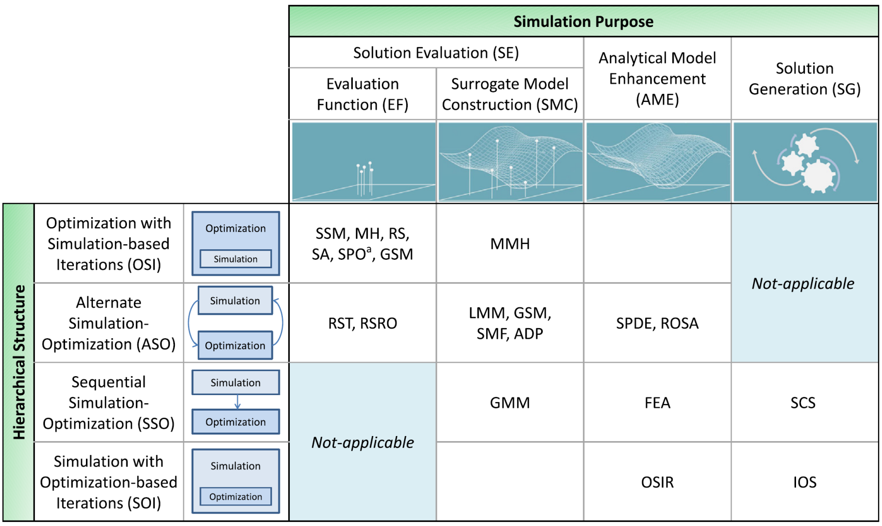


Figure 7.2: "Classification according to the interaction between simulation and optimization: simulation purpose and hierarchical structure." - Figueira and Almada-Lobo (2014)

The constructed solution method has an Optimisation with Simulation-based iterations (OSI) structure in the first dimension. For every iteration in the optimisation algorithm, a complete run of the simulation model is performed. In the second dimension, the current method falls into the Analytic Model Enhancement (AME) category. AME is defined by

Figueira and Almada-Lobo (2014) as using a "problem specific model which is modified according to the simulation output".

Search Scheme		Search Method: Exact (E)	Continuous-space Heuristic (CH): Derivative-Based Heuristic (DBH)	Continuous-space Heuristic (CH): Other Continuous-space Heuristic (OCH)	Discrete-space Heuristic (DH)
	One realization for each solution (1R1S)	IOS, OSIR	SA, GSM, LMM[a], GMM[a]	SMF[a]	RS[b], ADP
	Different realizations for each solution (DR1S)	SSM[a], SCS, ROSA[b]	LMM[b], GMM[b]	MH[b], MMH[b], SMF[b]	MH[a], MMH[a], RS[a]
	Common realizations for each solution (CR1S)	SSM[b], SPDE, FEA	SPO[a]		
	One realization for multiple solutions (1RMS)	RSRO[a], ROSA[a]		RST[a]	RST[b]

Figure 7.3: ""Classification according to the search design: method and scheme."" - Figueira and Almada-Lobo (2014)

In the Search scheme dimension, the algorithm is placed somewhere between Different realisations for each solution (DR1S) and Common realisations for each solution (CR1S). The solutions which are simulated in the same set are given the same realisations, but the realisations change from one set to the next. However, as described in Section 7.1, the idea is that the many realisations would equate the effect of common realisations, although this was proven to be false for the number used in the tests. Finally, in the Search method dimension, the solution method belongs to the Discrete-space Heuristic (DH) category. The solution space is discrete, and a Nested Partitions algorithm and a Genetic Algorithm are employed.

Subsequently, the solution method constructed in this report is an AME-OSI/DH-[DR1S,CR1S], according to Figueira and Almada-Lobo (2014). There are no general examples of similar solution methods in Figure 7.2 and Figure 7.3, with Memory-based metaheuristics (MMH) as the closest. Based on the classification of the algorithm, it is suitable for problems of the following type:

- It is practicable to construct *a priori* a useful analytic model, but it needs to be improved. (AME)

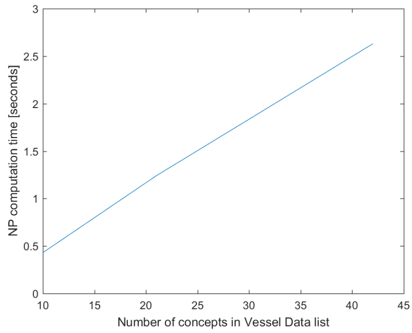
- The optimisation is highly dependent on feedback from the simulation model. (OSI)
- The problem is difficult and combinatorial. (DH)
- It's harder to travel the solution space than the probability space, and both diversification (DR1S) and convergence is important (CR1S).

This description fits well with the problem described in Chapter 2. Especially, AME, as pointed out by Figueira and Almada-Lobo (2014): "Typically, when a linear relationship between most input and output variables exists and is known, but there are particular aspects difficult or undesirable to be included in the analytic model, AME can be very efficient and effective". This is exactly the case for the problem in question; for elements such as the weather, the demands and the effect of the loading rates and opening hours on the flow of vessels.

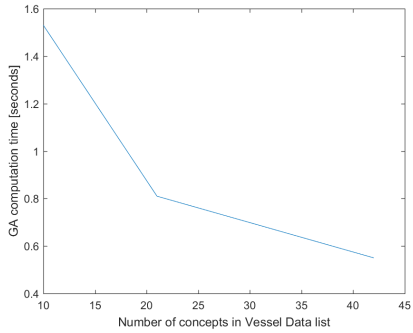
7.2.1 Computational Complexity

The aspect of reasonable time is also important for evaluating the suitability of this solution method. Because the size of the solution space may vary greatly, a linear increase in computational complexity is required for a solution method to be considered suited for the problem. Figure 7.4a shows how the computation time for the NP changes with the number of possible vessel concepts, and Figure 7.4b shows the same for the GA.

The linear increase of the NP indicates method suitability, not only because such a linear increase is required, but also because of what it represents. Since backtracking is included in the NP, there is no upper limit for how long time it could take to perform one round of the algorithm. The lower limit for the number of evaluations, and thus time, on the other hand, increases linearly with the size of the solution space. Therefore, this means that the NP efficiently searches the solution space. The downward trend for the computation time of the GA is somewhat of a mystery. A plausible explanation is that the intensification limits are closer for the cases with more vessel concepts. Therefore a lot more solutions are revisited in these cases, meaning that they jump out of the for-loops at line 40 and at line 64 in Appendix C.5.



(a) NP time averaged for each Vessel Data list, as a function of concepts in the list.



(b) GA time averaged for each Vessel Data list, as a function of concepts in the list.

Figure 7.4: Plots of how the computation times of each of the two algorithms change with an increasing number of possible vessel concepts.

Even though the time of the NP and GA searches increase slowly, the most important consideration is the change in the computational time of the solution method as a whole. In this respect, the number of necessary cycles, and the simulation time is also of importance. The solution times for the different cases are compared in Figure 7.5, as a function of the number of vessel concepts.

Figure 7.5 does not give much information about the development of the computation time, because only three different numbers of concepts are tested. However, it may be stated, with some certainty, that the development is slower than exponential, although varying greatly. This is because the great variation in the number of cycles performed, before the termination criteria were reached, in the different cases. Predict this behaviour is hard due to its dependency on the number of legal solutions found by the analytic function in each cycle. For big, complex vessel concept lists many cycles may be necessary because even the smallest of adjustments may introduce new, better solutions.

7.2.2 Flexibility and Intuitiveness

The flexibility of the solution method was tested by logging how long it took to prepare for solving new cases. The results of Table 6.4 is that it took less than two minutes for all

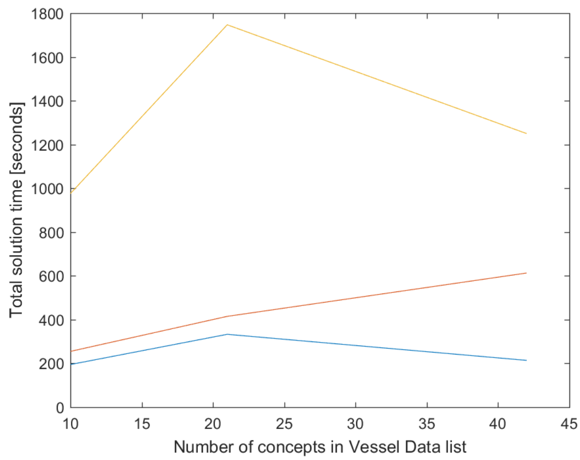


Figure 7.5: A plot of how the computation time of the complete solution method changes as a function of the length of the vessel concept list. The lower line is the 10 concepts list. The middle line is the 21 concepts list. The upper line is the 42 concepts list.

the tested cases. This is a manageable time, meaning that the solution method may be called flexible with respect to the tested cases. However, whether the diversity in the cases is sufficient to really call it flexible, is an open question. On the other hand, changing the objective evaluation and restrictions is also relatively easy, for example if Oil Spill Response (OSR) times are of importance. Both the NP and the GA is modular and may, to a large extent, be customised to fit the current problem.

As discussed in Section 7.1, the constructed solution method may simply be considered an automation of the traditional method for solving the problem. This is a strong argument for the method being intuitive. The structure of the process follows human intuition by making a qualified guess, testing it, and then improving the guess for the next round as more information is acquired. The search processes of the metaheuristics are also easy to follow; they are inspired by nature and simple partitioning. One aspect of the solution process that can be considered as little intuitive is the analytic evaluation of the concepts, although it is based on simple assumptions regarding the specific problem formulation. On the other hand, for someone with knowledge of the problem, this should not be an issue.

7.3 Alternative Implementations

Although the choices made in the construction of the solution method, there are many alternative implementations which are worth to discuss. Some of them may even be better than the current implementation, and some may be better for variations of the problem.

One obvious change that probably would improve the solution method is to change the treatment of illegal solutions. Rather than giving all illegal solutions a value of infinity, a finite punishment could be given, preferably as a function of the deviation of the solution from the thresholds of the restrictions. This would make the trend of the solution space clearer for the NP and GA and would not necessitate the user to assume the nature of the trend. In addition, it would be much more adaptive than the current approach manages.

While on the subject of the analytic function, adding loading rates and opening hours to the consideration probably would increase the accuracy of the predictions. However, this would mean a considerable complication of the analytic model. Also, the benefit from such an addition is not certain to outweigh the increased computation time. While the accuracy of the analytic model clearly isn't perfect, as indicated by the significant adjustment factors needed in Case 3, the issue is not that of miscalculating the loading times, but rather the stochastic elements.

To further reduce the computational time of the solution method, the vessel concept list could be "prepared" before the solution method is ran. That is, similar concepts could be removed from the list, reducing the size of the solution space. When good solutions are found, the concepts similar to those chosen in the best solutions could be re-entered into the list. The potential time savings are huge, but the quality of the solutions is hard to guarantee. First, the selection of concepts to remove must be very good. Second, there is no way to ensure that large areas of good solutions are not cut from the solution space.

In Cycle 4 of Case 6, all the solutions have exactly the same performance for all three cargo types. This is a result of how the performance is measured, there is only so much cargo to transport, therefore all solutions which transport all the cargo are given the same score. If the excess capacity of the fleet was returned from the simulation model, this could be used to compute a more accurate measure of the performance of the different "over performing" fleets. A consequence of the current setup is that it leads to the adjustment factors not reflecting the actual performance. This is the reason for situations like the one observed from Cycle 3 to Cycle 7 in Case 9, where the adjustment was set very low initially, then four cycles were needed to adjust back to the "intersection".

Chapter 8

Conclusion

The constructed solution method proved efficient at solving Maritime fleet size and mix problems on the form given by the ArcticLog-simulation model, provided by SINTEF Ocean. That is, a platform supply problem with one base, any number of platforms, any number of vessel concepts, and any cargo demand. The characteristics of the base and platforms may be changed in any way, in addition to the desired risk level. In addition to supporting this spectrum of variations to the problem, the flexibility of the solution method was proven by an average of 1 minute and 18 seconds needed for preparations for solving a new problem case. Notably, no changes were needed on the solution method, only input parameters and data sets.

The solution method was found to be intuitive, with the main argument being that the principal ideas of the method follow the same search procedure as the traditional approach of educated guessing. The details of the search are based on a nature inspired algorithm, the Genetic Algorithm, and a relatively simple partitioning algorithm, the Nested Partitions method. The solution method therefore succeeds in being practically tractable for decision makers without any expertise in the fields of simulation or optimisation.

The solution method succeeded at finding good solutions, both regarding to cost-optimality and reliability. However, optimality was not proven, and some issues regarding the reliability were revealed. That is, too few realisations were performed to ensure stable evaluations from one cycle to another.

Finally, the solution method proved to reach a termination criterion within reasonable time. The average solution time was 11 minutes and 6 seconds, and the solution method seems to have a polynomial time increase with respect to all relevant inputs. The computation time of both the exploration phase and the exploitation phase algorithms increases linearly with the number of dimensions in the problem.

Even though solution quality is not guaranteed with respect to optimality, the thesis objective is fulfilled: An S-O based solution method is established, which solves the main objective of the Arctic Offshore Logistics project. The solution method is intuitive and flexible and identifies good solutions within reasonable time.

Chapter 9

Recommendations for Further Work

The solution method is not complete in any way, but rather a proof of concept. In this chapter a selection of recommendations for further work are highlighted. Following these the method's applicability and suitability will be properly assessed for practical use.

Better Evaluation of the Usefulness

An evaluation of the actual usefulness of the solution method: A thorough analysis of the convergence properties of the method would be useful with respect to determining its suitability and benefits. Preferably, this includes a guaranteed worst-case computation time increase, and the establishment of an optimistic bound. In addition, the method should be compared to the traditional method as performed by professionals.

Better Evaluation of the Flexibility

The flexibility of the method should also be further investigated, preferably establishing the complete range of problems it may solve, and the necessary changes. Maritime fleet renewal problems, problems with different sets of restrictions and different objectives are some of the variations of immediate interest.

The solution method could be tested in combination with other models for evaluation, such as the model developed in Fagerholt et al. (2010), which requested a solution method of this exact type.

Better Implementation

The choices made in the current implementation may be re-evaluated, based on the discussion of Section 7.1 and Section 7.3. This includes a more comprehensive use of Optimal Computing Budget Allocation methods, and an improvement of the analytic function.

In order to fully exploit the benefits of the solution method, the transfer of information between the solution method and the simulation model should be made completely automatic. This reduces the solution time significantly. Also, the implementation should be done in a more efficient programming language, such as C++. The implementation could also be improved by streamlining the script, avoiding unnecessary loops.

Long Term

As quantum computers become increasingly powerful and popular, it would be interesting to see how this affects the efficiency and usefulness of this solution method. Both simulation and optimisation are likely to be revolutionised through the introduction of quantum computers. Implementing the solution method, with the necessary changes in a quantum environment, such as IBM Q Experience (International Business Machines (IBM), 2018), could disrupt the solution method efficiency.

Bibliography

Alrefaei, M. H., Andradóttir, S., 2001. A modification of the stochastic ruler method for discrete stochastic optimization. European Journal of Operational Research 133 (1), 160 – 182.

Alvarez, J. F., Tsilingiris, P., Engebrethsen, E. S., Kakalis, N. M. P., 05 2011. Robust fleet sizing and deployment for industrial and independent bulk ocean shipping companies. INFOR 49 (2), 93–107.

Amaran, S., Sahinidis, N. V., Sharda, B., Bury, S. J., 2017. Simulation optimization: A review of algorithms and applications. CoRR abs/1706.08591.
URL `http://arxiv.org/abs/1706.08591`

Andradóttir, S., October 2006. Simulation optimization with countably infinite feasible regions. ACM Transactions on Modeling and Computer Simulation 16 (4).

Andradóttir, S., Prudius, A. A., 2009. Balanced explorative and exploitative search with estimation for simulation optimization. INFORMS Journal on Computing 21 (2), 193 – 208.

Atlasis, A. F., Saltouros, M. P., Stassinopoulos, G. I., Vasilakos, A. V., 1998. An adaptive routing algorithm for atm networks using a learning automaton. In: IEEE GLOBECOM 1998 (Cat. NO. 98CH36250). Vol. 5. pp. 2669–2674 vol.5.

Aydın, N., 2014. A genetic algorithm on inventory routing problem. EMAJ: Emerging Markets Journal 3 (3), 59–66.

Bausch, D. O., Brown, G. G., Ronen, D., 1998. Scheduling short-term marine transport of bulk products. Maritime Policy & Management 25 (4), 335–348.

Bertsimas, D., Sim, M., 09 2003. Robust discrete optimization and network flows. Mathematical Programming 98 (1-3), 49–71.

Buchholz, P., 2009. Optimization of stochastic discrete event simulation models. In: Barnhart, C., Clausen, U., Lauther, U., Möhring, R. H. (Eds.), Models and Algorithms for Optimization in Logistics. No. 09261 in Dagstuhl Seminar Proceedings. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, Dagstuhl, Germany.

Chen, C.-h., 2011. Stochastic simulation optimization : an optimal computing budget allocation. Vol. v. 1 of Stochastic Simulation Optimization. World Scientific, Singapore ;.

Chen, E. J., Sep 2013. Some insights of using common random numbers in selection procedures. Discrete Event Dynamic Systems 23 (3), 241–259.

Chica, M., A. Juan PPrez, A., 01 2017. Why simheuristics? benefits, limitations, and best practices when combining metaheuristics with simulation.

Cousineau-Ouimet, K., 2002. A tabu search heuristic for the inventory routing problem. In: Proceedings of 37th Annual ORSNZ Conference.

Dengiz, B., Alabas, C., 2000. Simulation optimization using tabu search. In: Proceedings of the 32Nd Conference on Winter Simulation. WSC ’00. Society for Computer Simulation International, San Diego, CA, USA, pp. 805–810.

Eskandari, H., Mahmoodi, E., 12 2016. A simulation-based multi-objective optimization study of the fleet sizing problem in the offshore industry. Maritime Economics & Logistics 18 (4), 436–457, copyright - Copyright Palgrave Macmillan Dec 2016; Last updated - 2017-11-08.

Fagerholt, K., Christiansen, M., Hvattum, L. M., Johnsen, T. A., Vabø, T. J., 2010. A decision support methodology for strategic planning in maritime transportation. Omega 38 (6), 465 – 474.

Figueira, G., Almada-Lobo, B., 2014. Hybrid simulation–optimization methods: A taxonomy and discussion. Simulation Modelling Practice and Theory 46 (Supplement C), 118 – 134, simulation-Optimization of Complex Systems: Methods and Applications.

Fu, M. C., 2015. Handbook of simulation optimization. Vol. Volume 216 of International Series in Operations Research & Management Science.

Gosavi, A., 2015. Simulation-Based Optimization: Parametric Optimization Techniques and Reinforcement Learning. Vol. 55 of Operations Research/Computer Science Interfaces Series. Springer US, Boston, MA.

Halvorsen-Weare, E. E., Fagerholt, K., 2011. Robust Supply Vessel Planning. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 559–573.

Hoff, A., Andersson, H., Christiansen, M., Hasle, G., Løkketangen, A., 2010. Industrial aspects and literature survey: Fleet composition and routing. Computers & Operations Research 37 (12), 2041 – 2061.

Hong, L. J., Nelson, B. L., 2006. Discrete optimization via simulation using compass. Operations Research 54 (1), 115–129.

Hong, L. J., Nelson, B. L., Xu, J., 2010. Speeding up compass for high-dimensional discrete optimization via simulation. Operations Research Letters 38 (6), 550 – 555.

Ilaria, B., Lucia, C., Simone, F., Giuseppe, G. P., Aug 2016. Optimization of Stochastic Discrete Event Simulation Models Using AFO Heuristics. In: 2016 Third International Conference on Mathematics and Computers in Sciences and in Industry (MCSI). pp. 100–106.

Imai, A., IV, F. R., 2001. Strategic fleet size planning for maritime refrigerated containers. Maritime Policy & Management 28 (4), 361–374.

International Business Machines (IBM), 2018. IBM Q Experience. `https://quantumexperience.ng.bluemix.net/qx/experience`, Accessed: 2018-02-06.

Juan, A. A., Faulin, J., Grasman, S. E., Rabe, M., Figueira, G., 2015. A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. Operations Research Perspectives 2, 62 – 72.

Kennedy, J., Eberhart, R. C., 1995. Particle swarm optimization. In: Proceedings of the 1995 IEEE International Conference on Neural Networks. Vol. 4. Perth, Australia, IEEE Service Center, Piscataway, NJ, pp. 1942–1948.

Kirkpatrick, S., Gelatt, C. D., Vecchi, M. P., 1983. Optimization by simulated annealing. Science 220 (4598), 671–680.

Koenigsberg, E., Lam, R. C., 1976. Cyclic queue models of fleet operations. Operations Research 24 (3), 516.

Kokubugata, H., Kawashima, H., 2008. Application of Simulated Annealing to Routing Problems in City Logistics. I-Tech Education and Publishing, Vienna, Austria, pp. 131–154.

Kristinsdottir, B. P., Zabinsky, Z. B., Wood, G. R., 2002. Discrete Backtracking Adaptive Search for Global Optimization. Springer US, Boston, MA, pp. 147–174.

Larson, R. C., 1988. Transporting sludge to the 106-mile site: An inventory/routing model for fleet sizing and logistics system design. Transportation Science 22 (3), 186.

Larson, R. C., Minkofft, A. S., Gregory, P., 1991. A computer simulation model for fleet sizing for the marine division of the new york city department of sanitation. Waste Management & Research 9 (4), 267 – 276.

Lucas, T. W., Kelton, W. D., Sanchez, P. J., Sanchez, S. M., Anderson, B. L., 06 2015. Changing the paradigm: Simulation, now a method of first resort. Naval Research Logistics 62 (4), 293–303.

Lundgren, J., Rönnqvist, M., Värbrand, P., 2012. Optimization, 1st Edition. Studentlitteratur AB.

Marinakis, Y., Iordanidou, G.-R., Marinaki, M., 2013. Particle swarm optimization for the vehicle routing problem with stochastic demands. Applied Soft Computing 13 (4), 1693 – 1704.

Matta, A., Dec 2008. Simulation optimization with mathematical programming representation of discrete event systems. In: 2008 Winter Simulation Conference. pp. 1393–1400.

Pantuso, G., Fagerholt, K., Hvattum, L. M., 2014. A survey on maritime fleet size and mix problems. European Journal of Operational Research 235 (2), 341 – 349, maritime Logistics.

Pinho, A. F. d., Montevechi, J. A. A. B., Marins, F. A. S., Costa, R. F. A. d. S., Miranda, R. d. C., Friend, J. D., 12 2012. Evaluation of a proposed optimization method for discrete-event simulation models. Pesquisa Operacional 32, 543 – 560.

Richetta, O., Larson, R. C., 1997. Modeling the increase complexity of new york city's refuse marine transport system. Transportation Science 31 (3), 272.

Sanchez, S. M., Wan, H., Dec 2011. Better than a petaflop: The power of efficient experimental design. In: Proceedings of the 2011 Winter Simulation Conference (WSC). pp. 1436–1450.

Shi, L., 2009. Nested partitions method, theory and applications.

Shi, L., Ólafsson, S., 2000. Nested partitions method for global optimization. Operations Research 48 (3), 390–407.

Shyshou, A., Gribkovskaia, I., Barceló, J., 2010. A simulation study of the fleet sizing problem arising in offshore anchor handling operations. European Journal of Operational Research 203 (1), 230 – 240.

Siarry, P., 2016. Metaheuristics. Springer.

Syberfeldt, A., Grimm, H., Ng, A., Andersson, M., Karlsson, I., Dec 2008. Simulation-based optimization of a complex mail transportation network. In: 2008 Winter Simulation Conference. pp. 2625–2631.

Syberfeldt, A., Rogström, J., Geertsen, A., 2015. Simulation-based optimization of a real-world travelling salesman problem using an evolutionary algorithm with a repair function. International Journal of Artificial Intelligence and Expert Systems 6 (3), 27–39.

Tarantilis, C., Kiranoudis, C., Vassiliadis, V., 2002. A backtracking adaptive threshold accepting algorithm for the vehicle routing problem. Systems Analysis Modelling Simulation 42 (5), 631–664.

Thathachar, M. A. L., Sastry, P. S., Jan 1987. Learning optimal discriminant functions through a cooperative game of automata. IEEE Transactions on Systems, Man, and Cybernetics 17 (1), 73–85.

Vis, I. F. A., de Koster, R. M. B. M., Savelsbergh, M. W. P., 2005. Minimum vehicle fleet size under time-window constraints at a container terminal. Transportation Science 39 (2), 249–260.

Xu, J., Nelson, B. L., Hong, J. L., Feb. 2010. Industrial strength compass: A comprehensive algorithm and software for optimization via simulation. ACM Trans. Model. Comput. Simul. 20 (1), 3:1–3:29.

Yan, D., Mukai, H., 1992. Stochastic discrete optimization. SIAM Journal on Control and Optimization 30 (3), 594–612.

Ólafsson, S., 2006. Chapter 21 metaheuristics. In: Henderson, S. G., Nelson, B. L. (Eds.), Simulation. Vol. 13 of Handbooks in Operations Research and Management Science. Elsevier, pp. 633 – 654.

Appendix A

Input Files

There are two input files to the simulation model which are of interest in this report; DataSet1.xlsx and ConceptsFile.xlsx. In addition weather, but not changed...

A.1 DataSet1.xlsx

The input file containing all parameters determining the state, and response of the system, consisting of three sheets. The first containing A.1, the second containing either A.3, A.4 or A.5, and the third containing A.2.

	A	B
1	**Key campaign figures**	
2	Start Day	1
3	Start Month	5
4	Number of days	100
5	Fuel price [$/ton]	415
6	Harbour turnaround time worst case	24
7	Anchor handling [number of days]	0
8	**Weekly requirements (total all rigs)**	
9	Cargo	
10	- bulk (in) [m3]	8000
11	- deck (in) [m2]	4800
12	Crew [persons]	280
13	Spread departures [h]	8
14	**Oil spill response requirements**	
15	Time 2. vessel	17
16	Percentage 2. vessel on time	95
17	**Helicopter data**	
18	Helicopter base LAT	70.7
19	Helicopter base LON	23.7
20	Min turnaround at base	1
21	Min turnaround at installation	1
22	Earliest takeoff from base	7
23	Latest takeoff from base	19
24	Additional cost pr. flight hour heli [$]	2000
25	**Base location**	
26	Base LAT	70.7
27	Base LON	23.7
28	Open (Earliest load start)	9
29	Close (Latest load start)	14
30	Bulk load rate	480
31	Deck load rate	400
32	Additional turnaround time [h]	1
33	**Rig location**	
34	Rig LAT	74.7
35	Rig LON	36
36	**Simulation figures**	
37	Fog days (per year)	0
38	Simulations	250
39	OilSpills	100
40	Sea margin (%)	10
41	Metocean config file	C:\\Users\\

Figure A.1: Input data to the simulation model and the analytic function of the solution method.

	A	B	C	D	E	F	G
1	RigName	LAT	LON	BulkLoadRate	DeckLoadRate	StbyVessel	ditionalTurnaroundTi
2	Rig1	72	23	100	50	USANN	1
3	Rig2	73	20	100	50	USANN	1
4	Rig3	74	27	100	50	USANN	1
5	Rig4	75	32	100	50	USANN	1
6	Rig5	76	38	100	50	USANN	1
7	Rig6	73	37	100	50	USANN	1
8	Rig7	73	30	100	50	USANN	1

Figure A.2: Definitions of the platforms, for input to the simulation model and the analytic function of the solution method.

VesselType	Cat	Fixed/ mobilization cost [$]	Dayrate [$]	Fuel consumption normal [tons/day]	Fuel consumption max [tons/day]	Fuel cosmpt. standby/DP [tons/day	Fuel capacity [tons]	Speed max [knots]	Speed normal steam [knots]	Bulk capacity [m3]	Deck capacity [m2]	OSR	Time OSR mob	Max wave height Cargo [m]	Max wave height Pax [m]	Passenger capacity[persons]
PSV_StandardShuttle	P		20000	10.1	10.1	5	2000	11	11	3000	810	0	10	4		0
PSV_StandardShuttleOSR	P		25000	10.1	10.1	5	2000	11	11	1500	405	1		4		0
PSV_StandardStandby	S		20000	0	0	5	2000	0	0	0	0	0				0
PSV_High-EndShuttle	P		22000	10	10	5	2000	13	13	4000	1120	0	10	4	2	12
PSV_High-EndStandby	S		27000	0	0	5	2000	0	0	0	0	1				0
PSV_Long-distShuttle	P		36000	14.8	14.8	5	2000	16	16	2800	700	1		4	3	50
PSV_TrimaranShuttle	P		39000	30	30	5	2000	20	20	1200	1000	1		4	3	50
AHTS_Avg-sizeStandby	S		49000	0	0	10	2000	0	0	0	0	1				0
HelicopterShuttle	H		25000	0	0	0	2000	100	100	0	0	0				7
Crew boatShuttle	P		20000	15	15	5	2000	20	20	1	1	0		2	2	60

Figure A.3: Vessel data 1, 10 vessel concepts.

VesselType	Cat	Fixed/ mobilization cost [$]	Dayrate [$]	Fuel consumption normal [tons/day]	Fuel consumption max [tons/day]	Fuel cosmpt. standby/DP [tons/day	Fuel capacity [tons]	Speed max [knots]	Speed normal steam [knots]	Bulk capacity [m3]	Deck capacity [m2]	OSR	Time OSR mob	Max wave height Cargo [m]	Max wave height Pax [m]	Passenger capacity[persons]
PSV_StandardShuttle	P		20000	10.1	10.1	5	2000	11	11	3000	810	0	10	4	3	0
PSV_StandardShuttleOSR	P		25000	10.1	10.1	5	2000	11	11	1500	405	1		4	3	0
PSV_StandardStandby	P		20000	12.1	12.1	5	2000	11	11	1500	500	0		4	3	0
PSV_StandardOther	P		20000	10.1	10.1	5	2000	10	10	1600	405	0	10	4	3	0
PSV_High-EndShuttle	P		22000	10	10	5	2000	13	13	4000	1120	0		4	2	45
PSV_High-EndStandby	P		27000	10	10	5	2000	13	13	4000	1120	1		4	2	50
PSV_High-EndOther	P		29000	10	10	5	2000	13	13	4000	1200	1		4	2	55
PSV_Long-distShuttle	P		36000	14.8	14.8	5	2000	16	16	2800	700	1		4	3	50
PSV_Long-distStandby	P		37000	13	13	5	2000	16	16	2800	800	1		4	3	50
PSV_Long-distOther	P		38000	12	12	5	2000	16	16	2800	600	1		4	3	61
PSV_TrimaranShuttle	P		39000	30	30	5	2000	20	20	1200	1100	1		4	3	50
PSV_TrimaranStandby	P		40000	30	30	8	2000	22	22	1000	1000	1		4	3	52
PSV_TrimaranOther	P		39000	30	30	5	2000	20	20	1100	900	1		4	3	48
AHTS_Avg-sizeShuttle	P		50000	22	22	10	2000	12	12	2100	700	1		4	3	12
AHTS_Avg-sizeStandby	P		49000	22	22	10	2000	12	12	2100	700	1		4	3	11
AHTS_Avg-sizeOther	P		49000	22	22	10	2000	12	12	2100	655	1		4	3	13
HelicopterShuttle	H		25000	0	0	0	2000	90	90	0	0	0				8
HelicopterStandby	H		24000	0	0	0	2000	100	100	0	0	0				7
HelicopterSAR	H		25000	0	0	0	2000	110	110	0	0	0				5
Crew boatShuttle	P		21000	16	16	5	2000	22	22	0	0	0		3	3	65
Crew boatOther	P		20000	15	15	5	2000	20	20	0	0	0		3	3	60

Figure A.4: Vessel data 2, 21 vessel concepts.

VesselType	Cat	Fixed/ mobilization cost [$]	Dayrate [$]	Fuel consumption normal [tons/day]	Fuel consumption max [tons/day]	Fuel cosmpt. standby/DP [tons/day]	Fuel capacity [tons]	Speed max [knots]	Speed normal steam [knots]	Bulk capacity [m3]	Deck capacity [m2]	OSR	Time OSR mob	Max wave height Cargo [m]	Max wave height Pax [m]	Passenger capacity[persons]
PSV_StandardShuttle	P		20000	10.1	10.1	5	2000	11	11	3000	810	0	10	4	3	0
PSV_StandardShuttleOSR	P		25000	10.1	10.1	5	2000	11	11	1500	405	1		4	3	0
PSV_StandardStandby	P		20000	12.1	12.1	5	2000	11	11	1500	500	0		4	3	0
PSV_StandardOther	P		20000	10.1	10.1	5	2000	10	10	1600	405	0	10	4	3	0
PSV_High-EndShuttle	P		22000	10	10	5	2000	13	13	4000	1120	0		4	2	45
PSV_High-EndStandby	P		27000	10	10	5	2000	13	13	4000	1120	1		4	2	50
PSV_High-EndOther	P		29000	10	10	5	2000	13	13	4000	1200	1		4	2	55
PSV_Long-distShuttle	P		36000	14.8	14.8	5	2000	16	16	2800	700	1		4	3	50
PSV_Long-distStandby	P		37000	13	13	5	2000	16	16	2800	800	1		4	3	50
PSV_Long-distOther	P		38000	12	12	5	2000	16	16	2800	600	1		4	3	61
PSV_TrimaranShuttle	P		39000	30	30	5	2000	20	20	1200	1100	1		4	3	50
PSV_TrimaranStandby	P		40000	30	30	8	2000	22	22	1000	1000	1		4	3	52
PSV_TrimaranOther	P		39000	30	30	5	2000	20	20	1100	900	1		4	3	48
AHTS_Avg-sizeShuttle	P		50000	22	22	10	2000	12	12	2100	700	1		4	3	12
AHTS_Avg-sizeStandby	P		49000	22	22	10	2000	12	12	2100	700	1		4	3	11
AHTS_Avg-sizeOther	P		49000	22	22	10	2000	12	12	2100	655	1		4	3	13
HelicopterShuttle	H		25000	0	0	0	2000	90	90	0	0	0				8
HelicopterStandby	H		24000	0	0	0	2000	100	100	0	0	0				7
HelicopterSAR	H		25000	0	0	0	2000	110	110	0	0	0				5
Crew boatShuttle	P		21000	16	16	5	2000	22	22	0	0	0		3	3	65
Crew boatOther	P		20000	15	15	5	2000	20	20	0	0	0		3	3	60
PSV_StandardShuttle2	P		21000	10.1	10.1	5	2000	11	11	3000	810	0	10	4	3	0
PSV_StandardShuttleOSR2	P		26000	10.1	10.1	5	2000	11	11	1500	405	1		4	3	0
PSV_StandardStandby2	P		21000	12.1	12.1	5	2000	11	11	1500	500	0		4	3	0
PSV_StandardOther2	P		21000	10.1	10.1	5	2000	10	10	1600	405	0	10	4	3	0
PSV_High-EndShuttle2	P		23000	10	10	5	2000	13	13	4000	1120	0		4	2	45
PSV_High-EndStandby2	P		28000	10	10	5	2000	13	13	4000	1120	1		4	2	50
PSV_High-EndOther2	P		30000	10	10	5	2000	13	13	4000	1200	1		4	2	55
PSV_Long-distShuttle2	P		37000	14.8	14.8	5	2000	16	16	2800	700	1		4	3	50
PSV_Long-distStandby2	P		38000	13	13	5	2000	16	16	2800	800	1		4	3	50
PSV_Long-distOther2	P		39000	12	12	5	2000	16	16	2800	600	1		4	3	61
PSV_TrimaranShuttle2	P		40000	30	30	5	2000	20	20	1200	1100	1		4	3	50
PSV_TrimaranStandby2	P		41000	30	30	8	2000	22	22	1000	1000	1		4	3	52
PSV_TrimaranOther2	P		40000	30	30	5	2000	20	20	1100	900	1		4	3	48
AHTS_Avg-sizeShuttle2	P		51000	22	22	10	2000	12	12	2100	700	1		4	3	12
AHTS_Avg-sizeStandby2	P		50000	22	22	10	2000	12	12	2100	700	1		4	3	11
AHTS_Avg-sizeOther2	P		50000	22	22	10	2000	12	12	2100	655	1		4	3	13
HelicopterShuttle2	H		26000	0	0	0	2000	90	90	0	0	0				8
HelicopterStandby2	H		25000	0	0	0	2000	100	100	0	0	0				7
HelicopterSAR2	H		26000	0	0	0	2000	110	110	0	0	0				5
Crew boatShuttle2	P		22000	16	16	5	2000	22	22	0	0	0		3	3	65
Crew boatOther2	P		21000	15	15	5	2000	20	20	0	0	0		3	3	60

Figure A.5: Vessel data 3, 42 vessel concepts.

A.2 ConceptsFile.xlsx

The fleet concepts to be evaluated in the simulation model. Only containg this one table.

	A	B	C	D	E	F	G	H
1		1	2	3	4	5	6	7
2		K1	K2	K3	K4	K5	K6	K7
3								
4								
5		1	2	3	4	5	6	7
6	VesselConcept1	0	0	0	1	1	1	0
7	VesselConcept2	0	0	0	0	0	0	0
8	VesselConcept3	0	0	0	0	0	0	0
9	VesselConcept4	0	0	1	0	0	1	0
10	VesselConcept5	0	0	0	0	0	0	0
11	VesselConcept6	1	1	1	0	0	0	0
12	VesselConcept7	0	1	0	0	1	0	0
13	VesselConcept8	0	0	0	0	0	0	0
14	VesselConcept9	1	0	0	1	0	0	0
15	VesselConcept10	0	0	1	1	1	2	0

Figure A.6: An example of ConceptsFile.xlsx, showing 6 fleet concepts generated from Vessel Data 1. K7 is a zero-concept for reference when calculating the performance of the fleets. The table may have any number of fleet concepts(columns), consisting of any number of vessel concepts(rows).

Appendix B

Simulation Output

File Simulation Analysis Heuristic

General information | Vessels | Concepts | Installations | Scenario data | Simulation Results

Concept name	Run no.	Total costs [USD]	Ship costs [USD]	Helicopter costs [USD]	Deck cargo lifted [m2 per week]	Deck cargo waiting [m2 x days]	Bulk cargo lifted [ton per week]	Bulk cargo waiting [ton x days]	Voyages [per week]	PSV idle [hours per week]	Pax lifted [per week]	Pax waiting [pax x days]
K1		2 432 885	2 432 885	0	1 268	148 738	2 276	82 635	1,172	3	51	13 201
K1	1	2 435 679	2 435 679	0	1 294	143 141	1 952	56 227	1,19	3	50	10 152
K1	2	2 436 585	2 436 585	0	1 249	131 322	2 138	75 790	1,19	3	52	14 583
K1	3	2 427 641	2 427 641	0	1 332	195 856	2 594	101 108	1,19	3	52	14 427
K1	4	2 431 637	2 431 637	0	1 196	124 632	2 421	97 414	1,12	3	49	13 642
K2		2 932 885	2 932 885	0	1 268	148 738	2 276	82 635	1,172	3	56	9 678
K2	1	2 935 679	2 935 679	0	1 294	143 141	1 952	56 227	1,19	3	55	7 262
K2	2	2 936 585	2 936 585	0	1 249	131 322	2 138	75 790	1,19	3	58	10 778
K2	3	2 927 641	2 927 641	0	1 332	195 856	2 594	101 108	1,19	3	58	10 762
K2	4	2 931 637	2 931 637	0	1 196	124 632	2 421	97 414	1,12	3	55	9 912
K3		3 132 873	3 132 873	0	1 336	111 143	2 301	83 042	1,19	3	62	6 903
K3	1	3 135 596	3 135 596	0	1 309	97 235	1 952	58 432	1,19	3	59	5 709
K3	2	3 135 593	3 135 593	0	1 283	116 360	2 138	75 527	1,19	3	63	7 427
K3	3	3 129 139	3 129 139	0	1 415	143 090	2 594	102 180	1,19	3	63	8 349
K3	4	3 131 165	3 131 165	0	1 336	87 887	2 518	96 030	1,19	3	64	6 127
K4		3 920 643	3 920 643	0	924	378 005	2 281	71 362	1,33	3	63	5 438
K4	1	3 925 837	3 925 837	0	922	375 961	1 918	51 498	1,33	3	59	4 550
K4	2	3 923 042	3 923 042	0	924	326 633	2 107	60 592	1,33	3	64	5 389
K4	3	3 916 064	3 916 064	0	931	467 456	2 581	89 798	1,33	3	64	6 775
K4	4	3 917 631	3 917 631	0	920	341 968	2 518	83 560	1,33	3	65	5 039
K5		4 060 227	4 060 227	0	796	466 073	2 281	71 604	1,33	3	67	2 666
K5	1	4 064 535	4 064 535	0	796	462 461	1 918	54 374	1,33	3	65	2 632
K5	2	4 062 568	4 062 568	0	798	413 007	2 107	58 683	1,33	3	68	2 399
K5	3	4 056 268	4 056 268	0	798	561 456	2 581	89 798	1,33	3	70	3 338
K5	4	4 057 539	4 057 539	0	794	427 368	2 518	83 560	1,33	3	67	2 297
K6		0	0	0	0	1 040 246	0	1 617 598	0	0	0	49 652
K6	1	0	0	0	0	1 045 686	0	1 374 997	0	0	0	45 069
K6	2	0	0	0	0	996 807	0	1 432 467	0	0	0	51 738
K6	3	0	0	0	0	1 125 456	0	1 932 054	0	0	0	50 904
K6	4	0	0	0	0	993 036	0	1 730 873	0	0	0	50 898

Figure B.1: The results of five fleet concepts, simulated with four realisations each. A zero-concept is added at the end for reference.

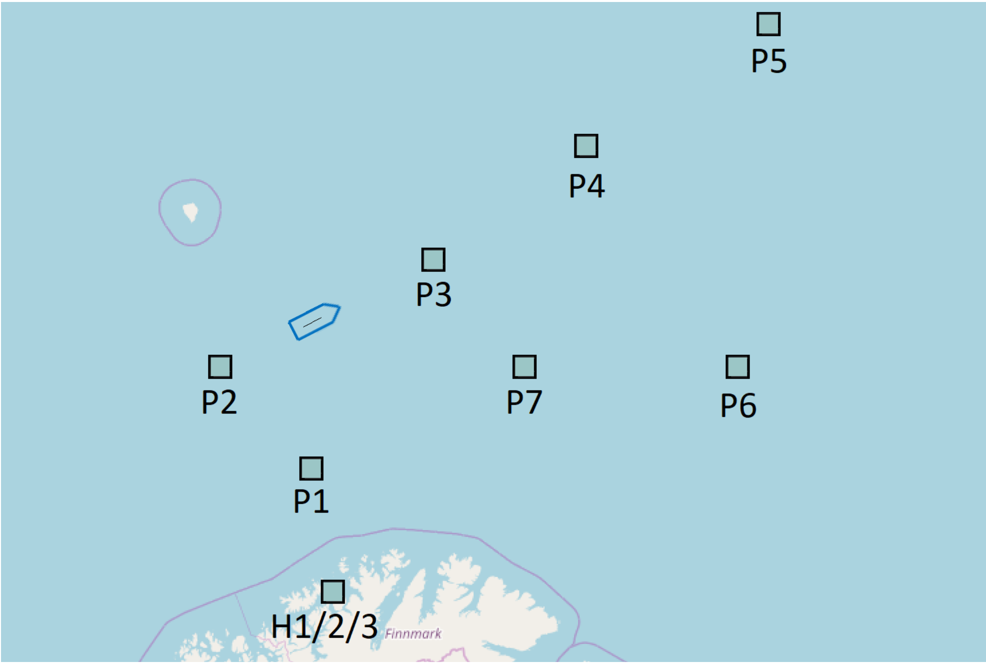


Figure B.2: An example of the visualisation feature of the simulation model. The platform and base names are added later.

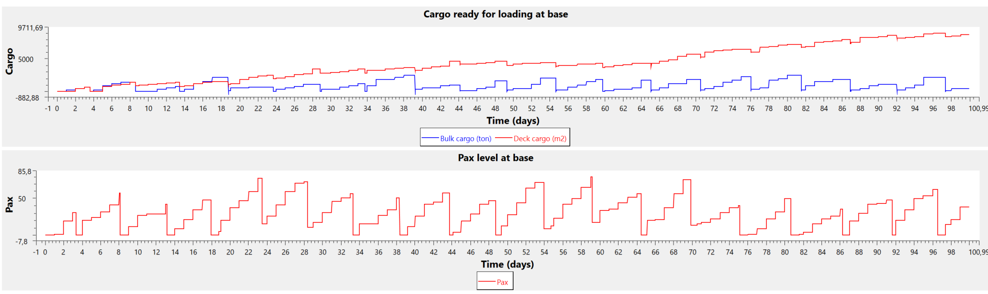
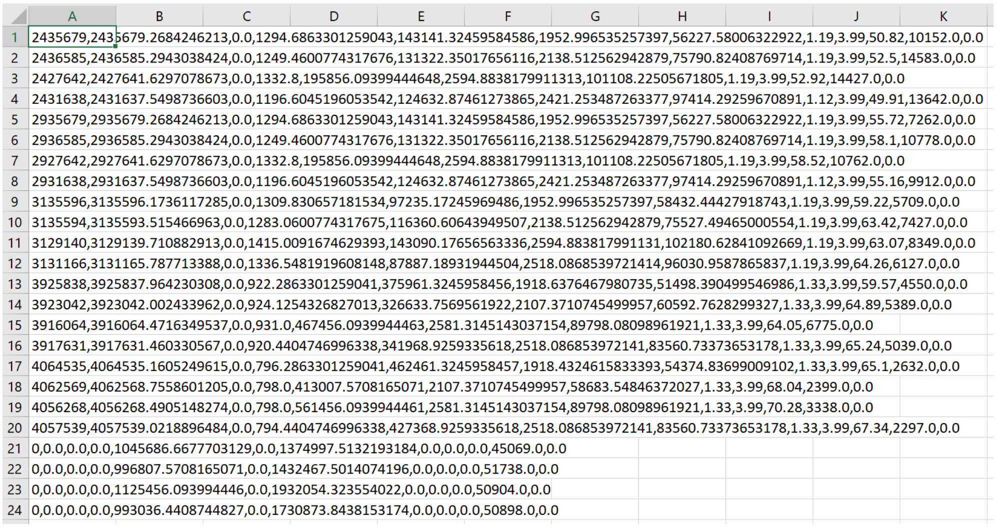


Figure B.3: An example of the graphical representation of the cargo and pax waiting to be picked up at the base.



	A	B	C	D	E	F	G	H	I	J	K
1	2435679,2435679.2684246213,0.0,1294.6863301259043,143141.32459584586,1952.996535257397,56227.58006322922,1.19,3.99,50.82,10152.0,0.0										
2	2436585,2436585.2943038424,0.0,1249.4600774317676,131322.35017656116,2138.512562942879,75790.82408769714,1.19,3.99,52.5,14583.0,0.0										
3	2427642,2427641.6297078673,0.0,1332.8,195856.09399444648,2594.8838179911313,101108.22505671805,1.19,3.99,52.92,14427.0,0.0										
4	2431638,2431637.5498736603,0.0,1196.6045196053542,124632.87461273865,2421.253487263377,97414.29259670891,1.12,3.99,49.91,13642.0,0.0										
5	2935679,2935679.2684246213,0.0,1294.6863301259043,143141.32459584586,1952.996535257397,56227.58006322922,1.19,3.99,55.72,7262.0,0.0										
6	2936585,2936585.2943038424,0.0,1249.4600774317676,131322.35017656116,2138.512562942879,75790.82408769714,1.19,3.99,58.1,10778.0,0.0										
7	2927642,2927641.6297078673,0.0,1332.8,195856.09399444648,2594.8838179911313,101108.22505671805,1.19,3.99,58.52,10762.0,0.0										
8	2931638,2931637.5498736603,0.0,1196.6045196053542,124632.87461273865,2421.253487263377,97414.29259670891,1.12,3.99,55.16,9912.0,0.0										
9	3135596,3135596.1736117285,0.0,1309.830657181534,97235.17245969486,1952.996535257397,58432.44427918743,1.19,3.99,59.22,5709.0,0.0										
10	3135594,3135593.515466963,0.0,1283.0600774317675,116360.60643949507,2138.512562942879,75527.49465000554,1.19,3.99,63.42,7427.0,0.0										
11	3129140,3129139.710882913,0.0,1415.0091674629393,143090.17656563336,2594.883817991131,102180.62841092669,1.19,3.99,63.07,8349.0,0.0										
12	3131166,3131165.787713388,0.0,1336.5481919608148,87887.18931944504,2518.0868539721414,96030.9587865837,1.19,3.99,64.26,6127.0,0.0										
13	3925838,3925837.964230308,0.0,922.2863301259041,375961.3245958456,1918.6376467980735,51498.390499546986,1.33,3.99,59.57,4550.0,0.0										
14	3923042,3923042.002433962,0.0,924.1254326827013,326633.7569561922,2107.3710745499957,60592.7628299327,1.33,3.99,64.89,5389.0,0.0										
15	3916064,3916064.4716349537,0.0,931.0,467456.0939944463,2581.3145143037154,89798.08098961921,1.33,3.99,64.05,6775.0,0.0										
16	3917631,3917631.460330567,0.0,920.4404746996338,341968.9259335618,2518.086853972141,83560.73373653178,1.33,3.99,65.24,5039.0,0.0										
17	4064535,4064535.1605249615,0.0,796.2863301259041,462461.3245958457,1918.4324615833393,54374.83699009102,1.33,3.99,65.1,2632.0,0.0										
18	4062569,4062568.7558601205,0.0,798.0,413007.5708165071,2107.3710745499957,58683.54846372027,1.33,3.99,68.04,2399.0,0.0										
19	4056268,4056268.4905148274,0.0,798.0,561456.0939944461,2581.3145143037154,89798.08098961921,1.33,3.99,70.28,3338.0,0.0										
20	4057539,4057539.0218896484,0.0,794.4404746996338,427368.9259335618,2518.086853972141,83560.73373653178,1.33,3.99,67.34,2297.0,0.0										
21	0,0.0,0.0,0.0,1045686.6677703129,0.0,1374997.5132193184,0.0,0.0,0.0,45069.0,0.0										
22	0,0.0,0.0,0.0,996807.5708165071,0.0,1432467.5014074196,0.0,0.0,0.0,51738.0,0.0										
23	0,0.0,0.0,0.0,1125456.093994446,0.0,1932054.323554022,0.0,0.0,0.0,50904.0,0.0										
24	0,0.0,0.0,0.0,993036.4408744827,0.0,1730873.8438153174,0.0,0.0,0.0,50898.0,0.0										

Figure B.4: An example of the output written to a .csv file, which is read to the solution method. Each row corresponds to a realisation. This is the same results as shown in Figure B.1, the first four rows are thus the same fleet concept, as are the four next. For each row, the outputs are ordered as follows (comma separated): Total cost, Ship cost, Helicopter cost, Deck cargo lifted, Deck cargo waiting, Bulk cargo lifted, Bulk cargo waiting, Voyages, PSV idle time, Pax lifted, Pax waiting, Number of flights.

File Simulation Analysis Heuristic

General information | Vessels | Concepts | Installations | Scenario data | Simulat

Day	Bulk	Dec	Pax	Fog	Wave Height	SeaMargin
1	226,126	103,389	18		0,704	0,095
2	168,311	376,475	16		0,987	0,016
3	269,861	95,655	11		1,522	0,078
4	282,015	210,269	1		1,315	0,229
5	241,457	702,514	10		1,49	0,076
6	173,262	105,417	10		1,028	0,056
7	0	146,19	3		1,157	0,083
8	232,856	102,688	2		1,154	0,22
9	316,248	713,156	0		1,498	0,026
10	326,17	713,796	6		1,605	0,068
11	0	201,77	6		0,949	0,169
12	0	170,091	2		1,2	0,049
13	293,289	202,502	0		2,32	0,229
14	0	121,148	11		3,02	0,071
15	270,459	154,842	16		1,734	0,106
16	206,373	378,727	13		1,683	0,028
17	292,416	37,457	0		2,165	0,125
18	648,3	160,275	3		1,76	0,052
19	328,617	98,066	13		1,512	0,04
20	0	372,238	16		1,329	0,08
21	192,258	102,533	8		1,164	0,164
22	0	112,667	10		1,572	0,186
23	940,69	93,084	11		3,503	0,074
24	341,553	70,701	1		2,21	0,079
25	0	67,723	9		1,403	0,014
26	1 169,886	114,512	3		1,537	0,004
27	314,902	552,413	2		1,925	0,049
28	0	172,235	15		1,659	0,13
29	303,525	157,452	17		1,809	0,186
30	613,403	364,941	5		1,195	0,001
31	235,533	103,773	9		2,472	0,075
32	1 213,876	112,754	13		2,653	0,041
33	0	153,868	7		1,853	0,033
34	626,542	173,144	4		1,411	0,119
35	0	176,895	10		1,807	0,086
36	233,524	178,715	5		1,587	0,047
37	0	124,134	12		1,04	0,004
38	1 179,83	514,568	5		1,052	0,371
39	165,721	63,543	10		2,446	0,008
40	231,936	112,709	8		1,899	0,051
41	320,144	534,981	2		1,848	0,115
42	0	371,41	11		2,854	0,208
43	347,5	180,803	16		2,373	0,029
44	314,046	191,059	1		1,433	0,057
45	0	354,193	14		1,115	0,097
46	1 180,133	380,482	0		1,496	0,166
47	294,548	181,219	15		1,517	0,131
48	636,861	100,391	7		0,869	0,148
49	166,102	157,986	14		1,139	0,015
50	168,964	65,074	16		1,379	0,023

Figure B.5: An example of the scenario data of a simulation realisation.

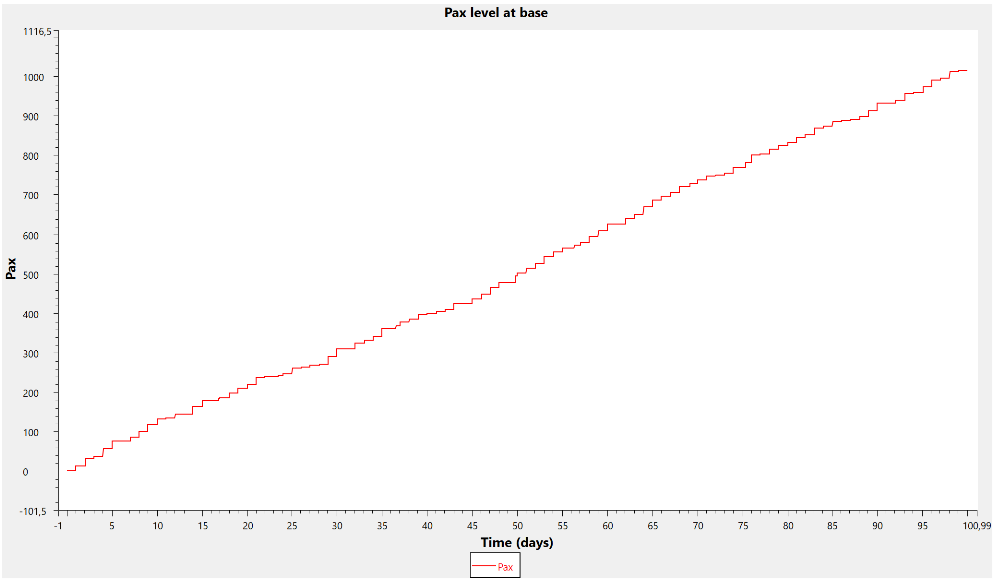


Figure B.6: A graph showing the increase in pax waiting at the base, for the zero-concept. That is, when there are no vessels in the solution. The simulation model would not allow a figure to be made for concepts that does not pick up bulk cargo or deck cargo, so the only zero-concept figure available is for pax.

Appendix C

Solution Method Implementation

C.1 AlgorithmV2D1.m

AlgorithmV2D1.m is the first script to run when running the solution method. This file sets all parameters which are used by all the other files in the solution method. Some of the parameters are loaded into DataSet1.xlsx from here, such as MinBulk and MinDeck, and some are read from DataSet1.xlsx and loaded into the MATLAB workspace. The ranges of the different dimensions (ARS) are set. Also, the distances between the different platforms and the base, and the round trip distance is calculated.

```

clear;%clc;
format long;

%CASE DEFINING PARAMETERS
MinBulk=8000; %Tons pr week.
MinDeck=4800; %M^2 pr week.
MinPax=280; %Persons pr week.
SpreadDep=6; %Hours.
BLR=480; %Bulk loading rate base.
DLR=400; %Deck loading rate base.
```

```
Duration=2400; %Hours.
UB=20; %Upper boundary for the number of vessels of one type.
T_r=[0.95 0.95 0.95]; %Threshold reliability. (the probability
    that the actual performance of the solution is better than the
     returned value.)
T_p=[0.9 0.9 0.9]; %Threshold performance. (the limit for what
    transportation coverage that is acceptable.)
C_r=0.95; %Cost reliability. (the probability that the actual
    cost will be lower than the returned value.)

%ALGORITHM DEFINING PARAMETERS
FB=1; FD=1; FP=1; %The initial values of the adjustment factors.
R=1000; %The number of "rounds" in the Nested Partitions
    algorithm.
NoFromNP=100; %The maximum number of unique solutions transferred
     from NP to GA.
NoPar=NoFromNP; %The population size in the GA.
Generations=200; %The number of generations in the GA.
NoSim=6; %The maximum number of unique solutions being simulated
    at a time.
f=1; %The counter for the adjustment factor.
%t=0;

%WRITING TO THE DATASET-FILE
xlswrite('DataSet1.xlsx',[MinBulk MinDeck MinPax]','Input','B10:
    B12');
xlswrite('DataSet1.xlsx',SpreadDep,'Input','B13');
xlswrite('DataSet1.xlsx',[BLR,DLR]','Input','B30:B31');

%READING FROM THE DATASET-FILE
A1=xlsread('DataSet1.xlsx','Input','B4:B40');
A2=xlsread('DataSet1.xlsx','Vessels','D2:Q11');
A3=xlsread('DataSet1.xlsx','Rigs','B2:C8');

%SETTING THE BOUNDARIES FOR THE DECISION VARIABLES
```

```
AL=length(A2(:,1));
for i=1:AL
    ARS(i,:)=[0,UB,1];
end

circ=40075/1.852; %Circumference of the earth in nautical miles

%ROUND TRIP DISTANCE
for i=0:length(A3(:,1))
    if i<1
        arclen(i+1) = distance('rh',[A1(15),A1(16)],[A3(i+1,1),A3
            (i+1,2)])*circ/360;
    elseif i>0 && i<length(A3(:,1))
        arclen(i+1) = distance('rh',[A3(i,1),A3(i,2)],[A3(i+1,1),
            A3(i+1,2)])*circ/360;
    else
        arclen(i+1) = distance('rh',[A3(i,1),A3(i,2)],[A1(15),A1
            (16)])*circ/360;
    end
end
RndDist=sum(arclen);

%AVERAGE DISTANCE FROM BASE TO PLATFORMS
for i=1:length(A3(:,1))
    arclen2(i) = distance('rh',[A1(15),A1(16)],[A3(i,1),A3(i,2)])
        *circ/360;
end
AvgDist=sum(arclen2)/length(A3(:,1));
```

C.2 AlgorithmV2D2.m

AlgorithmV2D2.m is the second script to run. Here the searches of the Nested Partitions algorithm and the Genetic Algorithm are performed, including sorting the outputs before forwarding them. The solutions to be simulated are written to ConceptsFile.xlsx, and the number of scenarios are set.

```
%RUNNING THE NESTED PARTITIONS ALGORITHM
tic;
BestNP = AlgorithmNP(Duration,AL,ARS,A1,A2,R,FB,FD,FP,MinBulk,...
MinDeck,MinPax,RndDist,AvgDist,SpreadDep); %Running the NP.
BestNPUnique = AlgorithmUnique(BestNP,AL,NoFromNP); %Removing all copies, so that the list consists only of unique entries.
toc;
%RUNNING THE GENETIC ALGORITHM
tic;
BestGA = AlgorithmGA3(Generations,NoPar,AL,A1,A2,BestNPUnique,...
    Duration,FB,FD,FP,MinBulk,MinDeck,MinPax,ARS,RndDist,AvgDist,SpreadDep); %Running the GA.
BestGAUnique = AlgorithmUnique(BestGA,AL,NoSim); %Removing all copies, so that the list consists only of unique entries.
toc;
%WRITING THE CHOSEN CONCEPTS TO THE CONCEPTS-FILE
Sim=[]; %Resetting the Sim-matrix.
Sim=BestGAUnique(:,1:AL); %Setting the solutions of the Sim-matrx equal to that of BestGAUnique.
FC= [BestGAUnique(:,1:AL); zeros(1,AL)]'; %Defining the solutions (Fleet Concepts) to be simulated, adding a zeros-concept for reference.

for SheetNum=1:1 %Removing previous entries from the concepts-file.
    [~, ~, Raw]=xlsread('ConceptsFile.xlsx','Ark1','B6:V47');
    [Raw{:, :}]=deal(NaN);
    xlswrite('ConceptsFile.xlsx', Raw, 'Ark1','B6:V47');
```

```
end
xlswrite('ConceptsFile.xlsx',FC,'Ark1','B6'); %Writing the new
    concepts to the concepts-file.

NoScen=75; %The number of scenarios simulated for each solution.
%75 until TC, then 250.
xlswrite('DataSet1.xlsx',NoScen,'Input','B38');


%NOW, RUN THE SIMULATIONS.
```

C.3 AlgorithmV2D3.m

AlgorithmV2D3.m is the third script to run. The simulation model is ran between AlgorithmV2D2.m and this script. Here, the outputs of the simulation model are read from the ConceptsFile_solutions.csv to the MATLAB workspace, and the relevant information is retrieved into different variables. The performances of the different solutions are computed, legal solutions are returned and adjustment factors are calculated.

After this script is ran, the user decides whether to run AlgorithmV2D2.m again, and repeat the cycle, or to terminate based on the TCs.

```
%READING THE SIMULATION RESULTS
tic;
Results=csvread('ConceptsFile_solution.csv');
toc;

J=length(Results(:,1))/(NoSim+1); %
i=0;j=0;
        TotalCost=zeros(NoSim,J);
        ShipCost=zeros(NoSim,J);
        HelicopterCost=zeros(NoSim,J);
        FracDeckCargoLifted=zeros(NoSim,J);
        DeckCargoWaiting=zeros(NoSim,J);
        FracBulkCargoLifted=zeros(NoSim,J);
        BulkCargoWaiting=zeros(NoSim,J);
        Voyages=zeros(NoSim,J);
        PSVidle=zeros(NoSim,J);
        FracPaxLifted=zeros(NoSim,J);
        PaxWaiting=zeros(NoSim,J);
        Flights=zeros(NoSim,J);
        DeckCargoLifted=zeros(NoSim,J);
        BulkCargoLifted=zeros(NoSim,J);
        PaxLifted=zeros(NoSim,J);

%COMPUTING A MEASURE OF THE AVERAGE WEEKLY DEMAND FOR EACH
```

```
    SCENARIO
WDB=Results(J*NoSim+1:J*(NoSim+1),7)/((Duration/24)^2)*7*2; % Weekly demand bulk.
WDD=Results(J*NoSim+1:J*(NoSim+1),5)/((Duration/24)^2)*7*2; % Weekly demand deck.
WDP=Results(J*NoSim+1:J*(NoSim+1),11)/((Duration/24)^2)*7*2; % Weekly demand pax.

%TRANSLATING SIMULATION RESULTS INTO CONFIDENCE INTERVALS AND FRACTIONS FOR
%ADJUSTMENT
for i=1:NoSim %For every fleet concept
    for j=1:J %For every scenario
        TotalCost(i,j)=Results((i-1)*J+j,1);
        ShipCost(i,j)=Results((i-1)*J+j,2);
        HelicopterCost(i,j)=Results((i-1)*J+j,3);
        DeckCargoLifted(i,j)=Results((i-1)*J+j,4);
        FracDeckCargoLifted(i,j)=Results((i-1)*J+j,4)/WDD(j);
        DeckCargoWaiting(i,j)=Results((i-1)*J+j,5);
        BulkCargoLifted(i,j)=Results((i-1)*J+j,6);
        FracBulkCargoLifted(i,j)=Results((i-1)*J+j,6)/WDB(j);
        BulkCargoWaiting(i,j)=Results((i-1)*J+j,7);
        Voyages(i,j)=Results((i-1)*J+j,8);
        PSVidle(i,j)=Results((i-1)*J+j,9);
        PaxLifted(i,j)=Results((i-1)*J+j,10);
        FracPaxLifted(i,j)=Results((i-1)*J+j,10)/WDP(j);
        PaxWaiting(i,j)=Results((i-1)*J+j,11);
        Flights(i,j)=Results((i-1)*J+j,12);
    end

    Performance(i,:)=[prctile(TotalCost(i,:),C_r) prctile(FracBulkCargoLifted(i,:),(1-T_r(1))) ...
        prctile(FracDeckCargoLifted(i,:),(1-T_r(2))) prctile(FracPaxLifted(i,:),(1-T_r(3))) ...
        mean(BulkCargoWaiting(i,:)) mean(DeckCargoWaiting(i,:))
```

```
            mean(PaxWaiting(i,:))];

    Fraction(i,:)=[prctile(FracBulkCargoLifted(i,:),(1-T_r(1)))/T_p(1) ...
        prctile(FracDeckCargoLifted(i,:),(1-T_r(2)))/T_p(2) ...
        prctile(FracPaxLifted(i,:),(1-T_r(3)))/T_p(3)];
end

%EVALUATING THE SIMULATION RESULTS
Sim(:,AL+1:AL+7)=Performance(1:length(BestGAUnique(:,1)),:); % Adding the simulation results to the solution-matrix.
i=1;
j=1;
Result=zeros(1,AL+7);
for i=1:length(Sim(:,1))
    if Sim(i,AL+2)>T_p(1) && Sim(i,AL+3)>T_p(2) && Sim(i,AL+4)>T_p(3) %At least 95% of the scenarios delivered more than 90% of what they should.
        Result(j,:)=Sim(i,:);
        j=j+1;
    end
end

f=f+1;
FB(f)=prctile(Fraction(:,1),50);
FD(f)=prctile(Fraction(:,2),50);
FP(f)=prctile(Fraction(:,3),50);

if Result(1,AL+1)~=0 %If this simulation returned a legal set.
      format long;
    Result=sortrows(Result,AL+1)
end

Sim
```

C.4 AlgorithmNP.m

This is the MATLAB implementation of the customised NP algorithm used in the solution method.

```
function [BestNP] = AlgorithmNP(Duration,AL,ARS,A1,A2,R,FB,FD,FP,
    MinBulk,...
    MinDeck,MinPax,RndDist,AvgDist,SpreadDep)

for r=1:R
    BestNP(r,:)=[zeros(1,AL) inf zeros(1,3)];
    cont=ones(AL,1);
    for i=1:AL
        if ARS(i,2)==ARS(i,1)
            cont(i)=0;
        end
    end
    contS=1;
    RS=ARS;
    while contS==1
        d=ceil(length(ARS(:,1))*rand);
        if cont(d)==1
            Limits(d,:)=[RS(d,1),floor((RS(d,2)-RS(d,1))/2)+RS(d
                ,1),...
                floor((RS(d,2)-RS(d,1))/2)+RS(d,1)+1,RS(d,2)];
            DP(d,1)=round((Limits(d,2)-Limits(d,1))*rand()+
                Limits(d,1);
            DP(d,2)=round((Limits(d,4)-Limits(d,3))*rand()+
                Limits(d,3);
            for k=1:AL
                part(k)=round((RS(k,2)-RS(k,1))*rand()+RS(k,1);
            end
            if RS(d,1)==ARS(d,1) && RS(d,2)==ARS(d,2)
                A=2;
            else
```

```
                A=4;
                if RS(d,2)<ARS(d,2)
                    DP(d,3)=ceil((ARS(d,2)-RS(d,2))*rand)+RS(d,2);
                    DP(d,4)=ceil((ARS(d,2)-RS(d,2))*rand)+RS(d,2);
                    if RS(d,1)>ARS(d,1)
                        a=ARS(d,2)-RS(d,2);
                        b=RS(d,1)-ARS(d,1);
                        tilf=rand;
                        if tilf<a/(a+b)
                            DP(d,3)=ceil((ARS(d,2)-RS(d,2))*rand)+RS(d,2);
                            DP(d,4)=ceil((ARS(d,2)-RS(d,2))*rand)+RS(d,2);
                        else
                            DP(d,3)=round((RS(d,1)-ARS(d,1)-1)*rand)+ARS(d,1);
                            DP(d,4)=round((RS(d,1)-ARS(d,1)-1)*rand)+ARS(d,1);
                        end
                    end
                elseif RS(d,1)>ARS(d,1)
                    DP(d,3)=round((RS(d,1)-ARS(d,1)-1)*rand)+ARS(d,1);
                    DP(d,4)=round((RS(d,1)-ARS(d,1)-1)*rand)+ARS(d,1);
                end
            end
            for a=1:A
                part(d)=DP(d,a);
                part=AlgorithmAnalytic(part,A1,A2,AL,Duration,...
                    FB,FD,FP,MinBulk,MinDeck,MinPax,RndDist,AvgDist,SpreadDep);
                SolVal(a)=part(AL+1);
```

```
            if part(AL+1)<BestNP(r,AL+1)
                BestNP(r,:)=part;
            end
        end
        if A==2
            if SolVal(1)<=SolVal(2)
                RS(d,1)=Limits(d,1);
                RS(d,2)=Limits(d,2);
            elseif SolVal(2)<SolVal(1)
                RS(d,1)=Limits(d,3);
                RS(d,2)=Limits(d,4);
           end
        elseif A==4
            if SolVal(1)<=SolVal(2) && SolVal(1)<=SolVal(3)
                && SolVal(1)<=SolVal(4)
                RS(d,1)=Limits(d,1);
                RS(d,2)=Limits(d,2);
            elseif SolVal(2)<SolVal(1) && SolVal(2)<SolVal(3)
                 && SolVal(2)<SolVal(4)
                RS(d,1)=Limits(d,3);
                RS(d,2)=Limits(d,4);
            elseif SolVal(3)<SolVal(1) && SolVal(3)<SolVal(2)
                 && SolVal(3)<SolVal(4)
                RS(d,1)=ARS(d,1);
                RS(d,2)=ARS(d,2);
            elseif SolVal(4)<SolVal(1) && SolVal(4)<SolVal(2)
                 && SolVal(4)<SolVal(3)
                RS(d,1)=ARS(d,1);
                RS(d,2)=ARS(d,2);
            end
        end
        RS3=RS;
        if RS(d,2)-RS(d,1)<ARS(d,3)
            cont(d)=0;
        end
```

```
                if sum(cont)==0
                    contS=0;
                end
            end
        end
end
BestNP=sortrows(BestNP,AL+1);
end
```

C.5 AlgorithmGA3.m

This is the MATLAB implementation of the customised GA used in the solution method.

```
function [BestGA] = AlgorithmGA3(Generations,AntPar,AL,A1,A2,part
    ,...
    Duration,FB,FD,FP,MinBulk,MinDeck,MinPax,ARS,RndDist,AvgDist,
        SpreadDep)

for j=1:AL
    size(j)=max(part(:,j))-min(part(:,j));
    Range(j,:)=[min(part(:,j)) min(part(:,j)+size(j))];
end

part=sortrows(part,AL+1);
BestGA=part;

contD=ones(AL,1);
for i=1:AL
    if Range(i,1)==Range(i,2)
        cont(i)=0;
    end
end

for k=1:Generations
    for i=1:AntPar
        %Choosing a dimension
        d=randi(AL);
        while contD(d)==0
            d=randi(AL);
        end
        %+1 or -1 for that variable.
        if part(i,d)==Range(d,1)
            part(i,d)=part(i,d)+1;
        elseif part(i,d)==Range(d,1)
```

```
            part(i,d)=part(i,d)-1;
        else
            part(i,d)=part(i,d)+(1-2*round(rand));
        end
        %Evaluate the solution.
        part(i,:)=AlgorithmAnalytic(part(i,:),A1,A2,AL,Duration
            ,...
                FB,FD,FP,MinBulk,MinDeck,MinPax,RndDist,AvgDist,
                    SpreadDep);
        if part(i,AL+1)<BestGA(end,AL+1)
            lik=0;
            for m=1:AntPar
                if sum(part(i,1:AL)==BestGA(m,1:AL))==AL
                    lik=1;
                    break;
                end
            end
            if lik==0
                BestGA(end,:)=part(i,:);
                BestGA=sortrows(BestGA,AL+1);
            end
        end
    end
    %Selection
    part=sortrows(part,AL+1);
    part=part(1:round(AntPar/4),:);
    %Reproduction
    for i=(round(AntPar/4)+1):AntPar
        part(i,1:ceil(AL/2))=part(ceil(round(AntPar/4)*rand),1:
            ceil(AL/2));
        part(i,ceil(AL/2)+1:AL)=part(ceil(round(AntPar/4)*rand),
            ceil(AL/2)+1:AL);
        part(i,:)=AlgorithmAnalytic(part(i,:),A1,A2,AL,Duration
            ,...
        FB,FD,FP,MinBulk,MinDeck,MinPax,RndDist,AvgDist,SpreadDep
```

```
        );
        if part(i,AL+1)<BestGA(end,AL+1)
            %If it's not in BestGA from earlier.
            lik=0;
            for m=1:AntPar
                if (part(i,1:AL)==BestGA(m,1:AL))>0
                    lik=1;
                    break;
                end
            end
            if lik==0
                BestGA(end,:)=part(i,:);
                BestGA=sortrows(BestGA,AL+1);
            end
        end
    end
    part=sortrows(part,AL+1);
end
end
```

C.6 AlgorithmAnalytic.m

This is the MATLAB implementation of the analytic function used in the solution method. See Chapter 4 for details.

```
function [particle] = AlgorithmAnalytic(part,A1,A2,AL,Duration,...
    FB,FD,FP,MinBulk,MinDeck,MinPax,RndDist,AvgDist,SpreadDep)
DayrateCost=0; FuelCost=0; Trips=0; PSVs=0; AvgCons=0; AvgSpeed=0;
Consumption=0; TransSpeed=0; FlightHours=0; FlightCost=0;

for j=1:AL
    DayrateCost=DayrateCost+part(j)*A2(j,1)*Duration/24;
    Consumption=Consumption+part(j)*A2(j,3); %Only sailing
    PSVs=PSVs+part(j);
    TransSpeed=TransSpeed+part(j)*A2(j,7);
end
AvgCons=Consumption/PSVs;
AvgSpeed=TransSpeed/PSVs;
Trips=min(Duration/SpreadDep,Duration*AvgSpeed*PSVs/RndDist);
FuelCost=Trips*AvgCons/24*RndDist/AvgSpeed*A1(2);
FlightHours=min(MinPax*100/7/A2(9,14)*(2*AvgDist)/A2(9,7)*1.3,2*part(9)*((2*AvgDist)/A2(9,7)+2)*Duration/24);
FlightCost=A1(21)*FlightHours;

Bulk=0;Deck=0;Pax=0;ShipPax=0;HoursinWeek=168;
for j=1:AL
    if A2(j,7)<80 %Distinguishes between ships and helicopters.
        Bulk=Bulk+part(j)*A2(j,8);
        Deck=Deck+part(j)*A2(j,9);
        ShipPax=ShipPax+part(j)*A2(j,14);
    elseif A2(j,7)>80
        Pax=Pax+A2(j,14)*FlightHours/((2*AvgDist)/A2(j,7));
    end
```

```
    if part(j)>0 && part(j)*A2(j,8)==0 && part(j)*A2(j,9)==0 &&
        part(j)*A2(j,14)==0 %Avoiding "dummy-ships" in the concept
         list.
        FuelCost=inf;
    end
end

Bulk=Bulk/PSVs*Trips/(100/7);
Deck=Deck/PSVs*Trips/(100/7);
ShipPax=ShipPax/PSVs*Trips;
Pax=(Pax+ShipPax)/(100/7);

Bulk=prod(FB)*Bulk;
Deck=prod(FD)*Deck;
Pax=prod(FP)*Pax;
part(AL+2)=Bulk;
part(AL+3)=Deck;
part(AL+4)=Pax;

if Bulk>MinBulk && Deck>MinDeck && Pax>MinPax
    part(AL+1)=DayrateCost+FuelCost+FlightCost; %Legal solution
else
    part(AL+1)=inf;%Illegal solution
end
particle=part;
end
```

C.7 AlgorithmUnique.m

This is the sorting algorithm used to ensure that only unique solutions are forwarded, both from the NP and the GA.

```
function [BestNPUnique] = AlgorithmUnique(BestNP,AL,NoFromNP)
j=0;
sol=zeros(1,AL+4);
for i=1:length(BestNP(:,1)) %For every solution in the input matrix.
    lik=0;
    for k=1:length(sol(:,1)) %For every solution previously stored from the input matrix.
        if sum(BestNP(i,1:AL)==sol(k,1:AL))==AL %If this solution equals one that is previously checked (that is, has a smaller index value).
            lik=1;
            break;
        end
    end
    if lik==0 %If this solution has not been seen before in this input matrix.
        j=j+1;
        sol(j,:)=BestNP(i,:); %Add to the list of seen solutions.
    end
    if j==NoFromNP
        break;
    end
end
BestNPUnique=sol;
end
```

Appendix D

Supply Distributions

Pay attention to the relation between the average weekly demands in the realisations, and the value input by the user. F.ex. in D.1 the user has set a basis for the weekly demand of deck cargo to be $1200m^2$. Clearly, this is not even close to the average, and cannot be said to be anything more than just a basis.

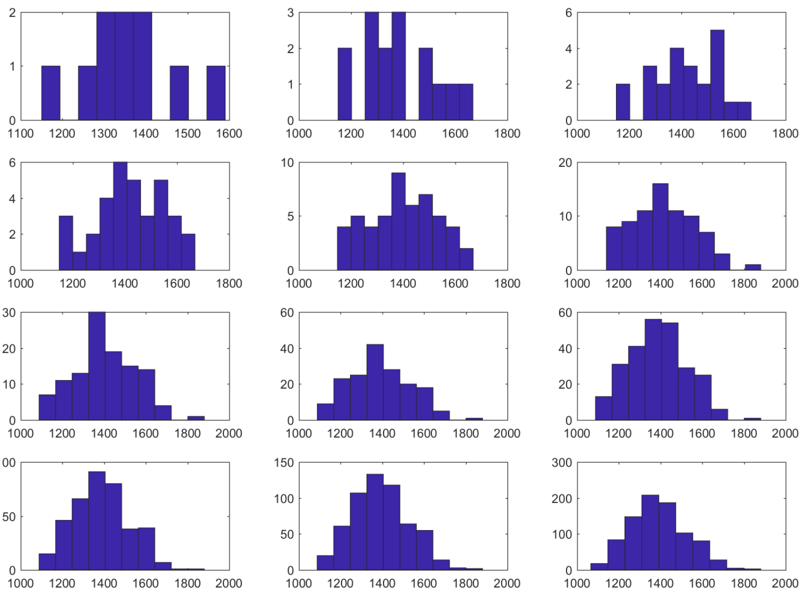


Figure D.1: Distribution of realisations of deck supply. D=1200. x=m^2/week, y=realisations. Total number of realisations, from upper left: 10,15,23,34,51,76,114,171,256,384,577,865.

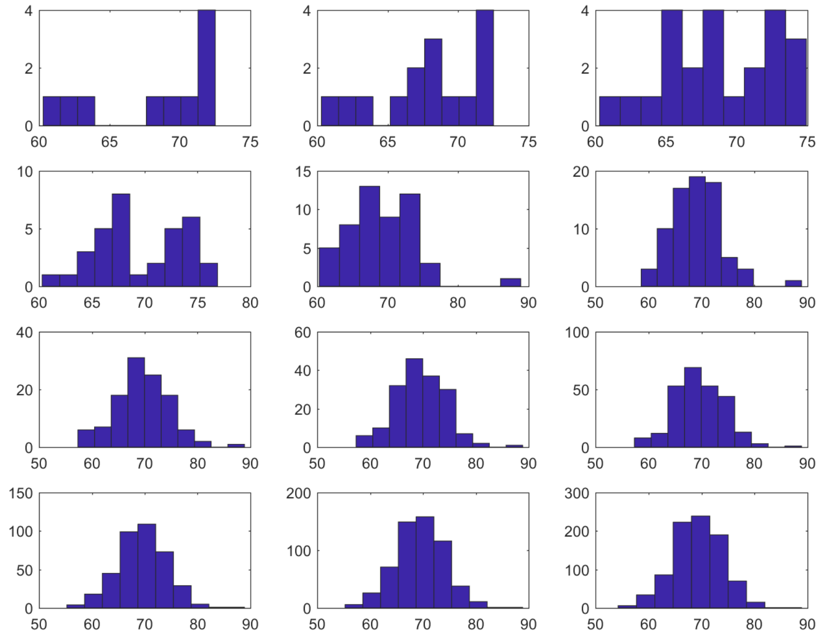


Figure D.2: Distribution of realisations of pax supply. P=70. x=pax/week, y=realisations. Total number of realisations, from upper left: 10,15,23,34,51,76,114,171,256,384,577,865.

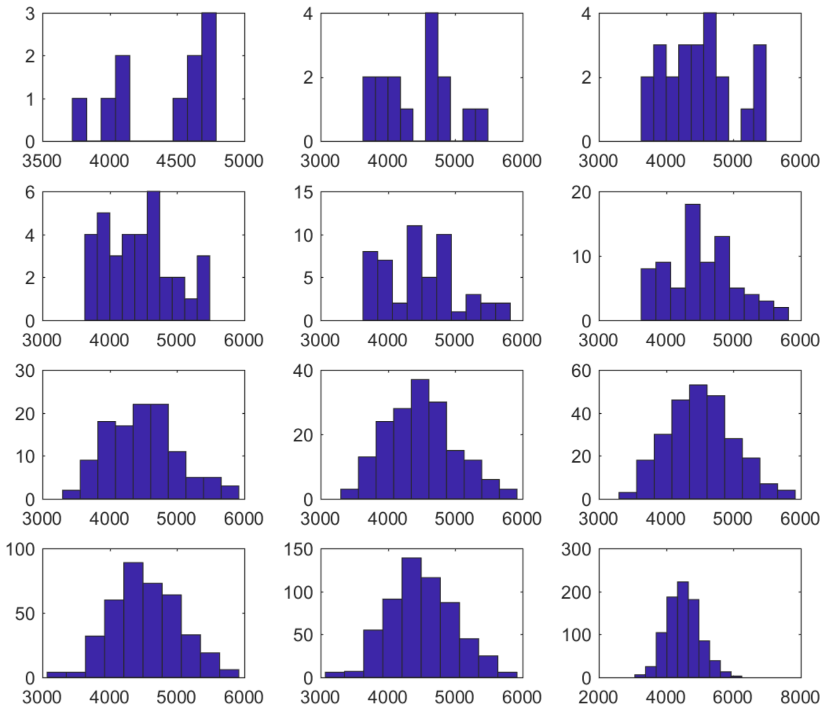


Figure D.3: Distribution of realisations of bulk supply. B=4000. x=tons/week, y=realisations. Total number of realisations, from upper left: 10,15,23,34,51,76,114,171,256,384,577,865.

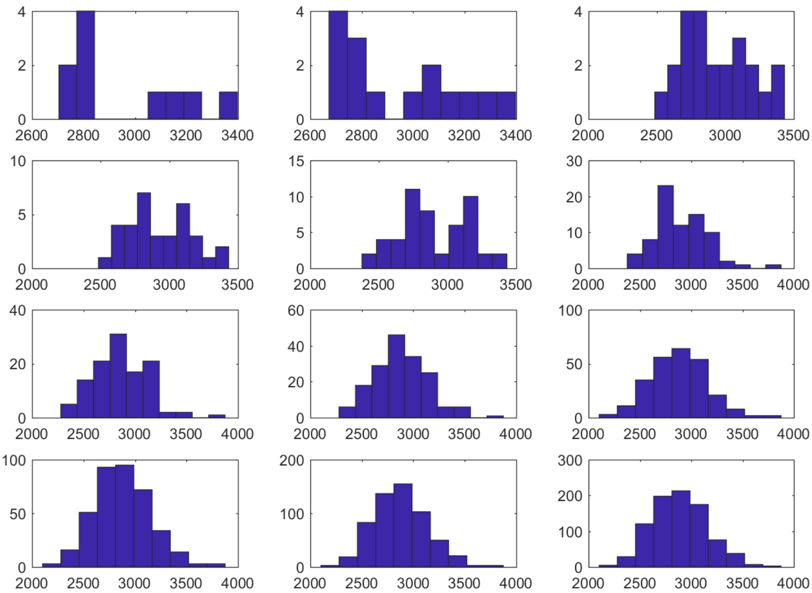


Figure D.4: Distribution of realisations of deck supply. D=2500. x=m^2/week, y=realisations. Total number of realisations, from upper left: 10,15,23,34,51,76,114,171,256,384,577,865.

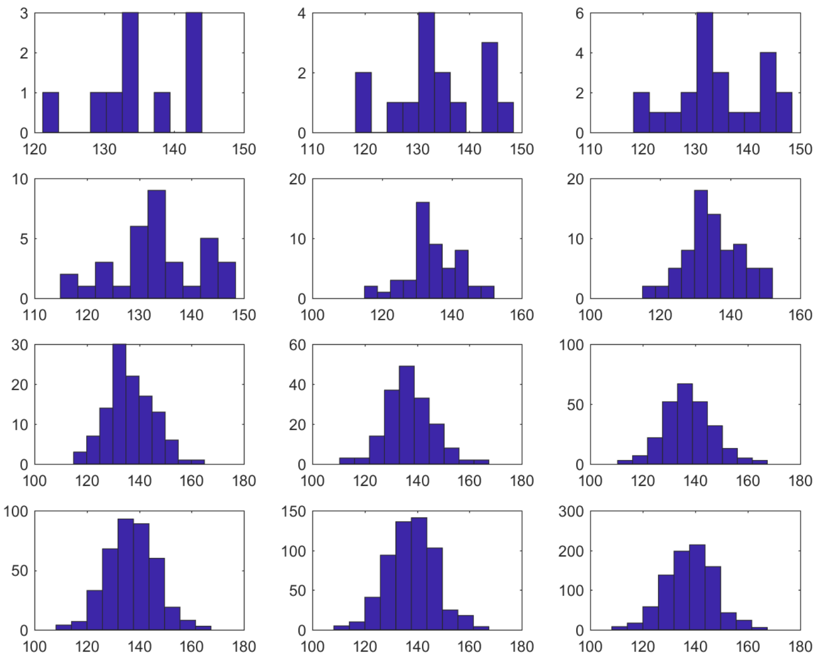


Figure D.5: Distribution of realisations of pax supply. P=140. x=pax/week, y=realisations. Total number of realisations, from upper left: 10,15,23,34,51,76,114,171,256,384,577,865.

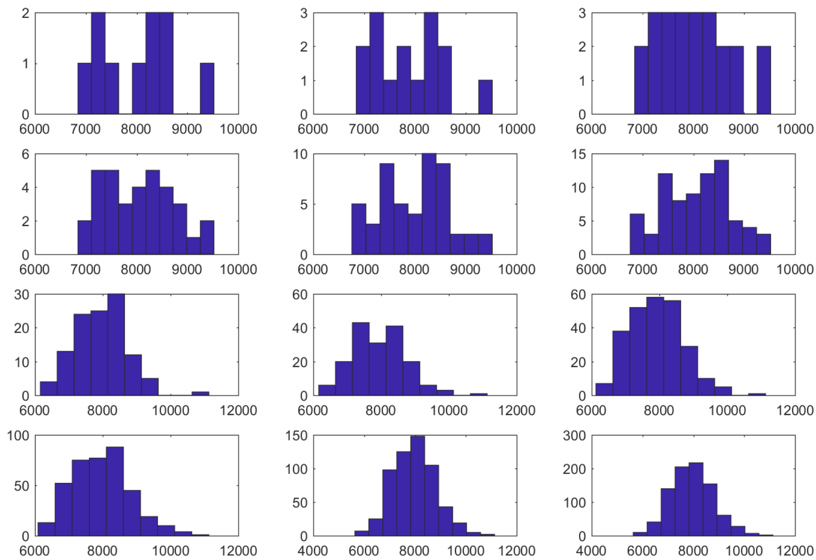


Figure D.6: Distribution of realisations of bulk supply. B=7000. x=tons/week, y=realisations. Total number of realisations, from upper left: 10,15,23,34,51,76,114,171,256,384,577,865.

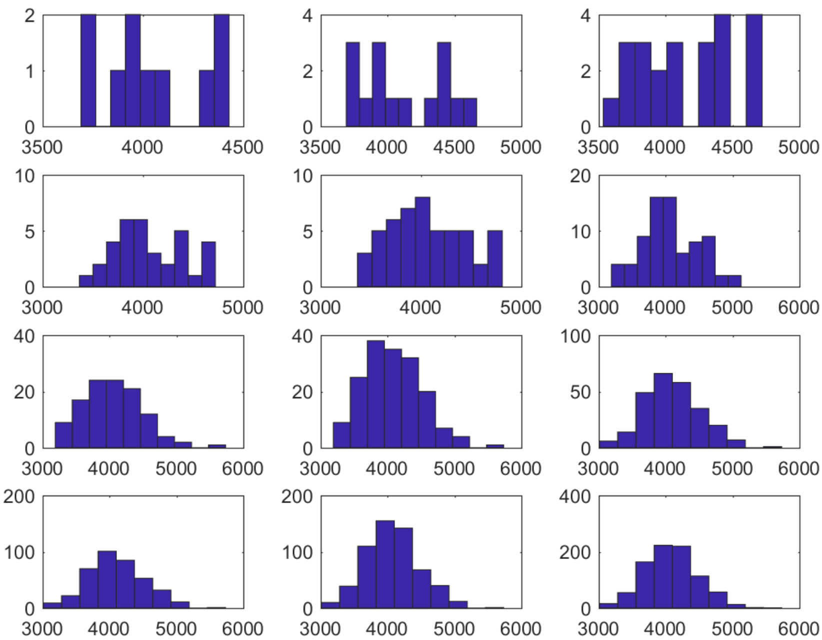


Figure D.7: Distribution of realisations of deck supply. D=3500. x=m^2/week, y=realisations. Total number of realisations, from upper left: 10,15,23,34,51,76,114,171,256,384,577,865.

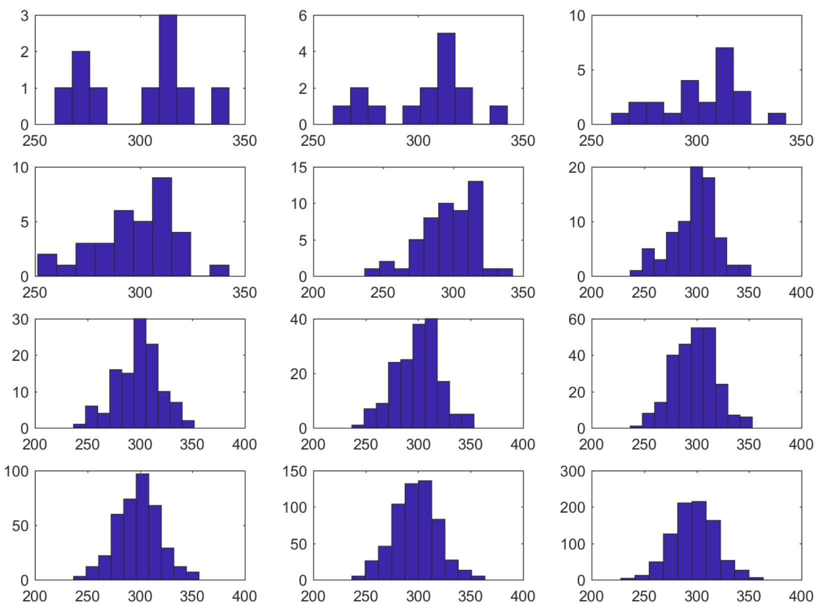


Figure D.8: Distribution of realisations of pax supply. P=300. x=pax/week, y=realisations. Total number of realisations, from upper left: 10,15,23,34,51,76,114,171,256,384,577,865.

Appendix E

Solution Process Logs

The log-table setup

The tables display all the cycles that were part of the solution process. For each cycle, the six solutions which were evaluated by simulation are presented. C_6=1 means that 1 vessel of vessel concept 6 was included in the solution. P_{95}(Cost) is the 95th percentile value of the cost of the solution, based on the different realisations the solutions were tested for. Similarly, the P_5(B),P_5(D) and P_5(P), are the 5th percentile values of the solution performances in the respective areas. Whether or not a solution is legal is presented in the fifth column. In the "Adjustment" column, for each cycle, the last set of adjustments which lead to the solutions of the cycle, are presented. As an example: in Table E.3, the solutions of cycle 3 were found based on the adjustment $FB = FB(0) \cdot FB(1) \cdot FB(2) = 1 \cdot 0.575 \cdot 0.946 = 0.544$, and clearly similarly for FD and FP.

E.1 Case1 log

Table E.1: A log of the solution process for Case 1.

Cycle nr.	Solutions			Legal?	Adjustment
	Vessels	P_{95}(Cost)	P_5(B), P_5(D), P_5(P)		
1	C_6=1	3.89e+06	0.94, 0.77, 0.95	No	FB(0)=1
	C_7=1	4.38e+06	0.78, 0.95, 0.95	No	
	C_4=1 C_{10}=1	4.63e+06	0.89, 0.92, 0.91	No	FD(0)=1
	C_4=1 C_6=1	6.22e+06	0.94, 0.95, 0.78	No	
	C_4=1 C_9=1	6.99e+06	0.94, 0.95, 0.95	Yes	FP(0)=1
	C_4=1 C_7=1	6.67e+06	0.94, 0.95, 0.80	No	
2	C_6=1	3.90e+06	0.93, 0.74, 0.95	No	FB(1)=1.040
	C_7=1	4.38e+06	0.79, 0.93, 0.95	No	
	C_4=1 C_{10}=1	4.63e+06	0.91, 0.92, 0.92	Yes	FD(1)=1.051
	C_4=1 C_6=1	6.22e+06	0.92, 0.92, 0.76	No	
	C_4=1 C_9=1	6.99e+06	0.92, 0.92, 0.95	Yes	FP(1)=1.034
	C_4=1 C_7=1	6.71e+06	0.92, 0.92, 0.85	No	
3	C_6=1	3.90e+06	0.92, 0.74, 0.95	No	FB=FB(0)
	C_7=1	4.38e+06	0.78, 0.91, 0.95	No	
	C_4=1 C_{10}=1	4.63e+06	0.91, 0.89, 0.89	No	FD=FD(0)
	C_4=1 C_6=1	6.22e+06	0.92, 0.91, 0.77	No	
	C_4=1 C_9=1	6.99e+06	0.91, 0.91, 0.95	Yes	FP=FP(0)
	C_4=1 C_7=1	6.70e+06	0.92, 0.91, 0.88	No	

E.2 Case2 log

Table E.2: A log of the solution process for Case 2.

Cycle nr.	Solutions: Vessels	P_{95}(Cost)	P_5(B), P_5(D), P_5(P)	Legal?	Adjustment
1	C_4=1 C_{10}=1	4.74e+06	0.89, 0.49, 0.68	No	FB(0)=1
	C_7=1	4.66e+06	0.47, 0.63, 0.69	No	
	C_4=1 C_6=1	6.42e+06	0.91, 0.82, 0.62	No	FD(0)=1
	C_4=1 C_7=1	6.94e+06	0.91, 0.90, 0.62	No	
	C_4=1 C_9=1	8.55e+06	0.89, 0.50, 0.95	No	FP(0)=1
	C_6=2	7.97e+06	0.91, 0.72, 0.95	No	
2	C_7=1	4.66e+06	0.48, 0.65, 0.69	No	FB(1)=1.001
	C_6=2	7.97e+06	0.92, 0.75, 0.94	No	
	C_6=1 C_7=1	8.50e+06	0.92, 0.88, 0.94	No	FD(1)=0.751
	C_4=1 C_7=1 C_{10}=1	9.20e+06	0.91, 0.92, 0.94	Yes	
	C_7=2	9.12e+06	0.78, 0.94, 0.94	No	FP(1)=0.764
	C_4=1 C_7=1 C_9=1	12.29e+06	0.92, 0.93, 0.94	Yes	
3	C_7=1,	4.66e+06	0.47, 0.64, 0.68	No	FB=FB(1)
	C_6=2	7.97e+06,	0.92, 0.73, 0.94	No	
	C_6=1 C_7=1	8.51e+06	0.91, 0.87, 0.94	No	FD=FD(1)
	C_4=1 C_7=1 C_{10}=1	9.20e+06,	0.91, 0.91, 0.94	Yes	
	C_7=2	9.14e+06	0.78, 0.92, 0.94	No	FP=FP(1)
	C_4=1 C_7=1 C_9=1	12.29e+06,	0.91, 0.91, 0.95	Yes	

E.3 Case3 log

Table E.3: A log of the solution process for Case 3.

Cycle	Solutions			Legal?	Adjustment
nr.	Vessels	P_{95}(Cost)	P_5(B), P_5(D), P_5(P)		
1	C_6=1 C_9=1	9.49e+06	0.38, 0.16, 0.60	No	FB(0)=1
	C_6=1 C_7=1	8.49e+06	0.55, 0.39, 0.47	No	
	C_4=1 C_6=1 $C_1$0=1	8.68e+06	0.79, 0.35, 0.53	No	FD(0)=1
	C_1=1 C_9=1 $C_1$0=1	10.10e+06	0.31, 0.14, 0.64	No	
	C_1=1 C_7=1 $C_1$0=1	9.12e+06	0.49, 0.39, 0.53	No	FP(0)=1
	C_1=1 C_4=1 $C_1$0=2	9.30e+06	0.75, 0.34, 0.58	No	
2	C_4=1 C_6=1 C_9=1	1.19e+07	0.80, 0.36, 0.63	No	FB(1)=0.575
	C_1=1 C_7=1 C_9=1	1.24e+07	0.51, 0.40, 0.63	No	
	C_4=1 C_7=1 C_9=1	1.25e+07	0.66, 0.47, 0.68	No	FD(1)=0.387
	C_1=1 C_4=1 C_6=1 C_9=1	1.42e+07	0.90, 0.51, 0.64	No	
	C_4=2 C_6=1 C_9=1	1.44e+07	0.90, 0.58, 0.68	No	FP(1)=0.621
	C_1=1 C_4=2 C_9=1 C_{10}=1	1.50e+07	0.90, 0.56, 0.73	No	
3	C_4=2 C_7=2 C_9=1	1.94e+07	0.93, 0.88, 0.93	No	FB(2)=0.946
	C_4=3 C_6=1 C_7=1 C_9=1	2.12e+07	0.93, 0.92, 0.93	Yes	
	C_1=1 C_4=2 C_7=2 C_9=1	2.17e+07	0.93, 0.92, 0.93	Yes	FD(2)=0.543
	C_4=3 C_7=2 C_9=1	2.18e+07	0.93, 0.93, 0.94	Yes	
	C_4=1 C_7=3 C_9=1	2.13e+07	0.91, 0.89, 0.94	No	FP(2)=0.732
	C_4=4 C_6=2 C_9=1	2.31e+07	0.93, 0.93, 0.94	Yes	
4	C_4=3 C_7=1 C_9=1	1.74e+07	0.92, 0.87, 0.78	No	FB(3)=1.034
	C_4=4 C_6=1 C_9=1	1.93e+07	0.92, 0.93, 0.78	No	
	C_4=5 C_9=1 C_{10}=1	2.00e+07	0.92, 0.93, 0.85	No	FD(3)=1.024
	C_4=4 C_7=1 C_9=1	1.99e+07	0.92, 0.93, 0.82	No	
	C_4=2 C_7=2 C_9=1	1.94e+07	0.92, 0.88, 0.94	No	FP(3)=1.040
	C_4=3 C_6=1 C_7=1 C_9=1	2.12e+07	0.92, 0.93, 0.94	Yes	
5	C_4=2 C_6=1 C_7=1 C_9=1	1.88e+07	0.92, 0.82, 0.93	No	FB(4)=1.020
	C_1=1 C_4=1 C_7=2 C_9=1	1.93e+07	0.91, 0.83, 0.92	No	
	C_4=2 C_7=2 C_9=1	1.94e+07	0.91, 0.90, 0.92	Yes	FD(4)=1.037
	C_4=3 C_6=2 C_9=1	2.07e+07	0.92, 0.91, 0.93	Yes	
	C_1=1 C_4=2 C_6=1 C_7=1 C_9=1	2.11e+07	0.92, 0.92, 0.92	Yes	FP(4)=0.928
	C_4=3 C_6=1 C_7=1 C_9=1	2.12e+07	0.92, 0.93, 0.94	Yes	
6	C_4=2 C_6=1 C_7=1 C_9=1	1.88e+07	0.90, 0.80, 0.93	No	FB(5)=1.023
	C_1=1 C_4=1 C_7=2 C_9=1	1.93e+07	0.90, 0.83, 0.93	No	
	C_4=2 C_7=2 C_9=1	1.94e+07	0.89, 0.89, 0.95	No	FD(5)=1.010
	C_4=3 C_6=2 C_9=1	2.07e+07	0.90, 0.90, 0.94	Yes	

	C_1=1 C_4=2 C_6=1 C_7=1 C_9=1	2.11e+07	0.89, 0.89, 0.92	No	FP(5)=1.032
	C_4=3 C_6=1 C_7=1 C_9=1	2.12e+07	0.90, 0.91, 0.94	Yes	
7	C_4=2 C_6=1 C_7=1 C_9=1	1.88e+07	0.90, 0.80, 0.91	No	FB=FB(1-5)
	C_1=1 C_4=1 C_7=2 C_9=1	1.93e+07	0.90, 0.83, 0.91	No	
	C_4=2 C_7=2 C_9=1	1.94e+07	0.91, 0.89, 0.93	No	FD=FD(1-5)
	C_4=3 C_6=2 C_9=1	2.07e+07	0.91, 0.91, 0.92	Yes	
	C_1=1 C_4=2 C_6=1 C_7=1 C_9=1	2.11e+07	0.91, 0.90, 0.91	Yes	FP=FP(1-5)
	C_4=3 C_6=1 C_7=1 C_9=1	2.12e+07	0.91, 0.92, 0.93	Yes	

E.4 Case4 log

Table E.4: A log of the solution process for Case 4.

Cycle nr.	Solutions			Legal?	Adjustment
	Vessels	P_{95}(Cost)	P_5(B), P_5(D), P_5(P)		
1	C_5=1	2.44e+06	0.92, 0.91, 0.94	Yes	FB(0)=1
	C_6=1	2.94e+06	0.92, 0.91, 0.94	Yes	
	C_7=1	3.14e+06	0.92, 0.91, 0.94	Yes	FD(0)=1
	C_8=1	3.89e+06	0.93, 0.75, 0.95	No	
	C_{10}=1	4.04e+06	0.93, 0.65, 0.95	No	FP(0)=1
	C_{11}=1	4.38e+06	0.78, 0.92, 0.95	No	
2	C_5=1	2.45e+06	0.91, 0.90, 0.93	Yes	FB(1)=1.025
	C_6=1	2.95e+06	0.91, 0.90, 0.93	Yes	
	C_7=1	3.15e+06	0.91, 0.90, 0.93	Yes	FD(1)=1.052
	C_8=1	3.90e+06	0.93, 0.75, 0.94	No	
	C_{10}=1	4.04e+06	0.93, 0.64, 0.94	No	FP(1)=1.051
	C_{13}=1	4.38e+06	0.72, 0.90, 0.94	No	
3	C_5=1	2.45e+06	0.91, 0.92, 0.93	Yes	FB(2)=1.013
	C_6=1	2.95e+06	0.91, 0.92, 0.94	Yes	
	C_7=1	3.15e+06	0.91, 0.92, 0.94	Yes	FD(2)=1.001
	C_8=1	3.90e+06	0.91, 0.74, 0.95	No	
	C_{10}=1	4.04e+06	0.91, 0.64, 0.95	No	FP(2)=1.037
	C_{13}=1	4.38e+06	0.72, 0.91, 0.94	No	
4	C_5=1	2.45e+06	0.92, 0.93, 0.94	Yes	FB=FB(1)
	C_6=1	2.95e+06	0.92, 0.93, 0.94	Yes	
	C_7=1	3.15e+06	0.92, 0.93, 0.94	Yes	FD=FD(1)
	C_8=1	3.90e+06	0.92, 0.74, 0.95	No	
	C_{10}=1	4.04e+06	0.92, 0.64, 0.95	No	FP=FP(1)
	C_{13}=1	4.38e+06	0.71, 0.92, 0.95	No	

E.5 Case5 log

Table E.5: A log of the solution process for Case 5.

Cycle nr.	Solutions			Legal?	Adjustment
	Vessels	**P_{95}(Cost)**	**P_5(B), P_5(D), P_5(P)**		
1	C_5=1 C_{21}=1	4.48e+06	0.90, 0.51, 0.43	No	FB(0)=1
	C_6=1 C_{20}=1	4.58e+06	0.90, 0.51, 0.43	No	
	C_{13}=1	4.66e+06	0.44, 0.58, 0.65	No	FD(0)=1
	C_{11}=1	4.65e+06	0.48, 0.71, 0.68	No	
	C_5=2	4.95e+06	0.93, 0.92, 0.85	No	FP(0)=1
	C_{12}=1	4.73e+06	0.42, 0.69, 0.75	No	
2	C_5=2	4.95e+06	0.92, 0.92, 0.84	No	FB(1)=0.765
	C_5=1 C_6=1	5.45e+06	0.92, 0.92, 0.89	No	
	C_5=1 C_7=1	5.65e+06	0.92, 0.92, 0.92	Yes	FD(1)=0.707
	C_6=2	5.95e+06	0.92, 0.92, 0.92	Yes	
	C_6=1 C_7=1	6.15e+06	0.92, 0.92, 0.92	Yes	FP(1)=0.737
	C_7=2	6.35e+06	0.92, 0.92, 0.94	Yes	
3	C_5=2	4.95e+06	0.91, 0.92, 0.85	No	FB=FB(1)
	C_5=1 C_6=1	5.45e+06	0.91, 0.92, 0.89	No	
	C_5=1 C_7=1	5.65e+06	0.91, 0.93, 0.92	Yes	FD=FD(1)
	C_6=2	5.95e+06	0.91, 0.92, 0.92	Yes	
	C_6=1 C_7=1	6.15e+06	0.91, 0.93, 0.94	Yes	FP=FP(1)
	C_7=2	6.35e+06	0.91, 0.93, 0.94	Yes	

E.6 Case6 log

Table E.6: A log of the solution process for Case 6.

Cycle nr.	Solutions			Legal?	Adjustment
	Vessels	P_{95}(Cost)	P_5(B), P_5(D), P_5(P)		
1	C_5=1 C_{18}=1	7.90e+06	0.46, 0.21, 0.54	No	FB(0)=1
	C_5=1 C_{19}=1	7.81e+06	0.46, 0.21, 0.44	No	
	C_5=1 C_{17}=1	8.23e+06	0.46, 0.21, 0.59	No	FD(0)=1
	C_6=1 C_{18}=1	8.39e+06	0.46, 0.21, 0.56	No	
	C_6=1 C_{17}=1	8.73e+06	0.46, 0.21, 0.61	No	FP(0)=1
	C_6=1 C_{19}=1	8.31e+06	0.46, 0.21, 0.46	No	
2	C_5=3 C_{19}=1	1.27e+07	0.91, 0.61, 0.79	No	FB(1)=0.513
	C_5=2 C_6=1 C_{19}=1	1.32e+07	0.91, 0.61, 0.81	No	
	C_5=2 C_7=1 C_{18}=1	1.35e+07	0.91, 0.62, 0.91	No	FD(1)=0.237
	C_5=2 C_7=1 C_{19}=1	1.34e+07	0.91, 0.62, 0.82	No	
	C_5=1 C_6=2 C_{19}=1	1.37e+07	0.91, 0.61, 0.82	No	FP(1)=0.615
	C_5=1 C_6=1 C_7=1 C_{18}=1	1.39e+07	0.91, 0.62, 0.93	No	
3	C_7=8 C_{19}=1	2.90e+07	0.93, 0.93, 0.96	Yes	FB(2)=1.016
	C_7=8 C_{18}=1	2.89e+07	0.93, 0.93, 0.96	Yes	
	C_7=9 C_{19}=1	3.19e+07	0.93, 0.93, 0.96	Yes	FD(2)=0.682
	C_7=9 C_{17}=1	3.19e+07	0.93, 0.93, 0.96	Yes	
	C_5=11 C_7=1	2.90e+07	0.93, 0.93, 0.95	Yes	FP(2)=0.915
	C_5=10 C_6=1 C_7=1	2.95e+07	0.93, 0.93, 0.95	Yes	
4	C_5=1 C_7=6 C_{19}=1	2.54e+07	0.92, 0.92, 0.96	Yes	FB(3)=1.031
	C_7=7 C_{19}=1	2.60e+07	0.92, 0.92, 0.96	Yes	
	C_7=7 C_{18}=1	2.59e+07	0.92, 0.92, 0.96	Yes	FD(3)=1.031
	C_6=1 C_7=6 C_{17}=1	2.59e+07	0.92, 0.92, 0.96	Yes	
	C_7=7 C_{17}=1	2.60e+07	0.92, 0.92, 0.96	Yes	FP(3)=1.062
	C_5=4 C_7=4 C_{18}=1	2.63e+07	0.92, 0.92, 0.96	Yes	
5	C_5=1 C_7=5 C_{19}=1	2.25e+07	0.92, 0.92, 0.96	Yes	FB(4)=1.025
	C_6=1 C_7=5 C_{19}=1	2.29e+07	0.92, 0.92, 0.96	Yes	
	C_6=1 C_7=5 C_{18}=1	2.28e+07	0.92, 0.92, 0.96	Yes	FD(4)=1.020
	C_7=6 C_{19}=1	2.31e+07	0.92, 0.92, 0.96	Yes	
	C_7=6 C_{18}=1	2.29e+07	0.92, 0.92, 0.96	Yes	FP(4)=1.071
	C_6=1 C_7=5 C_{17}=1	2.29e+07	0.92, 0.92, 0.96	Yes	
6	C_7=5 C_{19}=1	2.06e+07	0.91, 0.91, 0.93	Yes	FB(5)=1.017
	C_7=5 C_{18}=1	2.05e+07	0.91, 0.91, 0.93	Yes	
	C_5=3 C_7=3 C_{17}=1	2.12e+07	0.91, 0.91, 0.93	Yes	FD(5)=1.026
	C_5=2 C_6=1 C_7=3 C_{17}=1	2.16e+07	0.91, 0.91, 0.93	Yes	

	C_5=2 C_7=4 C_{17}=1	2.18e+07	0.91, 0.91, 0.93	Yes	FP(5)=1.065
	C_5=6 C_7=1 C_{18}=1	2.20e+07	0.91, 0.91, 0.93	Yes	
7	C_5=1 C_7=4 C_{19}=1	1.99e+07	0.92, 0.92, 0.95	Yes	FB(6)=1.012
	C_6=1 C_7=4 C_{19}=1	2.04e+07	0.92, 0.92, 0.95	Yes	
	C_6=1 C_7=4 C_{18}=1	2.03e+07	0.92, 0.92, 0.95	Yes	FD(6)=1.013
	C_7=5 C_{19}=1	2.06e+07	0.92, 0.92, 0.95	Yes	
	C_7=5 C_{18}=1	2.05e+07	0.92, 0.92, 0.95	Yes	FP(6)=1.035
	C_5=4 C_7=2 C_{19}=1	2.05e+07	0.92, 0.92, 0.95	Yes	
8	C_5=3 C_7=2 C_{18}=1	1.86e+07	0.92, 0.93, 0.95	Yes	FB(7)=1.023
	C_5=2 C_6=1 C_7=2 C_{18}=1	1.90e+07	0.92, 0.93, 0.95	Yes	
	C_5=2 C_7=3 C_{19}=1	1.93e+07	0.93, 0.93, 0.95	Yes	FD(7)=1.023
	C_5=2 C_7=3 C_{18}=1	1.92e+07	0.93, 0.93, 0.95	Yes	
	C_5=1 C_6=2 C_7=2 C_{18}=1	1.95e+07	0.92, 0.93, 0.95	Yes	FP(7)=1.060
	C_5=6 C_{19}=1	1.92e+07	0.93, 0.94, 0.95	Yes	
9	C_5=1 C_7=3 C_{19}=1	1.70e+07	0.91, 0.89, 0.94	No	FB(8)=1.026
	C_5=5 C_{19}=1	1.73e+07	0.90, 0.92, 0.94	Yes	
	C_5=5 C_{18}=1	1.73e+07	0.90, 0.92, 0.95	Yes	FD(8)=1.036
	C_5=4 C_6=1 C_{19}=1	1.78e+07	0.90, 0.92, 0.95	Yes	
	C_5=4 C_6=1 C_{18}=1	1.77e+07	0.90, 0.92, 0.95	Yes	FP(8)=1.057
	C_5=4 C_7=1 C_{19}=1	1.80e+07	0.90, 0.92, 0.94	Yes	
10	C_5=3 C_7=1 C_{19}=1	1.57e+07	0.93, 0.86, 0.94	No	FB(9)=1.052
	C_5=2 C_6=1 C_7=1 C_{19}=1	1.62e+07	0.93, 0.86, 0.94	No	
	C_5=2 C_7=2 C_{19}=1	1.64e+07	0.93, 0.87, 0.94	No	FD(9)=1.025
	C_6=2 C_7=2 C_{19}=1	1.73e+07	0.93, 0.87, 0.94	No	
	C_5=5 C_{19}=1	1.73e+07	0.94, 0.93, 0.94	Yes	FP(9)=1.050
	C_5=5 C_{18}=1	1.72e+07	0.94, 0.93, 0.94	Yes	
11	C_5=1 C_7=3 C_{19}=1	1.71e+07	0.91, 0.89, 0.95	No	FB(9)=1.052
	C_5=5 C_{19}=1	1.73e+07	0.91, 0.92, 0.95	Yes	
	C_5=5 C_{18}=1	1.72e+07	0.91, 0.92, 0.95	Yes	FD(9)=1.025
	C_6=1 C_7=3 C_{19}=1	1.75e+07	0.91, 0.89, 0.95	No	
	C_7=4 C_{19}=1	1.77e+07	0.91, 0.89, 0.95	No	FP(9)=1.050
	C_5=4 C_6=1 C_{19}=1	1.78e+07	0.91, 0.92, 0.95	Yes	
12	C_5=1 C_7=3 C_{19}=1	1.70e+07	0.92, 0.89, 0.95	No	FB=FB(1-8)
	C_5=5 C_{19}=1	1.73e+07	0.92, 0.93, 0.95	Yes	
	C_5=5 C_{18}=1	1.72e+07	0.92, 0.93, 0.95	Yes	FD=FD(1-8)
	C_6=1 C_7=3 C_{19}=1	1.75e+07	0.92, 0.89, 0.95	No	
	C_7=4 C_{19}=1	1.77e+07	0.92, 0.90, 0.95	Yes	FP=FP(1-8)
	C_5=4 C_6=1 C_{19}=1	1.78e+07	0.92, 0.93, 0.95	Yes	

E.7 Case7 log

Table E.7: A log of the solution process for Case 7.

Cycle nr.	Solutions			Legal?	Adjustment
	Vessels	P_{95}(Cost)	P_5(B), P_5(D), P_5(P)		
1	C_5=1	2.45e+06	0.92, 0.92, 0.94	Yes	FB(0)=1
	C_{26}=1	2.55e+06	0.92, 0.92, 0.94	Yes	
	C_6=1	2.95e+06	0.92, 0.92, 0.94	Yes	FD(0)=1
	C_{27}=1	3.05e+06	0.92, 0.92, 0.94	Yes	
	C_7=1	3.15e+06	0.92, 0.92, 0.94	Yes	FP(0)=1
	C_{28}=1	3.25e+06	0.92, 0.92, 0.94	Yes	
2	C_5=1	2.44e+06	0.90, 0.89, 0.94	No	FB(1)=1.022
	C_{26}=1	2.55e+06	0.90, 0.89, 0.94	No	
	C_6=1	2.95e+06	0.90, 0.89, 0.94	No	FD(1)=1.027
	C_{27}=1	3.05e+06	0.90, 0.89, 0.94	No	
	C_7=1	3.15e+06	0.90, 0.89, 0.94	No	FP(1)=1.049
	C_{28}=1	3.25e+06	0.90, 0.89, 0.94	No	
3	C_5=1	2.45e+06	0.91, 0.92, 0.94	Yes	FB=FB(0)
	C_{26}=1	2.55e+06	0.91, 0.92, 0.94	Yes	
	C_6=1	2.95e+06	0.91, 0.92, 0.95	Yes	FD=FD(0)
	C_{27}=1	3.05e+06	0.91, 0.92, 0.95	Yes	
	C_7=1	3.15e+06	0.91, 0.92, 0.95	Yes	FP=FP(0)
	C_{28}=1	3.25e+06	0.91, 0.92, 0.95	Yes	

E.8 Case8 log

Table E.8: A log of the solution process for Case 8.

Cycle nr.	Solutions			Legal?	Adjustment
	Vessels	P_{95}(Cost)	P_5(B), P_5(D), P_5(P)		
1	C_{30}=1	4.14e+06	0.87, 0.44, 0.56	No	FB(0)=1
	C_5=1 C_{21}=1	4.48e+06	0.89, 0.52, 0.43	No	
	C_5=1 C_{20}=1	4.58e+06	0.89, 0.52, 0.43	No	FD(0)=1
	C_5=1 C_{42}=1	4.58e+06	0.89, 0.52, 0.43	No	
	C_{21}=1 C_{26}=1	4.58e+06	0.89, 0.52, 0.43	No	FP(0)=1
	C_{11}=1	4.65e+06	0.48, 0.72, 0.68	No	
2	C_5=1 C_9=1	6.49e+06	0.92, 0.87, 0.93	No	FB(1)=0.999
	C_9=1 C_{26}=1	6.58e+06	0.92, 0.86, 0.93	No	
	C_6=1 C_9=1	6.98e+06	0.92, 0.87, 0.93	No	FD(1)=0.575
	C_9=1 C_{27}=1	7.08e+06	0.92, 0.86, 0.93	No	
	C_7=1 C_9=1	7.18e+06	0.92, 0.89, 0.94	No	FP(1)=0.474
	C_9=1 C_{28}=1	7.28e+06	0.92, 0.89, 0.93	No	
3	C_5=1 C_9=1	6.48e+06	0.91, 0.89, 0.93	No	FB(2)=1.021
	C_9=1 C_{26}=1	6.58e+06	0.91, 0.89, 0.93	No	
	C_6=1 C_9=1	6.98e+06	0.91, 0.89, 0.93	No	FD(2)=0.964
	C_9=1 C_{27}=1	7.08e+06	0.91, 0.89, 0.93	No	
	C_7=1 C_9=1	7.18e+06	0.91, 0.91, 0.94	Yes	FP(2)=1.033
	C_9=1 C_{28}=1	7.28e+06	0.91, 0.90, 0.94	Yes	
4	C_5=1 C_9=1	6.48e+06	0.91, 0.88, 0.93	No	FB=FB(1-2)
	C_9=1 C_{26}=1	6.58e+06	0.91, 0.88, 0.93	No	
	C_6=1 C_9=1	6.98e+06	0.91, 0.88, 0.94	No	FD=FD(1-2)
	C_9=1 C_{27}=1	7.08e+06	0.91, 0.88, 0.94	No	
	C_7=1 C_9=1	7.18e+06	0.91, 0.89, 0.94	No	FP=FP(1-2)
	C_9=1 C_{28}=1	7.28e+06	0.91, 0.89, 0.94	No	
5	C_5=1 C_9=1	6.48e+06	0.92, 0.89, 0.92	No	FB=
	C_9=1 C_{26}=1	6.58e+06	0.92, 0.89, 0.92	No	
	C_6=1 C_9=1	6.98e+06	0.92, 0.89, 0.94	No	FD=
	C_9=1 C_{27}=1	7.08e+06	0.92, 0.89, 0.94	No	
	C_7=1 C_9=1	7.18e+06	0.92, 0.90, 0.94	Yes	FP=
	C_9=1 C_{28}=1	7.28e+06	0.92, 0.89, 0.94	No	

E.9 Case9 log

Table E.9: A log of the solution process for Case 9.

Cycle nr.	Solutions			Legal?	Adjustment
	Vessels	P_{95}(Cost)	P_5(B), P_5(D), P_5(P)		
1	C_5=1 C_{18}=1	7.90e+06	0.45, 0.21, 0.53	No	FB(0)=1
	C_5=1 C_{17}=1	8.23e+06	0.45, 0.21, 0.58	No	
	C_5=1 C_{19}=1	7.81e+06	0.45, 0.21, 0.43	No	FD(0)=1
	C_5=1 C_{39}=1	8.00e+06	0.45, 0.21, 0.53	No	
	C_{18}=1 C_{26}=1	8.00e+06	0.45, 0.21, 0.53	No	FP(0)=1
	C_{19}=1 C_{26}=1	7.91e+06	0.45, 0.21, 0.43	No	
2	C_5=3 C_{19}=1	1.27e+07	0.90, 0.63, 0.81	No	FB(1)=0.503
	C_5=3 C_{40}=1	1.28e+07	0.90, 0.63, 0.81	No	
	C_5=2 C_{19}=1 C_{26}=1	1.28e+07	0.90, 0.63, 0.81	No	FD(1)=0.238
	C_5=1 C_{19}=1 C_{26}=2	1.29e+07	0.90, 0.63, 0.81	No	
	C_{19}=1 C_{26}=3	1.30e+07	0.90, 0.63, 0.81	No	FP(1)=0.586
	C_{26}=3 C_{40}=1	1.31e+07	0.90, 0.63, 0.81	No	
3	C_7=6 C_{19}=1	2.31e+07	0.92, 0.94, 0.94	Yes	FB(2)=1.002
	C_7=6 C_{40}=1	2.32e+07	0.92, 0.94, 0.94	Yes	
	C_7=6 C_{39}=1	2.31e+07	0.92, 0.94, 0.94	Yes	FD(2)=0.700
	C_7=6 C_{17}=1	2.32e+07	0.92, 0.94, 0.94	Yes	
	C_7=6 C_{38}=1	2.33e+07	0.92, 0.94, 0.94	Yes	FP(2)=0.897
	C_{19}=1 C_{28}=6	2.37e+07	0.92, 0.94, 0.94	Yes	
4	C_5=1 C_7=3 C_{28}=1 C_{40}=1	2.01e+07	0.94, 0.93, 0.95	Yes	FB(3)=1.028
	C_7=3 C_{26}=1 C_{28}=1 C_{40}=1	2.02e+07	0.94, 0.93, 0.95	Yes	
	C_7=2 C_{26}=1 C_{28}=2 C_{40}=1	2.31e+07	0.94, 0.93, 0.95	Yes	FD(3)=1.041
	C_7=5 C_{19}=1	2.06e+07	0.94, 0.93, 0.95	Yes	
	C_6=1 C_7=3 C_{28}=1 C_{40}=1	2.06e+07	0.94, 0.93, 0.95	Yes	FP(3)=1.045
	C_7=5 C_{18}=1	2.05e+07	0.94, 0.93, 0.95	Yes	
5	C_5=3 C_7=1 C_{28}=1 C_{40}=1	1.88e+07	0.93, 0.91, 0.94	Yes	FB(4)=1.044
	C_5=3 C_{28}=2 C_{40}=1	1.89e+07	0.93, 0.91, 0.94	Yes	
	C_5=1 C_{26}=1 C_{27}=2 C_{28}=2 C_{40}=1	1.96e+07	0.93, 0.91, 0.94	Yes	FD(4)=1.034
	C_5=2 C_{28}=3 C_{40}=1	1.97e+07	0.93, 0.91, 0.94	Yes	
	C_5=6 C_{39}=1	1.92e+07	0.93, 0.91, 0.94	Yes	FP(4)=1.052
	C_5=1 C_{26}=1 C_{28}=3 C_{40}=1	1.98e+07	0.93, 0.91, 0.94	Yes	
6	C_5=4 C_7=1 C_{18}=1	1.79e+07	0.90, 0.92, 0.94	Yes	FB(5)=1.032
	C_7=1 C_{26}=4 C_{39}=1	1.84e+07	0.90, 0.92, 0.94	Yes	
	C_5=1 C_{18}=1 C_{26}=2 C_{28}=2	1.89e+07	0.90, 0.92, 0.94	Yes	FD(5)=1.010
	C_6=1 C_{18}=1 C_{26}=2 C_{27}=1 C_{28}=1	1.92e+07	0.90, 0.92, 0.94	Yes	

	C_5=2 C_7=3 C_{39}=1	1.93e+07	0.90, 0.92, 0.94	Yes	FP(5)=1.050
	C_5=1 C_6=1 C_{18}=1 C_{26}=1 C_{28}=2	1.93e+07	0.90, 0.92, 0.94	Yes	
7	C_5=5 C_{18}=1	1.72e+07	0.93, 0.94, 0.96	Yes	FB(6)=1.003
	C_5=4 C_{19}=1 C_{26}=1	1.74e+07	0.93, 0.93, 0.96	Yes	
	C_7=4 C_{19}=1	1.77e+07	0.91, 0.89, 0.96	No	FD(6)=1.022
	C_7=4 C_{40}=1	1.78e+07	0.91, 0.89, 0.96	No	
	C_5=1 C_{18}=1 C_{26}=4	1.76e+07	0.93, 0.93, 0.96	Yes	FP(6)=1.048
	C_{18}=1 C_{26}=5	1.77e+07	0.93, 0.93, 0.96	Yes	
8	C_5=3 C_7=1 C_{40}=1	1.58e+07	0.91, 0.98, 0.95	No	FB(7)=1.029
	C_5=3 C_{28}=1 C_{40}=1	1.59e+07	0.91, 0.98, 0.95	No	
	C_5=1 C_7=1 C_{26}=2 C_{40}=1	1.60e+07	0.91, 0.98, 0.94	No	FD(7)=1.039
	C_7=1 C_{26}=3 C_{40}=1	1.62e+07	0.91, 0.98, 0.95	No	
	C_5=2 C_6=1 C_7=1 C_{40}=1	1.63e+07	0.91, 0.98, 0.95	No	FP(7)=1.070
	C_5=2 C_7=1 C_{27}=1 C_{40}=1	1.64e+07	0.91, 0.98, 0.95	No	
9	C_6=2 C_7=1 C_{19}=1 C_{28}=1	1.74e+07	0.90, 0.86, 0.95	No	FB(8)=1.016
	C_5=5 C_{18}=1	1.72e+07	0.90, 0.93, 0.95	Yes	
	C_5=5 C_{40}=1	1.74e+07	0.90, 0.93, 0.95	Yes	FD(8)=0.978
	C_5=5 C_{39}=1	1.74e+07	0.90, 0.93, 0.95	Yes	
	C_5=3 C_{19}=1 C_{26}=2	1.75e+07	0.90, 0.93, 0.95	Yes	FP(8)=1.050
	C_5=3 C_{18}=1 C_{26}=2	1.75e+07	0.90, 0.93, 0.95	Yes	
10	C_5=1 C_7=3 C_{19}=1	1.70e+07	0.92, 0.90, 0.95	Yes	FB=FB(1-6)
	C_7=3 C_{26}=1 C_{40}=1	1.73e+07	0.92, 0.90, 0.95	Yes	
	C_5=5 C_{19}=1	1.73e+07	0.92, 0.93, 0.95	Yes	FD=FD(1-6)
	C_5=3 C_{19}=1 C_{26}=2	1.75e+07	0.92, 0.93, 0.95	Yes	
	C_7=3 C_{19}=1 C_{27}=1	1.76e+07	0.92, 0.90, 0.95	Yes	FP=FP(1-6)
	C_7=4 C_{19}=1	1.77e+07	0.92, 0.91, 0.95	Yes	