Dissertation

Decision Support for Liner Shipping Network Decisions

M.Sc. Daniel Müller

Submitted to the Department of Business Information Systems
in fulfillment of the requirements for the degree of
Doctor rerum politicarum (Dr. rer. pol.)
at the University of Paderborn

Dean of the Faculty of Business Administration and Economics:
Prof. Dr. Caren Sureth-Sloane

Examiners:
1. Prof. Dr. Leena Suhl
2. Prof. Dr. Kevin Tierney (Universität Bielefeld)

July 16, 2018

Abstract

Liner shipping is one of the most important building blocks of modern international trade. Throughout the year, a liner shipping company adjusts its network to seasonal and general economic trends. Such a modification is a heavy intervention with large financial burdens, making it necessary to use sophisticated planning techniques. Although many researchers developed various optimization models for the liner shipping industry, planners in liner shipping companies are still relying on inadequate decision support tools that are not exactly tailored to their planning problems. Therefore, it is of high importance that requirements of real-world planners are considered in mathematical optimization models and decision support systems. To this end, extensions of state-of-the-art optimization models and solution approaches are presented as well as an analysis of real-world vessel data to evaluate travel times and a decision support system with a newly developed problem-specific visualization technique. This work shows that the gap between the practical world and the research world can be bridged by incorporating practical requirements into optimization models as well as making these models and their results more accessible, which increases the chances of planners adopting these models in their daily operations.

Keywords: maritime transportation, liner shipping, fleet repositioning, decision support system, visualization, cargo allocation, fleet deployment, service design, biased random-key, genetic algorithm

Scientific papers included in this thesis

- **Chapter 4.** Müller, D., Tierney, K.: Decision support and data visualization for liner shipping fleet repositioning. *Information Technology and Management*, 18(3):203-221, 2017.
- **Chapter 5.** Müller, D., Guericke, S., Tierney, K.: Integrating fleet deployment into the liner shipping cargo allocation problem. In *International Conference on Computational Logistics*, pages 306-320, 2017.
- **Chapter 6.** Tierney, K., Ehmke, J. F., Campbell, A. M., Müller, D.: Liner Shipping Single Service Design Problem with Arrival Time Service Levels. *Flexible Services and Manufacturing Journal*, 2018.
- **Chapter 7.** Müller, D.: A Biased Random-Key Genetic Algorithm for the Liner Shipping Fleet Repositioning Problem. *DS&OR Lab Working Paper*, no. 1801, 2018.

Acknowledgments

Almost exactly four years ago, the journey to this thesis started. Just like a journey at sea, this one had its rough parts as well as its enjoyable parts. Today, I would not be arriving at my destination, if it wasn't for the following group of people, who supported me along the way.

First of all, I would like to thank Kevin Tierney for supervising me. It has been a great honor to be his very first PhD student. I'm happy that he introduced me to the exciting field of maritime transportation. I want to thank Leena Suhl for sparking my interest in the field of Operations Research. All the best for your well-deserved retirement. Without Peter Volmich and Carina Uhde, the Decision Support & Operations Research Lab would be in chaos, so I thank both for all their help in various situations. It was always a pleasure to work at the DS&OR Lab with the freedom to propose students projects, enjoy the weekly Chocolate Wednesday, various field trips or conferences and the occasional discussion on our research. It wouldn't have been the same without Marius Merschformann, Henry Wolf, Corinna Hallmann, Daniela Guericke, Christoph Weskamp, Lars Beckmann, Kostja Siefen, Florian Isenberg, Florian Stapel, Michaela Wissing, Stefan Kuhlemann, André Hottung and Artus Krohn-Grimberghe. Thank you all for your support over the last years.

I would like to express my special thanks to Marius Merschformann and Henry Wolf. The discussions with Marius have been a vital part of this work and his support at harder times was what brought me back on track. I thank Henry for letting me join him at his EU projects and for bringing me together with some incredible people.

It was a pleasure to work with Stefan Guericke, Jan Fabian Ehmke and Ann Campbell as co-authors. I thank Dario Pacino for supporting me at various conferences. I had many amazing experiences during Blended-Aim with Nuno Escudeiro, Kristien van Assche, Wio D'Hespeel, Edwin Gray, Davy De Winne and Saskia Wierinck. I want to thank them for some crazy evenings in Porto, Glasgow, Rovereto and Paderborn. In addition, I thank Daniel Eisberg for his comments on this thesis.

I would like to thank my Mum and Dad for their support in everything I do and for clearing any doubts I had during the last years. Also, I thank my sisters and the rest of my family for their motivation to finish this thesis.

Finally, I'm deeply thankful for the loving support of Christin and I couldn't be happier to spend my future with you.

Daniel Müller
Paderborn, July 2018

Contents

List of Figures

Part I.

Synopsis

1. Introduction

With a tradition of more than 5,000 years, maritime transportation has seen plenty of developments and technologies over the last thousands of years (Stopford (2009)). Today, the amount of transported cargo per year surpassed 10 billion tons, making maritime transportation one of the most important building blocks of modern international trade. In order to transport these large quantities, a fleet of about 49,000 vessels of all kinds are sailing around the world (UNCTAD (2016)).

Compared to other modes of transportation like air-based or land-based transportation, maritime transportation provides the highest transportation capacity with several tens of thousands of tons on a single ship (Vahrenkamp and Kotzab (2012)). With such a high transportation capacity, the transportation costs for a single cargo unit becomes comparably small to the other transportation modes. Since the invention of standardized cargo containers in the 1950s, liner shipping has become a reliable way to transport cargo to locations around the world.

Similar to bus routes in public transport, vessels in liner shipping networks visit a fixed sequence of ports at specified times in order to load and unload cargo. Just like it is possible to switch from one bus line to another at certain stops, vessel schedules allow for transshipments of cargo at selected ports of their routes. The liner shipping schedules are publicly available such that customers can synchronize their transportation with the vessel's arrival and departure times. Throughout the year, these schedules and the service networks are modified by liner carriers in order to adjust to seasonal or general economic trends. Integrating these modifications into the service network is a complex planning task. Unlike bus services that are usually restricted to a certain part of the day before the buses drive back to their depot, vessels on liner services are constantly sailing on their routes.

A modification of a service is a heavy intervention on the service network, usually with large financial ramifications included. Sophisticated planning techniques should be considered to keep the financial burden of such an intervention as low as possible. In reality, many planners in liner shipping companies are still relying on their experience and tools like spreadsheet software, which is not completely tailored to support such a complex decision process.

The focus of this work lies on extending the state-of-the-art library of models and algorithms to solve planning problems, where the basis of the decision process is the modification of a liner service or the evaluation of such a modification. For this, four research goals were posed:

1. Extend the scope of selected optimization models from liner shipping

2. Develop techniques to better visualize optimization problems in liner shipping
3. Analyze real-world shipping data to improve the precision of existing optimization models
4. Propose a decision support system to integrate developed models and visualization techniques

The remainder of this thesis is structured as follows: Part I contains the synopsis of this work, which starts with this introductory chapter. Chapter 2 contains background information about relevant topics for this thesis like maritime transportation (Section 2.1), decision support systems (Section 2.2) and the methodology that was used in this thesis (Section 2.3). An overview of the four research papers of this thesis is given in Chapter 3. After this overview, Part II presents the research papers. It starts with a work on liner shipping fleet repositioning in Chapter 4, followed by an extension of the liner shipping cargo allocation problem in Chapter 5. The third paper of this thesis presents several mathematical models for the liner shipping single service design problem (Chapter 6). The final paper of this thesis presents a biased random-key genetic algorithm for the liner shipping fleet problem, which is combined with a hill climbing algorithm (Chapter 7). Finally, this thesis is concluded in Chapter 8 with an overview of the contributions of this thesis as well as an outlook on possible future research.

2. Background

2.1. Maritime transportation

2.1.1. General overview

According to UNCTAD (2016), maritime transportation cargo can be split into three categories: oil and gas, main bulk commodities and dry cargo. In 2015, about 2.95 billion tons of oil and gas and 2.95 billion tons of main bulk commodities have been transported around the world. The amount of dry cargo in the same year was about 4.15 billion tons, resulting in a total sum of more than 10 billion tons. This is more than double compared to what was transported in 1980, where the total amount of transported goods reached 3.7 billion tons. From that time, this number has been steadily growing. Only in 1985 and 2009 were some pullbacks[1]. The reason for the most recent drop in 2009 was the worldwide financial crisis. This particular pullback was already evened out in 2010 with a higher transportation volume than in 2008. Although there was a growth in every area of maritime transportation, the largest growth in this time frame has happened in the area of liner shipping, which belongs to the category of dry cargo.

2.1.2. Liner shipping

In liner shipping, vessels are carrying standardized containers and sail on predefined routes with a specified schedule (Stopford (2009)). The transportation of containerized goods began in the 1950s when Malcolm McLean shipped the first containers on a specialized vessel and continued with the standardization by the International Standardization Organization (ISO) in the late 1960s. This invention significantly improved the handling of cargo at ports, as it unified several boxes or bags into a single unit. Nowadays, the amount of transported containers has reached 175 Mio. TEU[2] (UNCTAD (2016)).

Liner shipping companies operate cyclical services in order to transport containerized goods. A service connects several ports, which are visited in a specific order. A service can be used to connect different trade regions, for example Europe and Asia. Ports in a service are visited at fixed time periods, following a certain frequency (usually weekly or biweekly).

Figure 2.1 presents an example of a liner shipping service network with three services: Chennai Express, Asia-CA3 and Intra-WCSA. Each service has its own route

[1]the measurements from 1980 to 2005 are only available in 5-year-steps, see UNCTAD (2016)
[2]TEU = twenty foot equivalent units

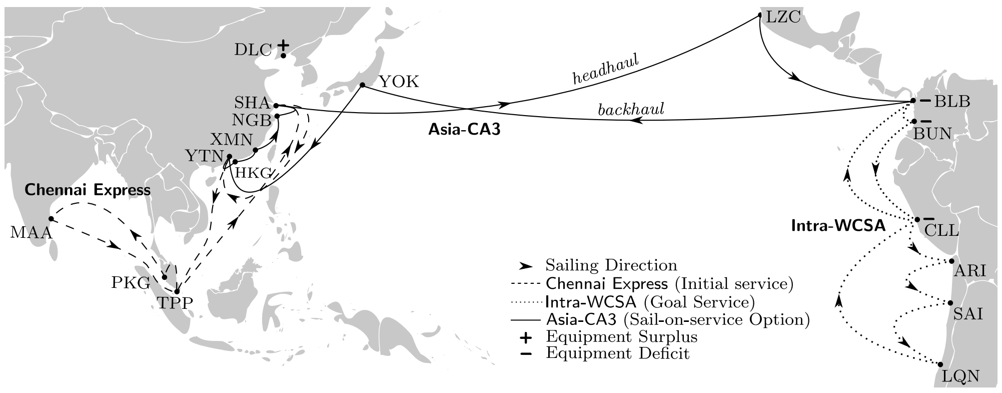


Figure 2.1.: An example of a liner shipping service network from Tierney et al. (2014).

with multiple ports. Chennai Express is only serving ports in Asia and Intra-WCSA is only serving ports in South America. The service Asia-CA3 connects these two services by visiting ports on both continents.

Multiple vessels are assigned to a service, each taking a specific slot. Each slot has its own schedule to visit the ports of the service. Depending on the desired frequency of the service, the number of slots is selected such that the vessels of the service maintain the frequency.

Whenever a vessel stops at a port, cargo is loaded or unloaded. This operation is always performed by container cranes. Containers that are unloaded and are not directly transported to the hinterland or transshipped to another vessel are stored in container yards. Transshipments usually occur whenever the original port and the destination port are not on the same service. In this case, the container needs to be routed over the different services of the liner shipping company to bring it to its destination. For shipments to the hinterland, containers are picked up by trucks, put onto trains or on barges to bring them to their final destination. Trucks, trains and barges are also moving containers from the hinterland to the ports to load them onto container vessels.

As the container trade covers a large range of products that are transported, it is heavily influenced by worldwide trends in consumer behavior. Furthermore, the seasons of the year also have a great influence on the amount of containers to be transported on specific liner services. For example, harvest times or holidays can lead to a peaks in container demand for services connecting the harvest or holiday region. Liner carriers need to adjust to these peaks in order to fulfill this temporary higher demand. Not just at the time of these peaks, but also for the rest of the year, there is a certain degree of cargo imbalances. Some regions have a higher export of containers then other regions, where import is higher. This raises the problem, that liner carriers have to reposition empty containers to locations, where they are needed.

Characteristics of liner shipping

There are specific container types for different purposes: containers with tanks to transport liquids, containers with ventilation, 20 foot containers, extra high containers or reefer containers to name a few examples. Reefer containers can carry cargo that needs to be cooled down. For this, electricity is needed to power the cooling system of the container. Container vessels provide a certain amount of places, called plugs, for reefer containers, where they can be connected to electricity.

Since the first container vessels, the total capacity for containers has increased from around 500 TEU to more than 20,000 TEU on the largest vessels. One of the main reasons for this increase in vessel size is the economies of scale. According to Stopford (2009), operating costs, capital costs and bunker costs "do not increase proportionally with the transport capacity of the ship". Stopford (2009) also points out that the economies of scale diminish as the ships are getting bigger.

These three cost factors are items from a longer list of costs associated with running a liner shipping business. In total the costs can be classified in five categories (Stopford (2009)):

1. Operating costs
2. Periodic maintenance
3. Voyage costs (including bunker costs)
4. Cargo-handling costs
5. Capital costs

Running a liner shipping company is a capital intensive business. According to Stopford (2009), a 1,200 TEU vessel costs $25 million in 2006. Due to the fierce competition in the liner shipping business, carriers are forced to optimize their costs such that they are able to offer competitive prices to their clients, while still trying to achieve a certain degree of profitability. For this, it is critical for liner carriers to make use of advanced planning methods like mathematical optimization to improve their network structures, schedules and solve other planning problems related to operating a liner business.

Planning problems in liner shipping

An overview of the planning problems that occur in liner shipping is given in Christiansen et al. (2007), Christiansen et al. (2013), Brouer et al. (2017) and Meng et al. (2014). These publications classify planning problems according to their planning horizons into strategic, tactical and operational problems.

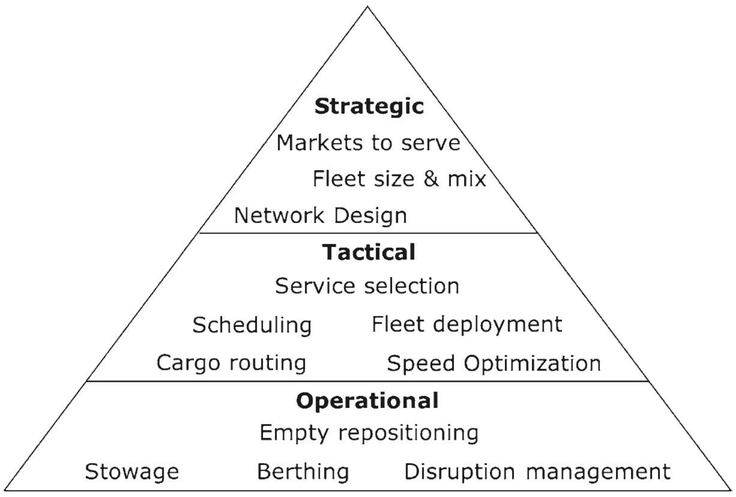


Figure 2.2.: Classification of selected liner shipping planning problems from Brouer et al. (2017).

Figure 2.2 gives an overview of selected planning problems from the field of liner shipping. It should be noted that these problems often depend on each other and a decision on a strategic level will have implications to other decisions on a tactical or even operational level. As an introduction into the general topic of planning problems in liner shipping, some of the planning problems from Figure 2.2 will be discussed in the following.

Due to the fact that a newly built vessel can be used for up to 30 years, it is obvious that any decision involving the fleet is a strategic decision. One major strategical decision is to determine the *size of the fleet and its mix*. With this planning problem, the existing vessel types and sizes are adjusted as well as the number of vessels of each type or size. The basis for this decision is the already existing fleet, the service network of the liner shipping company and its future expectations of demands and economic developments. With this, a management strategy for the fleet can be developed and decisions like buying or selling vessels are supported (Christiansen et al. (2013)).

Another planning problem at the strategic level is the *liner shipping network design*, as it determines the routes of the vessels. For this, it is important to identify ports that should be visited and the sequence in which the port visits should happen. There should be an estimate of possible future cargo that can be loaded or unloaded at the ports in order to be able to evaluate possible services. Furthermore, the frequency of the service needs to be fixed, because this also determines the number of vessels that are needed on the service (Agarwal and Ergun (2008), Brouer et al. (2017)).

In order to evaluate any network modifications, planners can use the *liner shipping cargo allocation problem* to compute the profit-maximum cargo flows possible on their services, given deterministic cargo demands (Guericke and Tierney (2015)). Although this problem is highly dependent on other planning problems like the network design or the fleet deployment, optimization models often consider these problems separately (Müller et al. (2017)).

At a tactical level, liner shipping companies need to make decisions about the assignment of vessels to routes, which is called the *fleet deployment problem* (Gelareh and Meng (2010)). Vessel characteristics, the structure of the services and expected demands have to be considered to be able to determine a good deployment. In practice, liner shipping companies are able to charter in additional vessels for their services or they can charter out their vessels to other companies. The fleet deployment problem supports liner carriers in these decisions.

Related to this problem is the *liner shipping fleet repositioning problem*. Whenever the fleet deployment changes, vessels need to be repositioned to another service (Tierney et al. (2014)). Usually, the majority of the fleet is actively sailing on a service as it is too expensive to keep vessels at a port for a long time. Therefore, any fleet deployment change needs to be implemented during the actual operations of the services. For this, the liner shipping fleet repositioning problem identifies optimal repositioning activities in order to keep the costs for such a reassignment at a minimum.

At a short term horizon, liner shipping companies need to make decisions regarding the actual operations of their vessels. For example, it is necessary to determine the actual positioning of the containers on the vessel when containers are loaded, which is called the *stowage planning problem* (Pacino (2012)). The stowage planning is used to avoid unnecessary container movements at ports as well as balancing the container load, guaranteeing its seaworthiness. Another goal of liner shipping companies is the reduction of bunker fuel consumption to save costs by adjusting the vessel speed. Due to the fact that bunker fuel consumption increases cubically with the speed of the vessel (Brouer et al. (2013)), *speed optimization* is used to reduce bunker costs while still holding the schedules of vessels. Furthermore, many liner shipping companies have introduced the concept of "slow steaming", where vessels sail below their design speed (Meyer et al. (2012)).

Due to the size of the liner shipping network of a carrier, its deployed vessels and the long time scales in liner shipping, all these planning problems are complex and need detailed information of the fleet, the ports of the services and other conditions like weather or expected demands. For this, sophisticated tools and methods are needed to support the planners in solving these problems. Therefore, the use of mathematical optimization models and decision support systems (DSS) becomes more and more attractive for liner shipping companies.

2.2. Decision support systems

Planners, managers and executives in maritime transportation need to make many decisions in a complex and dynamic environment, each with possibly large financial impacts. Therefore, decision makers should have easy access to relevant information and tools available that support them in their decision making process. Computer-based decision support systems have been developed to improve the planning capabilities of planners.

A definition of a DSS can be found in Sauter (2010): “a DSS is a computer-based system that supports choice by assisting the decision maker in the organization of information and modeling of outcomes”. Technologically, such a system consists of several components: data handling, model handling and the graphical user interface (Shim et al. (2002)). These components serve special purposes like storing data or support the user in modeling his planning problems. With the first development of decision support systems in the 1970s, the focus of DSS has been on improving the efficiency of the decision making process, as well as on improving the effectiveness of a decision (Shim et al. (2002)). Since then, computer systems have become cheaper and easier to access for the general public. Today, many DSS are web-based, where the tools of the system are presented on a website. Such an approach improves the maintainability of the system, as the user only has to install a browser on his client and the core of the application can be maintained on a server. For a company with offices or planners dispersed over several locations, it becomes easy to serve their planners the same functions as well as the same data if necessary.

Despite the complexity of planning problems in maritime transportation, there are only a few studies about DSS in maritime transportation available. Mansouri et al. (2015) summarize the reasons for this lack of studies with the industry’s conservative thinking and the attitude against the investment of resources. The same authors identify 12 relevant papers discussing DSS for maritime transportation. The underlying planning problem for most of these papers is the vessel scheduling problem. Other problems include liner shipping network design, risk assessment and the impact on the economy and the environment.

Fisher and Rosenwein (1989), Kim and Lee (1997) and Bausch et al. (1998) all present optimization models to solve the vessel scheduling problem. Their proposed DSSs are focused on the idea of user friendly interfaces and use visualization techniques like Gantt charts and geographical maps. Fagerholt (2004) proposes the DSS “TurboRouter”, which aims at an intuitive and interactive system. For example, the user is able to manually adjust computed schedules in TurboRouter. A web-based DSS for for ship scheduling is presented in Lam (2010), supporting the analysis of what-if scenarios on oil prices, cargo demands and other financial factors.

2.3. Methodology

2.3.1. Operations Research

It is generally accepted that the field of Operations Research (OR) was established during World War II (Hillier and Liebermann (2001)). The allocation of troops and resources in this war posed a complex problem for military planners. Therefore, they invited scientists to come up with solutions for these problems. These solutions were the first steps in the field of operations research. By making use of mathematical optimization, military planners had a science-based approach to solve their planning problems.

Even after the war ended in 1945, it still had its effects on the industry in Europe and other countries years later. The industrial problems were similar to those that were solved by the OR scientists during war time: Limited resources had to be allocated to the right places (Hillier and Liebermann (2001)). By making use of the developed OR methods in a civilian context, it was possible to solve complex planning problems.

Although many of the standard theories and techniques were developed until the 1950s, a strong impact on the development of OR was made by the evolution of the computer. In order to solve these complex problems, large quantities of computational power are needed. The access to sophisticated systems to solve these planning problems became easier in the last decades. Today, with modern cloud technologies as well as powerful personal computers or smartphones, everybody can have access to tools using OR methods. Most of the time, this access is not obvious to the normal user as it is perfectly integrated in today's applications.

Mathematical Optimization Models

The basis for many OR applications are mathematical optimization models. These models are used to represent planning problems with their relevant characteristics. Depending on the nature of the mathematical expressions, different types of mathematical models can be identified. If all expressions (in constraints and the objective function) are linear expressions, the model is called a linear programming model (LP). Expressions 2.1 to 2.4 from Suhl and Mellouli (2006) show a general definition of such a linear programming model:

$$\text{minimize } z = \sum_{j=1}^{n} c_j x_j \tag{2.1}$$

s.t.

$$\sum_{j=1}^{n} a_{ij} x_j \leqslant b_i \qquad \forall i \in \{1, ..., m\} \tag{2.2}$$

$$l_j \leqslant x_j \leqslant u_j \qquad \forall j \in \{1, ..., n\} \tag{2.3}$$

$$x_j \in \mathbb{R} \tag{2.4}$$

The given model definition contains an objective function in Expression 2.1, constraints in Expression 2.2 and bounds in Expression 2.3. The objective function is used to define the optimization goal of the model. In this case, the definition shows a minimization goal, but it can be easily converted to a maximization model. The mathematical expression for the objective function (Expression 2.1) is built by using parameters (here c_j) as well as decision variables (here x_j). Parameters are fixed values that cannot be changed during the optimization process. In contrast to this, decision variables have to be changed during the optimization process to find the best possible value for such a variable. The concrete values of all decision variables in the model make up the final solution of an optimization problem. Based on the number of decision variables there is a large pool of possible combinations to choose from, called the search space (Blum and Roli (2003)). Usually, not every combination of decision variables is possible and makes practical sense. Therefore, constraints (Expression 2.2) and bounds (Expression 2.3) are formulated to restrict the search space to only those solutions that are feasible. Expression 2.4 shows that the decision variable in this definition belongs to the domain of real numbers.

If the domain of every decision variable is defined as integer numbers, the model becomes an integer programming model (IP). This is often the case for decisions, where it is not possible to split something into the fractions of a single entity. Furthermore, it is also possible to have a mixture of real number decision variables and integer variables, resulting in a mixed integer programming model (MIP). As described by Papadimitriou and Steiglitz (1982), both IPs and MIPs have a higher mathematical complexity than an LP. There is no known algorithm at the moment that solves IPs and MIPs in polynomial time, making it impractical to try to solve them optimally in a practical application in many cases. Therefore, sophisticated solution techniques are needed to solve these problems in a more practical time frame.

Another difficulty arises from any non-linearities that may be added to a model. In order to use standard techniques to solve these models, the non-linearities need to be linearized. This makes it possible to achieve a linear programming model, but also has the disadvantage that there are precision losses in the solution. This is an important factor in modeling liner shipping problems, as the bunker consumption varies approximately cubically with the speed of a vessel and proportionally to factors such as load and trim (Brouer et al. (2014)). Due to this, multiple linearization techniques like piecewise linearization (Guericke and Tierney (2015), Reinhardt et al. (2016)) or second-order cone programming (Du et al. (2015)) have found their way into liner shipping optimization models.

2.3.2. Metaheuristics

As listed by Blum and Roli (2003), there are two types of algorithms for mathematical optimization problems: complete or approximate algorithms. When a complete algorithm is used, it can be guaranteed that the optimal solution of the underlying problem will be found at some point. Approximate algorithms sacrifice the guarantee for optimal solutions in order to provide better runtimes in solving the problem. As described in Section 2.3.1, the complexity of IP and MIP models often makes it impossible to find the solution in a practical time frame. Therefore, approximate algorithms are used to solve these models. This is a critical issue during the development of a computer-based solution procedure for an optimization problem, as it influences the practicality of the overall decision support process.

According to Blum and Roli (2003), approximate algorithms can be categorized into constructive methods and local search methods. The authors explain that constructive methods generate a solution from scratch, while local search methods try to modify an existing solution in order to find better ones. Metaheuristics can be seen as a framework to guide constructive methods and local search methods. They combine different strategies to explore the search space (Laporte and Osman (1996)), where the search space is defined as the set of all feasible solutions (Blum and Roli (2003)). The strategies to explore the search space can include complex learning processes to efficiently converge to better solutions. Furthermore, they also include techniques to avoid local optima. Local optima are the best solutions of a specific search space region, but can be far from the global optimum. Many metaheuristic concepts are based on analogies stemming from fields like mechanical engineering or nature.

The following parts of this section consist of concepts and characteristics of metaheuristics that have been used for the research presented in this thesis. For a deeper understanding of these selected concepts or other concepts not listed here, we refer to Blum and Roli (2003) or Gendreau and Potvin (2010).

Simulated Annealing

The Simulated Annealing algorithm (SA) was first presented by Kirkpatrick et al. (1983). It has its origins in statistical mechanics and is based on the thermodynamic process of cooling down matter (e. g. a metal) to the point where it forms a crystalline structure. By making use of the Metropolis acceptance criterion, the algorithm decides whether a solution with a worse quality than the current solution is accepted or not (Gendreau and Potvin (2010)). Due to the cooling function, the probability to accept worse solutions decreases during the execution of this algorithm. As described by Blum and Roli (2003), there are different ways to define a cooling regime for an SA and the selection of a regime is a crucial decision when this algorithm is used to solve an optimization problem. It plays an important role in regard of computational runtime and the extent of the search space exploration.

Tabu Search

In contrast to SA, where every iteration of the algorithm is considered independently of previous iterations, Tabu Search (TS) follows a different strategy. TS uses local search techniques to explore the search space and combines that with a memory, storing information about the previous moves through the search space (Glover (1991)). At each iteration, different neighborhood operations can be used to transform the current solution to the next solution. A tabu list stores specific information about the exploration in order to prevent “revisiting” already found solutions. This makes it easier to avoid getting trapped in local optima in the search space. What kind of information is stored depends on the configuration of the algorithm. It is possible to store complete solutions or only some information about the last transformations. It is also important to define the possibility to remove elements from the tabu list, called an aspiration criterion. Gendreau and Potvin (2010) gives some details on how a tabu list can be structured, what kind of information can be stored on a tabu list and how an aspiration criterion can be formulated.

Genetic Algorithms

Genetic algorithms (GA) are based on the analogy of recombining individuals of a population. Biological terms from genetics and evolutionary theory have found their way into this metaheuristic concept. As presented by Holland (1975), the basic idea of this concept is to have a population of individuals, where each individual has a certain set of chromosomes, representing the solution of the individual. By using an evaluation function to determine the quality of the solutions, specific individuals are selected in order to recombine them and produce offspring for the next generation of individuals. Such a recombination is called a crossover and there are thousands of techniques to perform such a crossover (De Jong (1975)). Generally speaking, a crossover takes the chromosomes of the parent individuals and selects particular elements from these chromosomes in order to form a new offspring. Mutations might change specific genes of randomly selected individuals to another value. This process leads to a new generation of individuals where the next iteration of crossover and mutation takes place. Individuals with better solutions have a higher chance to be selected for a crossover, leading to the fact that over the iterations of this algorithm the overall best solution improves until a stop criterion ends the GA.

2.3.3. Optimization projects

While there are some similarities to the course of an IT project, an OR project has some specific characteristics. Generally, such a project can be divided in six, partly overlapping, phases. These phases are described by Hillier and Liebermann (2001) as:

1. Problem definition and gathering of relevant data
2. Creating a mathematical model for the problem
3. Develop a computer-based solution procedure
4. Testing the model
5. Final implementation & deployment

Problem definition & data. Phase 1 is the fundamental basis for an OR project. Together with the actual planners, the planning process is analyzed and domain specific requirements are gathered. Furthermore, the planning goals are fixed and what part of the process should be optimized by the optimization model. In addition, indicators are set up to measure the success of the model. It is also necessary to understand the IT architecture, to develop relevant interfaces to other systems. This is obligatory for the design of a DSS at a later point in such a project.

Modeling the problem. With the information of the actual planners, a mathematical optimization model can be formulated. This model should aim to represent the planning question at hand as good as possible. For this, the elements of a mathematical optimization model should be carefully designed and described. It might be necessary to involve the actual planners at this stage to discuss the model together with them. It should be noted though that planners are not always familiar with the mathematical expressions of a model. Therefore, such a discussion should be carefully prepared.

Computer-based solution procedure. After the mathematical model has been developed, it needs to be implemented such that the planning problem can be solved programmatically. This phase is crucial for the following steps of an OR project. Without an implementation of a solution procedure, it is not possible to test the model or to prepare a future deployment. One important consideration should be the time frame in which an optimal or near-optimal solution should be found. Depending on the complexity of the problem, it can take up to several hours, days or even longer to find the optimal solution (see Section 2.3.1). Often this is not practical in a real-world setting, where decisions might need to be taken several times a day. It can also be satisfactory to find a good heuristic solution to present some kind of guidance on how to solve the underlying problem.

Testing the model. In Phase 4, the actual planners have to be consulted again. Now as the model is implemented and first results are generated, these results need to be evaluated. The domain experts should check the results of the model and verify the correctness of the solution. In case an error is found in the solutions, the source of the error should be identified. Depending on the reason for the error, this starts another iteration of modeling and testing. For the tests of the model, actual real-world data should be used in order to have significant results that are worth discussing.

Final implementation & deployment. When the tests are completed and the solutions have been accepted as reasonable, the final phase of an OR project starts. Depending on the nature of the project, the goal and the current technical infrastructure of the customer, the implemented model needs to be made ready, such that it can be used in the planning tasks of the customer. A DSS tool needs to be implemented, such that the user will be able to use the developed model and its solutions. This phase wraps up the implementation before the system goes live on the customer side.

3. Overview of research papers

This section provides a summary for each paper presented in Part II. These papers have addressed multiple project phases as discussed in Section 2.3.3. An overview of the addressed phases is given in Table 3.1. It should be noted that the final implementation and deployment of a decision support system has not been a part of this research.

	Müller and Tierney (2017)	Müller et al. (2017)	Tierney et al. (2018)	Müller (2018)
Problem definition	(✓)	✓	✓	(✓)
Modeling	✓	✓	✓	(✓)
Computer-based solution procedure	✓	✓	✓	✓
Testing the model	✓	✓	✓	✓
Final implementation & deployment	-	-	-	-

Table 3.1.: Phases that have been addressed in the research of this thesis.

Chapter 4: Decision support and data visualization for liner shipping fleet repositioning. This paper contains an extension of the optimization model for the liner shipping fleet repositioning problem (LSFRP) presented by Tierney et al. (2014). It proposes a web-based decision support system, which contains an extended version of a state-of-the-art simulated annealing solution approach. Furthermore, it contains a study on how to display information such as cargo flows and interactions between vessels. It was published in print in 2017 by the journal Information Technology and Management.

Müller, D., Tierney, K.: Decision support and data visualization for liner shipping fleet repositioning. *Information Technology and Management*, 18(3):203-221, 2017.

Conference presentations:

- International Conference on Operations Research, September 2015, Vienna
- Multikonferenz Wirtschaftsinformatik, March 2016, Ilmenau (as a peer-reviewed extended abstract)

Contributions:

- Extension of a state-of-the-art optimization model of LSFRP to increase interactive decisions of the planner

- Proposition of a web-based DSS for the LSFRP
- Introduction of a new visualization technique for liner services and evaluation with other visualization techniques

Chapter 5: Integrating Fleet Deployment into the Liner Shipping Cargo Allocation Problem. Due to the fact that liner carriers perform modifications on their networks on a regular basis, there is a need for advanced tools to analyze the implications of such a change. In this paper, an extension of a state-of-the-art mixed integer programming model for the liner shipping cargo allocation problem (LSCAP) is proposed, which incorporates the optimization of vessel count and vessel classes for each service. By using a computational analysis it is shown that such an integration can increase the profitability of a carrier's network. The paper was published in print in the proceedings of the International Conference on Computational Logistics in 2017.

Müller, D., Guericke, S., Tierney, K.: Integrating fleet deployment into the liner shipping cargo allocation problem. In *International Conference on Computational Logistics*, pages 306-320. Springer, 2017.

Conference presentations:

- International Conference on Computational Logistics, September 2017, Southampton

Contributions:

- Extension of a state-of-the-art optimization model for the LSCAP
- Integration of the LSCAP and the fleet deployment problem

Chapter 6: Liner Shipping Single Service Design Problem with Arrival Time Service Levels. This paper introduces three mathematical optimization models for designing liner shipping services that guarantee the punctual arrival of vessels at a specified service level. For this, an empirical analysis of vessel travel times of a real liner shipping network has been performed. A simulation procedure based on real-world data is used to demonstrate the effectiveness of the presented approach. In July 2018, the journal Flexibel Services and Manufacturing accepted it for publication.

Tierney, K., Ehmke, J.F., Campbell, A.M., Müller, D.: Liner Shipping Single Service Design Problem with Arrival Time Service Levels. *Flexible Services and Manufacturing Journal*, 2018.

Contributions:

- Three mathematical models with increasing complexity to optimize the liner shipping single service design problem, including arrival time service levels
- A simulation model to evaluate the performance of the optimization models
- Empirical analysis of real-world travel times of vessels on liner services
- Proposition of a distribution for delays of vessels on liner services

Chapter 7: A Biased Random-Key Genetic Algorithm for the Liner Shipping Fleet Repositioning Problem. The state-of-the-art algorithm to solve the liner shipping fleet repositioning problem is a hybrid of the heuristics Tabu Search and Simulated Annealing presented by Becker and Tierney (2015). Although the algorithm is able to solve the majority of the presented instances, there is still room for improvement considering the gap to the optimal solution. Therefore, this paper proposes a biased random-key genetic algorithm (BRKGA) to solve the liner shipping fleet repositioning problem. The BRKGA is extended with a hill climbing algorithm to solve instances of various sizes. This paper is published as a working paper at the DS&OR Lab (Working Paper no. 1801) in July 2018.

Müller, D.: A Biased Random-Key Genetic Algorithm for the Liner Shipping Fleet Repositioning Problem. DS&OR Lab Working Paper, no. 1801.

Conference presentations:

- International Conference on Operations Research, September 2017, Berlin

Contributions:

- Proposition of a random-key decoder for the LSFRP
- Proposition of four random-key definitions for the LSFRP
- Computational evaluation the four definitions
- Proposition of a BRKGA combined with hill climbing

Bibliography

Agarwal, R., Ergun, Ö.: Ship Scheduling and Network Design for Cargo Routing in Liner Shipping. *Transportation Science*, 42(2):175–196, 2008.

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

Becker, M., Tierney, K.: A hybrid reactive tabu search for liner shipping fleet repositioning. In: F. Corman, S. Voß, R. Negenborn (eds.) Computational Logistics, *Lecture Notes in Computer Science*, vol. 9335, pp. 123–138. Springer International Publishing (2015).

Blum, C., Roli, A.: Metaheuristics in Combinatorial Optimization - Overview and Conceptual Comparison. *ACM Computing Surveys*, 35(3):268–308, 2003.

Brouer, B. D., Dirksen, J., Pisinger, D., Plum, C. E. M., Vaaben, B.: The Vessel Schedule Recovery Problem (VSRP) – A MIP model for handling disruptions in liner shipping. *European Journal of Operational Research*, 224(2):362–374, 2013.

Brouer, B. D., Alvarez, J. F., Plum, C. E. M., Pisinger, D., Sigurd, M. M.: A Base Integer Programming Model and Benchmark Suite for Liner-Shipping Network Design. *Transportation Science*, 48(2):281–312, 2014.

Brouer, B. D., Karsten, C. V., Pisinger, D.: Optimization in Liner Shipping. *4OR*, 15(1):1–35, 2017.

Christiansen, M., Fagerholt, K., Nygreen, B., Ronen, D. Chapter 4 Maritime Transportation. In *Transportation*, volume 14 of *Handbooks in Operations Research and Management Science*, pages 189–284. Elsevier, 2007.

Christiansen, M., Fagerholt, K., Nygreen, B., Ronen, D.: Ship routing and scheduling in the new millennium. *European Journal of Operational Research*, 228(3):467–483, 2013.

De Jong, K. A.: Analysis of the Behavior of a Class of Genetic Adaptive Systems. University of Michigan, 1975.

Du, Y., Chen, Q., Lam, J. S. L., Xu, Y., Cao, J. X.: Modeling the Impacts of Tides and the Virtual Arrival Policy in Berth Allocation. *Transportation Science*, 49(4): 939–956, 2015.

Fagerholt, K.: A computer-based decision support system for vessel fleet scheduling—experience and future research. *Decision Support Systems*, 37(1):35–47, 2004.

Fisher, M. L., Rosenwein, M. B.: An interactive optimization system for bulk-cargo ship scheduling. *Naval Research Logistics*, (36):27–42, 1989.

Gelareh, S., Meng, Q.: A novel modeling approach for the fleet deployment problem within a short-term planning horizon. *Transportation Research Part E: Logistics and Transportation Review*, 46(1):76–89, 2010.

Gendreau, M., Potvin, J. Y.: *Handbook of metaheuristics*, volume 146 of *International Series in Operations Research & Management Science.* Springer, 2010.

Glover, F.: *Tabu search for nonlinear and parametric optimization*, 1991.

Guericke, S., Tierney, K.: Liner shipping cargo allocation with service levels and speed optimization. *Transportation Research Part E: Logistics and Transportation Review*, 84:40–60, 2015.

Hillier, F. S., Lieberman, G. J.: *Introduction to operations research.* McGraw-Hill series in industrial engineering and management science. McGraw-Hill, 2001.

Holland, J. H.: *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence.* Univ. of Michigan Press, Ann Arbor, 1975.

Kirkpatrick, S., Gelatt, C. D., Vecchi, M. P.: Optimization by Simulated Annealing. *Science*, (220):671–680, 1983.

Lam, J. S. L.: An integrated approach for port selection, ship scheduling and financial analysis. *NETNOMICS: Economic Research and Electronic Networking*, 11(1):33–46, 2010.

Laporte, G., Osman, I. H.: *Metaheuristics in combinatorial optimization*, volume 63 of *Annals of operations research.* Baltzer, Amsterdam, 1996.

Mansouri, S. A., Lee, H., Aluko, O.: Multi-objective decision support to enhance environmental sustainability in maritime shipping: A review and future directions. *Transportation Research Part E: Logistics and Transportation Review*, 78: 3–18, 2015.

Meng, Q., Wang, S., Andersson, H., Thun, K.: Containership Routing and Scheduling in Liner Shipping: Overview and Future Research Directions. *Transportation Science*, 48(2):265–280, 2014.

Meyer, J., Stahlbock, R., Voß, S.: Slow Steaming in Container Shipping. In *2012 45th Hawaii International Conference on System Sciences (HICSS)*, pages 1306–1314, 2012.

Müller, D.: A Biased Random-Key Genetic Algorithm for the Liner Shipping Fleet Repositioning Problem. DS&OR Lab Working Paper, no. 1801, 2018.

Müller, D., Tierney, K.: Decision support and data visualization for liner shipping fleet repositioning. *Information Technology and Management*, 18(3):203–221, 2017.

Müller, D., Guericke, S., Tierney, K.: Integrating fleet deployment into the liner shipping cargo allocation problem. In *International Conference on Computational Logistics*, pages 306–320. Springer, 2017.

Pacino, D.: *Fast Generation of Container Vessel Stowage Plans: using mixed integer programming for optimal master planning and constraint based local search for slot planning.* IT-Universitetet i København, 2012.

Papadimitriou, C. H., Steiglitz, K.: *Combinatorial optimization: Algorithms and complexity.* Prentice-Hall, Englewood Cliffs, NJ, 1982.

Reinhardt, L. B., Plum, C. E. M., Pisinger, D., Sigurd, M. M., Vial, G. T. P.: The liner shipping berth scheduling problem with transit times. *Transportation Research Part E: Logistics and Transportation Review*, 86:116–128, 2016.

Sauter, V. L.: *Decision support systems for business intelligence.* Wiley, 2010.

Shim, J. P., Warkentin, M., Courtney, J. F., Power, D. J., Sharda, R., Carlsson, C.: Past, present, and future of decision support technology. *Decision Support Systems*, 33(2):111–126, 2002.

Kim, S. H., Lee, K. K.: An optimization-based decision support system for ship scheduling. *Computers & Industrial Engineering*, 33(3-4):689–692, 1997.

Stopford, M.: *Maritime economics.* Routledge, 2009.

Suhl, L., Mellouli, T.: *Optimierungssysteme: Modelle, Verfahren, Software, Anwendungen.* Springer, 2006.

Tierney, K., Áskelsdóttir, B., Jensen, R. M., Pisinger, D.: Solving the liner shipping fleet repositioning problem with cargo flows. Transportation Science 49, 652 – 674 (2014)

Tierney, K., Ehmke, J. F., Campbell, A. M., Müller, D.: Liner Shipping Single Service Design Problem with Arrival Time Service Levels. Flexible Services and Manufacturing, 2018.

United Nations Conference on Trade and Development: Review of maritime transport. United Nations, 2016.

Vahrenkamp, R., Kotzab, H.: *Logistik: Management und Strategien*. Walter de Gruyter, 2012.

Part II.

Research papers

4. Data Visualization and Decision Support for the Fleet Repositioning Problem

Daniel Müller* Kevin Tierney*

*University of Paderborn
Warburger Straße 100, 33098 Paderborn, Germany
mueller@dsor.de

Liner carriers move vessels between routes in their networks several times a year in a process called fleet repositioning. There are currently no decision support systems to allow repositioning coordinators to take advantage of recent algorithmic advances in creating repositioning plans. Furthermore, no study has addressed how to visualize repositioning plans and liner shipping services in an accessible manner. Displaying information such as cargo flows and interactions between vessels is a complex task due to the overlap of container demands and long time scales. To this end, we propose a web-based decision support system designed specifically for liner shipping fleet repositioning that integrates an extended version of a state-of-the-art simulated annealing solution approach. Our system supports users in evaluating different strategic settings and scenarios, allowing liner carriers to save money through better fleet utilization and cargo throughput, as well as reduce their environmental impact by using less fuel.

Keywords: liner shipping, fleet repositioning, decision support system, visualization

4.1. Introduction

Seaborne trade plays a major role in the world economy and is responsible for the transportation of about 9.6 billion tons of goods per year UNCTAD (2014). In particular, the transport of containerized goods grew by 4.6 percent in 2013 and the total volume reached 1.5 billion tons or 160 million 20-foot equivalent units (TEU). Except for a short pullback during the financial crisis of 2008, the liner trade has been growing for decades, meaning that more and more goods are transported by sea UNCTAD (2014). To participate in the seaborne trade, liner carriers have to make large investments in vessels and equipment and are confronted with high daily operating costs Fagerholt (2004). Sophisticated planning of a carrier's operations is a necessity for competing in this growing market. However, throughout the liner shipping industry the planning of networks, including route construction and vessel movements, is still mostly performed manually. The emergence of new mathematical models and algorithms offers liner carriers the opportunity to improve their processes with decision support systems (DSS).

Containerized goods are transported on cyclical routes called services. Liner shipping companies operate a number of these services in order to connect different trade regions within their network. Figure 4.1 shows an example of a subset of a service network of an industrial collaborator Tierney (2015). In this example, three services serve ports in Asia, North America and South America. While the services "Chennai Express" and "Intra-WCSA" each serve a single trade region, the service "Asia-CA3" connects the regions in Asia and America.

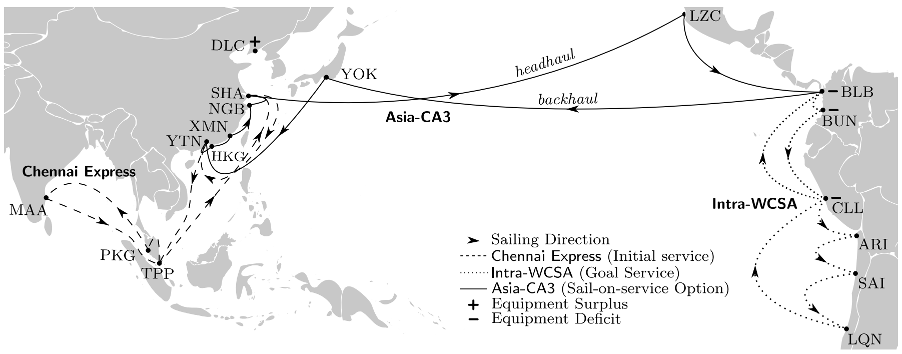


Figure 4.1.: An example of a service network from a case study with an industrial collaborator, from Tierney et al. (2015).

Services are generally operated with a certain periodicity (usually weekly or biweekly) such that ports of a service are visited at a fixed time each period. Furthermore, services consist of multiple slots, where each slot is associated with a specific vessel and a fixed schedule for the port calls. Depending on the periodicity of the service, there are fixed time intervals between the schedules of the slots. In the example of Figure 4.1, this could mean that a vessel is arriving at the port of Yokohama (YOK) while another vessel is in the middle of the Pacific Ocean, and a third vessel would be at the port of Balboa (BLB). In other words, several vessels are all sailing on the service at the same time to maintain the desired frequency of the service.

Liner carriers must adjust their service networks on a regular basis due to trends in the world economy and seasonal variations in cargo volumes. Carriers add, remove and modify services to adjust to these changes. To carry out these modifications to the network, vessels must be moved, or *repositioned*, to other services. Repositioning coordinators generate an extensive plan for each repositioning of one or more vessels with the goal of minimizing the costs for moving the vessel(s) while maximizing the revenue earned from transporting customer cargo. This problem is referred to as the liner shipping fleet repositioning problem (LSFRP) Tierney (2015).

Despite algorithmic advances in recent years to support repositioning coordinators, these advances are not yet accessible for non-researchers. To the best of our knowledge, there is no DSS for the LSFRP in use at any carrier. Repositioning coordinators are therefore still using spreadsheets (or other manual mechanisms) to create their plans. The process of creating a repositioning plan can therefore take up to a couple of days Tierney (2015), which is significantly more than the several minutes required by modern heuristics Tierney et al. (2015). Although systems exist for other problems in the area of maritime transportation Mansouri et al. (2015), they lack the specific requirements of the repositioning problem and thus only offer limited guidance in creating a DSS for the LSFRP.

A DSS that combines access to algorithmic approaches for the LSFRP as well as a comprehensible and usable approach to display relevant data would create enormous benefits for liner shipping companies. As fleet repositioning coordinators have significant expert knowledge, they should be able to influence the behavior of such a system to an extent where they can make ad-hoc adjustments of models, constraints or other relevant aspects of the system.

In this paper, we present an interactive DSS for the LSFRP. We discuss various visualization techniques to display relevant data of the liner shipping fleet repositioning and general aspects of liner shipping. The visualization strategies we present have a wide application within decision support systems in the maritime sector beyond fleet repositioning. The main contributions of this paper can therefore be summarized as follows:

1. A web-based DSS for the LSFRP
2. An extension of the state-of-the-art simulated annealing algorithm for the LSFRP to incorporate interactive decisions of the planner
3. Several visualization techniques to represent liner shipping services

This paper is structured as follows. Section 2 contains an overview of the literature about the LSFRP followed by a description of liner shipping fleet repositioning in Section 3. In Section 4, we present our system and discuss the interactivity of the system as well as options for visualizing the fleet repositioning problem. Finally, in Section 5, we conclude this paper and give an overview of our future plans for this research topic.

4.2. Literature Review

4.2.1. Decision Support Systems in Maritime Transportation

Despite the fact that the shipping industry plays a major role in the world economy, the number of DSSs in the literature for this industry is surprisingly low Fagerholt (2004), Mansouri et al. (2015). In the area of bulk shipping, Fisher and Rosenwein (1989) describe a system that finds optimal fleet schedules for bulk cargo pickup and delivery with a dual algorithm. This system uses Gantt-charts, showing the individual schedules of the vessels, and maps for the itinerary of a selected vessel. Another system by Bausch et al. (1998) for the transport scheduling of bulk products uses spreadsheets and also Gantt diagrams to display and manage the proposed schedules. This application uses a simulation to generate vessel schedules, from which a linear set partitioning model selects the best alternatives. In Kim and Lee (1997), the focus also lies on vessel scheduling in the bulk industry. In addition to a spreadsheet view, this system uses a world map to display optimal schedules.

One of the more recent DSSs is TurboRouter, presented in Fagerholt (2004) and Fagerholt and Lindstad (2007). This system focuses on scheduling and routing decisions of liner shipping companies. TurboRouter supports manual planning and optimization algorithms for automatic planning. Fagerholt reports a significant improvement in Fagerholt and Lindstad (2007) of the quality of the schedules and the time required for the planning process of the companies using their system. In Fagerholt et al. (2009), the capabilities of the TurboRouter system are extended to solve fleet deployment problems. A case study with industrial partners is presented, where the authors show that the system produces improvements compared to the previous manual planning.

The “MARISA” system, presented in Balmat et al. (2009) and Balmat et al. (2011), incorporates a fuzzy approach to define risk factors for individual ships. The approach considers static aspects of the vessels as well as dynamic aspects like meteorological conditions. The MARISA system uses simulation techniques to calculate individual risk factors for each vessel using this data. The tool uses a world map to present the simulation results.

4.2.2. Design Studies

When a decision support system is created, it is important to use representations of data that are relevant for domain experts and possible users of the system. Due to the fact that the area of design studies considers visualization research from a problem-driven perspective, it leads to systematic approaches on how to create visualizations for a DSS.

In an extensive literature review, Sedlmair et al. (2012) search for methodologies that can be used in the field of design studies. Using these sources, a general framework is created for the realization of a design study. This framework consists of nine stages, which contain the work before a design study, the core work of the design study and the analytical reasoning after a design study. In addition to this general framework, a list of 32 pitfalls is presented that can occur when realizing a design study.

Brehmer and Munzner (2013) argue that the core of visualizations are specific tasks that can be defined by three aspects: why is the task performed, how is the task performed and what are the inputs and outputs. They present a typology of visualization tasks that contains a set of generalized tasks for each of these aspects, so that new visualization approaches can be classified according to this framework.

Only a few publications in the field of design studies focus specifically on tasks related to maritime transportation. Considering situational awareness in the maritime area, Lavigne et al. (2011) identify three possible opportunities where new visualizations could improve the status quo. In addition to the identification of these opportunities, concrete designs are proposed for each task. By considering the proposed systematic approaches, it is possible to analyze the planning processes of a problem and create supporting visualizations for a DSS.

4.2.3. Liner Shipping Fleet Repositioning

Similar to the small amount of literature about DSSs in liner shipping, there are only a few publications that focus on the LSFRP. Even in their extensive list of relevant publications in the field of liner shipping, Christiansen et al. (2013) do not refer to the LSFRP, showing that this area of research has been explored only recently. The problem itself and the first mathematical approaches for solving the LSFRP were first introduced in Tierney et al. (2012). Furthermore, Tierney et al. (2015) develops a simulated annealing algorithm with cargo flows that scales to large, real-world problems. The approach finds higher profits compared to a reference scenario from industry. The runtime of this algorithm is small enough such that the authors claim it is suitable for a DSS, but do not provide such a system.

There is some significant work for other problems in the field of liner shipping like the network design problem (NDP), as described in Agarwal and Ergun (2008) and Brouer et al. (2014), and the vessel schedule recovery problem (VSRP), presented in Brouer et al. (2013). The goal of the NDP is to define profitable services for a set of demands, ports and a given fleet. The decisions of this problem are on a strategic level and do not handle how vessels should be repositioned to generated services. In the VSRP, the focus lies on recovering from disruptions due to weather conditions, port congestion or other reasons. For the recovery, an action is chosen that might increase costs and have an impact on cargo and customers. The set of activities available to coordinators in the VSRP differs from the LSFRP, as does the timescale of the problem.

In contrast to liner shipping, tramp shipping does not require services and fixed schedules as tramp shipping companies engage in contracts for specific amounts of cargo Brønmo et al. (2007), Korsvik et al. (2011). Therefore, it is not possible to transfer problem-specific aspects from tramp shipping to the LSFRP even though the objectives of tramp shipping problems are similar.

4.3. Liner Shipping Fleet Repositioning

4.3.1. Problem Description

The liner shipping industry is a competitive and dynamic environment in which companies adjust their networks multiple times throughout the year in order to survive. By adjusting their networks, these companies adapt to seasonal variations in cargo demand and other economic trends. Adjustments of the service network consist of adding new services, removing services or modifying existing services. Whenever these changes occur, vessels must be reassigned between the different services of the company, thus these vessels have to be *repositioned* from one service to another service.

Each repositioning of a vessel is associated with costs due to lost revenue (from cargo flow disruptions) and bunker fuel consumption. This causes further costs in the context of the repositioning of a vessel. Therefore, the goal of liner shipping companies is to minimize these costs by using profit generating activities during the repositioning. Profit can be generated by carrying customers' cargo and moving empty containers in the network. In order to avoid cargo flow disruptions, it is important for models to take these cargo flows into account.

A key cost-saving activity is *slow steaming*, in which a vessel sails below its design speed Meyer et al. (2012). Since the fuel consumption of vessels is roughly cubic according to the speed the vessel travels, slowing down can save considerable amounts of money. Slow steaming has become a standard practice among the world's liner carriers. Another option is to use so-called sail-on-service opportunities to sail between the original service and the goal service. Here, the repositioning vessel replaces another vessel (the *on-service vessel*) in a specified service. The on-service vessel can then be laid up or chartered to another company. These activities as well as the whole repositioning process have to be planned within a horizon bounded by the earliest time the vessel may stop sailing according to its original schedule and the time it must begin operations on its new service. The time when the repositioning process starts is called the *phase-out* and the time it ends the repositioning is called the *phase-in*.

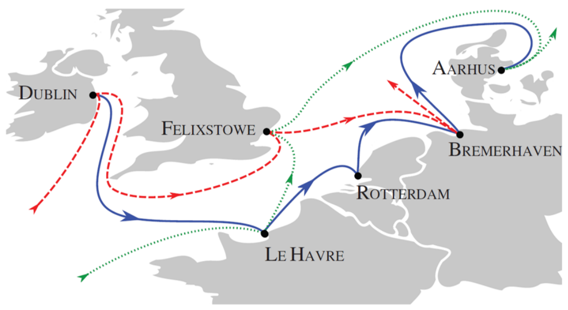


Figure 4.2.: A repositioning (blue) from an original service (dashed red) to a goal service (dotted green).

Figure 4.2 shows an example of a repositioning. In this figure, a vessel repositioned from a phase-out service (dashed red line) to a phase-in service (dotted green line). The repositioning path is represented by a blue line. The figure shows that the phase-out happens in the port of Dublin (DUB). After the phase-out, the vessel stops for port calls in the ports of Le Havre (LEH), Rotterdam (RTM) and Bremerhaven (BRV) before it arrives in Aarhus (AAR). Here, the vessel phases in in the goal service. The port call in Le Havre also belongs to the goal service, but the repositioning coordinator decides that the final phase-in should happen in Aarhus. This can be associated with the schedule of the goal service or with cargo flow restrictions. In addition, the figure shows that repositioning coordinators are able to *induce* or *omit* (add or remove) ports during the repositioning. Rotterdam is neither called by the original service nor the goal service, so the repositioning coordinator added this port to the repositioning path. On the other hand, the port call of Felixstowe (FXT) has been omitted although it can be found in the original and the goal services.

To solve the LSFRP, the problem is modeled as a graph Tierney (2015), where nodes represent a visit of a vessel at a particular port at a specified time and arcs are the allowed sailings between these visits. The graph includes all activities that are relevant for the specific repositioning period. A solution for a repositioning problem is then a path through this graph, which contains all activities that a vessel has to undertake during a repositioning. We provide a formal mathematical model of the problem in Appendix 4.6, and a more detailed description of the LSFRP can be found in Tierney (2015).

Phase-Out and Phase-In

Vessels that are to be repositioned have to stop their normal operations on their original service at a specific point in time to start the repositioning phase. This time is the phase-out time of the vessel. Until the phase-out, the vessel sails according to its schedule and visits ports as planned. After the phase-out, the vessel stops sailing according to its schedule and undertakes different activities to reach the goal service. When the repositioned vessel reaches the goal service, it has to phase into a slot of the goal service. In order to minimize disruptions of customer cargo, the repositioning coordinator sets a fixed time at which the latest phase-in can happen. After this time, the vessel must start its regular operations according to the schedule of the new service. The time between the earliest phase-out and the latest phase-in encompasses the time frame for the repositioning.

As shown in Figure 4.2, the repositioning coordinator has the ability to decide whether ports should be induced into the repositioning path that are not in the initial service or in the goal service. Additionally, the coordinator can omit ports from the initial service or the goal service in the repositioning if this would save money or allow more containers to be carried.

Several countries have laws regarding how cargo may be transported within its borders, called *cabotage restrictions*. These laws generally prohibit vessels registered in foreign countries from transporting domestic cargo, and must be obeyed by any repositioning. Due to the fact that each country has its own version of these restrictions, the consideration of all these restrictions is rather complicated. We therefore only address them in certain cases within the LSFRP where they can be simply modelled as omitting a port from the schedule of a vessel.

Sail-on-Service Opportunities

During the repositioning, the repositioning coordinator can decide to use sail-on-service opportunities (SoS) to save costs. A SoS refers to a different service than the initial service or the goal service. Therefore, there are at least two vessels involved in a SoS. There is the repositioning vessel, which uses the SoS to connect two parts of the service network and, in addition, there is the on-service vessel, which normally sails on the SoS, but can be replaced by a repositioning vessel. When using an SoS, the repositioning vessel uses the slot of the on-service vessel to sail temporarily on the SoS. As described by Tierney et al. (2015), this activity saves significant amounts of bunker fuel as there is only one vessel where there would have otherwise been two, as the on-service vessel can be laid up at a port or it can be leased out for other operations.

There are two options for starting an SoS. One option is to perform a transshipment between the repositioning vessel and the on-service vessel. For this transshipment, both vessels have to be at the same port, because all the cargo of the on-service vessel will be unloaded and transshipped to the repositioning vessel. The second option is to perform a parallel sailing for a specific amount of time. Here, both vessels sail "in tandem", with the on-service vessel only discharging its cargo, and the repositioning vessel only loading cargo. Although the parallel sailing causes double bunker costs and port fees, this procedure is sometimes less expensive than transshipping large amounts of containers between the vessels.

Cargo and Equipment

Liner carriers earn money from transporting containers from one port to another port. Cargo has a starting port and a destination port, as well as a latest delivery time at the destination port. Carriers also carry empty equipment, which generally refers to empty containers. Empty equipment is available in certain ports and is required in other ports. Unlike cargo, which has a specific origin and destination, empty equipment can be transported from any port with a surplus to any other port with a deficit. Due to the fact that the total number of containers throughout the liner shipping industry is very high, there is no need to specify exactly the size of the surplus or the deficit of empty equipment at the ports. The transportation of equipment generates revenue, as money is saved that would have been spent to transport the equipment with other options.

For their model of the LSFRP, Tierney et al. (2015) consider dry and refrigerated (*reefer*) cargo. Specialized containers with refrigerator units (or cooling hookups) are used to store cargo that needs to be cooled. These containers require electric outlets or cooling connections on board a vessel, meaning they must be placed in special slots that are correctly outfitted for this purpose. In contrast to this, dry cargo uses standard containers without any further requirements. Vessels provide different capacities for each of these cargo types. During the repositioning these capacities have to be considered as dry cargo containers and reefer cargo containers are not interchangeable for customers.

During the repositioning, it is possible to visit ports that are not on the initial service, the goal service or a SoS if they have empty equipment. As these ports are not included in any relevant schedules for the repositioning vessel and therefore do not have a fixed time frame for visits, a call in these ports is *flexible*. In contrast to this, all other port calls are *inflexible* as they have a fixed time for the visit.

4.3.2. Heuristic Solution Approaches

When LSFRP instances become too large, they become too difficult to solve with standard mathematical solvers, even when using advanced techniques like column generation. Therefore, Tierney et al. (2015) present a simulated annealing algorithm to solve the larger instances. In addition to this, a late acceptance hill climbing algorithm is presented in Tierney (2015), although its performance is limited in comparison to simulated annealing. A reactive tabu search and hybrid simulated annealing approach is introduced in Becker and Tierney (2015). We note that this approach can be inserted into our DSS without any additional work thanks to its modular design.

Simulated Annealing

Simulated annealing, introduced by Kirkpatrick et al. (1983), is a local search heuristic for solving combinatorial optimization problems. The algorithm changes its greediness over time according to a temperature parameter that is reduced in each iteration of the algorithm. The algorithm thus starts off accepting nearly any solution it sees, and slowly starts accepting worsening solutions only with some probability. Due to the decline of the temperature and the decline of the acceptance probability, the search converges to a local optimum. Simulated annealing is considered as a general tool for the field of operations research Eglese (1990) and has been used in diverse applications such as in the planning of power systems Chen and Ke (2004), scheduling problems Seçkiner and Kurt (2007) and routing problems Tavakkoli-Moghaddam et al. (2007). Further information about the application of the simulated annealing algorithm can be found in Dowsland and Thompson (2012).

The implementation of Tierney et al. (2015) uses a standard simulated annealing implementation that includes the Metropolis criterion for accepting solutions along with an exponential cooling strategy. Reheating and restart strategies are used to overcome local optima. During reheating, the temperature of the simulated annealing is increased to a factor of the initial temperature of the algorithm. Multiple reheatings can be performed before the algorithm is restarted from a new solution. Figure 4.3 summarizes this process. Candidate solutions are compared by evaluating the objective functions of each solution. Two different methods are proposed to evaluate the objective: a greedy approach and a linear program (LP). The greedy algorithm can only be used if there are no flexible arcs on the path of a vessel and if it is not possible to exceed the capacity of the vessel, even if all container demands on a path are carried. The greedy approach greatly decreases the computation time required for the objective function, as it avoids a call to an external solver. However, in more complex vessel paths it is necessary to use an LP to evaluate the candidate solution. In these situations, the flexible visitations have to be scheduled along the path, and profitable cargo must be identified. A penalty function is used to allow violations of certain constraints (such as multiple vessels using the same visitation) for infeasible solutions to allow the search to move between feasible regions of the search space.

There are several neighbourhood operators available for the LSFRP. These neighborhoods are composed of the addition of a specific port visit, the removal of a port visit, a swap of two port visits, a random path completion of a randomly selected vessel and demand destination completion, which adds a port to a path in order to fulfill demands. These neighborhoods are extensively evaluated and described in Tierney et al. (2015) and Becker and Tierney (2015).

Since the simulated annealing approach for the LSFRP uses a penalty function and allows infeasible solutions, integrating repositioning coordinator requests is relatively easy. These requests may make the problem infeasible, however the search quickly restores feasibility and finds good solutions. By creating these ad-hoc adjustments, the repositioning coordinator influences the structure of the graph representation of the problem and therefore also the search space. If, for example, the repositioning coordinator forbids a specific port, all visit nodes of this port will be removed from the graph. Furthermore, the coordinator might change enter and exit times for a specific visit. As the arcs of the graph are dependent on these times, they will also change. In addition to the adjustments of the data objects, the general settings for the simulated annealing algorithm can also be changed by the repositioning coordinator. Some examples for such changes are adjustments on the temperature reduction factor, the convergence temperature or the number of reheats before the solution is set back to the initial solution.

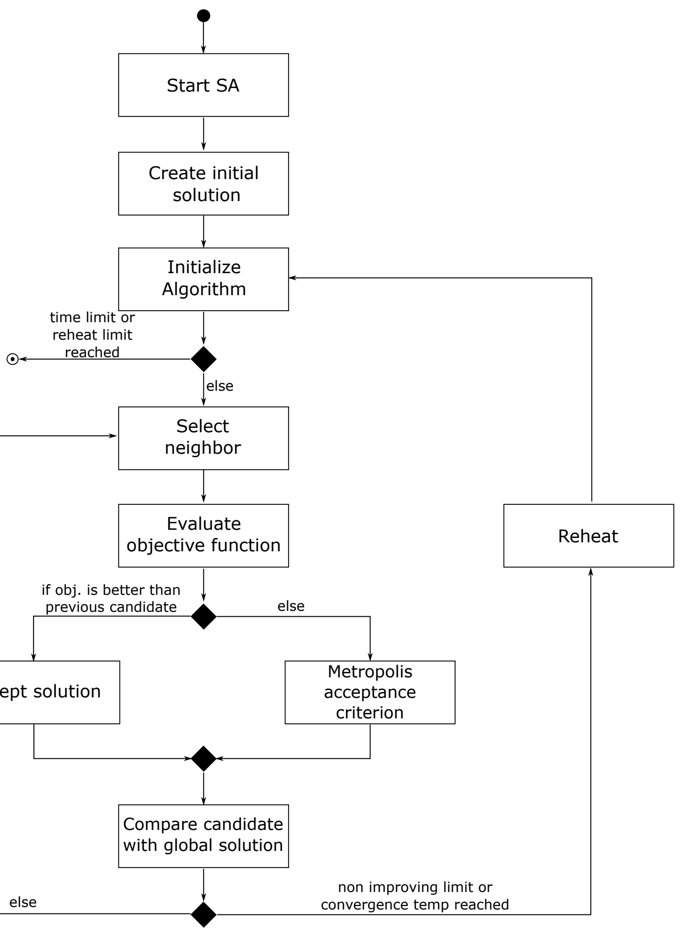

Start SA
Create initial solution
Initialize Algorithm
time limit or reheat limit reached
else
Select neighbor
Evaluate objective function
Reheat
if obj. is better than previous candidate
else
Accept solution
Metropolis acceptance criterion
Compare candidate with global solution
else
non improving limit or convergence temp reached

Figure 4.3.: Visual flow of the simulated annealing algorithm from Tierney et al. (2015)

4.4. Decision Support System

A decision support system for the LSFRP is greatly needed by the industry, as repositioning coordinators are still using spreadsheet software to manage their operations. This limits their ability to visualize and comprehend complicated repositioning procedures, even when such plans would save money. Due to the long time scales in liner shipping and the interactions between services and ports, fleet repositioning is a challenging problem for a decision support system. Nonetheless, recent algorithmic advances Tierney et al. (2015) have made it possible to create such a system for the LSFRP.

We propose a DSS using the simulated annealing approach in Tierney et al. (2015) with some small modifications to allow for fast resolving and interaction that supports

Table 4.1.: Requirements for a DSS for the fleet repositioning problem.

Data	Performance	Functions	Usability
Import/Export	Loading times	Planning	User Guidance
Storage	Fast optimization times	Optimization	Visualizations
Accessibility	System requirements	Reporting	Interactivity

the requirements of the industry. We list key requirements for the DSS in Table 4.1 from the perspective of the carrier, followed by a discussion of several visualization options, culminating with a recommendation for which one to use.

One of the key challenges in deploying a DSS at a company is integrating it within the organization's data management architecture. Today, most companies have an existing IT infrastructure, where processes and policies are already fixed. To ensure a DSS is used, it must be well integrated into the organization's environment, as it is very hard to integrate the company's data into the DSS. Depending on the position of the DSS in the processes of the company, the results of the DSS may be needed for further operations. Therefore, the system should be able to import and export relevant data in an existing data format or in a data format that can easily be converted to an existing format. The LSFRP in particular draws on many different data sources within a company and therefore this data exchange with other systems is an essential part of a DSS for the LSFRP. Furthermore, decisions about data storage and data accessibility in the DSS have to be considered. Usually, companies identify different user groups with different user rights. The system should be able to incorporate such an existing user hierarchy.

Additionally, the DSS needs to obey system requirements that are set by the company's infrastructure and the problem specific requirements. A DSS has to solve the LSFRP fast enough to allow the user to interact with the system. Furthermore, the system needs to have fast loading times to encourage user adoption, otherwise the user will think twice before starting a process or changing anything in the system.

Essential functions of a DSS can be categorized into planning, optimization and reporting functions. In a system for the LSFRP, the planning functions will be used by the repositioning coordinator to set up their environment, i.e. creating the master data of vessels, services and port calls. In addition to this, the coordinator has to set up important relations between this data. Using this basis, the repositioning coordinator needs to be able to run optimizations and get solutions for his problems. The optimization function contains the implementation of exact or heuristic methods that are able to solve the underlying problem. For a discussion of the solutions of an optimization problem, it is necessary to have reports about results, which our system provides through visualizations.

The usability of the system is also critically important. Without good usability, the system will not be accepted by the key users of a company. User guidance is important for the usability of a DSS, especially when existing processes rely on manual planning. Without an intuitive structure and guidance within the system, planners will have a hard time to adapt their processes to such a system. Furthermore, the DSS needs to be able to visualize the problem in a way that is intuitive and clear. These visualizations have to cover all relevant aspects of the problem. Ideally, the DSS provides the user several possible ways to do their planning. The visualizations and the system itself need to provide a degree of interactivity, allowing the user to use the system flexibly.

Our system uses a web-based approach, as this is operating system independent and easy to deploy. This makes it possible to quickly create repositioning prototypes that can be easily realized. Furthermore, user desktops may not be powerful enough to solve the LSFRP. Thus, a client-server architecture ensures that user's computers are not tasked with arduous computations. These advantages of web-based approaches are the basis for the increasing development of web-based systems (Wang et al. (2011), Bhargava et al. (2007)).

On the server side, the system uses the Python framework Django[1] to handle client requests. On the client-side, the system makes use of standard web technologies like HTML, JavaScript and CSS, which are supported by all modern browsers. In order to solve the LSFRP in this system, we integrate the simulated annealing approach as described by Tierney (2015). This algorithm is implemented in C++ to ensure fast solution times, so the system uses the Cython[2] library to wrap the C++ code and make it accessible within Django.

4.4.1. Interactivity

In most DSSs, the process of solving a problem consists of the following steps: instance setup, setting of solver parameters, starting the solver, waiting for the solution and finally interpreting the solution. Depending on the solution techniques used, this process is fixed and cannot be altered Piramuthu and Shaw (2009). It can often be the case that the answers provided by an optimization system are not exactly applicable to the real world, either due to outdated data or the level of abstraction of the model. In the worst case, the planner has to invest additional time to alter his setup and reoptimize his problem. These alterations in the setup create unnecessary disruptions in the workflow of the planner. Especially, for a sensitivity analysis of a given optimization result, this is time consuming.

Our system allows for interactivity during the solution process to avoid time consuming disruptions in the workflow of the repositioning coordinator. This is possible thanks to the local search heuristic used for finding solutions, as it can start from any valid or even only partially valid starting point and find a solution. Our system allows for the adjustment of port calls, demands and general information of a scenario.

[1]https://www.djangoproject.com/
[2]http://cython.org/

There are multiple views in the system that help the coordinator manipulate data and restart their optimization process.

The DSS uses a scenario based approach. These scenarios can be used to evaluate planned repositionings or strategic settings/“what if” scenarios set in the far future. Each scenario consists of its own set of vessels, port calls and services. Before an optimization, the coordinator has to prepare the relevant scenario data and the optimization settings (CPU time, memory limits, etc.). When the repositioning coordinator has finished his settings, he can start an optimization process using the front-end interface of the system (Figure 4.4). This request is sent to the server where relevant data is collected from the database. The data is then prepared such that it can be transferred to the simulated annealing module. After the simulated annealing algorithm has found a solution, this solution is stored in the database and a view with the result is created. This view is then presented to the repositioning coordinator. The results of the optimization are stored in the database so that the system can access these results later, for re-optimization or modifications of scenarios.

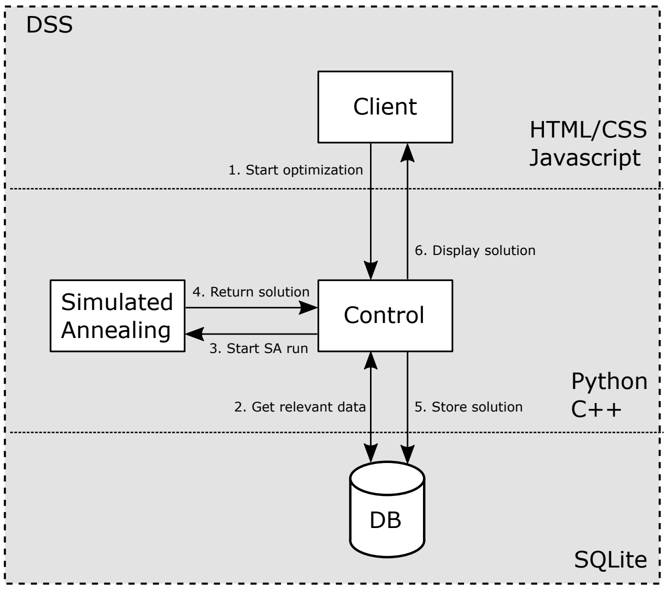


Figure 4.4.: Schema of the optimization process in the DSS.

Our prototype presents several options for ad-hoc adjustments/interactivity to the user, requiring some modifications of the simulated annealing algorithm of Tierney et al. (2015). A brief description of these options is given in Table 4.2. Further details of these options and their implications are described in the following subsections.

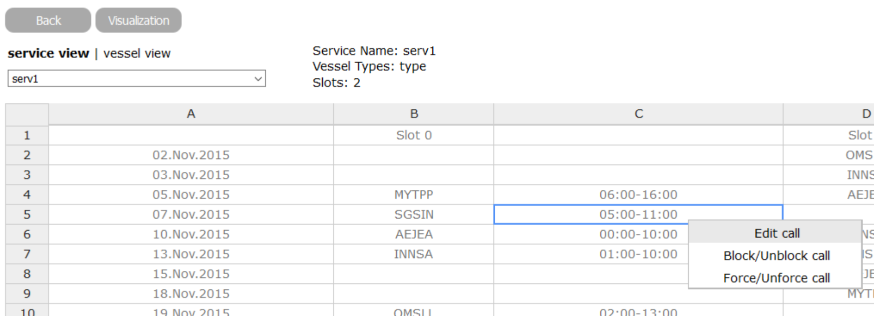
Table 4.2.: Ad-hoc adjustments and their implications on the SA approach.

Adjustment	Extension of the SA approach
Changing the time of port calls	-
Adding/Removing/Changing demands	-
Blocking/Requiring a specific port calls	Penalty function
Requiring a particular demand	Penalty function
Using fixed time windows for flexible port calls	Penalty function & MIP

Changing the time of port calls

Repositionings are planned several weeks or months before they are actually executed. Due to this lead time, information will likely change before the repositioning takes place, especially information about port calls in the services that are relevant for the repositioning coordinator. When information about the times of port calls change, the coordinator needs to evaluate his current plan in order to see if the plan is still valid, and that the plan remains of high quality.

To adjust the times of port calls in our prototype, the user is able to select a specific port call from a spreadsheet view and edit the arrival and departure time of this call (Fig. 4.5). As these adjustments only impact parameters of the model Tierney et al. (2015), the mathematical formulation and the simulated annealing algorithm remain unchanged.



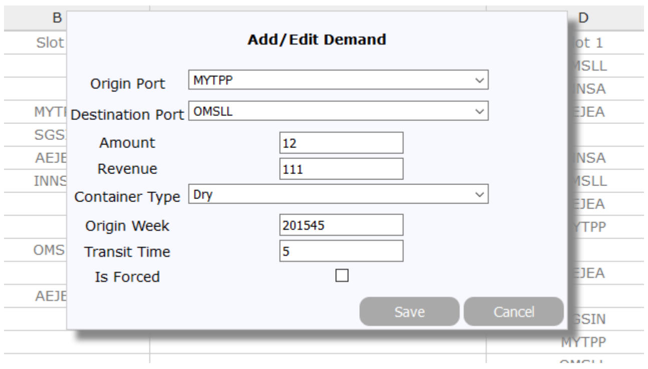
Service Name: serv1
Vessel Types: type
Slots: 2

	A	B	C	D
1		Slot 0		Slot
2	02.Nov.2015			OMS
3	03.Nov.2015			INNS
4	05.Nov.2015	MYTPP	06:00-16:00	AEJE
5	07.Nov.2015	SGSIN	05:00-11:00	
6	10.Nov.2015	AEJEA	00:00-10:00	NS
7	13.Nov.2015	INNSA	01:00-10:00	IS
8	15.Nov.2015			JE
9	18.Nov.2015			MYTI
10	19.Nov.2015	OMSLL	02:00-13:00	

Figure 4.5.: Screenshot of the detail view of scenarios from the prototype system.

Adding/Removing/Changing demands

As the world economy is a highly dynamic market, container demands change greatly over time. Our system prototype therefore enables the planner to adjust the demands in a scenario, such as allowing for spot cargo in a particular location. Demands can be modified such that their time frame is changed, as well as their amount, revenue and container type (Figure 4.6). These changes only affect the parameters of the



Add/Edit Demand

Origin Port	MYTPP
Destination Port	OMSLL
Amount	12
Revenue	111
Container Type	Dry
Origin Week	201545
Transit Time	5
Is Forced	☐

Figure 4.6.: Screenshot of the edit dialog for demands of a specific scenario from the DSS.

mathematical model. Therefore, there is again no need to change the formulation of the model or the implementation of the heuristic.

Blocking/Requiring specific port calls

Repositioning coordinators can have multiple reasons to block or require a specific port call, such as an expected port strike or other concerns that would cause a delay in the voyage of the repositioning vessel. In contrast to this, a repositioning coordinator may want to require a port call if they are expecting a higher demand at a port during the repositioning (e.g. the Christmas season). By adjusting a plan in this fashion, the repositioning coordinator is able to evaluate the consequences compared to their original repositioning plan.

As this adjustment directly influences decision variables of the mathematical model, we need to adjust the model and the implementation of the simulated annealing algorithm. For this we define two new sets:

$V^B \subseteq V'$	Set of blocked visits.
$V^R \subseteq V'$	Set of required visits.

We add the following constraints to the mathematical model, and enforce these within the simulated annealing through a penalty function.

$$\sum_{s \in S} \sum_{i \in In(j)} y_{ij}^s = 1 \qquad \forall j \in V^R \tag{4.1}$$

$$\sum_{s \in S} \sum_{i \in Out(j)} y_{ji}^s = 1 \qquad \forall j \in V^R \tag{4.2}$$

$$\sum_{s \in S} \sum_{i \in In(j)} y_{ij}^s = 0 \qquad \forall j \in V^B \tag{4.3}$$

$$\sum_{s \in S} \sum_{i \in Out(j)} y_{ji}^s = 0 \qquad \forall j \in V^B \tag{4.4}$$

The variable y_{ij}^s is set to 1 if ship s uses arc (i, j). Constraints (4.1) and (4.2) enforce that there is exactly one arc to a required visit and one arc from a required visit that must be part of the solution. Constraints (4.3) and (4.4) do the opposite, ensuring that there is no arc in the solution that leads to or from a blocked visit.

Requiring a particular demand

During a repositioning it might be relevant to transport a particular demand, for example if a high valued customer's cargo might be disrupted by the repositioning. Especially when the repositioning vessel is using a sail-on-service opportunity, this option can be used to minimize disruptions to the cargo flow of the involved services. The flow of demand is part of the decision of the mathematical model and requires a slight modification. We simply require that the flow of a particular demand is set to the maximum of the demand, i.e., $x^{(o,d,q)} = a^{(o,d,q)}$, where o is the origin of the demand, d is a set of destinations, q is the container type, and $a^{(o,d,q)}$ is the amount of containers available. In the simulated annealing algorithm, we penalize solutions not carrying required cargo flows.

Using fixed time windows for flexible port calls

Flexible port calls are visits at ports that are not scheduled on any service directly involved in a repositioning. These ports have excess cargo and equipment and therefore it can be profitable to visit these ports. Using flexible visits in a repositioning plan is associated with some risk, as there is no guarantee that the time of such a visit is possible. To the best of our knowledge, there are currently no integrated systems between container terminals and shipping lines (even when both are owned by the same group). Repositioning coordinators must therefore manually confirm whether berthing times are available as planned Tierney (2015). In case the container terminal proposes time windows to the repositioning coordinator, they can use these options in our system to evaluate the consequences of these time windows for the whole repositioning plan.

Allowing time window adjustments is more difficult to implement in the simulated annealing algorithm, as the objective function evaluation changes significantly. The algorithm normally uses an LP to schedule flexible visits along a fixed path. However, time windows require binary variables, which turn it into a mixed-integer program (MIP). This extension has serious consequences for the complexity of the model. While an LP is solvable in polynomial time Khachiyan (1980), MIPs are NP-hard Karp (1972). However, since only a few binary variables are required, in practice these problems are easy to solve.

We propose the following mathematical formulation of time windows. We first provide the parameters, followed by the variables and the constraints. We implement the time window constraints in a soft fashion so that the user always receives a solution.

This also allows us to penalize infeasible solutions, as the simulated annealing solution may not always be feasible during search.

Sets and parameters:

$V^{TW} \subseteq V'$	Set of visits with time windows.
TW_i	Set of time window tuples (w_i^E, w_i^X)) for visit $i \in V^{TW}$.
P^E, P^X	Penalties for when the time windows are violated.
$w_i^E, w_i^X \in \mathbb{R}$	Time window enter and exit times for visit $i \in V^{TW}$.

Variables:

z_i^E, z_i^X	The enter and exit time at flexible node i, respectively.
y_{ik}^{TW}	Indicates whether time window $k \in TW_i$ for visit $i \in V^{TW}$ is being used (1) or not (0).
d_i^E, d_i^X	Amount of time that the time window of visit $i \in V^{TW}$ is missed by (both for the enter and exit times of the window).

$$\sum_{k \in TW_i} w_i^E y_{ik}^{TW} \leqslant z_i^E + d_i^E - d_i^X \qquad \forall i \in V^{TW} \tag{4.5}$$

$$z_i^E + d_i^E - d_i^X \leqslant \sum_{k \in TW_i} w_i^X y_{ik}^{TW} \qquad \forall i \in V^{TW} \tag{4.6}$$

$$\sum_{k \in TW_i} w_i^E y_{ik}^{TW} \leqslant z_i^X + d_i^E - d_i^X \qquad \forall i \in V^{TW} \tag{4.7}$$

$$z_i^X + d_i^E - d_i^X \leqslant \sum_{k \in TW_i} w_i^X y_{ik}^{TW} \qquad \forall i \in V^{TW} \tag{4.8}$$

$$d_i^E, d_i^X \geqslant 0 \qquad \forall i \in V^{TW} \tag{4.9}$$

Constraints (4.5) and (4.6) are used to set the enter time at visit i, z_i^E, according to the existing time windows. Constraints (4.7) and (4.8) function similarly for the exit times. We calculate the amount of time the windows are missed using d_i^E and d_i^X, and add these values to the objective function. We note that since the objective is maximized, the model will always try to minimize the penalties.

4.4.2. Case Study

We conduct a case study to show the effects of our interactive options and the workflow of our system. For this study we recreated a typical decision process using publicly available data. This data is derived from the public LSFRP instances from Tierney et al. (2015) and the LINERLIB, established by Brouer et al. (2014) for benchmarking models for the network design and fleet deployment problems in the liner shipping industry. We have not deployed the system at a liner carrier, and therefore show several hypothetical use cases for the system based on our experience and contact with carriers.

We examine three different LSFRP instances. We first solve them using the simulated annealing algorithm. We then examine what happens when we perform sample interactions with the system that a repositioning coordinator might reasonably undertake. The instances we choose vary mainly in the number of vessels and the structure of the services. Table 4.3 provides an overview of the instance properties, giving information about the number of services as well as the number of vessels of each instance and furthermore the number of port calls of each service in the three instances. The maximum number of port calls of the services are 15, 17 and 26 for instances 1, 2 and 3, respectively. All instances share two common services, with 13 and eight port calls each. All vessels used are Panamax sized vessels from the same vessel class. We note that many repositioning scenarios deal with such vessels in practice as they are very common. Furthermore, all the vessels that are indicated in Table 4.3 need to be repositioned for each instance. While these vessels might be positioned on different initial services, the fleet repositioning will bring them to the same goal service. We now describe each instance in detail.

Table 4.3.: Properties of the three instances for our case study.

	No. of services	No. of vessels	No. of port calls per service
Instance 1	5	3	13, 6, 8, 8, 15
Instance 2	5	4	13, 15, 17, 8, 14
Instance 3	4	9	13, 8, 8, 26

Instance 1. The first instance has three original services, one goal service and one service, which can be used as an SoS opportunity. On each of the original services, there is a single vessel that needs to be repositioned. These three vessels have a capacity of about 5800 TEU for dry cargo and about 500 TEU for reefer cargo (Table 4.4). The regions that are visited by the original services are Asia, North America, Europe and the Middle East. All three vessels need to be repositioned to the goal service, which only visits ports in South America. The connection to the other services is the port of Balboa (Panama). In the final repositioning plan, the latest start of operations of the goal service, should be in week 34 at the port of Balboa.

Table 4.4.: Properties of the vessels in Instance 1.

	Dry/reefer capacity	Original service	Earliest PO week
Vessel 1	5814/490	Asia - N. America	33
Vessel 2	5814/490	Middle East - N. America - Europe	28
Vessel 3	5814/490	Asia	29

Instance 2. The second instance consists of five services with port calls in Asia, North America, Europe, the Middle East and North Africa. The original services of the four vessels of this scenario operate in the regions Middle East, North America, Europe and Asia (see Table 4.5). Vessel 1 and Vessel 2 are originally assigned to the same service. Vessel 3 and Vessel 4 also have the same assignment. In the final repositioning plan, all four vessels have to be repositioned to the goal service, which only serves Asian ports. The latest start of operations for this goal service is in week 30 in the port of Chennai (India). For the creation of the repositioning plan, the two remaining services can be used as SoS opportunities.

Table 4.5.: Properties of the vessels in Instance 2.

	Dry/reefer capacity	Original service	Earliest PO week
Vessel 1	5814/490	Middle East - N. America - Europe	30
Vessel 2	5814/490	Middle East - N. America - Europe	29
Vessel 3	5814/490	Asia - N. America	30
Vessel 4	5814/490	Asia - N. America	29

Instance 3. In the third instance there are three initial services and one goal service. Three vessels are assigned to each initial service, so that in this scenario nine vessels need to be considered for the repositioning (see Table 4.6). The initial services are visiting ports in Asia, South America, North America and Europe. The goal service in this scenario shares most of these regions and is serving ports in Asia, North America and South America. In the final repositioning plan, the latest start of operations is at the port of Kaohsiung (Taiwan) in week 32.

Table 4.6.: Properties of the vessels in Instance 3.

	Dry/reefer capacity	Original service	Earliest PO week
Vessel 1	5814/490	Asia	32
Vessel 2	5814/490	Asia	31
Vessel 3	5814/490	Asia	30
Vessel 4	5814/490	S. America - Asia	34
Vessel 5	5814/490	S. America - Asia	33
Vessel 6	5814/490	S. America - Asia	32
Vessel 7	5814/490	Europe - N. America - Asia	33
Vessel 8	5814/490	Europe - N. America - Asia	32
Vessel 9	5814/490	Europe - N. America - Asia	31

We evaluated the case study on a virtual machine with OpenSuse 13.2 and an Intel Core 2 Quad processor (2.83GHz) with 4 GB of memory. The configuration of a real server for the DSS would be much more powerful in order to handle multiple processes at the same time. However, since the system was used by a single user during the case study, it was not necessary to choose another system setup.

Table 4.7.: Runtime comparison for the three instances of our case study in seconds.

	Instance 1	Instance 2	Instance 3
Initial Plan	77.43	309.82	436.33
Block a port	125.43	392.36	393.96
Add time windows	80.35	341.95	600.00
Force a demand	239.89	494.22	600.00

An extensive computational evaluation of the simulated annealing algorithm is available in Tierney (2015), and our goal is not to duplicate this. Instead, we wish to give insight into the interactive components of the system.

Table 4.7 shows the runtimes for the four planning steps of each instance in seconds. As in the non-interactive case, more ports and vessels results in more difficult instances. For these three scenarios, it takes on the average about 5.7 minutes to generate a repositioning plan. The planning steps with the longest runtimes are step three and four of Instance 3. These planning steps reached the time limit of ten minutes. These runtimes likely represent an improvement over the manual planning time by the repositioning coordinator. Nonetheless, without the knowledge of possible requirements for such a system, it is hard to evaluate whether this is fast enough for a real system.

4.4.3. Visualizations

The plans created by the DSS must be easy to use and modify. As stated in Power and Kaparthi (2002), the major tasks for a development of a web-based decision support system are the data model and the user interface. Especially the design of appropriate visualization approaches for relevant planning steps is a crucial task. By making use of recent advances in the area of design studies, we propose several new visualizations of repositioning plans.

Design studies are suitable to find a visualization technique for the LSFRP as they are meant for problem-driven research Sedlmair et al. (2012). According to Sedlmair et al. (2012), there are multiple steps that constitute a design study. Among other things, these steps involve the analysis of the problem, the abstraction of data and tasks, the design and implementation of the visualization technique and the evaluation of the visualization. By using the information of repositioning coordinators we were able to abstract relevant aspects of the problem (ports, port sequences, services, networks, time, vessels, phase-out/phase-in and sail-on-services) and use them as evaluation criteria for different visualization approaches of the fleet repositioning.

Spreadsheet View

The spreadsheet view as presented by Fagerholt and Lindstad (2007) for the TurboRouter system uses columns to represent time and vessels, and individual cells to display port calls. With some changes, this view can be adapted to demonstrate services and their port sequences as well. Although this view displays time, it is not possible to include phase-outs, phase-ins and sail-on-services without making the view too complex. In addition to this, the connection between vessels and their services is not exactly clear, let alone the connections between services. We note, however, that the spreadsheet view is the one repositioning coordinators are most familiar with. Thus, we integrate these views into our user interface whenever possible, but add extra constraints on the data that can be entered into the sheets.

Simple Graph

A simple approach to visualize services is to use a sequential graph for each service as is displayed in Figure 4.7. Each graph consists of nodes for the port calls and arcs for the sailings between calls. In order to add information about time, nodes and arcs can be labeled. Nevertheless, these labels lack the precision to determine the chronological order of phase-outs, phase-ins and the repositioning path. An example for this lack of precision are the repositionings from Shanghai (SHA) to Ningbo (NGB) and from Ningbo to Port Klang (PKG): The first repositioning ends at 16:00 and the second starts at 12:30. It is not clear whether the second repositioning starts on one of the next days or if it has started some time before. The same holds for the repositioning from Santos (SSZ) to Savannah (SAV).

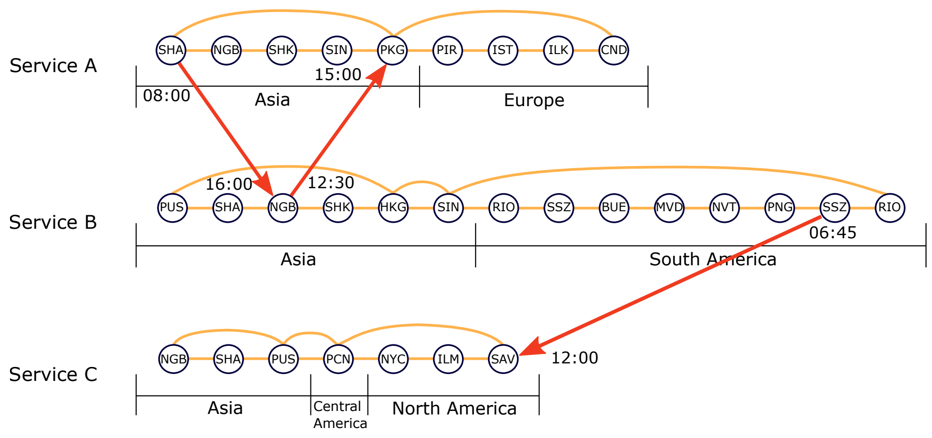


Figure 4.7.: Sequential graphs to illustrate services and repositionings

Connected Graph

By connecting separated graphs, it is possible to create a visualization that demonstrates the network structure of services and the connecting ports (Figure 4.8). In this view, the repositioning coordinator also has an overview of possible ports for the repositioning path. Such a path could be displayed using additional arcs. However, in this type of view it is problematic to add the dimension of time. There is the possibility to add time labels to the arcs for repositionings, but this information has the same problems as in the simple graph. Therefore, the repositioning coordinator is not able to see the chronological relation between the possible port calls and the fixed schedules of the services.

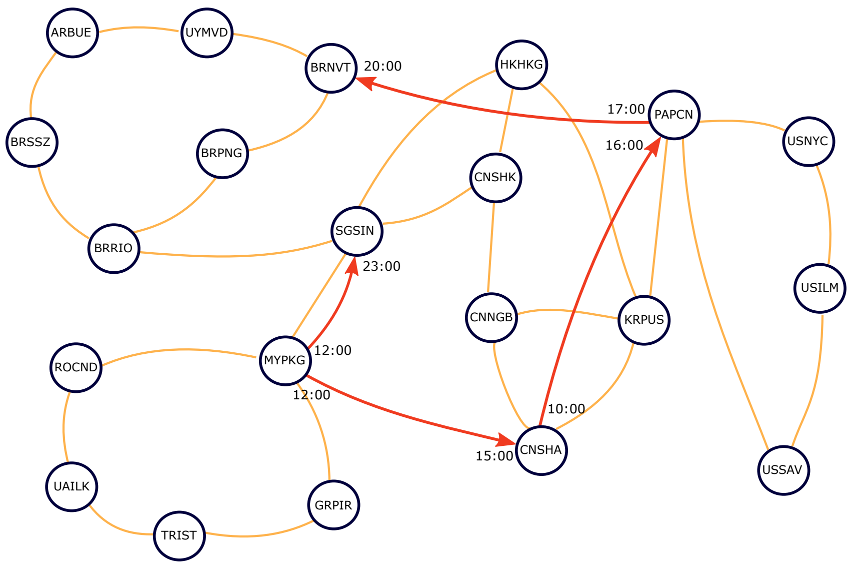


Figure 4.8.: A connected graph to illustrate services and repositionings

Transit Graph

A typical way to visualize networks of stations and paths in public transport networks is the “transit map” (Figure 4.9). This type of map is used for public transit plans to show the intersections of services and the relation between stations. Transit maps can give information about the time it takes to travel between two stations, but in the case of the LSFRP this information is not enough. In contrast to public transit, time distances between port calls are much longer. The repositioning coordinator needs precise information about the schedules in order to fully understand the repositioning plan, meaning this visualization provides insufficient information. This becomes obvious by adding time labels to repositioning paths. These time labels are not clear about their chronological ordering (see simple and connected graph). In addition, they lack the relation to scheduled port visits and slots of the services, which makes it impossible for the repositioning coordinator to make good decisions.

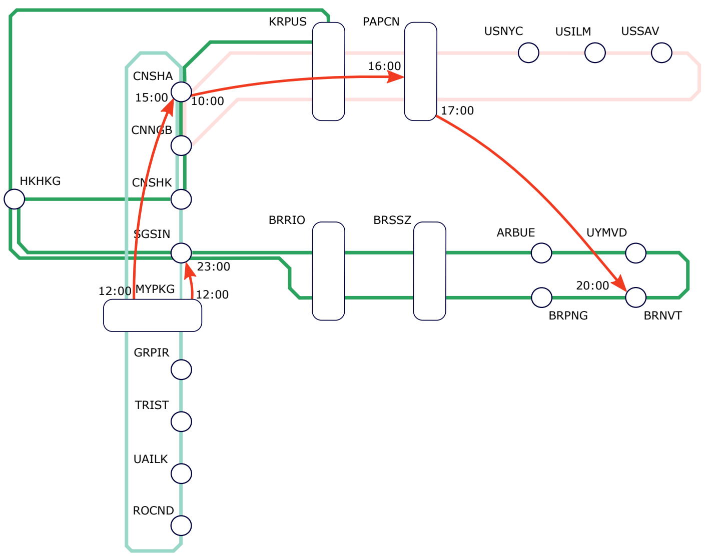


Figure 4.9.: A transit graph to illustrate services and repositionings

Service Graph

As the other approaches lack the ability of including the dimension of time, Figure 4.10 shows a visualization that is based on a time scale with days as the unit. By using this scale, the view presents the port calls of the services as nodes according to their occurrence. Additionally, arcs are used to represent sailings between ports. Services are displayed as closed boxes with multiple compartments, representing the associated vessels of each service. With this information, it is clear on which day the vessels are performing their port calls and how much time it takes to sail between two ports. There is no need for any kind of additional labels that might reduce the usability of the view. Repositionings can be visualized by using extra arcs to show the port nodes for the phase-out and phase-in. In this specific example, which is a small extract from bigger service schedules, these repositioning arcs are visualized by the red arrows from Service B to Service A and from Service A to Service C. Problem specific aspects like the sail-on-service opportunities can also be added to this view as can be seen in Figure 4.10.

Table 4.8 shows a summary of the previous comparison of the visualizations. The techniques of Figures 4.7-4.9 all lack the possibility to represent repositioning specific content like phase-outs (PO), phase-ins (PI) and sail-on-service opportunities (SOS).

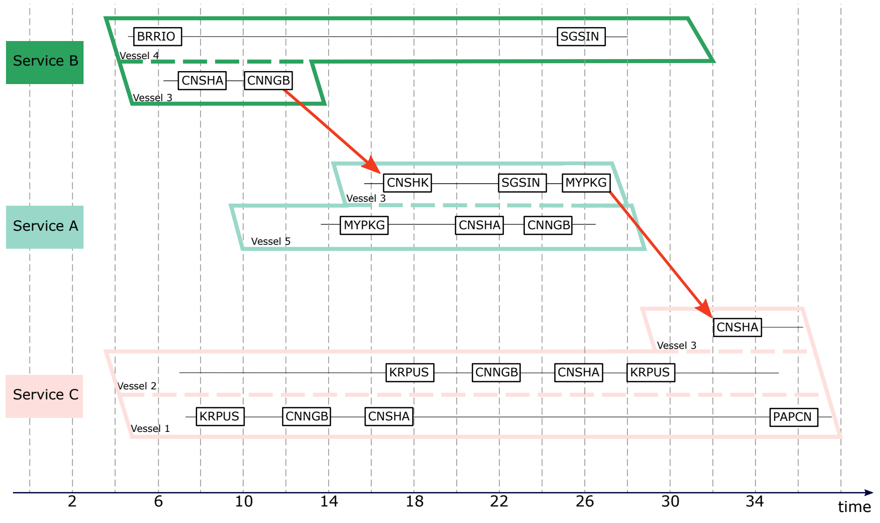


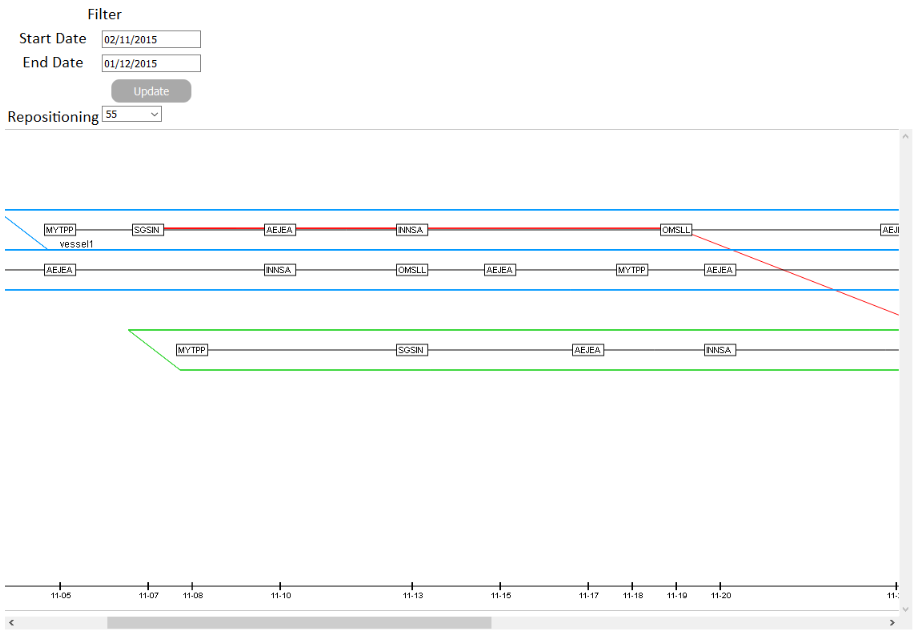
Figure 4.10.: Our approach to illustrate services and repositionings.

Only in Figure 4.10 is it possible to include this relevant information without making the view too complex or losing any usability.

Table 4.8.: Comparison of the visualization techniques

	Ports	Sequence	Service	Network	Time	Vessels	PO/PI	SOS
Spreadsheet	✓	✓	✓	×	✓	✓	×	×
Sequential	✓	✓	✓	×	∼	×	×	×
Connected	✓	∼	∼	✓	×	×	×	×
Transit	✓	✓	✓	✓	×	×	×	×
Service	✓	✓	✓	×	✓	✓	✓	✓

Due to the fact that the service graph approach is best able to visualize the problem, we implement it in our prototype. Figure 4.11 shows this implementation for a simple test scenario. In this scenario, “vessel 1” needs to reposition from the initial service (blue bars) to the goal service (green bar). The screenshot shows the planned port calls for each of the services. The red line shows the repositioning path of “vessel 1”. The filter menu on the left side of the screen the planner is able to adjust the time horizon.



Start Date: 02/11/2015
End Date: 01/12/2015
Repositioning: 55

Figure 4.11.: Screenshot of the service graph visualization in the prototype system.

4.5. Conclusion and future research

In this paper we design a DSS for the liner shipping fleet repositioning problem and provide a novel visualization technique for displaying repositioning plans. Our prototype shows that algorithms from the literature are ready for implementation in real-world systems at liner carriers.

In addition to this, we improve the interactivity of the current state-of-the-art simulated annealing approach. By not constraining the repositioning coordinator in his work, the process of creating repositioning plans with our system provides an extensive planning environment. Several options for an extended interactivity have been presented as well as their consequences for the most recent mathematical formulation. Despite the increased complexity of solving the objective function evaluation subproblem, we show that interactivity is not overly computationally expensive.

For future work, we intend to improve the system by providing the possibility to analyze and compare the strategic scenarios of the repositioning coordinator. Furthermore, it is our goal to discuss additional requirements with repositioning coordinators of liner shipping companies for this system.

Bibliography

Agarwal, R., Ergun, Ö.: Ship scheduling and network design for cargo routing in liner shipping. Transportation Science 42(2), 175–196 (2008).

Balmat, J. F., Lafont, F., Maifret, R., Pessel, N.: Maritime risk assessment (MARISA), a fuzzy approach to define an individual ship risk factor. Ocean Engineering 36(15-16), 1278–1286 (2009).

Balmat, J. F., Lafont, F., Maifret, R., Pessel, N.: A decision-making system to maritime risk assessment. Ocean Engineering 38(1), 171–176 (2011).

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

Becker, M., Tierney, K.: A hybrid reactive tabu search for liner shipping fleet repositioning. In: F. Corman, S. Voß, R. Negenborn (eds.) Computational Logistics, *Lecture Notes in Computer Science*, vol. 9335, pp. 123–138. Springer International Publishing (2015).

Bhargava, H. K., Power, D. J., Sun, D.: Progress in web-based decision support technologies. Decision Support Systems 43(4), 1083–1095 (2007).

Brehmer, M., Munzner, T.: A multi-level typology of abstract visualization tasks. IEEE Transactions on Visualization and Computer Graphics 19(12), 2376–2385 (2013).

Brønmo, G., Christiansen, M., Fagerholt, K., Nygreen, B.: A multi-start local search heuristic for ship scheduling—a computational study. Computers & Operations Research 34(3), 900–917 (2007).

Brouer, B. D., Alvarez, J. F., Plum, Christian E. M., Pisinger, D., Sigurd, M. M.: A base integer programming model and benchmark suite for liner-shipping network design. Transportation Science 48(2), 281–312 (2014).

Brouer, B. D., Dirksen, J., Pisinger, D., Plum, C. E., Vaaben, B.: The vessel schedule recovery problem (VSRP) – A MIP model for handling disruptions in liner shipping. European Journal of Operational Research 224(2), 362–374 (2013).

Chen, Y. L., Ke, Y. L.: Multi-objective VAr planning for large-scale power systems using projection-based two-layer simulated annealing algorithms. IEE Proceedings – Generation, Transmission and Distribution 151(4), 555 (2004).

Christiansen, M., Fagerholt, K., Nygreen, B., Ronen, D.: Ship routing and scheduling in the new millennium. European Journal of Operational Research 228(3), 467–483 (2013).

Dowsland, K. A., Thompson, J. M.: Simulated annealing. In: G. Rozenberg, T. Bäck, J.N. Kok (eds.) Handbook of Natural Computing, pp. 1623–1655, (2012).

Eglese, R. W.: Simulated annealing: A tool for operational research. European Journal of Operational Research 46(3), 271–281 (1990).

Fagerholt, K.: A computer-based decision support system for vessel fleet scheduling—experience and future research. Decision Support Systems 37(1), 35–47 (2004).

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

Fagerholt, K., Lindstad, H.: Turborouter: An interactive optimisation-based decision support system for ship routing and scheduling. Maritime Economics & Logistics 9(3), 214–233 (2007).

Fisher, M. L., Rosenwein, M. B.: An interactive optimization system for bulk-cargo ship scheduling. Naval Research Logistics (36), 27–42 (1989).

Karp, R. M.: Reducibility among combinatorial problems. Complexity of Computer Computations pp. 85–103 (1972)

Khachiyan, L. G.: Polynomial algorithms in linear programming. USSR Computational Mathematics and Mathematical Physics 20(1), 53–72 (1980).

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

Korsvik, J. E., Fagerholt, K., Laporte, G.: A large neighbourhood search heuristic for ship routing and scheduling with split loads. Computers & Operations Research 38(2), 474–483 (2011).

Lavigne, V., Gouin, D., Davenport, M.: Visual analytics for maritime domain awareness. IEEE International Conference on Technologies for Homeland Security (HST) pp. 49–54 (2011).

Mansouri, S. A., Lee, H., Aluko, O.: Multi-objective decision support to enhance environmental sustainability in maritime shipping: A review and future directions. Transportation Research Part E: Logistics and Transportation Review 78, 3–18 (2015).

Meyer, J., Stahlbock, R., Voß, S.: Slow steaming in container shipping. In: 45th Hawaii International Conference on System Science (HICSS), 2012, pp. 1306–1314. IEEE (2012).

Piramuthu, S., Shaw, M. J.: Learning-enhanced adaptive dss: a design science perspective. Information Technology and Management 10(1), 41–54 (2009).

Power, D. J., Kaparthi, S.: Building web-based decision support systems. Studies in Informatics and Control 11(4), 291–302 (2002).

Seçkiner, S. U., Kurt, M.: A simulated annealing approach to the solution of job rotation scheduling problems. Applied Mathematics and Computation 188(1), 31–45 (2007).

Sedlmair, M., Meyer, M., Munzner, T.: Design study methodology - reflextions from the trenches and the stacks. IEEE Transactions on Visualization and Computer Graphics 18(12), 2431–2440 (2012).

Si-Hwa Kim, Kyung-Keun Lee: An optimization-based decision support system for ship scheduling. Computers & Industrial Engineering 33(3-4), 689–692 (1997).

Tavakkoli-Moghaddam, R., Safaei, N., Kah, M., Rabbani, M.: A new capacitated vehicle routing problem with split service for minimizing fleet cost by simulated annealing. Journal of the Franklin Institute 344(5), 406–425 (2007).

Tierney, K.: Optimizing Liner Shipping Fleet Repositioning Plans, *Operations Research/Computer Science Interfaces Series*, vol. 57. Springer International Publishing (2015).

Tierney, K., Áskelsdóttir, B., Jensen, R. M., Pisinger, D.: Solving the liner shipping fleet repositioning problem with cargo flows. Transportation Science 49, 652 – 674 (2015).

Tierney, K., Coles, A., Coles, A., Kroer, C., Britt, A., Jensen, R.: Automated planning for liner shipping fleet repositioning. In: L. McCluskey, B. Williams, J. Silva, B. Bonet (eds.) Proceedings of the 22nd International Conference on Automated Planning and Scheduling, pp. 279–287 (2012).

United Nations Conference on Trade and Development: Review of maritime transport. United Nations, New York and Geneve (2014).

Wang, X., Wang, H., Zhang, L., Cao, X.: Constructing a decision support system for management of employee turnover risk. Information Technology and Management 12(2), 187–196 (2011).

4.6. Appendix: LSFRP arc flow model

We now provide the arc flow model from Tierney (2015) with a brief explanation. The model is based off of a graph with embedded LSFRP constraints, which we do not reproduce here for brevity. It suffices to be given a graph $G = (V^i \cup V^f, A^i \cup A^f)$, where V^i (A^i) is the set if inflexible visits (arcs) and V^f (A^f) is the set of flexible visits. As a reminder, inflexible visits have fixed enter and exit times for vessels and inflexible arcs have fixed sailing times and costs, whereas the amount of time a vessel spends on a flexible visits or arc is a decision variable. We now present the table of parameters for the mathematical model, followed by the objective function and constraints.

S	Set of ships.
V'	Set of visits minus the graph sink.
V^i, V^f	Set of inflexible and flexible visits, respectively.
A^i, A^f	Set of inflexible and flexible arcs, respectively.
A'	Set of arcs minus those arcs connecting to the graph sink, i.e., $(i, j) \in A$, $i, j \in V'$.
Q	Set of equipment types. $Q = \{dc, rf\}$.
M	Set of demand triplets of the form (o, d, q), where $o \in V', d \subseteq V'$ and $q \in Q$ are the origin visit, destination visits and the cargo type, respectively.
$V^{q+} \subseteq V'$	Set of visits with an equipment surplus of type q.
$V^{q-} \subseteq V'$	Set of visits with an equipment deficit of type q.
$V^{q^*} \subseteq V'$	Set of visits with an equipment surplus or deficit of type q ($V^{q^*} = V^{q^+} \cup V^{q^-}$).
$u_s^q \in \mathbb{R}^+$	Capacity of vessel s for cargo type $q \in Q$.
$M_i^{Orig}, (M_i^{Dest}) \subseteq M$	Set of demands with an origin (destination) visit $i \in V$.
$v_s \in V'$	Starting visit of ship $s \in S$.
$t_{si}^{Mv} \in \mathbb{R}$	Move time per TEU for vessel s at visit $i \in V'$.
$t_i^E \in \mathbb{R}$	Enter time at inflexible visit $i \in V'$.
$t_i^X \in \mathbb{R}$	Exit time at inflexible visit $i \in V'$.
$t_i^P \in \mathbb{R}$	Pilot time at visit $i \in V'$.
$r_q^{Var} \in \mathbb{R}^+$	Revenue for each TEU of equipment of type $q \in Q$ delivered.
$r^{(o,d,q)} \in \mathbb{R}^+$	Amount of revenue gained per TEU for the demand triplet.
$c_{sij}^{Sail} \in \mathbb{R}^+$	Fixed cost of vessel s utilizing arc $(i, j) \in A'$.
$c_{sij}^{VarSail} \in \mathbb{R}^+$	Variable hourly cost of vessel $s \in S$ utilizing arc $(i, j) \in A'$.
$c_i^{Mv} \in \mathbb{R}^+$	Cost of a TEU move at visit $i \in V'$.
$c_{si}^{Port} \in \mathbb{R}$	Port fee associated with vessel s at visit $i \in V'$.
$d_{ijs}^{Min} \in \mathbb{R}^+$	Minimum duration for vessel s to sail on flexible arc (i, j).
$d_{ijs}^{Max} \in \mathbb{R}^+$	Maximum duration for vessel s to sail on flexible arc (i, j).
$a^{(o,d,q)} \in \mathbb{R}^+$	Amount of demand available for the demand triplet.
$In(i) \subseteq V'$	Set of visits with an arc connecting to visit $i \in V$.

$Out(i) \subseteq V'$	Set of visits receiving an arc from $i \in V$.
$\tau \in V$	Graph sink, which is not an actual visit.

$w_{ij}^s \in \mathbb{R}_0^+$	The duration that vessel $s \in S$ sails on flexible arc $(i,j) \in A^f$.
$x_{ij}^{(o,d,q)} \in \mathbb{R}_0^+$	Amount of flow of demand triplet $(o,d,q) \in M$ on $(i,j) \in A'$.
$x_{ij}^q \in \mathbb{R}_0^+$	Amount of equipment of type $q \in Q$ flowing on $(i,j) \in A'$.
$y_{ij}^s \in \{0,1\}$	Indicates whether vessel s is sailing on arc $(i,j) \in A$.
$z_i^E \in \mathbb{R}_0^+$	Defines the enter time of a vessel at visit i.
$z_i^X \in \mathbb{R}_0^+$	Defines the exit time of a vessel at visit i.

$$\max \; -\sum_{s \in S} \left(\sum_{(i,j) \in A'} c_{sij}^{Sail} y_{ij}^s + \sum_{(i,j) \in A^f} c_{sij}^{VarSail} w_{ij}^s \right) \tag{4.10}$$

$$+ \sum_{(o,d,q) \in M} \left(\sum_{j \in d} \sum_{i \in In(j)} \left(r^{(o,d,q)} - c_o^{Mv} - c_j^{Mv} \right) x_{ij}^{(o,d,q)} \right) \tag{4.11}$$

$$+ \sum_{q \in Q} \left(\sum_{i \in V^{q+}} \sum_{j \in Out(i)} \left(r_q^{Eqp} - c_i^{Mv} \right) x_{ij}^q - \sum_{i \in V^{q-}} \sum_{j \in In(i)} c_i^{Mv} x_{ji}^q \right) \tag{4.12}$$

$$- \sum_{j \in V'} \sum_{i \in In(j)} \sum_{s \in S} c_{sj}^{Port} y_{ij}^s \tag{4.13}$$

$$\text{subject to} \sum_{s \in S} \sum_{i \in In(j)} y_{ij}^s \leqslant 1 \qquad \forall j \in V' \tag{4.14}$$

$$\sum_{j \in Out(i)} y_{ij}^s = 1 \qquad \forall s \in S, i = v_s \tag{4.15}$$

$$\sum_{i \in In(\tau)} \sum_{s \in S} y_{i\tau}^s = |S| \tag{4.16}$$

$$\sum_{i \in In(j)} y_{ij}^s - \sum_{i \in Out(j)} y_{ji}^s = 0 \qquad \forall j \in \{V' \setminus \bigcup_{s \in S} v_s\}, s \in S \tag{4.17}$$

$$\sum_{(o,d,rf) \in M} x_{ij}^{(o,d,rf)} \leqslant \sum_{s \in S} u_s^{rf} y_{ij}^s \qquad \forall (i,j) \in A' \tag{4.18}$$

$$\sum_{(o,d,q) \in M} x_{ij}^{(o,d,q)} + \sum_{q' \in Q} x_{ij}^{q'} \leqslant \sum_{s \in S} u_s^{dc} y_{ij}^s \qquad \forall (i,j) \in A' \tag{4.19}$$

$$\sum_{i \in Out(o)} x_{oi}^{(o,d,q)} \leqslant a^{(o,d,q)} \sum_{i \in Out(o)} \sum_{s \in S} y_{oi}^s \qquad \forall (o,d,q) \in M \tag{4.20}$$

$$\sum_{i \in In(j)} x_{ij}^{(o,d,q)} - \sum_{k \in Out(j)} x_{jk}^{(o,d,q)} = 0 \forall (o,d,q) \in M, j \in V' \setminus (o \cup d) \tag{4.21}$$

$$\sum_{i \in In(j)} x_{ij}^q - \sum_{k \in Out(j)} x_{jk}^q = 0 \qquad \forall q \in Q, j \in V' \backslash V^{q*} \tag{4.22}$$

$$d_{ijs}^{Min} y_{ij}^s \leqslant w_{ij}^s \leqslant d_{ijs}^{Max} y_{ij}^s \qquad \forall (i,j) \in A^f, s \in S \tag{4.23}$$

$$z_i^E = t_i^E \sum_{s \in S} \sum_{j \in In(i)} y_{ij}^s \qquad \forall i \in V^i \tag{4.24}$$

$$z_i^X = t_i^X \sum_{s \in S} \sum_{j \in Out(i)} y_{ij}^s \qquad \forall i \in V^i \tag{4.25}$$

$$z_i^X + \sum_{s \in S} w_{ij}^s \leqslant z_j^E \qquad \forall (i,j) \in A^f \tag{4.26}$$

$$\begin{aligned} &\sum_{(o,d,q) \in M_i^{Orig}} \sum_{j \in Out(o)} t_{so}^{Mv} x_{oj}^{(o,d,q)} + \sum_{(o,d,q) \in M_i^{Dest}} \sum_{d' \in d} \sum_{j \in In(d')} t_{sd}^{Mv} x_{jd'}^{(o,d,q)} \\ &+ \sum_{q \in Q} \left(\sum_{i' \in \{V^{q+} \cap \{i\}\}} \sum_{j \in Out(i')} t_{si'}^{Mv} x_{i'j}^q + \sum_{i' \in \{V^{q-} \cap \{i\}\}} \sum_{j \in In(i')} t_{sj}^{Mv} x_{ji'}^q \right) \\ &- \quad z_i^X + z_i^E + t_i^P \sum_{j \in In(i)} y_{ij}^s \leqslant 0 \qquad \forall i \in V^f, s \in S \end{aligned} \tag{4.27}$$

The objective first consists of the sailing cost (4.10) that takes into account the precomputed sailing costs on arcs between inflexible visitations, as well as the variable cost for sailings to and from flexible visitations. Note that the fixed sailing cost on an arc includes fuel costs, canal fees or even the time-charter bonus for entering an SoS. The profit from delivering cargo (4.11) is computed based on the revenue from delivering cargo minus the cost to load and unload the cargo from the vessel. Equipment profit is taken into account in (4.12), and, finally, port fees are deducted in (4.13).

Multiple vessels are prevented from visiting the same visitation in (4.14). The flow of each vessel from its source node to the graph sink is handled by (4.15), (4.16) and (4.17), with (4.16) ensuring that all vessels arrive at the sink.

Arcs are assigned capacities if a vessel utilizes the arc in (4.18), which assigns the reefer container capacity, and in (4.19), which assigns the total container capacity, respectively. Note that constraints (4.18) do not take into account empty reefer equipment, since empty containers do not need to be turned on, and can therefore be placed anywhere on the vessel. Cargo is only allowed to flow on arcs with a vessel in (4.20). The flow of cargo from its source to its destination, through intermediate nodes, is handled by (4.21). Constraints (4.22) balance the flow of equipment in to and out of nodes. In contrast to the way cargo is handled, equipment can flow from any port where it is in supply to any port where it is in demand. Since the amount of equipment carried is limited only by the capacity of the vessel, no flow source/sink constraints are required.

Flexible arcs have a duration constrained by the minimum and maximum sailing time of the vessel on the arc in (4.23). The enter and exit time of a vessel at inflexible ports is handled by (4.24) and (4.25), and we note that in practice these constraints are only necessary if one of the outgoing arcs from an inflexible visitation ends at a flexible visitation. Constraints (4.26) sets the enter time of a visitation to be the duration of a vessel on a flexible arc plus the exit time of the vessel at the start of the arc. Constraints (4.27) controls the amount of time a vessel spends at a flexible visitation. The first part of the constraint computes the time required to load and unload cargo and equipment, with the final term of the constraint adding the piloting time to the duration only if one of the incoming arcs is enabled (i.e., the flexible visitation is being used).

5. Integrating Fleet Deployment into the Liner Shipping Cargo Allocation Problem

Daniel Müller* Stefan Guericke** Kevin Tierney*

*University of Paderborn
Warburger Straße 100, 33098 Paderborn, Germany
mueller@dsor.de, tierney@dsor.de

**A.P. Moller Maersk, Copenhagen, Denmark
stefan.guericke@maersk.com

Liner carriers change their network on a regular basis, and they are therefore interested in a practical evaluation of the impact these changes have on the cargo flows in their networks. Despite great advancements in the practical applicability of network evaluators in recent years, vessel deployment continues to be considered as an input into the problem, rather than a decision. In this paper, we propose an extension of a state-of-the-art mixed integer programming model for the LSCAP that incorporates the optimization of vessel count and vessel classes for each service. We perform a computational analysis on liner shipping networks of different sizes and compare our optimized results to fixed deployment scenarios. By integrating fleet deployment decisions into the cargo allocation problem, liner carriers can increase the profitability of their networks by at least 2.8 to 16.9% and greatly enhance their decision making.

Keywords: liner shipping, cargo allocation, fleet deployment

5.1. Introduction

Seaborne trade plays a critical role in global markets, and is responsible for transporting more than 10 billion tons of goods per year UNCTAD (2016). Furthermore, since the year 2000 the number of containers transported each year has almost tripled UNCTAD (2016). The challenge of designing, adjusting and operating liner shipping networks is thus becoming increasingly difficult to solve with current tools.

Containerized goods are transported in liner shipping networks on cyclical routes called services. Liner shipping companies operate a number of these services to connect different trade regions within their network. Services are generally operated with a certain periodicity (usually weekly or biweekly) such that ports of a service are visited at a fixed time each period. At each port of the service, cargo is loaded, unloaded or transshipped. Furthermore, cargo is transported in varying container types and sizes (*equipment types*).

Liner carriers must regularly make changes to their network, such as adding new services, changing the ports visited on a service, or removing services that are no longer profitable. These network modifications can have far-reaching and non-obvious effects on the network, such as changing the capacity or connection time between ports that are not part of the subset of the network being changed. Liner carriers are therefore interested in examining the impact of network changes on the cargo flows of their networks, which has been formulated by the operations research community as the liner shipping cargo allocation problem (LSCAP) (see, e.g., Guericke and Tierney (2015)).

The LSCAP targets the strategic and tactical planning horizon of a carrier. It computes the profit-maximal cargo flows on a predefined service network to provide carriers with a holistic view of their network, under the assumption that the cargo flows are deterministic. Models in the literature consider a number of detailed aspects, such as transit time requirements and variable vessel speeds.

Another tactical planning problem in the liner shipping industry is the fleet deployment problem. This problem is solved to determine the number of vessels and the type of vessel (*vessel class*) on a service. The assignment of vessels to services has a direct influence on the capacity of the services and the possible vessel speeds. Currently, these problems are solved independently of each other, with the output of the fleet deployment problem being fed into the LSCAP as fixed numbers of vessels and vessel types. However, the allocation of cargo is dependent on the capacity of the service vessels and the schedule of the services, meaning adjusting the deployment could yield higher profit in the LSCAP for the carrier. Simultaneously optimizing the cargo allocation and deployment offers tactical level guidance to carriers for which types of vessels should go where, and could even be used in a strategic context to determine how many vessels of a particular type should be built or chartered.

We extend the LSCAP model in Guericke and Tierney (2015). The complete model contains the following components: path restricted multicommodity flow, transshipments, complex routes, transit time requirements, leg based speed optimization and empty container repositioning. We add the assignment of vessel classes to services and the determination of the number of vessels of each service to the model.

With the help of our model, liner carriers can, for example, evaluate whether "upgrading" a service to a bigger vessel class is profitable or not. Furthermore, carriers can evaluate the effects of selecting different vessel classes and numbers of vessels for their services. These decisions influence individual leg speeds and the overall schedule of the service. This, in turn, changes how many containers can be transported, as faster ships can better meet customer's transit time requirements. We show in our computational experiments, considering fleet deployment leads to an increase in profit of several million dollars.

This paper is organized as follows. First, we review the related literature in Section 5.2. Then, Section 5.3 presents the cargo allocation problem with fleet deployment and vessel class selection. In Section 5.4, the model is presented. Section 5.5 contains the results of our computational experiments. Finally, Section 5.6 concludes and offers an outlook on future work.

5.2. Literature Review

There is a wealth of work considering cargo allocation/cargo routing and fleet deployment, however very little that addresses the intersection between these problems. Table 5.1 presents a summary of relevant work. For details, we refer to Guericke and Tierney (2015). The upper half of the table contains publications about cargo allocation, while the lower half contains fleet deployment publications. The table only refers to publications that consider complex service types. For an extensive overview of other optimization problems in liner shipping, we refer to Brouer et al. (2017).

Table 5.1.: Overview of relevant publications about the fleet deployment problem and the cargo allocation problem[1].

Paper	MCF	TS	TT	TSD	LBSO	ER	VC	VCL
Akyüz and Lee (2014)	✓	✓	✓	✓	-	-	-	-
Karsten et al. (2015)	✓	✓	✓	✓	-	-	-	-
Guericke and Tierney (2015)	✓	✓	✓	(✓)	✓	✓	-	-
Branchini et al. (2015)	-	-	-	-	-	-	(✓)	✓
Wang et al. (2016)	✓	✓	-	(✓)	-	✓	-	-
Gelareh and Meng (2010)	(✓)	-	✓	-	(✓)	(✓)	✓	✓
Meng and Wang (2010)	-	-	-	-	-	-	✓	✓
Wang and Meng (2012)	✓	✓	-	-	-	-	✓	✓
Meng et al. (2013)	✓	✓	-	(✓)	-	-	✓	✓
This article	✓	✓	✓	(✓)	✓	✓	✓	✓

5.2.1. Cargo Allocation

In the work of Akyüz and Lee (2014), a column generation approach is used to solve the cargo allocation problem with service levels, which are defined as combinations of vessel capacity and vessel speed. Although in Karsten et al. (2015) speed optimization is not considered, they extend the previous mentioned literature by including transit times and transshipment durations.

The publication of Guericke and Tierney (2015) integrates transit times and transshipment durations as well as speed optimization on individual legs and empty container repositioning. By integrating these requirements and aspects into a single model, Guericke and Tierney (2015) provide a high level of realism, making it valuable for liner carriers. Since varying vessel speed results in a non-linear optimization problem, the bunker consumption functions need to be linearized. We refer to Psaraftis

[1]MCF = Multiple cargo flows, TS = Transshipment, TT = Transit times, TSD = Transshipment duration, LBSO = Leg based speed optimization, ER = Empty repositioning, VC = Vessel count, VCL = Vessel class

and Kontovas (2016) for a taxonomy of speed optimization publications, and various formulations of the fuel consumption function.

The integration of contractual cargo and spot cargo is considered in Branchini et al. (2015). A mixed-integer programming model is presented that optimizes cargo assignments as well as the deployment and scheduling of vessels. Speed optimization and empty container repositioning are not included in this model.

In the work of Wang et al. (2016), a chance-constrained optimization model is presented that considers deterministic and stochastic demand as well as various shipping activities like container loading/unloading, transshipments and waiting times. The model does not include any kind of speed optimization, but it does consider selecting the best vessel class for a specific service.

5.2.2. Fleet Deployment

The fleet deployment problem is integrated with the optimization of vessel speed and service frequency in Gelareh and Meng (2010). A mixed integer model considers transit time restrictions and is evaluated on a set of randomly generated instances.

Another chance constrained model to solve the fleet deployment problem is presented by Meng and Wang (2010). This model assumes a normal distribution for the cargo demand of each service. Without considering leg based speed optimization, empty container repositioning or transit times, the mixed integer programming model is solved on nearly realistic instances.

The model of Wang and Meng (2012) considers transshipment operations combined with the fleet deployment problem. For this, transshipments are allowed to be carried out multiple times without restrictions. Vessels can either be used from the carrier's fleet or chartered to be deployed on services. The authors assume predetermined vessel speeds for each service leg. In Meng et al. (2013), a two-stage stochastic programming model that considers uncertainties in container demand and transshipment options is used to solve the fleet deployment problem.

5.3. Problem description

Before we describe the details of our mixed-integer programming model we discuss the aspects of the integrated cargo allocation and fleet deployment problem addressed in this paper.

5.3.1. Cargo allocation problem

Cargo allocation models are used for a strategic or tactical evaluation of a given liner shipping network. By considering possible cargo flows as well as fixed and variable costs, it can determine the profitability of an entire network. Furthermore, liner carriers are able to use the results of this model to refine their service schedules and improve coordination with container terminals.

The solution of the cargo allocation model determines how much cargo from each demand is carried and how that cargo is routed from its origin to its destination.

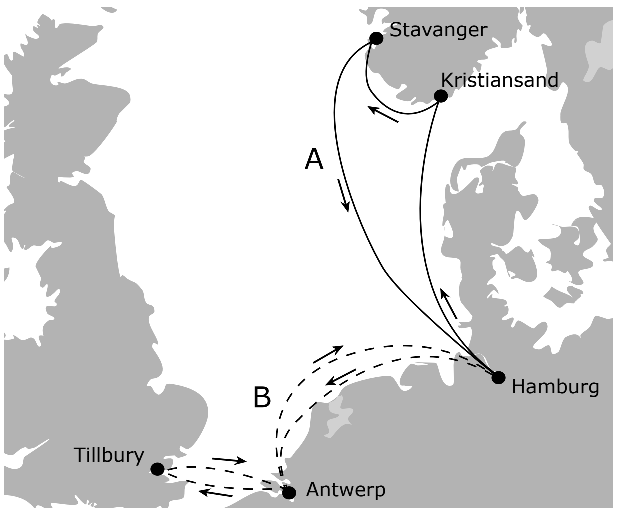


Figure 5.1.: A simple liner shipping network in Northern Europe with two services connecting ports in Germany, Belgium, Denmark and Norway, from Guericke and Tierney (2015).

When a single service contains both the origin and destination of a cargo demand, only loading and unloading operations along with the routing within the service need to be determined. For other cargo in which the origin and destination ports are distributed over separate services, both services need to be connected by transshipping the cargo at a shared port. If there is no port present in both services, multiple transshipments will occur until the cargo arrives at its destination port. We allow split cargo flows, but restrict the maximum number of paths the cargo may take.

Figure 5.1 shows a simple liner shipping network with two services. Both services are connected by the transshipment port Hamburg, such that cargo can be transported from Stavanger to Antwerp or Tillbury.

Cargo demands are associated with a maximum transit time between the demand's origin and destination. The transit time consists of the time cargo spends traveling by ship added to the time it spends in port during transshipment. We take into account the movement time of the containers on and off the ship, but use a constant transshipment time due to the complexity of allowing this to vary.

Depending on the overall frequency of the service and the total call time in ports, the remaining time can be spent for sailing between ports. The speed of the vessels is adjusted such that this remaining total sailing time is enough to maintain the periodicity of the service. Since the duration at sea is closely connected to the available time for vessels to move cargo in ports, including speed optimization in the cargo allocation problem is necessary. Additionally, speed optimization reduces bunker fuel consumption, one of the main costs of operating a seagoing vessel Stopford (2009).

Vessels have a minimum and maximum speed in which the vessel can be operated.

Buffer can be added between port visits on a schedule if the sailing time between the ports is longer than the vessel would require even at its minimum speed. Buffer can be used to hedge against uncertainty, although we do not directly consider this in our model.

5.3.2. Fleet deployment

The assignment of vessels to services is a tactical planning problem in which, typically, an entire shipping season is planned Christiansen et al. (2013). An assignment consists of the selection of a vessel class for a service as well as the determination of the total number of vessels for a service. Liner carriers use the fleet deployment problem to regularly assess the cost-efficiency of their network structure based on current rates of the charter-market. Dependent on the charter and bunker market, new vessels can be hired or existing off-hired, or own vessels chartered out.

In our model, we integrate the selection of vessel classes for the services and the determination of the number of vessels with the cargo allocation decisions. The goal is to benefit from the close connection between these fleet deployment decisions, the possible leg speeds and cargo moves. This integration also provides a more precise estimation of possible profits Wang et al. (2016).

By optimizing the vessel class for each service, we take advantage of the specifications of these classes. In our case, vessel classes are defined by different minimum and maximum speeds as well as differing capacities (called *resource groups*), port call costs and charter costs. Resource groups can be, for example, the maximum weight the vessel can transport, the maximum number of container slots or the maximum number of reefer container plugs. We assume that only a single vessel class can be assigned to a service, which is a reasonable assumption in practice.

5.4. A mixed integer programming model

In this section we provide the formal definition of the mixed-integer programming model for the cargo allocation problem with speed optimization and fleet deployment. This model is based on the formulation of Guericke and Tierney (2015), which uses a directed graph as a representation of the problem. Nodes represent port calls and arcs represent the legs between ports of a service. The graph includes a layered structure to model multiple visits to the same port in a single service. For more details about the graph formulation we refer to Guericke and Tierney (2015). In addition to adding fleet deployment to the model, we also change the piece-wise linearization of the bunker consumption costs to the more efficient approach of Reinhardt et al. (2016).

In the following, we define the relevant index sets of the model. It should be noted that the majority of the index sets are equal to the original formulation. We added two sets to define Cartesian products of other index sets.

Sets	
Q	Set of vessel classes
P	Set of ports
S	Set of services
$\mathbb{L}_s$	Set of layers for service s
$P_s \subseteq P \times \mathbb{L}_s$	Set of ports for service s
$L_s \subseteq P_s \times P_s$	Set of legs for service s
R	Set of resources
G	Set of resource groups
$R_g \subseteq G$	Set of resources in resource group g
N	Set of cargo flows
N_p^{OD}	Set of cargo flows whose origin or destination is port p, $N_p^{OD} = \{n \in N \mid o_n = p \vee d_n = p\}$
E	Set of equipment types for empty container balancing
$\Pi = \{0, 1, ...\}$	Set of container paths
N_e	Set of cargo flows and container paths of equipment type $e \in E$, $N_e = \{(n, \pi) \in N \times \Pi \mid e_n = e\}$
$H^N = S \times P \times Q$	Set of the Cartesian product of services, ports and vessel classes
$H^F = N \times \Pi$	Set of the Cartesian product of cargo flows and container paths

We now introduce the parameters of the model. We add parameters for the secant based linearization of the model. It is necessary to determine these secants for each vessel class as the classes have different bunker cost functions.

Parameters	
$f_s \in \mathbb{N}_+$	Frequency in days of service s
$\delta^+(s,p,l) \in L_s$	Incoming leg of service s to port p,l
$\delta^-(s,p,l) \in L_s$	Outgoing leg of service s from port p,l
$C_{gq} \in \mathbb{N}_+$	Capacity of vessel type q of resource group g
$k_q^{Min} \in \mathbb{R}_+$	Minimum speed in knots of vessel type q
$k_q^{Max} \in \mathbb{R}_+$	Maximum speed in knots of vessel type q
$a_{rc} \in \mathbb{R}_+$	Utilization of resource r of container c
$o_n \in P$	Origin of cargo flow n
$d_n \in P$	Destination of cargo flow n
$q_n^{Max} \in \mathbb{R}_+$	Maximum quantity of cargo flow n in the planning horizon
$e_n \in E$	Equipment type of cargo flow n
$r_n \in \mathbb{R}_+$	Revenue in US$ of cargo flow n
$\theta_n \in \mathbb{R}_+$	Maximum transit time of cargo flow n in days
$t_{pq}^E \in \mathbb{R}_+$	Duration in hours to move one container at port p with vessel type q. A move is a loaded or unloaded container
$t_p^{Add} \in \mathbb{R}_+$	Additional constant duration (for pilotage, bunkering etc.) required at port p in hours
$t \in \mathbb{N}_+$	Length of the planning horizon in days
w_s	Weekly volume adjustment parameter, $w_s = \frac{f_s}{t}$
$t_p^F \in \mathbb{R}_+$	Fixed container storage duration in port p in days
$\phi_{pq}^{PC} \in \mathbb{R}_+$	Port call cost in US$ per call of vessel type q at port p
$\phi_q^D \in \mathbb{R}_+$	Depreciation/time charter cost of vessel type q
$\phi_p^{CH} \in \mathbb{R}_+$	Container handling cost at port p per unit in US$
$\phi_p^{TS} \in \mathbb{R}_+$	Transshipment cost at port p per unit in US$
$\phi_e^C \in \mathbb{R}_+$	Depreciation cost for one unit of equipment type $e \in E$
$\phi^P \in \mathbb{R}_+$	Penalty cost for services that have too few vessels deployed
g_{lq}^v	Gradient of secant for bunker consumption approximation of leg $l \in L_S$ for vessel type $q \in Q$
i_{lq}^v	y-intercept of secant for bunker consumption approximation of leg $l \in L_S$ for vessel type $q \in Q$
$\mathbb{M}_q^{Max}$	Max. amount of vessels of vessel type $q \in Q$
$\mathbb{M}_{kq}^{SMax}$	Max. costs to sail leg $k \in L_s$ with vessel type $q \in Q$
$\mathbb{M}_{spl}^P$	Maxiumum port duration for service s, port p,l
$\mathbb{M}_{sk}^S$	Maximum sailing duration for service s, leg k
$\mathbb{M}_n^C$	Maximum. duration for all container paths of cargo $n \in N$
ϵ	Small value for adjusting the transshipment indicator variables

Finally, we present the variables of the model. To support vessel class and count decisions, several sets of new variables are required. As discussed previously, the different vessel classes have different consumption functions, leading to different secant sets. Consequently, the variables for leg durations and bunker costs per leg also need a dependency on the vessel class. The main variables that are introduced to reflect the new possible decisions of vessel class selection and setting the vessel count are y_{sq}^V and γ_{sq}. Previously, γ_{sq} was a parameter that was set by the planner beforehand. Now, it can freely range on an integer scale larger than zero. The binary variable y^V_{sq}, however, describes whether a specific vessel is assigned to a service or not.

Variables

y^V_{sq}	Indicates whether vessel type $q \in Q$ is used on service $s \in S$
$\gamma_{sq} \in \mathbb{N}_+$	Vessel count of service $s \in S$ of vessel type $q \in Q$
$b_{skq} \in \mathbb{R}_+$	Bunker cost of leg k of service s for vessel type q
ϕ^F	Fixed cost of all services in the planning horizon
$\alpha_{n\pi} \in \mathbb{R}_+$	The quantity of cargo flow n on container path π over the entire planning horizon
$x_{skn\pi} \in \mathbb{R}_+$	The quantity of cargo flow $n \in N$ for path π on leg $k = (i, l, j, l') \in L_s$ of service s
$x_{ske} \in \mathbb{R}_+$	The amount of flow of equipment type $e \in E$ on leg $k = (i, l, j, l') \in L_s$ of service s
$l_{spln\pi q}$, $u_{spln\pi q} \in \mathbb{R}_+$	The amount of laden containers loaded and unloaded of flow $n \in N$ to and from liner service s at port $(p, l) \in P_s$ on container path π for vessel type $q \in Q$
l_{spleq}, $u_{spleq} \in \mathbb{R}_+$	The amount of empty equipment loaded and unloaded of empty flow $e \in E$ to and from liner service s at port $(p, l) \in P_s$ for vessel type $q \in Q$
$y^{CPL}_{skn\pi} \in \{0, 1\}$	Indicates whether leg k of service s is used to route cargo n on path π
$y^{CP}_{n\pi}$	Indicates whether cargo flow is $n \in N$ is routed on container path $\pi \in \Pi$
$y^T_{spln\pi} \in \{0, 1\}$	Indicates whether cargo n on path π is transshipped at port (p, l) on service s
$\tau^{TT}_{spln\pi} \in \mathbb{R}_+$	The time in days per visit to unload a cargo n on path π at service s port (p, l) for transshipment operations
$\tau^{TF}_{spln\pi} \in \mathbb{R}_+$	The time in days per visit to forward cargo flow n from service s lth call of port p to the succeeding port (and no cargo transshipment is performed)
$\tau^{CP}_{skn\pi} \in \mathbb{R}_+$	The total duration in days to route cargo flow n on path π on service s leg k
τ_s	Round trip time in days of service s
τ^V_s	Total relevant duration of all vessels of service s (number of vessels times the planning duration)
$\tau^L_{skq} \in \mathbb{R}_+$	Duration in days to travel leg k in service s for vessel type q
$\tau^S_{sk} \in \mathbb{R}_+$	Auxiliary variable specifying the duration in days it takes service s to steam leg $k \in L_s$ over the whole planning horizon. $\tau^S_{s,k} \leqslant t^{SMax}_{s,k}$
$\tau^P_{spl} \in \mathbb{R}_+$	Auxiliary variable giving the duration in days that service s calls port (p, l)
$\tau^B_{spl} \in \mathbb{R}_+$	The additional buffer for service s at port (p, l) in days to hold the round trip time
$\rho^{VS}_s \in \mathbb{R}_+$	Slack variable allowing vessels in service s to steam above maximum speed

We now introduce the objective and constraints of the model.

$$\begin{aligned}
max \; = & \sum_{n \in N} \sum_{\pi \in \Pi} \left(r_n - \phi^C_{en}\right) \alpha_{n\pi} - \phi^F - \sum_{s \in S} \phi^P \rho^{VS}_s && (5.1)\\
& - \sum_{s \in S, (p,l) \in P_s} \sum_{n \in N^{OD}_p} \sum_{\pi \in \Pi} \sum_{q \in Q} \phi^{CH}_p \left(u_{spln\pi q} + l_{spln\pi q}\right) && (5.2)\\
& - \sum_{s \in S, (p,l) \in P_s} \sum_{n \in N \setminus N^{OD}_p} \sum_{\pi \in \Pi} \sum_{q \in Q} \phi^{TS}_p \left(u_{spln\pi q} + l_{spln\pi q}\right) && (5.3)\\
& - \sum_{s \in S, (p,l) \in P_s} \sum_{e \in E} \sum_{q \in Q} \phi^{TS}_p \left(u_{spleq} + l_{spleq}\right) && (5.4)\\
& - \sum_{s \in S} \sum_{k \in L_s} \sum_{q \in Q} b_{skq} && (5.5)
\end{aligned}$$

The objective function contains terms for revenue (5.1), container handling costs (5.2), transshipment costs for laden (5.3) and empty containers (5.4) as well as for bunker consumption costs (5.5). Furthermore, term (5.1) considers depreciation costs for containers and adds fixed and penalty costs to the objective function. In comparison to the objective function of Guericke and Tierney (2015), the dependency on the selected vessel class was added to the terms (5.2) through (5.5). The fixed costs term ϕ^F contains costs that are dependent on the selected vessel class, making it a variable in our model. The bunker cost calculation in this model is simplified to the sum of all b_{skq} variables. The constraints of the model are as follows.

$$\begin{aligned}
& \sum_{\pi \in \Pi} \alpha_{n\pi} \leqslant q^{Max}_n && \forall n \in N && (5.6)\\
& \sum_{p \in P_s} \sum_{l \in \mathbb{L}_s} \tau^P_{spl} + \sum_{k \in L_s} \tau^S_{sk} - \rho^{VS}_s = \tau^V_s && \forall s \in S && (5.7)\\
& \tau^S_{sk} \leqslant t^{SMax}_{sk} && \forall s \in S \; k \in L_s && (5.8)\\
& \tau^{TT}_{spln\pi} \geqslant \frac{w_s}{2} \tau^P_{spl} + t^F_p - \mathbb{M}^P_{spl}(1 - y^T_{spln\pi}) && \forall s \in S, (p,l) \in P_s, \; n \in N, \pi \in \Pi && (5.9)\\
& \tau^{TF}_{spln\pi} \geqslant w_s \tau^P_{spl} - \mathbb{M}^P_{spl} y^T_{spln\pi} && \forall s \in S, (p,l) \in P_s, \; n \in N, \pi \in \Pi && (5.10)\\
& x_{skn\pi} \leqslant q^{Max}_n y^{CPL}_{skn\pi} && \forall s \in S, k \in L_s, \; n \in N, \pi \in \Pi && (5.11)\\
& \tau^{CP}_{skn\pi} \geqslant \tau^S_{sk} w_s - \mathbb{M}^S_{sk} y^{CPL}_{skn\pi}
\end{aligned}$$

$$+\begin{cases}\tau^{TT}_{siln\pi}, & if i \neq o_n \wedge i \neq d_n \\ \tau^{TT}_{sjl'n\pi} + \tau^{TF}_{sjl'n\pi}, & if j \neq o_n \wedge j \neq d_n \\ \frac{w_s}{2}\tau^{P}_{siln\pi}, & if i = o_n \\ \frac{w_s}{2}\tau^{P}_{sjl'n\pi}, & if j = d_n \\ 0, & otherwise\end{cases} \qquad \begin{array}{l}\forall s \in S,\\ k=(i,l,j,l') \in L_s,\\ n \in N, \pi \in \Pi\end{array} \tag{5.12}$$

$$\alpha_{n\pi} \leqslant q^{Max}_n y^{CP}_{n\pi} \qquad \forall n \in N, \pi \in \Pi \tag{5.13}$$

$$\sum_{s\in S, k\in L_s} \tau^{CP}_{skn\pi} \leqslant \theta_n + \mathbb{M}^C_n(1 - y^{CP}_{n\pi}) \qquad \forall n \in N, \pi \in \Pi \tag{5.14}$$

$$\alpha_{n\pi} \leqslant \alpha_{n\pi+1} \qquad \forall n \in N, \pi \in \{0, ..., |\Pi| - 1\} \tag{5.15}$$

Constraints (5.6) to (5.15) are unchanged compared to Guericke and Tierney (2015) as there are no direct dependencies on the vessel class selection or the determination of the vessel count. Constraints (5.6) constrains the maximum volume on all container paths of a single cargo flow. The period structure of the services is modeled in the Constraints (5.7), including the durations at sea and the durations at the ports and an upper bound for the total leg duration is given in Constraints (5.8). Constraints (5.9) and (5.10) are used to calculate the time it takes to transship a container to another service or to simply forward the cargo to the next port of the service. In Constraints (5.11), cargo flows are allowed if a particular leg is used and restrict the leg capacity to the maximum cargo flow in the planning horizon. Constraints (5.12) compute the transport duration for cargo flows for all service legs, considering transshipment, forwarding and sea durations as well as port durations. Constraints (5.13) are used to set the variable $y^{CP}_{n\pi}$ to one if a cargo flow is routed on a container path. In Constraints (5.14), the maximum transit time is used to bound the sum of all single leg durations. Constraints (5.15) is a symmetry breaking constraint for k-splittable flow problems (see Petersen (2011)).

$$x_{s\delta^+(s,k)n\pi} + \sum_{q\in Q} l_{skn\pi q} = x_{s\delta^-(s,k)n\pi} + \sum_{q\in Q} u_{skn\pi q} \qquad \begin{array}{c}\forall s \in S, k \in P_s,\\ \pi \in \Pi, n \in N\end{array} \tag{5.16}$$

$$x_{s\delta^+(s,p,l)e} + \sum_{q\in Q} l_{seplq} = x_{s\delta^-(s,p,l)e} + \sum_{q\in Q} u_{seplq} \qquad \begin{array}{l}\forall s \in S, (p,l) \in P_s,\\ e \in E\end{array} \tag{5.17}$$

$$\sum_{z\in H^N} (l_{zn\pi} - u_{zn\pi}) = \begin{cases}\alpha_{n\pi}, & if\ p = o_n \\ -\alpha_{n\pi}, & if\ p = d_n \\ 0, & otherwise\end{cases} \qquad \begin{array}{l}\forall p \in P,\\ n \in N,\\ \pi \in \Pi\end{array} \tag{5.18}$$

$$\sum_{z \in H^N} (l_{ze} - u_{ze}) = \begin{cases} -\sum_{(n,\pi) \in N_e} \alpha_{n\pi}, & \textit{if } p = o_n \\ \sum_{(n,\pi) \in N_e} \alpha_{n\pi}, & \textit{if } p = d_n \\ 0, & \textit{otherwise} \end{cases} \qquad \forall p \in P, e \in E \tag{5.19}$$

$$\sum_{r \in R_g} \sum_{l,n \in H^F} a_{rn} x_{skl} + \sum_{e \in E} \sum_{r \in R_g} a_{re} x_{ske} \leqslant \sum_{q \in Q} C_{gq} y^V_{sq} \qquad \forall s \in S, k \in L_s, g \in G \tag{5.20}$$

$$\tau^P_r = \sum_{q \in Q} \left[\left[\sum_{l \in H^F} (u_{rlq} + l_{rlq}) + \sum_{e \in E} (u_{req} + l_{req}) \right] t^E_{pq} \right] + t^{Add}_p + \tau^B_r \qquad \forall s \in S, (p,l) \in P_s, r \in S \times P_s \tag{5.21}$$

$$\epsilon y^T_{spln\pi} \leqslant \sum_{q \in Q} u_{spln\pi q} \leqslant q^{Max}_n y^T_{spln\pi} \qquad \forall s \in S, (p,l) \in P_s, n \in N, \pi \in \Pi \tag{5.22}$$

$$\epsilon y^T_{spln\pi} \leqslant \sum_{q \in Q} l_{spln\pi q} \leqslant q^{Max}_n y^T_{spln\pi} \qquad \forall s \in S, (p,l) \in P_s, n \in N, \pi \in \Pi \tag{5.23}$$

$$g^v_{kq} \tau^L_{skq} + i^v_{kq} y^V_{sq} \leqslant b_{skq} \qquad \forall s \in S, k \in L_s, q \in Q, 0 \leqslant v \leqslant \lambda \tag{5.24}$$

$$b_{skq} \leqslant \mathbb{M}^{SMax}_{kq} \qquad \forall s \in S, k \in L_s, q \in Q \tag{5.25}$$

$$\tau^L_{skq} \leqslant \mathbb{M}^S_{sk} y^V_{sq} \qquad \forall s \in S, k \in L_s, q \in Q \tag{5.26}$$

$$\phi^F = \sum_{s \in S} \left(\sum_{(p,l) \in P_s} \sum_{q \in Q} \phi^{PC}_{pq} \frac{t}{f_s} y^V_{sq} + \sum_{q \in Q} \phi^D_q \gamma_{sq} \right) \tag{5.27}$$

$$\tau_s = \sum_{q \in Q} \gamma_{sq} f_s \qquad \forall s \in S \tag{5.28}$$

$$\tau^V_s = \sum_{q \in Q} \gamma_{sq} t \qquad \forall s \in S \tag{5.29}$$

$$\sum_{q \in Q} y^V_{sq} = 1 \qquad \forall s \in S \tag{5.30}$$

$$\gamma_{sq} \leqslant \mathbb{M}^{Max}_q y^V_{sq} \qquad \forall s \in S, q \in Q \tag{5.31}$$

$$t^{SMax}_{sk} = (l_{ij} \frac{t}{f_s}) / (\sum_{q \in Q} 24 k^{Min}_q y^V_{sq}) \qquad \forall s \in S, k \in L_s \tag{5.32}$$

Constraints (5.16) through (5.23) are adjusted to mirror the extension of deployment and duration variables. Constraints (5.16) and (5.17) are used to balance the flows of laden and empty containers. Constraints (5.18) and (5.19) compute the loading and unloading of laden and empty containers. The capacity limitation on service legs is defined in Constraints (5.20). It should be noted that the capacity strongly depends on the selected vessel class and therefore needs to be considered for the calculation of the maximum capacity. Constraints (5.21) computes the port call duration. The indicator variable for transsshipments is set in Constraints (5.22) and (5.23).

Constraints (5.24) to (5.32) are added to the model to include the new linearization approach as well as the fleet deployment decisions. Constraints (5.27) perform the fixed cost calculation while Constraints (5.28) compute the round-trip time. Constraints (5.29) determine the total relevant duration and Constraints (5.32) the maximum duration of a service. Constraints (5.24) and (5.25) are used for the calculation of the bunker costs. The linearization with secants can be seen in Constraints (5.24) for a given number of secants λ. Constraints (5.25) are used to restrict the bunker consumption costs for vessel classes that are not used. There are two additional constraints that handle fleet deployment decisions. First, Constrains (5.30) only allows a single vessel class per service to be selected, and Constraints (5.31) restrict the number of vessels of a specific class to zero if the class is not selected for a service. If a vessel class is not selected for a particular service, the duration to travel a leg with that specific vessel class is set to zero in Constraints (5.26).

5.5. Computational results

Our computational analysis is based on the same instance sets that are used in Guericke and Tierney (2015), which consists of data from the public LINER-LIB database. They represent small to medium-sized service networks in three different regions. For each region, there are 30 different network variations. Table 5.2 provides detailed information about the size of these regions regarding the number of ports, number of legs, cargo flows and available vessel types. While the Baltic and WAF instances represent small feeder networks, the Mediterranean instances portray a medium sized network. Furthermore, we fixed the number of available container paths for our experiments to one. The analysis of Guericke and Tierney (2015) provides more details of the consequences if the number of container paths is increased.

The goal of our analysis is to determine to what extent integrating fleet deployment with cargo allocation can improve the overall profit of a network. For this, we evaluate our instances with dual six-core Intel Xeon X5650 2.67GHz CPUs and 32GB of RAM per instance. We use Gurobi 7.0 with a time limit of 24 hours to solve our model. To evaluate the effects of the proposed model, we fix the fleet deployment to the

Table 5.2.: LINER-LIB instance information (see Brouer et al. (2014)).

Instance	Ports	Cargo flows	Legs	Vessel types	min/max services
Baltic	12	22	132	2	1/3
WAF	20	37	380	2	5/10
Mediterranean	39	365	1482	3	1/3

Table 5.3.: Average runtime in seconds, average MIP gaps and number of solved instances for free and fixed runs.

	Optimized			Fixed		
	Avg. time	Gap	Solved	Avg. time	Gap	Solved
Baltic	0.67	0.00	30/30	0.10	0.00	30/30
WAF	3618.06	0.46	29/30	44.92	0.00	30/30
Mediterranean	86400.00	356.81	0/30	58978.33	14.14	10/30

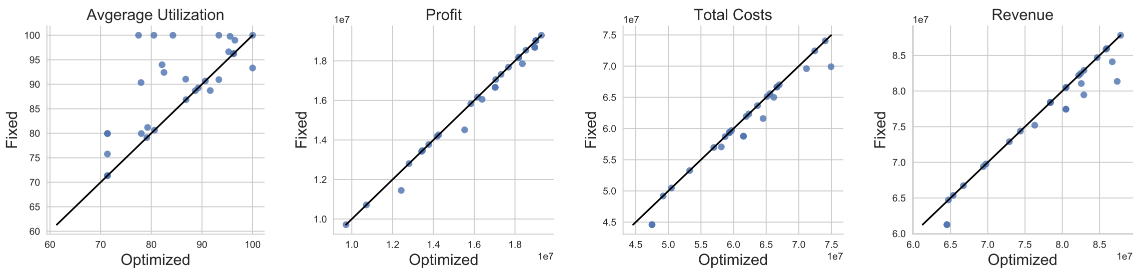


Figure 5.2.: Performance of fixed and optimized deployments with Baltic instances.

originally planned vessel assignments. In the following analysis, we refer to these fixed assignments using the term “fixed deployment”. The results from our newly proposed model are referred to as “optimized deployment”.

Table 5.3 shows the average runtime in seconds, the average MIP gaps to the optimal solution in percent, and the number of solved instances for optimized and fixed runs over all instance regions.

For the WAF region, there is a single instance for the optimized deployment that was not solved within the timelimit of 24 hours, having a gap of 13.7% to the optimal solution. The last feasible solution in this case was found about 14.5 hours before termination. In the given time frame, none of the Mediterranean instances could be solved to optimality, and only three instances had a gap of less than 20% at the time of termination.

In the fixed deployment, the same WAF instance that could not be solved in the optimized case was solved in less than 13 minutes. Also, only 7 of the 30 Mediterranean instances have an optimality gap of more than 10%.

Although feasible instances have been found in some of the Mediterranean instances, in most cases the MIP gap is too big such that a further analysis of these feasible solutions would not benefit this work. Therefore, the following analysis will only evaluate the results of the Baltic and the WAF instances.

Figure 5.2 shows scatter plots of performance indicators for fixed and optimized deployments with Baltic instances. Each point represents an instance. The leftmost scatter plot shows the average utilization of the vessels as a percentage, the other three display financial indicators in tens of million USD. The performance of the

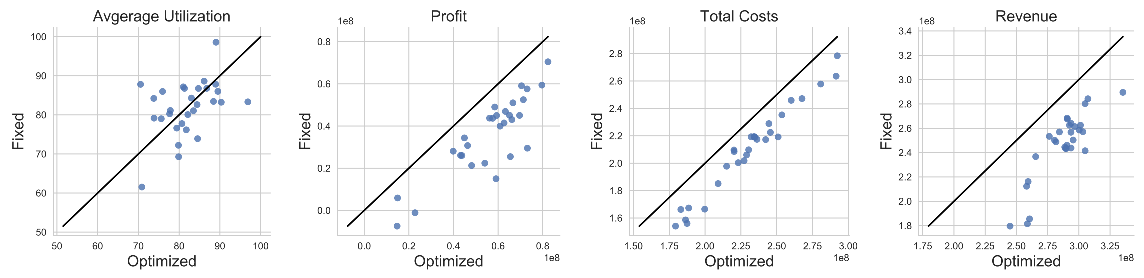


Figure 5.3.: Performance of fixed and optimized deployments with WAF instances.

optimized deployments is plotted on the x-axis and the performance of the fixed deployments is plotted on the y-axis. The diagonal line illustrates data points in which the performance of the fixed deployment is equal to the performance of the optimized deployment, while points below the line mean the optimized solution had a higher value than the fixed deployment (and a lower solution value for points above the line).

In the scatter plot showing the average utilization, only three instances have a better utilization in the optimized case for the Baltic region. By repositioning more empty containers, these instances increase the amount of used capacity of the vessels. It can be observed that some instances have a different strategy when to unload cargo, resulting in higher container path durations for their cargo. In these cases, cargo is transported on additional legs compared to the optimized case, increasing the usage of vessel capacity on these legs. In five of all the 53 services of the Baltic region (about 9.4%), the vessel class has been changed to increase the capacity of the service. In most of these cases, the utilization of these services decreased, although the total amount of transported cargo was increased in these instances, leading to higher revenues. The overall additional profit in the Baltic instances is on average about 150,000 USD and ranges from no difference at all to an increase of 1 million USD.

The scatter plots of the fixed and optimized deployment performance with WAF instances are displayed in Figure 5.3. The structure of this figure is the same as for the Baltic instances, except that the financial indicators are represented in units of 100 million USD.

About 45% of all the services (108 out of 238 services in total) show an adjustment to the deployment. Due to these deployment changes, all WAF instances are able to carry more cargo and generate higher revenues. In many cases, vessel classes with higher capacity are assigned to the services, resulting in less utilization. There are also cases in which smaller vessel classes are selected, which usually leads to higher utilization of the vessels. It can also be observed that in some instances, the number of vessels is increased to take advantage of slow steaming. Despite raising costs by assigning more vessels and transporting more cargo, the overall additional profit is on average about 21 million USD, ranging from 9 million to 46 million USD.

The evaluation of the selected performance indicators (average vessel utilization, profit, total costs and revenue), shows that by optimizing the fleet deployment the profit of a service network can be increased in many cases. Our proposed model allows for increases in the overall capacity of a service network by adding more vessels to a service or by switching vessels classes. Due to the higher capacity, more cargo can be transported.

5.6. Conclusion and future research

In this paper we integrated fleet deployment decisions into a state-of-the-art cargo allocation model. This extension combines two closely connected problems into a single model, giving liner carriers a decision support tool that enables a practically relevant analysis of their liner shipping service network. By providing the flexibility of optimizing vessel classes of a service as well as the vessel count of a service, services can be further improved regarding the profitability. To demonstrate this, we evaluated instances of two LINER-LIB regions by comparing the previously planned deployment with the results of our optimized deployment. We showed that the integrated optimization of cargo allocation and fleet deployment leads to higher numbers of transported cargo, therefore resulting in overall increases of profits of an average of 150,000 USD in the Baltic instances and an average of 21 million USD for the WAF instances.

Future research can be performed on improving the solution time of this model, as even the relatively small WAF instances take a long time to solve. For this, a column generation approach or a heuristic approach could be implemented. Furthermore, the current model includes assumptions about cargo handling and piloting times that could be relaxed in a stochastic model.

Acknowledgements

The authors thank the PC^2 at the University of Paderborn for the use of the Arminius Cluster.

Bibliography

Akyüz, M. H., Lee, C. Y.: Service level assignment and container routing for liner shipping service networks. Proceedings of the International MultiConference of Engineers and Computer Scientists (2) (2014).

Branchini, R. M., Armentano, V. A., Morabito, R.: Routing and fleet deployment in liner shipping with spot voyages. Transportation Research Part C: Emerging Technologies 57, 188–205 (2015).

Brouer, B. D., Alvarez, J. F., Plum, C. E. M., Pisinger, D., Sigurd, M.M.: A base integer programming model and benchmark suite for liner-shipping network design. Transportation Science 48(2), 281–312 (2014).

Brouer, B. D., Karsten, C.V., Pisinger, D.: Optimization in liner shipping. A Quarterly Journal of Operations Research 15(1), 1–35 (2017).

Christiansen, M., Fagerholt, K., Nygreen, B., Ronen, D.: Ship routing and scheduling in the new millennium. European Journal of Operational Research 228(3), 467–483 (2013).

Gelareh, S., Meng, Q.: A novel modeling approach for the fleet deployment problem within a short-term planning horizon. Transportation Research Part E: Logistics and Transportation Review 46(1), 76–89 (2010).

Guericke, S., Tierney, K.: Liner shipping cargo allocation with service levels and speed optimization. Transportation Research Part E: Logistics and Transportation Review 84, 40–60 (2015).

Karsten, C. V., Pisinger, D., Ropke, S., Brouer, B. D.: The time constrained multi-commodity network flow problem and its application to liner shipping network design. Transportation Research Part E: Logistics and Transportation Review 76, 122–138 (2015).

Meng, Q., Wang, T.: A chance constrained programming model for short-term liner ship fleet planning problems. Marit. Policy Manag. 37(4), 329–346 (2010).

Meng, Q., Wang, T., Wang, S.: Multi-period liner ship fleet planning with dependent uncertain container shipment demand. Marit. Policy Manag. 42(1), 43–67 (2013).

Petersen, B.: Shortest Path and Vehicle Routing, Ph.D. Thesis. DTU Management Engineering (2011).

Psaraftis, H. N., Kontovas, C. A.: Green maritime transportation: Speed and route optimization. In: Psaraftis, H.N. (ed.) Green Transportation Logistics, International Series in OR & MS, vol. 226, pp. 299–349. Springer (2016).

Reinhardt, L. B., Plum, C. E., Pisinger, D., Sigurd, M. M., Vial, G. T.: The liner shipping berth scheduling problem with transit times. Transportation Research Part E: Logistics and Transportation Review 86, 116–128 (2016).

Stopford, M.: Maritime economics. Routledge (2009).

United Nations Conference on Trade and Development: Review of maritime transport (2016).

Wang, H., Zhang, X., Wang, S.: A joint optimization model for liner container cargo assignment problem using state-augmented shipping network framework. Transportation Research Part C: Emerging Technologies 68, 425–446 (2016).

Wang, S., Meng, Q.: Liner ship fleet deployment with container transshipment operations. Transportation Research Part E: Logistics and Transportation Review 48(2), 470–484 (2012).

6. Liner Shipping Single Service Design Problem with Arrival Time Service Levels

Kevin Tierney* Jan Fabian Ehmke**
Ann Melissa Campbell*** Daniel Müller*

*University of Paderborn
Decision Support & Operations Research Lab
Warburger Straße 100, 33098 Paderborn, Germany
tierney@dsor.de, mueller@dsor.de

**University of Magdeburg
Magdeburg, Germany
jan.ehmke@ovgu.de

***University of Iowa
Department of Management Sciences
Tippie College of Business
Iowa City 52242, USA
ann-campbell@uiowa.edu

We introduce three mathematical models of increasing complexity for designing liner shipping services that guarantee the punctual arrival of vessels at a specified service level. On-time reliability is an important performance indicator for many liner carriers, but current approaches for creating new routes in liner shipping networks do not consider data-driven uncertainty. We perform an empirical analysis of vessel travel times in a real liner shipping network to develop probability distributions that we use within novel, chance-constrained mathematical models for liner shipping service design. Our models are also the first to support variable vessel speeds for service design. In our experiments, we use real-world data from 22 liner shipping routes and evaluate the designed services using a simulation procedure that demonstrates the effectiveness of our approach for reducing lateness. We show that our models can be effectively used for decision support at a tactical level not only for designing services, but also potentially for negotiating maximum demand transit times and prices with customers.

Keywords: maritime transportation, liner shipping optimization, service design, on-time guarantee

6.1. Introduction

The liner shipping industry is situated at the center of global trade, providing efficient and secure freight transportation on over 5,000 seagoing vessels (United Nations Conference on Trade and Development (UNCTAD), 2015). Each year, more and more freight is transported with standardized steel containers in liner shipping networks. In 2015, over 171 million TEU[1] of freight was carried across the oceans, representing over 1.6 billion tons of goods (United Nations Conference on Trade and Development (UNCTAD), 2015).

Liner shipping differentiates itself from other forms of maritime transportation, such as tramp or industrial shipping, due to the periodic and cyclical nature of the routes in liner shipping networks. In liner shipping, these routes, called *services*, visit a sequence of ports within specified time windows on a periodic (usually weekly) basis. Once a vessel reaches the last port in the sequence, it travels to the first port in the sequence and starts again. As it may take more than a week to finish a round-trip, it is often necessary to deploy more than one vessel to fulfill the periodicity at the ports. The periodicity and reliability of liner shipping services have become key selling points, resulting in liner shipping services forming the backbone of many modern supply chains (Notteboom and Rodrigue, 2008). However, the planning of efficient and reliable liner shipping services is challenging, as there are many sources of delay that can send a vessel off-schedule, which can have expensive repercussions throughout the supply chain (Notteboom, 2006). Sources of delays for vessels include, for example, bad weather, port congestion, equipment breakdowns, labor disputes and medical emergencies. These can cause a vessel to be late for its weekly time window or even have to cancel its stop at a particular port. Moreover, it is not only important that vessels arrive at ports when scheduled, but that the shipped containers (*demands*) reach their destination in the time period that was guaranteed by the liner carriers, as arrival uncertainty causes extra costs in the supply chain (Vernimmen et al., 2007).

Due to the periodicity and cyclical nature of liner shipping services, there are interdependencies between the expected travel times of a service, the number of vessels assigned to a service, and the resulting reliability of a service. When designing services for their networks, liner carriers often add buffer time before each port call to reduce the effect of such delays and increase the reliability of schedules. The amount of buffer is usually determined based on the experience of planners or simple rules of thumb. Liner carriers often have no analytical basis in the historical data for each port for decision making, nor is the order of the service optimized to provide a certain amount of reliability. Analytical tools are missing that allow for an exploration of costs and reliability of liner shipping services considering the specific structure of the schedules and the complex trade-offs.

[1] A single twenty foot equivalent unit (TEU) represents one twenty foot container, with two TEU representing the commonly found forty foot container.

This paper focuses on the planning of a single service within a liner shipping network that explicitly incorporates uncertainty in travel times. The ports that should be visited and the berthing windows identifying when they should be visited are given as inputs for all models. In addition to optimizing the route a service takes to ports, we also consider the number of vessels necessary for running the service in conjunction with the planned speed of the vessels. Our models provide an arrival time service level to planners in a similar fashion to the service levels in Ehmke et al. (2015) for the well known vehicle routing problem with time windows, with one key difference. In Ehmke et al. (2015), lateness accumulates over the course of each vehicle's tour. In the case of liner shipping service design, during the execution of a route, planners have a range of actions they can take to get a vessel back on schedule (see Brouer et al. (2013b)). We do not explicitly model recourse actions in our tactical model, but assume that they can be taken to avoid propagation of lateness. However, we analyze the impact that propagation can have on the reliability of a service through simulation.

We present three mathematical models of increasing complexity for planning liner shipping services to understand how different features influence the results. To establish these models, we begin with a data-driven investigation into the distribution of travel times between different kinds of ports in liner shipping (see Section 6.3). This provides an analytical background for how we model uncertainty in liner shipping. In Section 6.4, we present the mathematical models and how they are able to accommodate the derived travel time distributions. In Section 4.1, we start with analyzing the impact of different service level guarantees on the arrival time at different ports on the design of cost-minimizing services, assuming the vessels plan to travel at their design speed. This helps us quantify the impact of modeling uncertainty. In Section 4.2, we expand the first model to explicitly consider the impact of varying speed levels on the number of required vessels and the resulting arrival time service levels. Contrasting the model of Section 4.1, we can now trade a smaller number of vessels for higher speeds in service execution or vice versa. In order to provide acceptable service levels with regard to port time windows, our first two models often create large buffers in the services' schedules. Thus, they ignore delivery time guarantees on freight. Our third model, in Section 4.3, alleviates these limitations and helps us understand how different levels of delivery time guarantees can impact the amount of freight that can be carried when there are limitations on maximum container transit times.

In Section 5, we experiment with the use of the presented mathematical models in a series of computational experiments. To this end, we apply the presented travel time distributions to instances based on data from the well-known LINERLIB (Brouer et al., 2013a) and compute the optimal solutions for the different models. Considering different service levels through chance constraints in our optimization models, we analyze the trade-off between a larger fleet of vessels and adaptations of speed levels from a tactical network design perspective. Then, we assess the operational performance of these optimized services with a discrete-event simulation. The simulation evaluates the optimized services by imitating individual service runs to determine the "actual" realized cost and quality of a service, analysing the impact of propagation of lateness in the course of a round-trip. We provide a set of managerial insights based on these results in Section 6.

In summary, our paper provides the following contributions:

1. a data-driven investigation of distributions of travel time for liner shipping services,

2. the modeling of service guarantees on arrival times through chance constraints for service design,

3. a model for service routing optimization with variable vessel speed (tactical perspective),

4. and a simulation study to evaluate the effectiveness of the models (operational perspective) and the impact of lateness propragation when executing the services.

In the following section, we delineate our modeling approach from related literature in this field.

6.2. Literature Review

The liner shipping service design problem with arrival time service levels has connections to a number of different problem domains, including the areas of vehicle routing, tramp shipping, and liner shipping. We now discuss the similarities and differences of this work with related work in the literature, referring to Christiansen et al. (2013) and Christiansen et al. (2004) for an overview of the entire area of maritime transportation and Brouer et al. (2017) for an overview of liner shipping optimization.

6.2.1. Liner Shipping

We divide our discussion of related work within the area of liner shipping into two subsections. First, we discuss related problems such as network design and fleet deployment and then move into work addressing single service scheduling and routing. Given the prevalence of delays in maritime applications due to storms, labor disputes, and breakdowns, considering uncertainty is a natural extension to operations research models for tramp, industrial and liner shipping. Nonetheless, there is little literature considering stochastic elements in these problems.

Related Liner Shipping Problems

Liner shipping single service design can be considered a subproblem of the overall network design problem and is considered a tactical problem in Meng et al. (2013). The network design problem is very difficult to solve with exact solution approaches (Agarwal and Ergun, 2008, Álvarez, 2009, Brouer et al., 2013a) and even for heuristics (Brouer et al., 2014). Due to its difficulty, network design problems usually leave out or abstract a number of important side constraints that we are able to include, such as the handling of container transit times. We note that to the best of our knowledge, there has not been any work on combining uncertainty with liner shipping network design models.

Fleet deployment models are also related (see, e.g., Powell and Perakis (1997)), as these assign a heterogeneous fleet of vessels to a set of services. In contrast to these models, we do not try to size the vessel to the service we are designing. In our approach, the vessel class is an input parameter. Fleet deployment models are subject to different types of uncertainty than service/network design problems. A key source of uncertainty for these types of problems is the amount of demand that is shipped each week on each service. A chance-constrained model takes this into consideration in Meng and Wang (2010), finding a minimal cost allocation of vessels to services. We do not take demand uncertainty into account in our model, but this would be a logical extension of our work. Transit time restrictions for demands were recently considered in terms of the cargo allocation problem, which assigns containers to routes in a network in Guericke and Tierney (2015), as well as for a time-constrained multicommodity flow problem in Karsten et al. (2015). We solve a simplified version of this problem as a part of our single service design. We do not consider transshipments, making our cargo allocation easier in comparison.

Single Service Scheduling

Several papers determine vessel speeds and schedules for one (or more) service(s) under uncertainty given a pre-defined port sequence. The key difference between these works and our model is that we determine both the port sequence and the schedule. Delays at ports are considered in Qi and Song (2012), in which delayed arrivals are penalized in the objective function, reflecting a potential loss of goodwill. The authors discuss how to compute the service level at each customer, acknowledging that prior delays can accumulate. Furthermore, the paper focuses on special cases, such as with 100% service levels for all customers. A drawback of this work is that the authors do not use real data for their port time distributions, instead assuming uniform and normal distributions. In a parallel work, Wang and Meng (2012a), the authors also seek vessel speeds and a schedule under uncertainty for a new liner shipping service. They create a mixed-integer non-linear stochastic programming model for the problem that minimizes ship and fuel costs while maintaining a given service level. The problem is solved using a cutting-plane algorithm.

The most relevant work in the area of liner shipping in terms of focusing on delays is Lee et al. (2015). The paper assumes the routes are given and looks at the impact of different steaming speeds for executing the route, given the stochastic nature of port

Article	Routing	Lateness dist.	Speed opt.	Method
Wang and Meng (2012a)	✗	Any Truncated	Non-linear	Non-lin. stoch. prog.
Wang and Meng (2012b)	✗	Uniform & Normal	Non-linear	Non-lin. stoch. prog.
Qi and Song (2012)	✗	Uniform/Normal	Non-linear	Sim. stoch. approx.
Song and Dong (2013)	✓	✗	Non-linear	Heuristic Decomposition
Plum et al. (2014)	✓	✗	✗	Branch-Cut-and-Price
Lee et al. (2015)	✗	Normal/Any	Non-linear	Markov chains
Song et al. (2015)	✗	Trunc. Normal	Non-linear	NSGA-II Deb et al. (2002)
Reinhardt et al. (2016)	✗	✗	Disc. secants	MILP
Wang and Wang (2016)	✗	✗	Non-linear	Polynomial time algorithm
Santini et al. (2017)	✓	✗	Disc. graph	Branch and price
This paper	✓	log-logistic 3P	Disc. secants	MILP

Table 6.1.: A categorization of related work within the area of liner shipping service/schedule design.

operations. Buffers are planned into liner shipping schedules in order to maintain a specified arrival service level. In contrast, not only do we perform route planning, we also do not limit ourselves to only considering port delay. While port delays make up a large percentage of the sources of delay (Notteboom, 2006), a number of other sources exist that we are able to account for in our model.

A feeder network design problem is presented in Santini et al. (2017) that routes and schedules several services within a geographically limited area. The authors use an expanded time space graph so that vessel speeds can be precomputed and assigned as costs directly to the arcs. The number of vessels used on the routes if fixed to a value K, meaning the set of speeds for vessels and amount of buffer that can be inserted into the schedule, is limited. Thus, Santini et al. (2017) does not consider arrival time service levels.

The paper that motivates the basic form of our model is Plum et al. (2014). The goal in this work is to design a single service given a set of ports that must be visited. A set of container demands between ports must be satisfied while taking into account the capacity restrictions of the vessels. Our first deterministic model (Phase 1) resembles this work very closely, adding only the optimization of the number of ships on the service. Furthermore, our models extend the work of Plum et al. (2014) with variable vessel speed and arrival time guarantees, which represent important characteristics for the liner shipping industry. A summary of the features of our models in relation to the literature is given in Table 6.1.

6.2.2. Vehicle Routing with Time Windows

The well-known vehicle routing problem with time windows deals with the efficient utilization of a fleet of vehicles to deliver goods in a transportation network (Kolen et al., 1987). In particular, a set of customers is assigned to a fleet of vehicles such that the number of vehicles is minimal, the order of customer visits is cost optimal, and every customer is visited exactly once within its time window. Ehmke et al. (2015) is among the most recent and related papers; they explicitly consider the interplay of stochastic travel times and time windows, ensuring service levels in routing through chance constraints. For them, the largest challenge stems from the computation of the combined arrival time distribution along a route, which we ignore in our approach since we assume statistical independence between the individual sequences on a route. Contrasting with vehicle routing approaches, liner shipping service design involves a number of constraints usually not present in vehicle routing, such as a maximum container transit time, the selection of feasible demand, and the consideration of varying commodities.

6.2.3. Tramp and Industrial Shipping

Tramp and industrial shipping involve the transportation of bulk or liquefied goods (but rarely containers) in a vehicle routing-like fashion. Tramp/industrial shipping problems differ greatly from liner shipping problems in terms of the schedule structure. Whereas liner shipping has a periodic, fixed schedule like a public bus network, tramp/industrial shipping more resembles taxis, in which ships sail wherever there is a demand to be satisfied. However, despite their differences, all of these types of shipping involve vessels that can be delayed in the same way, as well as have similar cost profiles for sailing. Speed optimization has become a standard feature of tramp shipping models (Norstad et al., 2011).

The most recent work in this area is Agra et al. (2015), which considers a maritime inventory routing problem for liquefied natural gas with stochastic sailing and port times. The authors determine routes and the quantity of gas to load/unload a priori and model the problem as a stochastic program with recourse using scenarios. In contrast, we compute distributions over the arrival times of vessels at ports. In Agra et al. (2016), this work is extended to use a log-logistic distribution to model delays as in our work. Uncertain sailing times are considered in Halvorsen-Weare et al. (2013), motivated by a real energy company, Statoil, as well as uncertain production rates for a natural gas provider. In Halvorsen-Weare et al. (2013), the same problem is solved as in Halvorsen-Weare and Fagerholt (2013), but with robustness strategies added. The sailing times are fitted to a log-logistic probability distribution, based on information gathered for a gas tanker in Kauczynski (1994).

A method for creating replenishment schedules for offshore installations is proposed in Halvorsen-Weare and Fagerholt (2011). The paper defines four weather states and their impact on sailing speed and service time in ports at an offshore delivery location. A key difference with our work is that the authors assume a set of routes already exist, and they must choose which routes to use.

6.3. Data Exploration

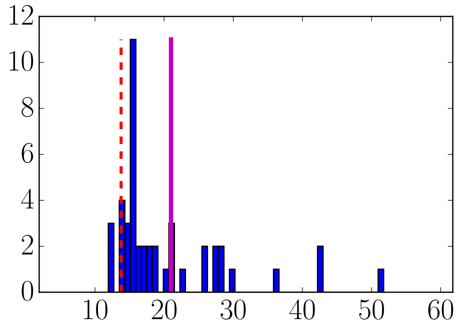
A key challenge that arises in guaranteeing arrival time service levels for a liner shipping service is determining what distribution underlies the travel time of the vessels. As discussed in the previous section, the maritime literature has several suggestions on appropriate travel time distributions, including the log-logistic distribution (Halvorsen-Weare et al., 2013). However, it is not clear a priori that this distribution is the best fit for travel times of liner shipping services, and furthermore, it is not clear that this distribution is appropriate for trips between all ports worldwide. To this end, we gathered data from operating liner shipping service routes, including profiles of the vessels used and the actual transit times of the vessels between ports. We first describe the data we use for distribution fitting in more detail, including a frank discussion of the strengths and weaknesses of our dataset, and finish with a presentation of the distributions we found.

6.3.1. Data Sources and Integration

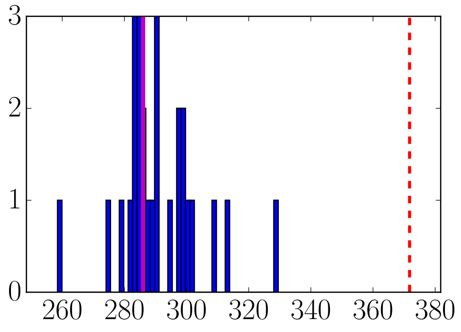
We have taken care to make our dataset as realistic as possible. To do this, we use service information from the liner carrier COSCO and combine this with AIS (positioning) data from MarineTraffic regarding the vessels on 25 selected liner shipping services. We note that we have no relationship with COSCO and use data regarding their network because they offer detailed arrival time information for their services. We gathered positioning data from the vessels' transponders for the year 2014 and use this data to check whether the vessels are on time or not by comparing it with the planned schedule from COSCO (COSCO Shipping Lines, 2017). In total, our dataset contains 40 ports, 118 vessels, 125 port to port connections, and records for 1872 transits between ports. This provides us with a list of transits between the ports visited by the 25 services and an indication of how late (or early) the vessel was as compared to the planned travel time.

Figure 6.1 shows three sample transits and their associated histograms, chosen because they represent a range of connection types: an intra-region connection ((a)), a Pacific Ocean connection ((b)) and an Indian Ocean connection ((c)). While it is not possible to generalize from the data of only a few connections, it is clear that even on short connections, such as crossing the English channel, significant delays above the scheduled sailing time are possible. Furthermore, we can see that some schedules are planned with enough buffer that even delays of a couple of days do not cause lateness, as in the case of the backhaul from Long Beach to Ningbo. Finally, some schedules are tightly planned, as in the case of sailing from Singapore to the southern side of the Suez canal. Here, significant delays of up to almost five days can occur. We note that it is not unusual for connections to be assigned a speed faster than the vessel's design speed as in Figures 6.1a and 6.1c. These are likely connections where shippers wish to have fast transit times and the carrier must meet these requests to carry shipper's cargo.

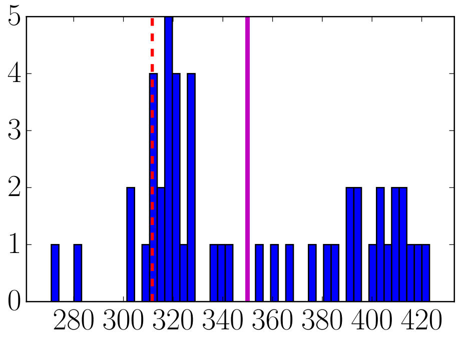
The data we use from the AIS transponders is the best data we can obtain, as carriers we have spoken to do not keep track of more accurate statistics regarding travel time. There are, however, weaknesses that we want to outline. An obvious



(a) Rotterdam, NL to Le Havre, FR



(b) Long Beach, US to Ningbo, CN



(c) Singapore, SG to Suez, EG

Figure 6.1.: Histograms of travel times (in hours) of several port to port pairs. The x-axis shows the varying travel times in hours between the ports. The y-axis gives the frequency of a particular travel time. The scheduled transit time is shown with a dashed red line and the travel time at the vessel's design speed with a solid magenta line.

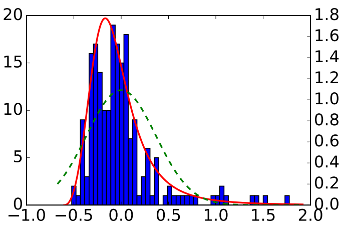
issue is that we do not know what recourse actions were taken to avoid or reduce lateness during operations. In particular, if a vessel is running late, a carrier may instruct the captain to speed up to stay on schedule. We note that AIS data does include the speed of the vessel, but even with the speed, we cannot assess whether a speed-up was part of the original service schedule or not. Another source of error in our data is that extremely late vessels may simply skip port calls to save time, meaning we do not end up knowing how late they actually were. Furthermore, if no vessel performs a particular port to port transit, we will have no data for it, and this makes it difficult to know what distribution to use to model travel time for such a connection when considering the creation of a new service. We attempt to counteract some of the weaknesses in our approach by clustering the port to port connections and by computing distributions between regions rather than individual ports. This alleviates the problem for many connections.

6.3.2. Distribution Fitting

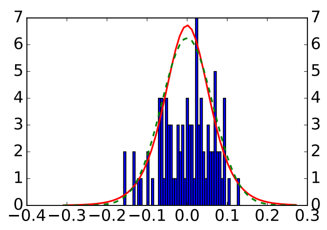
We compute distributions based on intra and inter-region transit times, normalized according to the average travel time for each region-to-region pair. To create distributions for any pair of ports worldwide, we aggregate and generalize the operational data as follows. First, to generalize the port-to-port observations, we follow a common idea in the network design literature (e.g. Mulder and Dekker (2014)) and cluster ports into 16 regions with the well known k-means algorithm, adjusting the ports between some regions slightly by hand. The resulting clusters provide the input for our distribution fitting. Second, between two regions or within a single region, for each pair of ports, we normalize the travel times based on the mean empirical travel time between (or within) the regions containing the ports. That is, given ports i and j from regions r and s, we compute the average travel time between the regions and divide the travel time between i and j by this value. Since not every pair of regions has sufficient data to allow for an empirical fit of a distribution, we also compute a distribution across all data that can be used for such port pairs.

We use the software EasyFit (MathWave Technologies, 2016) to generate a list of distributions for each inter and intra region pair to narrow down the type of distribution to use across our data. In 17 out of 28 pairs with enough port to port transfers to fit a distribution, the three-parameter log-logistic (ll3p) distribution is one of the top three best fitting distributions, and in all other cases was still one of the best fitting distributions. Other good fits included the Cauchy distribution and the four-parameter Burr distribution. Given the previous use of the ll3p distribution in the literature, we select it for the remainder of this work.

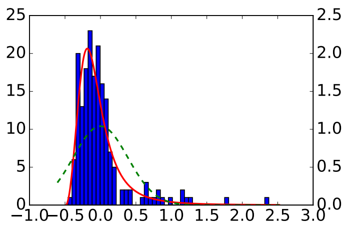
Figure 6.2 shows histograms and the best fitting normal and log-logistic probability distribution functions for several region-to-region pairs. The x-axis provides the normalized travel time, with the y-axis reflecting the frequency of a particular value. There is a clear trend across all of the regions we consider. The normal distribution has less probability mass than the ll3p distribution around on-time transits at time 0, which is a good indicator of why the log-logistic distribution is often the one with the best fit. In many cases, the log-logistic distribution puts more mass near on-time arrivals and is skewed to the right, reflecting a large number of vessels arrive late and across a wide range of values. For a few region-to-region pairs, the log-logistic distribution looks almost like the normal distribution, as is the case in Figure 6.2b, and this further emphasizes the importance of fitting specific parameters for different parts of the world, instead of only one set of parameters.



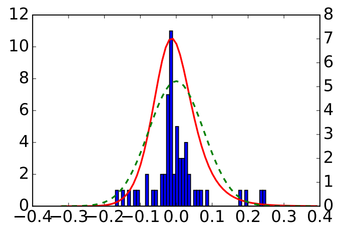
(a) Intra Europe



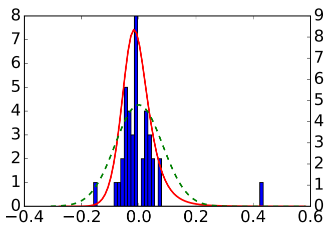
(b) China/Japan to Southern Mediterranean



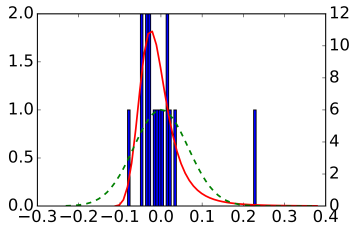
(c) Intra Western North America



(d) Western North America to China/Japan



(e) Central/South America to North America



(f) Indian Ocean to Southern Mediterranean

Figure 6.2.: Histogram and probability distribution functions for several region-to-region pairs. Blue bars show the histogram, a solid red line shows the fit of the ll3p distribution, and the dashed green line the fit of a normal distribution.

6.4. Mathematical Models

We construct an arc flow model that extends the one presented in Plum et al. (2014) to find a cyclical route through ports given fixed port time windows. We decompose the modeling of the extended problem into three phases, each one with added complexity. This helps us understand the impact of different problem features and also their impact on the solution time. Unlike in Plum et al. (2014), all of our models contain chance constraints requiring vessels arrive at ports on-time with a specified arrival time service level. In the first phase, we address the basic liner shipping service design problem, requiring all demand to be transported without the maximum transit times on vessels sailing at their design speeds used in Plum et al. (2014). The design speed is used to provide schedules that can serve as a reasonable baseline for Phase 2 and Phase 3. In the second phase, we relax the fixed sailing speed requirement and allow the vessel to vary its speed, taking this into account in the objective function. In the third phase, we impose maximum container transit times, but allow demand to be rejected if it is not possible to meet the transit time limitations. The objective function in this phase switches to profit maximization from cost minimization in Phases 1 and 2. As mentioned in Section 6.1, our models do not consider propagation of lateness over the course of the service. This is because route planners have a range of actions they can take to get a vessel back on schedule (see Brouer et al. (2013b)) if needed, including skipping ports, negotiating new time windows, etc. We do not explicitly model these recourse actions in our tactical model, because we do not know which ones could be negotiated in each case. We assume, though, that such actions can be taken to avoid propagation of lateness, but also create expenses that a company would want to avoid if possible through on-time arrivals.

6.4.1. Phase 1: Design Speed Model

In Phase 1, we impose the following assumptions:

1. All ports are assigned a fixed port time window per week in which they are called. The vessel must arrive before the port time window starts (planned arrival) and may not leave until the window ends (planned departure).
2. Vessels must wait until the start of their time window if they arrive early at a port.
3. Time windows must be satisfied with a confidence level of α (arrival time service level). For example, α may be a value such as 75% or 90%.
4. Travel times between ports are statistically independent.
5. All vessels must plan on traveling at their design speed between ports.
6. All pickups and deliveries of demand must be served, assuming that the vessel capacities are sufficiently large.
7. Forty foot containers are broken down into two twenty foot containers, and their flow can be modeled in a continuous fashion.
8. All ports must be called exactly once.
9. The objective is to minimize the cost of the single route and the number of vessels required.
10. The service frequency is weekly.

Note that the requirement that all ports must be called exactly once means we cannot design butterfly or conveyor belt style routes. We note our model can be adjusted to create such routes and in its current form can support having a port specified multiple times. According to Song and Dong (2013), cycle routes (i.e., those where all ports are visited once) are the most common topological structure and can be found in about 45% of services. We assume that one of the ports is arbitrarily identified as the start port for the service, and it is also the end port since the route is cyclical. The starting port is assumed, without loss of generality, to be a port that has a time window that is fully contained during the week, i.e., the time window does not intersect with Monday at 12:00 AM.

We now define the mixed-integer linear program for the Phase 1 model.

Parameters

$P = \{1, \dots, p, p+1\}$	Set of ports to call; ports 1 and $p+1$ represent the initial port (and its return)
A	Set of arcs (i, j); we assume that the network is complete, with no arcs into the initial port or out of the ending port
A'	All of the arcs of A with the addition of a single arc connecting the ending port to the initial port
K	Set of demands
o_k	Origin of demand $k \in K$
d_k	Destination of demand $k \in K$
a_k	Amount of cargo of demand $k \in K$
u	Capacity of a vessel in TEU
t_i^s, t_i^e	Time window start and end (respectively) within a week for port $i \in P$, where time 0 is Monday at 12:00 AM. The values for the time window start and end time represent hours. The latest arrival time has a domain $t_i^s \in \{0, \dots, 167\}$, whereas the earliest departure time $t_i^e \in \{t_i^s, \dots, 335\}$. This allows port time windows to start and end in consecutive weeks.
c_{ij}^f	Fixed sailing cost for arc $(i, j) \in A$
t_{ij}^α	Time for sailing arc $(i, j) \in A$ at the vessel's design speed to provide a service level of α
c^v	Charter cost per vessel per week

Variables

w_i	Service week at port $i \in P$, where $w_1 = 0$
x_{ij}	Indicates whether arc $(i, j) \in A$ is included in the service
τ_i^s	Start service time of a vessel at port $i \in P$ (hours from beginning of service)
τ_i^e	End service time of a vessel at port $i \in P$ (hours from beginning of service)
τ_i^a	Arrival time at port $i \in P$ (hours from beginning of service)
f_{kij}	Flow of demand $k \in K$ on arc $(i, j) \in A$

Objective and constraints

$$\min\ c^v w_{p+1} + \sum_{(i,j)\in A} c^f_{ij} x_{ij} \tag{6.1}$$

$$
\begin{aligned}
\text{subject to } \sum_{(i,j)\in A} x_{ij} &= 1 && \forall j \in P\backslash\{1\} && (6.2)\\
\sum_{(i,j)\in A} x_{ij} &= 1 && \forall i \in P\backslash\{p+1\} && (6.3)\\
\tau^s_i &= t^s_i + 168w_i && \forall i \in P && (6.4)\\
\tau^e_i &= t^e_i + 168w_i && \forall i \in P && (6.5)\\
\tau^a_i &\leqslant \tau^s_i && \forall i \in P && (6.6)\\
\tau^e_i + t^\alpha_{ij} &\leqslant \tau^a_j + M(1 - x_{ij}) && \forall (i,j) \in A && (6.7)\\
w_1 &= 0 && && (6.8)\\
\sum_{j\in P\backslash\{o_k\}} f_{k o_k j} &= a_k && \forall k \in K && (6.9)\\
\sum_{j\in P\backslash\{d_k\}} f_{k j d_k} &= a_k && \forall k \in K && (6.10)\\
\sum_{(j,i)\in A'} f_{kji} &= \sum_{(i,j)\in A'} f_{kij} && \forall k \in K, i \in P\backslash\{o_k, d_k\} && (6.11)\\
\sum_{k\in K} f_{kij} &\leqslant u x_{ij} && \forall (i,j) \in A' && (6.12)\\
x_{p+1,1} &= 1 && && (6.13)\\
x_{ij} &\in \{0,1\} && \forall (i,j) \in A' && (6.14)\\
\tau^a_i, \tau^s_i, \tau^e_i &\geqslant 0 && \forall i \in P && (6.15)\\
f_{kij} &\geqslant 0 && \forall (i,j) \in A', k \in K && (6.16)\\
w_i &\in \mathbb{Z}^+ && \forall i \in P\backslash\{1\} && (6.17)
\end{aligned}
$$

The objective function (6.1) includes the cost for vessels and the sailing cost for the complete service. The variable w_{p+1} contains the week of the last port visit. Since we are assuming a weekly service frequency, the last week also specifies the number of vessels. Constraints (6.2) and (6.3) require that the service enters and leaves each port exactly once, except for the initial and ending port. The time the vessel starts and ends its visit at each port is determined in Constraints (6.4) and (6.5). The term $168w_i$ converts the week selected for customer i to the appropriate hour from the start of service. Constraints (6.6) restrict the arrival time to be before the start of the visit at a port.

We use a chance constraint to ensure that the service route has enough buffer at each port to achieve the level of punctuality requested. Constraints (6.7) model a general constraint that can be used with any probability distribution in which the inverse CDF can be computed. These constraints force the arrival time at the next port to always be greater than the departure time at the previous port plus the travel time with an adequate buffer. We compute a bound for M as follows. First, we compute a maximum path length (worst case tour) traveling salesman problem (TSP) through all of the ports[2]. Given this "slow" TSP path, we then add extra buffer based on our service level formulas to the path if required to reach a particular service level, and further add buffer that considers when the vessel arrives at a port and how long it must wait until the port is available.

To compute the value of t_{ij}^{α}, consider, for example, the case of the normal distribution where we want to guarantee a confidence level of α that we arrive on time. Assume we have a parameter z^{α} that provides the number of standard deviations associated with a service level of α. We can then set t_{ij}^{α} to be the mean travel time between i and j plus z^{α} times the standard deviation. Consider now the case of a non-normal distribution such as the three-parameter log-logistic distribution. Given the scale parameter σ, shape parameter ξ and location parameter μ, we want to find the value of t_{ij}^{α} providing the minimum required travel time between i and j enabling a service level α. We can compute this from the inverse CDF as follows:

$$F^{-1}(\alpha;\sigma,\xi,\mu) = \left(\frac{1}{\alpha}-1\right)^{-1/\sigma}\left(\xi+\mu\left(\frac{1}{\alpha}-1\right)^{1/\sigma}\right).$$

If we are interested in comparing the results with a deterministic model, as we do in our experiments, we can simply set t_{ij}^{α} to a fixed value t_{ij} representing the travel time at the design speed with no added buffer.

The visit to the initial port is specified to be in the initial week by Constraints (6.8). Constraints (6.9), (6.10), and (6.11) are used to handle the flow of demands between their supply and destination points as well as between transhipment points. These are modeled in a similar way to Plum et al. (2014), but adapted to require that all demand is served. The capacity of the vessel is enforced by Constraints (6.12) on each arc, but only if a vessel is sailing on the arc. Should an arc not be used in the vessel path, the flow capacity is set to zero. Constraint (6.13) connects the last port to the first port, ensuring that the service's route is complete. Lastly, Constraints (6.14), (6.15), (6.16), and (6.17) define the variables appropriately.

[2]First, we note that this version of the TSP is NP-complete, but since our problems do not contain many ports, it is easy to solve. Second, in later phases when speed can be adjusted we use the slowest vessel speed allowed.

6.4.2. Phase 2: Optimized Speed Model

While the Phase 1 model covers a number of basic properties of liner shipping service design, in practice, planners can adjust the speed of the vessels to reduce sailing costs or increase the probability of meeting tight deadlines. Hence, we adjust the fourth assumption to the following:

4. Vessels may sail between their minimum and maximum speed, and the cost of doing so will be reflected in the objective function.

We implement variable vessel speeds with several new sets of variables and parameters. Since the distance between all ports is known in advance, the model can simply decide the sailing duration for each leg of the service and the speed is known (duration / distance). The duration is used in a piecewise linearization to determine the approximate bunker costs. Three new sets of variables are required:

γ_{ij}	The cost of sailing between i and j for arc $(i,j) \in A$
ρ_{ij}	The duration for sailing between ports i and j for arc $(i,j) \in A$
β^{α}_{ij}	The amount of buffer added into the schedule between ports i and j for arc $(i,j) \in A$

We also need to add further parameters:

δ	The speed of a vessel
δ^*	The design speed of a vessel
c^B	Cost per ton of bunker fuel
ϕ^g_{ij}	Slope of the secant of bunker consumption cost function approximation
ω^g_{ij}	y-intercept of the secant of the bunker consumption cost function approximation
t_{ij}	The sailing time at the vessel's design speed between i and j for arc $(i,j) \in A$
$t^{Min}_{ij}, t^{Max}_{ij}$	The minimum or maximum sailing time between i and j for arc $(i,j) \in A$

Since the fuel consumption function of vessels is roughly cubic (Brouer et al., 2013a), we use a piecewise linearization of the function in our Phase 2 model. One approximation for the non-linear bunker consumption function (Brouer et al., 2013a) is given by

$$B(\delta) = \left(\frac{\delta}{\delta^*}\right)^3 B(\delta^*),$$

where δ is the vessel's speed, δ^* is the vessel's design speed and $B(\delta^*)$ is the bunker consumption in tons of fuel per hour at the design speed. We use the secant approximation presented in Reinhardt et al. (2016) to linearize this function and note that, due to the convexity of the function, no binary variables are necessary. We now show their approximation, adjusting the notation to fit our model. First, we modify the bunker consumption function to accept a travel duration between two ports with

$$\hat{B}(\rho_{ij}) = \left(\frac{t_{ij}}{\rho_{ij}}\right)^3 B(\delta^*)c^B,$$

where ρ_{ij} is the travel duration, t_{ij} is the sailing time at the vessel's design speed as previously defined, and c^B is the cost per ton of bunker fuel. This function can be approximated with a given number of secants, θ. Each secant g is defined with the function

$$\hat{c}_{ij}(\rho_{ij}) = \phi^g_{ij}\rho_{ij} + \omega^g_{ij},$$

where ϕ^g_{ij} is the slope of the secant and ω^g_{ij} is the y-intercept. Then, we can add the following linear constraints to our model for computing the sailing cost:

$$\gamma_{ij} \geqslant \phi^g_{ij}\rho_{ij} + \omega^g_{ij}x_{ij} \qquad \forall (i,j) \in A, 0 \leqslant g < \theta. \tag{6.18}$$

Since the variables x_{ij} indicate whether an arc is being used or not, we ensure that the sailing costs on the arc are only constrained when the arc is being used. Furthermore, we replace the objective function (6.1) with the following:

$$\min\ c^v w_{p+1} + \sum_{(i,j)\in A} \gamma_{ij}. \tag{6.19}$$

To handle the arrival time service levels with varying speeds, we modify Constraints (6.7) to include the decision variable ρ_{ij}:

$$\tau^e_i + \rho_{ij} + \beta^\alpha_{ij} \leqslant \tau^a_j + M(1 - x_{ij}) \qquad \forall (i,j) \in A. \tag{6.20}$$

In other words, the travel time from Phase 1 is replaced with the variable specifying the travel duration on the arc, if the arc is used. We then require constraints ensuring that the minimum duration according to the chosen service level is enforced:

$$\rho_{ij} + \beta^\alpha_{ij} \geqslant t^\alpha_{ij}x_{ij} \qquad \forall (i,j) \in A. \tag{6.21}$$

For a deterministic model, the term β^α_{ij} can be set to zero. We then constrain the minimum and maximum duration of the voyage using the parameters t^{Min}_{ij} and t^{Max}_{ij} with the following constraint:

$$t^{Min}_{ij}x_{ij} \leqslant \rho_{ij} \leqslant t^{Max}_{ij}x_{ij} \qquad \forall (i,j) \in A. \tag{6.22}$$

These constraints ensure that vessels do not sail faster or slower than is allowed should the arc be chosen. Finally, we impose bounds on the travel time, travel costs, and buffer variables:

$$\rho_{ij} \geqslant 0, \gamma_{ij} \geqslant 0, \beta^{\alpha}_{ij} \geqslant 0 \qquad \forall (i,j) \in A. \tag{6.23}$$

6.4.3. Phase 3: Optimized Speed with Maximum Transit Times Model

For our Phase 3 model, we require that demands are not carried for longer than their maximum transit time. Requiring that all demand be served in combination with transit times limitations can lead to infeasibility, especially when a high level of on-time arrival is desired. A further assumption of our Phase 3 model is that the demands at a port are the same each week. This is, of course, not the case in practice, but it is sufficient for tactical level planning. In Phase 1 and 2, we wanted to see the impact of service levels and speed optimization on serving the same demands, so we relaxed this transit time requirement. Here, to enforce the maximum transit times, we will relax the fifth assumption of Phase 1, namely that all demands must be carried. We now have the following assumptions in addition to the Phase 2 assumptions (minus assumption 5 from Phase 1):

10. All demands have a maximum transit time. This is a common assumption in practice.

11. Demands can be rejected if their maximum transit times cannot be met.

12. The objective is to maximize the profit, i.e. the revenue from carrying cargo minus the cost of sailing and cost of vessels.

These changes require two new variables:

y_k	This equals 1 if demand k is transported or 0 if it is not transported for $k \in K$
δ_k	This equals 0 when the service start time of the destination is greater than the end service time of the cargo source for demand k for $k \in K$. However, due to the cyclical structure of liner shipping routes, it is possible that the start service time of the destination is actually less than the end service time of the source if the path of the cargo crosses the (arbitrary) “end” of the service. When this occurs, this variable will be 1.

We also need to add further parameters:

r_k	The revenue of delivering a demand k
l_k	The maximum transit time of a demand k

For our new profit maximizing objective, we add the revenue of delivering a demand, r_k, to the objective function, and subtract the vessel and sailing costs from it:

$$\max \sum_{k \in K} r_k y_k - c^v w_{p+1} - \sum_{(i,j) \in A} \gamma_{ij}. \tag{6.24}$$

We next replace Constraints (6.9) and (6.10), which control the start and end of the flow of demand k, with the following:

$$\sum_{j \in P \setminus \{o_k\}} f_{k o_k j} = a_k y_k \qquad \forall k \in K \tag{6.25}$$

$$\sum_{j \in P \setminus \{d_k\}} f_{k j d_k} = a_k y_k \qquad \forall k \in K. \tag{6.26}$$

Constraints (6.25) and (6.26) set the amount of containers to be carried for a particular demand to 0 when y_k is 0, ensuring that the demand is either taken in its entirety or not at all.
Given the maximum transit time of a demand k, l_k, we use the following constraints to restrict the transit time:

$$\tau^s_{d_k} \geqslant \tau^e_{o_k} - M\delta_k \qquad \forall k \in K \tag{6.27}$$

$$\tau^s_{o_k} \geqslant \tau^e_{d_k} - M(1-\delta_k) \qquad \forall k \in K \tag{6.28}$$

$$\tau^s_{d_k} - \tau^e_{o_k} \leqslant l_k + M\delta_k + M(1-y_k) \qquad \forall k \in K \tag{6.29}$$

$$\tau^s_{d_k} - \tau^e_{o_k} + 168 w_{p+1} \leqslant l_k + M(1-\delta_k) + M(1-y_k) \qquad \forall k \in K \tag{6.30}$$

Constraints (6.27) and (6.28) set the value of δ_k depending on the schedule of the vessel. When $\delta_k = 0$ (the "normal" case), the destination of demand k is scheduled later than the origin. However, when $\delta_k = 1$, the destination comes after the "end" of the service, meaning the scheduled time of the origin is actually later than the destination in the model. Constraints (6.29) and (6.30) set the transit time limitation depending on the value of δ_k. We then add one more set of constraints to the model to connect the flow of each demand to its corresponding 0/1 variable:

$$f_{kij} \leqslant a_k y_k \qquad \forall k \in K, (i,j) \in A. \tag{6.31}$$

Finally, we add bounds constraints for the new variables:

$$y_k \in \{0,1\} \qquad \forall k \in K \tag{6.32}$$

$$\delta_k \in \{0,1\} \qquad \forall k \in K. \tag{6.33}$$

6.5. Computational Experiments

We will first describe how we create our instances and experimental design in Section 6.5.1. For our Phase 1 and 2 models, we simulate operations of the created schedules to evaluate their performance with regard to cost and service quality. We describe our simulation in Section 6.5.2. Then, we present our results of tactical planning for the three models in Section 6.5.3. Finally in Section 6.5.4, we present sample schedules generated by our different models.

6.5.1. Instances

For our experiments, we want to compare the results from deterministic and stochastic variants (with different service levels) of each model across a variety of instances. This will help us understand the impact of different service levels, the ability to optimize speed in the planning phase, and the impact of the maximum transit times. We also want to understand how the underlying features of the created services, such as the locations of the ports being considered, impact the results.

The instances are constructed using real demand data and vessel information from the LINERLIB (Brouer et al., 2013a), which we combine with empirical travel time distributions constructed from COSCO liner shipping services (as discussed in Section 6.3). We build 44 instances based on 22 existing COSCO services as follows. For each COSCO service, we make a "standard" instance and a "tight" instance using the service's port calls. The standard instance matches the time windows from the COSCO data, while the tight instance modifies the time windows to provide a schedule with time windows as close together as possible. That is, all buffer between ports i and j in the service's port sequence is removed such that a vessel would have to sail at maximum speed to be on time. However, we are only able to remove buffer time up to a certain point, because the sum of all travel and port times has to be divisible by the number of hours in a week. We therefore remove all buffers between all port calls except for the return trip of the vessel to the origin port. For each instance, we extract relevant demands from the LINERLIB, including the amount of containers and the maximum transit time. For Phase 3, we also create a "relaxed" version that allows the cargo to be delivered up to 1.5 times later than the original transit time. This creates a total of 88 instances for Phase 3.

Since the LINERLIB data has been designed for the liner shipping network design problem, i.e., for problems that may contain several services, it is not perfectly suited towards single service design. This is because a significant amount of cargo is transported through hubs rather than along direct routes. To ensure our instances have a realistic amount of cargo, we aggregate ports into 25 clusters (similar to the aggregation performed in Mulder and Dekker (2014)). We then assume each of those clusters has some feeder services that will carry cargo to and from ports on the service we are designing. In some cases, especially with cargo originating from Asia, this can result in too much demand for any vessel to carry. We then randomly reduce the size of the demands. We subtract the amount of time required to carry the cargo to the hub from the transit time limit, along with three days for transshipment as done in Guericke and Tierney (2015).

An overview of our instances is given in Table 7.3. The service name corresponds to the COSCO service name. Note that our services in some cases do not exactly correspond with the original services since we remove duplicate port calls. In addition

Service	Ports	Demands	Minimum vessels	Service vessels	Vessel type
abx	9	13	7	8	Post Panamax
aesa	12	14	10	13	Super Panamax
awe1	7	20	8	10	Super Panamax
awe2	7	12	9	10	Super Panamax
awe3	10	37	8	10	Super Panamax
awe4	7	18	9	11	Super Panamax
awe8	9	33	9	11	Super Panamax
cen	7	8	6	7	Super Panamax
ces	9	27	9	10	Super Panamax
ese	11	18	7	8	Super Panamax
fal1	11	18	9	11	Super Panamax
fax	5	6	7	7	Super Panamax
fwas	9	19	9	12	Super Panamax
fwax	10	22	12	12	Super Panamax
ne2	10	31	9	10	Super Panamax
ne6	11	29	10	11	Super Panamax
ne7	10	28	9	10	Super Panamax
psw1	4	6	5	6	Super Panamax
psw5	6	8	7	6	Super Panamax
tas1	7	16	4	4	Super Panamax
wsa	10	34	9	10	Super Panamax
wsa2	11	30	9	10	Super Panamax

Table 6.2.: Instance properties for the instances tested in this work. We note that on psw5, the vessels from the carrier are slightly faster than the vessel we use from the LINERLIB, hence one less vessel is required.

to the number of ports and port-to-port demands, we compute the minimum number of vessels necessary to sail on the service using the original schedule and provide the number of vessels the carrier assigned to each service. The vessel type dictates the design speed, the minimum and maximum speed, as well as the (fixed) bunker costs in tons per day at design speed (which we get from LINERLIB). For vessels of the Post Panamax and Super Panamax classes, these values are as follows: design speed 16.5/17.0 knots, minimum speed 12 knots, maximum speed 23/22 knots, bunker costs 82.2/126.9 tons/day. We also assume a fuel price of $400 per ton.

We solve all instances with Gurobi version 7.0.2 (Gurobi Optimization, 2015) on eight Intel Xeon E5506 CPUs at 2.13 GHz with a maximum runtime of 24 hours. For each instance, we solve all of the models in Section 6.4. We experiment with modeling the underlying travel time data as a normal distribution and as an ll3p distribution we identified to be a better fit. We experiment with both to see if modeling travel time data with a normal distribution, which is easy to use and fit, offers a similar quality performance as the more accurate distribution. For both distributions, we experiment with service levels of 70 and 90%, and, for the ll3p distribution, also for a service level of 95% because of its long tail. We also test a deterministic version with no added buffer. This yields 8 runs for each instance and for each model. The output of each run is an optimized schedule that contains the ordering of ports of a service (a service design), the associated number of vessels that minimize the particular objective, the optimized objective value, and the scheduled arrival and departure times at ports. In Phase 2 and Phase 3, the optimized speed level between each pair of ports is also reported.

6.5.2. Simulation

For each service that is created by Phase 1 and 2 models, we simulate the resulting schedule to evaluate the impact of the varying service levels on “actual" costs and reliability, imitating operations of individual service runs. The purpose of the simulation is to analyze the impact of the stochastic travel time information and the possibility of adapting speeds to fulfill service levels in liner shipping network design. From a tactical perspective, the optimization models can create liner shipping services considering uncertainty through chance constraints as presented in Section 6.4. From an operational perspective, the simulation now tries to realize the optimal plans from the tactical level. We imitate that by drawing random travel times as an input for the realization of a service. We let the simulation react with speed adaptation according to the same degrees of freedom and cost parameters as our mathematical models, which causes differences to the planned speeds and to the planned variable sailing costs. Then, we can directly compare the planned costs (from optimization) with the “actual” realized costs (from simulation) for the different optimization models.

To create this simulation, for each arc of a service, we sample the travel time between ports using the ll3p distribution that has been fit to the particular OD pair as discussed in Section 6.3. If the sampled travel time for an arc would cause a delay at the destination port, the simulation increases the speed of the vessel (and reduces the travel time) to make up for the delay, inducing higher sailing costs. If the sampled travel time is less than what is required to arrive at the destination port on time, the simulation decreases the speed (and increases the travel time) to reduce fuel consumption and save money. Following the degrees of freedom of our optimization model in Phase 2, the simulated speed and travel times are limited by the minimum and maximum speeds as defined by the vessel type from the LINERLIB. As a consequence, in case of speed ups, the simulated vessel may not be able to make up for all of the delay at the destination port. The remaining lateness is then propagated to the next arc until the service terminates. We propagate lateness in the simulation because, as mentioned earlier, it is hard to anticipate which recourse action would be taken at each port where lateness might occur. By letting it accumulate in the simulation, we can easily evaluate the quality of the tactical planning and get an idea of the worst case performance of the service.

More formally, the simulation creates individual services runs from an optimized schedule (including order of ports, scheduled departure and arrival times at ports, and optimized speeds [only Phase 2]) as provided by Phase 1 and Phase 2 optimization models. The discrete-event simulation process is as follows:

1. For each arc between ports i and j, a travel time t_{ij}^{rand} is sampled from the appropriate travel time distribution. For the ll3p distribution, if $t_{ij}^{rand} > 10\mu_{ij}$, where μ_{ij} is the empirical mean travel time from i to j, then t_{ij}^{rand} is set to $10\mu_{ij}$. We truncate the ll3p-generated travel times because the fitted ll3p distributions often yield a very long tail, inducing unrealistically long sampled travel times. Even if such a long travel time may occur in practice, e.g., due to a storm or a port strike, we would not be able to deal with these by means of a more reliable schedule or an adaption of speeds on a tactical planning level. In these cases, ports would need to be skipped and other significant changes would need to be made, which we do not want to consider here to allow for a fair comparison between tactical planning and operations.

2. The delay at port j when sailing from i to j is computed as $e_{ij} := -(t_j^s - t_{ij}^{rand} - t_i^e)$, assuming that departure is at t_i^e.

3. To determine the arc-specific speed δ_{ij} that a vessel would need to sail to arrive at port j on time, the simulation computes the sailing duration as follows:

   $$\rho_{ij} = t_j^s - (t_i^e + e_{ij}).$$

   Here, we assume that today's sailing time between two ports i and j is random, but known when leaving port i so we can react with an optimized speed level. Implementing the idea of minimum and maximum sailing times from Constraints (6.22), the sailing duration is restricted by minimum and maximum sailing times:

   $$t_{ij}^{Min} \leqslant \rho_{ij} \leqslant t_{ij}^{Max}.$$

   Finally, the arc-specific speed can be derived from the distance between the ports Δ_{ij}, resulting in

   $$\delta_{ij} = \Delta_{ij}/\rho_{ij}.$$

4. In case of a large positive delay, the resulting speed up may still not be enough to achieve an on-time arrival, and the remaining delay will then be propagated to the next arc.

Let us look at a numerical example for a schedule with $t_j^s = 1471.0$, $t_i^e = 918.0$ and scheduled travel time of 549.71. The sample travel time from ll3p distribution is $t_{ij}^{rand} = 558.786$. Next, $e_{ij} = -(1471.0 - 558.786 - 918.0) = 5.786$. Last, $\rho_{ij} = 1471.0 - (918.0 + 5.786) = 547.214$, i.e. $\delta_{ij} = 9345.0/547.214 = 17.08$.

The simulation was coded in Java 8 and run on a 64-bit Windows 10 machine. We run 100,000 simulations per schedule.

6.5.3. Results

First, we present the results of computational experiments for Phase 1 and Phase 2 models including the results of the simulation. We then provide the results from experiments with the Phase 3 model.

Phase 1 and 2 Results

In Table 6.3, we present averaged results for Phase 1 and 2 models grouped by standard and tight instances. The first column is the service level and the second column specifies the group of instances. The next three columns present results from the Phase 1 optimization (tactical perspective). The column “Ves” represents the number of vessels required for a service, “Var" is the variable sailing cost, and “Total" gives the total costs including variable and vessel costs from the optimization. Recall that all Phase 1 optimization runs assume the vessel sails at the design speed. The next five columns represent the averaged results from the 100,000 simulations of each schedule (operational perspective). This includes the variable sailing costs (“Var”), the number of late ports per service that remains in spite of speed ups (“# Late”), the average hours late per port when arrival to a port is late (“Late/p”), the percent of the time the vessel sailed above the design speed (“Faster”), and the average speed traveled in the simulation (“Spd”). The simulation assumes the use of the number of vessels selected by the optimization, which is why the total cost in the simulation is not reported. For Phase 2, the same results are reported. For the optimization, the additional column “Spd" shows the average speed level chosen by the optimization, which allows for a comparison of tactical and operational speed levels. We report the full set of Phase 1 and Phase 2 results in the electronic appendix.

First, we will examine the optimization outputs from Phase 1. As expected, the number of vessels and total costs increase with service level guarantees as compared with the deterministic results. For the standard instances, the number of vessels required for ll3p with 95% service level is almost double the number in the deterministic results (17.64 vs 9.73). For the tight instances, the vessels required for ll3p with 95% service level are slightly more than double the number in the deterministic results (17.86 vs 8.68). The corresponding total costs increase by 49% for the standard and 61% for the tight instances. The average number of vessels is slightly larger for the tight instances than for the standard instances except when service levels are considered. This indicates the deterministic solutions do not have many natural buffers in the solutions.

Table 6.3.: Simulation results for Phase 1 and 2 for different instance types.

		Phase 1									Phase 2								
		Optimization			Simulation						Optimization				Simulation				
Instance	Type	Ves	Var	Total	Var	#Late	Late/p	Faster	Spd		Ves	Var	Total	Spd	Var	#Late	Late/p	Faster	Spd
Deterministic	Standard	9.73	24.19	61.13	24.89	2.09	27.42	45.38%	16.85		9.91	16.48	54.06	14.01	19.59	1.80	22.01	29.52%	15.13
Norm 0.7	Standard	11.73	24.52	69.03	17.55	0.65	17.11	11.93%	14.06		11.76	14.20	58.82	12.92	15.18	0.50	13.36	9.37%	13.27
Norm 0.9	Standard	13.18	24.27	74.26	15.66	0.40	10.10	6.38%	13.37		13.19	13.31	63.29	12.56	13.96	0.29	7.06	4.73%	12.78
ll3p 0.7	Standard	11.27	24.31	67.14	18.49	0.77	16.62	16.46%	14.58		11.24	14.96	57.56	13.29	16.36	0.57	13.00	12.70%	13.78
ll3p 0.9	Standard	13.91	24.28	76.94	15.09	0.25	2.78	4.50%	13.18		13.81	13.17	65.40	12.52	13.48	0.12	1.89	3.04%	12.64
ll3p 0.95	Standard	17.64	24.59	91.28	14.02	0.01	0.11	1.55%	12.73		17.29	13.28	78.57	12.48	13.34	0.01	0.10	1.17%	12.50
Deterministic	Tight	8.68	24.22	57.14	29.16	4.22	42.96	64.73%	18.38		9.64	17.15	53.74	14.29	21.03	3.64	31.70	38.13%	15.67
Norm 0.7	Tight	12.32	24.66	71.32	18.13	0.58	14.51	12.00%	14.11		12.35	13.87	60.58	12.76	14.77	0.40	10.41	7.21%	13.07
Norm 0.9	Tight	13.55	24.36	75.74	15.90	0.37	7.67	5.41%	13.44		13.73	13.56	65.64	12.57	14.06	0.28	6.27	3.32%	12.74
ll3p 0.7	Tight	11.59	24.60	68.53	19.99	0.79	18.76	19.07%	14.86		11.71	15.46	59.83	13.32	17.13	0.60	15.12	13.30%	13.88
ll3p 0.9	Tight	14.41	24.29	78.87	15.30	0.15	2.08	3.91%	13.26		14.29	13.29	67.36	12.55	13.54	0.10	1.42	2.58%	12.64
ll3p 0.95	Tight	17.86	24.48	92.05	14.48	0.01	0.24	1.54%	12.94		17.62	13.22	79.78	12.46	13.28	0.01	0.18	1.25%	12.49

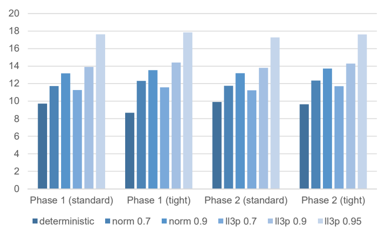
Next, we examine what happens when we simulate the operations of service runs for these Phase 1 results. For deterministic instances, simulated variable costs are slightly higher, which reflects the need of vessels having to sail above design speed to minimize lateness. With deterministic values, the vessel would have to sail on average 45.38% (standard) or 64.73% (tight) faster than design speed to minimize lateness, but this still leads to an average late arrival of 2.09 (standard) and 4.22 (tight). The number of late ports per service is under 1 for all instances with service levels, but when vessels are late, the average hours of lateness is over five hours except for *ll3p 0.9* and *ll3p 0.95* service levels. This indicates the utility of the better fitting distribution.

In Phase 2 optimization results, the variable sailing costs decrease for all instance types, and the number of vessels remains close to the values found in the Phase 1 optimization results. The average planned tactial speed level drops from 16.5 or 17 knots in Phase 1 to as low as 12.46 knots with the ll3p 0.95 service level. In the Phase 2 simulation, the variable costs, the number of late arrivals per service and the average lateness is less than in the Phase 1 simulation, indicating the value of the more detailed optimization model. The percentage of the service needing speed ups in operations is significantly smaller than in the Phase 1 simulation, indicating the gap between tactical and realized (simulated) speed tends to become smaller than in Phase 1.

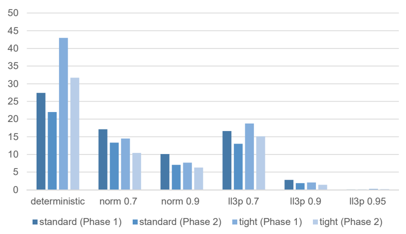
Figure 6.3 presents the discussed metrics graphically for another view. Figure 6.3a compares the average optimal number of vessels for the different instances and phases. It becomes clear that with the deterministic instances the number of required vessels is the smallest, and higher service levels require larger number of vessels. Furthermore, the deterministic tight instances require less vessels than their standard counterparts, but this changes when introducing buffers, indicating the impact of stochastic information.

Figure 6.3b displays the results of the simulations with the different travel time distributions with regard to average hours of lateness at ports. It is obvious that planning in a deterministic way leads to a significant amount of lateness on average, especially for the tight instances (up to 43 hours in Phase 1 and up to 32 hours in Phase 2). Interestingly, optimizing for speed levels (Phase 2) produces schedules that help reduce the average amount of lateness. However, the best option to reduce lateness is the inclusion of buffers. With *norm 0.7* or *ll3p 0.7* optimized schedules, lateness can be reduced significantly. Considering the long tail of the ll3p distribution, planning with a buffer based on *ll3p 0.9* makes lateness almost disappear, of course at the cost of a large number of required vessels. Here, the value of the more realistic distribution, especially its long tail, comes into play for all instances and phases.

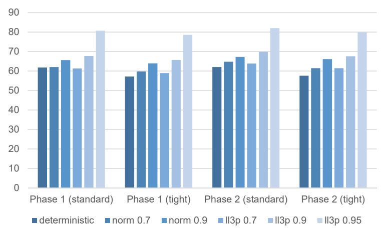
Finally, Figure 6.3c visualizes the total simulated costs, which include variable costs arising from realizing schedules by simulation and fixed vessel costs provided by optimization. This metric yields the operational costs and should match the total costs of tactical planning in an idealized setting. For almost all instances and phases, a higher service quality comes at a higher total cost, mainly due to the larger number



(a) Number of vessels from optimization in Phases 1 and 2



(b) Analysis of hours late for different travel time distributions



(c) Total simulated costs

Figure 6.3.: Visualization of simulation results

of required vessels. Interestingly, for standard instances in Phase 1, including buffers based on *ll3p 0.7* can reduce total simulated costs a bit compared with the deterministic schedule. Enforcing a high service level based on *ll3p 0.95* increases total costs greatly, though.

Phase 3 Results

The main difference between Phase 3 and Phase 2 is the inclusion of revenue generation and transit time limitations for container demands. For our experiments, we focus on how much demand is carried under different service guarantees with these new limitations. Table 6.4 shows the percentage of total available demand carried for various ll3p service levels, along with the average increase in port-to-port travel time over the deterministic Phase 3 model solution. We provide results for two different settings of maximum transit times. The “LL” results use the maximum transit time for container demands as listed in the LINERLIB, whereas “LL 1.5” uses the LINERLIB maximum transit times multiplied by 1.5. As in Phases 1 and 2, we evaluate each service with a standard (S) and a tight (T) instance. We provide these longer durations because the ones in the LINERLIB are tailored to Maersk Line's network, which is somewhat different than COSCO's network. The longer durations also show how much more demand can be carried when the restrictions are relaxed.

Our results show that when using the transit times in the LINERLIB, even a service guarantee of 0.7 results in less than half of the containers of the deterministic solution being carried on 12 services. The increase in transit times between ports to achieve service levels is why cargo is not being carried, as in many cases the transit time is over 50% higher for a service level of 0.7. Raising the maximum transit times by 50% allows significantly more containers to be transported, resulting in only one service where both the S and T variants at the 0.7 service level carry less than half of the containers of the deterministic version. We note that in the case of *awe2* at service level 0.9, the average transit times slightly decrease over the deterministic solution due to the selection of a different route.

The message for carriers from these results is clear: implementing a punctuality guarantee requires them to either charge higher freight rates to shippers or to convince shippers to accept higher maximum transit times. However, this work provides a mechanism for quantitatively assessing how much extra revenue or transit time would be necessary to run a profitable service. For example, on the *abx* service in the standard case, the carrier only needs to convince their customers to accept 11% longer transit times (on average around the service) for a 70% service guarantee. Our model is particularly useful in this respect in comparison to models that do not optimize the vessel route, as such models will not yield any insights when adjusting transit time limitations or cargo revenues.

Runtime

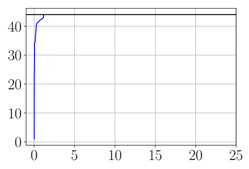
Our mathematical model is solved in most cases relatively quickly using the Gurobi solver. Figure 6.4 shows the number of instances solved at each point in time over 24 hours of runtime. In Phase 1, all 44 instances can be solved within a couple of hours, with the vast majority solved in only a few minutes. The inclusion of speed optimization in Phase 2 increases the difficulty of the model, however we note that nearly all of our instances are solved within 10 hours regardless of the service level desired. In Phase 3, the difficulty increases due to the extra freedom of the model to optimize the cargo intake. Here, we are able to solve over half of the instances within the first hour, with all but four instances solved before the timeout in the deterministic case, and all but two instances for *ll3p* 0.95. For planning services that have a lifespan of months or even years, a runtime of 24 hours or more is acceptable.

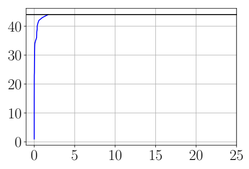
6.5.4. Sample Service Plans

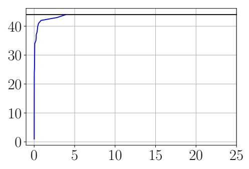
Depending on the optimization model of the particular phase, the resulting schedules may differ significantly, even when total costs are similar. We present some of the

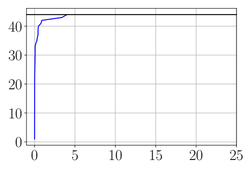
		Cargo carried (%)								Transit time increase (%)					
		LL				LL 1.5				LL			LL 1.5		
Service	Cat.	Det	0.7	0.9	0.95	Det	0.7	0.9	0.95	0.7	0.9	0.95	0.7	0.9	0.95
abx	S	93	86	70	69	100	97	80	76	11	956	1446	9	854	1173
	T	97	81	73	73	99	97	80	76	93	133	237	125	1132	289
aesa	S	78	32	22	20	97	78	38	36	44	102	163	21	109	161
	T	78	22	22	22	97	78	36	32	74	134	187	37	111	179
awe1	S	39	15	14	14	79	40	39	22	42	107	149	102	169	188
	T	39	16	16	15	42	39	26	26	26	45	96	162	206	253
awe2	S	15	15	15	13	29	16	16	15	12	-1	35	30	38	69
	T	15	15	15	13	36	18	16	15	11	-1	32	34	56	70
awe3	S	19	11	6	6	36	19	15	10	67	125	192	47	75	132
	T	18	10	8	8	38	16	13	10	103	136	167	68	102	158
awe4	S	31	11	8	8	38	31	21	21	63	173	278	128	134	197
	T	21	20	12	8	36	31	21	19	13	93	257	76	109	202
awe8	S	11	8	8	2	34	10	11	8	252	307	422	148	201	271
	T	11	6	5	2	35	18	12	10	238	122	473	51	99	143
cen	S	79	16	7	7	100	72	65	58	68	90	144	20	55	93
	T	72	11	11	7	100	69	62	40	99	134	145	36	76	200
ces	S	90	47	45	43	93	83	65	59	161	201	247	73	176	150
	T	62	28	6	6	93	76	72	69	55	143	192	30	51	78
ese	S	70	15	14	10	85	85	42	28	33	87	98	0	42	57
	T	63	18	14	14	85	68	61	60	122	153	172	54	63	91
fal1	S	88	44	44	7	100	83	77	47	75	109	193	33	75	145
	T	79	28	18	18	100	77	67	45	86	113	212	31	78	160
fax	S	99	35	33	33	100	99	67	64	28	123	260	52	111	226
	T	100	33	33	1	100	99	67	34	74	107	196	40	115	165
fwas	S	98	69	35	15	99	99	74	57	19	210	328	5	68	100
	T	98	76	36	11	99	100	74	57	37	105	332	140	256	375
fwax	S	60	30	20	12	81	79	57	39	72	94	188	71	329	357
	T	61	30	20	12	81	80	73	73	86	76	154	37	166	266
ne2	S	81	41	38	38	98	87	74	60	155	207	268	34	150	230
	T	83	47	45	38	97	83	75	59	35	215	188	25	138	173
ne6	S	77	12	8	7	98	97	70	53	79	179	213	41	232	391
	T	76	27	20	7	98	83	71	53	58	83	186	47	242	401
ne7	S	82	47	58	38	98	84	77	63	30	245	215	28	109	208
	T	83	37	28	28	98	86	67	63	71	96	128	26	166	208
psw1	S	45	1	1	0	100	40	40	39	49	76	92	27	43	60
	T	40	1	1	1	100	40	39	32	76	93	93	42	59	79
psw5	S	64	22	22	2	96	74	64	62	66	114	138	45	89	145
	T	29	2	2	2	94	33	25	24	209	340	528	61	125	181
tas1	S	79	45	39	35	85	70	70	64	73	70	109	82	82	109
	T	70	39	39	35	85	70	69	69	55	55	93	81	81	101
wsa	S	94	39	37	37	97	86	69	44	163	268	284	50	143	228
	T	94	20	17	4	98	86	65	43	169	207	514	43	214	350
wsa2	S	85	57	18	18	95	88	57	50	22	115	154	227	447	470
	T	87	58	18	18	95	86	53	51	57	161	205	49	225	131

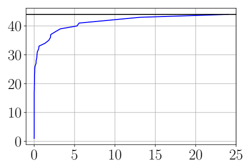
Table 6.4.: The percentage of total cargo carried in the Phase 3 model and increase in port-to-port transit times over the deterministic solution for several service levels using the ll3p distribution

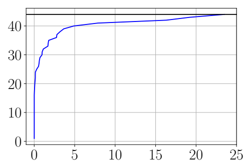

(a) Phase 1 deterministic

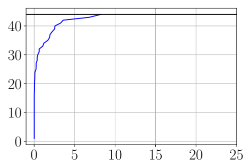

(b) Phase 1 *ll3p 0.7*

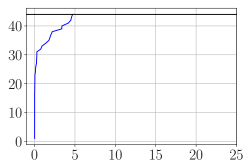

(c) Phase 1 *ll3p 0.9*

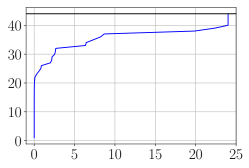

(d) Phase 1 *ll3p 0.95*

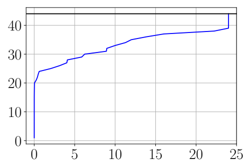

(e) Phase 2 deterministic

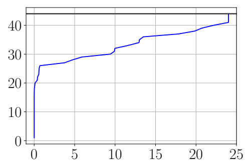

(f) Phase 2 *ll3p 0.7*


(g) Phase 2 *ll3p 0.9*


(h) Phase 2 *ll3p 0.95*


(i) Phase 3 deterministic


(j) Phase 3 *ll3p 0.7*


(k) Phase 3 *ll3p 0.9*

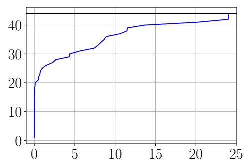

(l) Phase 3 *ll3p 0.95*

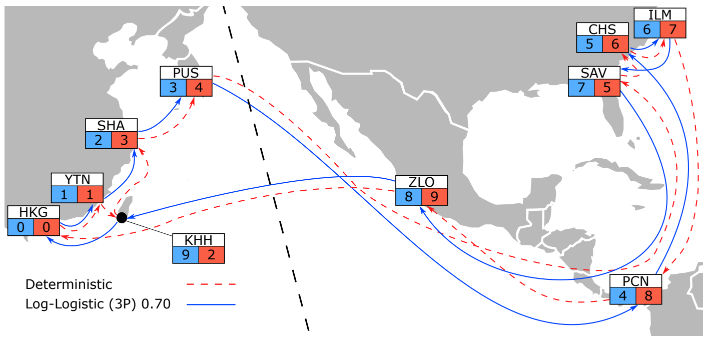
Figure 6.4.: The number of instances solved (y-axis) versus the runtime of Gurobi (wall time) in hours in all phases for the deterministic model and all *ll3p* models.

created schedules to understand the impact of the different assumptions on them.

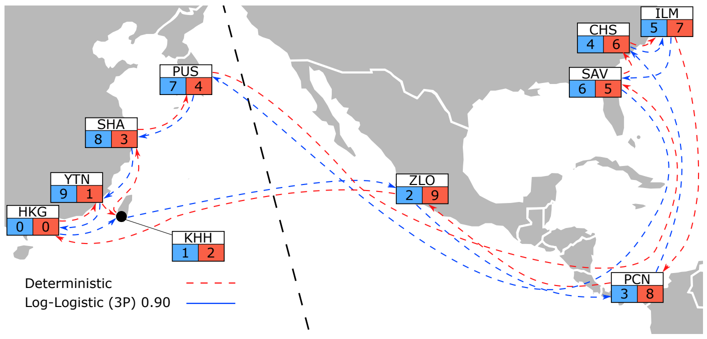
Figures 6.5a – 6.5d compare the optimal schedules for the different phases of service *awe3* (standard). This service connects ports in North America with ports in East Asia. In Figure 6.5, the dashed line reflects the result of the corresponding deterministic optimization, and the solid blue line shows a given service level. The sequence number for each port is shown in a box with a color matching either the deterministic or 0.7/0.9/0.95 service level solution. For Phase 1 (see Figure 6.5a), the optimal deterministic port order is HKG–YTN–KHH–SHA–PUS–SAV–CHS–ILM–PCN–ZLO–HKG, requiring a total of 11 vessels and sailing costs of \$2.871 million per rotation. Including buffers based on *ll3p 0.9* completely changes the order of the ports in the optimal schedule, increasing the number of vessels to 16 and slightly reducing sailing costs to \$2.837 million per rotation. Simulation of these schedules reveals a significant reduction of the speed ups required to ensure punctuality (from 40% to 4%). Interestingly, although a high service level means investing in a large number of vessels, due to slow steaming, the simulated sailing cost is so small that the simulated total costs can be reduced a bit ($7.264 million for deterministic vs. $7.072 million for *ll3p 0.9*).

The results of optimization of this instance for Phase 2 can be seen in Figure 6.5c. Both schedules are different from their Phase 1 counterparts in the order of port visits, and the schedules operate in opposite directions. The number of vessels remains constant compared with Phase 1, but the optimized and realized speeds vary. For the deterministic solution, we have a planned average speed of 13.8 knots, and for the *ll3p 0.9* based solution a value of 12.41 knots. When optimizing speed levels, required speed ups can be reduced from 24% to 3%, reducing planned and simulated variable sailing costs. However, due to the larger number of vessels, total simulated costs increase significantly from 65.14 to 77.35, which is accompanied by an average reduction of lateness from 2.7 to 0.2 ports per trip.

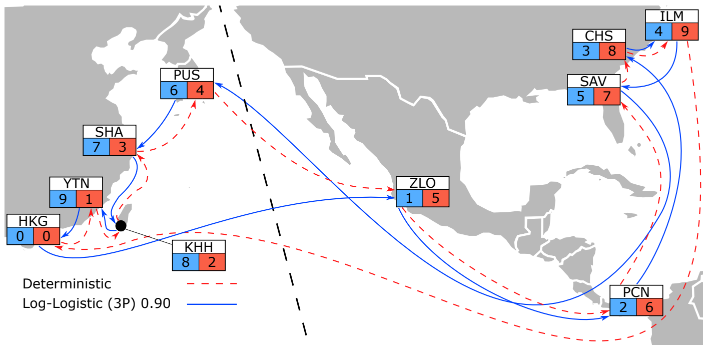
The Phase 3 solutions show the drop-off in cargo on the individual legs of the solution for the 0.9 service level. Both the fronthaul and backhaul see significant reductions in containers carried, with service level 0.9 taking 2.6x less cargo on the fronthaul and 7.8x less on the backhaul. While many carriers would likely be willing to accept reductions in utilization on the backhaul in exchange for high punctuality guarantees (90% on-time would lead the industry by a significant margin), reductions in fronthaul utilization are especially bad for a carrier's profits.



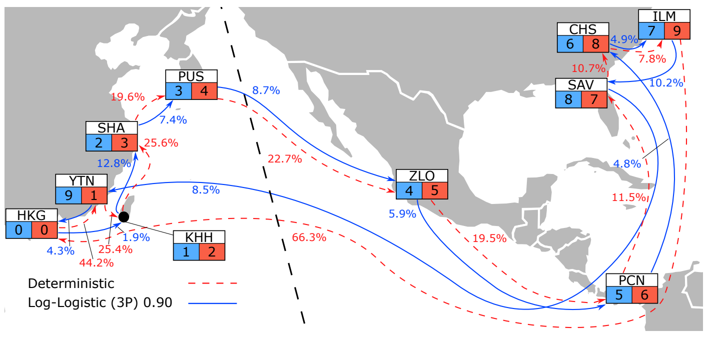
(a) Phase 1 (Det/0.7)



(b) Phase 1 (Det/0.9)



(c) Phase 2



(d) Phase 3 (LL 1.5)

Figure 6.5.: Visualizations of the deterministic and 0.9 ll3p service level solutions for service awe3

6.6. Conclusions

We solved the liner shipping service design problem with arrival time service levels. We showed that a three parameter log-logistic distribution fits the journeys of liner ships better than other distributions, such as the uniform or normal distribution, that were previously used in the literature. Our mathematical models of the service design problem use chance constraints to ensure that vessels arrive on time from a tactical planning perspective. Simulation of schedule operations allows for the comparison of planned total costs with “actual" total costs. Despite modelling a realistic version of service design, the model is nonetheless computationally tractable with a state-of-the-art mixed-integer programming solver. Our results show that services can be designed with on-time guarantees and that the route a vessel takes heavily influences the service level required. Furthermore, our model allows carriers to quantitatively assess what level of service they wish to offer and can support them during negotiations with shippers over prices and container transit time limitations.

This model provides a basis for several directions of future work. First, more detailed disruption recovery actions could be considered not only in the simulation of the model solution, but also in the model itself. Actions such as skipping ports or not fully loading/unloading containers when severely delayed could be included, however these would likely make the model significantly harder to solve. Furthermore, faster solution techniques could be considered such as branch & price or heuristic approaches. These would make the model more useful for operational decision support when negotiating berthing windows with terminals. Finally, time windows could be made optional, i.e., instead of a planner specifying time windows, the model could determine optimal time windows that the planner could then attempt to negotiate with a terminal.

Acknowledgements

We thank the Paderborn Center for Parallel Computation (PC^2) for the use of their high throughput cluster for the experiments in this work. We further thank Stefan Guericke for his advice and comments regarding this work.

Bibliography

Agarwal, R., Ergun, Ö., 2008. Ship scheduling and network design for cargo routing in liner shipping. Transportation Science 42 (2), 175–196.

Agra, A., Christiansen, M., Delgado, A., Hvattum, L., 2015. A maritime inventory routing problem with stochastic sailing and port times. Computers & Operations Research 61, 18 – 30.

Agra, A., Christiansen, M., Hvattum, L., Rodrigues, F., 2016. A MIP based local search heuristic for a stochastic maritime inventory routing problem. In: Paias, A., Ruthmair, M., Voß, S. (Eds.), 7th International Conference on Computational Logistics. Springer International Publishing, pp. 18–34.

Álvarez, J., June 2009. Joint routing and deployment of a fleet of container vessels. Maritime Economics and Logistics 11 (2), 186–208.

Brouer, B., Alvarez, J., Plum, C., Pisinger, D., Sigurd, M., 2013a. A base integer programming model and benchmark suite for liner-shipping network design. Transportation Science 48 (2), 281–312.

Brouer, B., Desaulniers, G., Pisinger, D., 2014. A matheuristic for the liner shipping network design problem. Transportation Research Part E: Logistics and Transportation Review 72, 42–59.

Brouer, B., Dirksen, J., Pisinger, D., Plum, C., Vaaben, B., 2013b. The Vessel Schedule Recovery Problem (VSRP) – A MIP model for handling disruptions in liner shipping. European Journal of Operational Research 224 (2), 362–374.

Brouer, B., Karsten, C., Pisinger, D., 2017. Optimization in liner shipping. 4OR 15 (1), 1–35.

Christiansen, M., Fagerholt, K., Nygreen, B., Ronen, D., 2013. Ship routing and scheduling in the new millennium. European Journal of Operational Research 228 (3), 467 – 483.

Christiansen, M., Fagerholt, K., Ronen, D., 2004. Ship routing and scheduling: Status and perspectives. Transportation Science 38 (1), 1–18.

COSCO Shipping Lines, 2017. http://www.coscon.com/ourservice/toService.do.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation 6 (2), 182–197.

Ehmke, J., Campbell, A., Urban, T., 2015. Ensuring service levels in routing problems with time windows and stochastic travel times. European Journal of Operational Research 240, 539–550.

Guericke, S., Tierney, K., 2015. Liner shipping cargo allocation with service levels and speed optimization. Transportation Research Part E: Logistics and Transportation Review 84, 40–60.

Gurobi Optimization, 2015. Gurobi optimizer reference manual.
URL http://www.gurobi.com

Halvorsen-Weare, E., Fagerholt, K., 2011. Robust supply vessel planning. In: Pahl, J., Reiners, T., Voß, S. (Eds.), Network Optimization. Vol. 6701 of Lecture Notes in Computer Science. Springer Berlin Heidelberg, pp. 559–573.

Halvorsen-Weare, E., Fagerholt, K., 2013. Routing and scheduling in a liquefied natural gas shipping problem with inventory and berth constraints. Annals of Operations Research 203, 167–186.

Halvorsen-Weare, E., Fagerholt, K., Ronnqvist, M., 2013. Vessel routing and scheduling under uncertainty in the liquefied natural gas business. Computers & Industrial Engineering 64, 290–301.

Karsten, C., Pisinger, D., Ropke, S., Brouer, B., 2015. The time constrained multi-commodity network flow problem and its application to liner shipping network design. Transportation Research Part E: Logistics and Transportation Review 76, 122–138.

Kauczynski, W., 1994. Study of the reliability of the ship transportation. Proceeding of the International Conference on Ship and Marine Research, 15.

Kolen, A. W. J., Kan, A. H. G. R., Trienekens, H. W. J. M., 1987. Vehicle routing with time windows. Operations Research 35 (2), 266–273.

Lee, C.-Y., Lee, H. L., Zhang, J., 2015. The impact of slow ocean steaming on delivery reliability and fuel consumption. Transportation Research Part E 76, 176 – 190.

MathWave Technologies, 2016. EasyFit.

Meng, Q., Wang, S., Andersson, H., Thun, K., 2013. Containership routing and scheduling in liner shipping: overview and future research directions. Transportation Science 48 (2), 265–280.

Meng, Q., Wang, T., 2010. A chance constrained programming model for short-term liner ship fleet planning problems. Marit. Pol. Mgmt. 37 (4), 329–346.

Mulder, J., Dekker, R., 2014. Methods for strategic liner shipping network design. European Journal of Operational Research 235 (2), 367 – 377, maritime Logistics.

Norstad, I., Fagerholt, K., Laporte, G., 2011. Tramp ship routing and scheduling with speed optimization. Transportation Research Part C: Emerging Technologies 19 (5), 853–865.

Notteboom, T., 2006. The time factor in liner shipping services. Maritime Economics & Logistics 8 (1), 19–39.

Notteboom, T., Rodrigue, J.-P., 2008. Containerisation, box logistics and global supply chain: The integration of ports and liner shipping networks. Maritime Economics & Logistics, 152–174.

Plum, C., Pisinger, D., Salazar-González, J., Sigurd, M., 2014. Single liner shipping service design. Computers & Operations Research 45, 1 – 6.

Powell, B., Perakis, A., Spring 1997. Fleet deployment optimization for liner shipping: An integer programming model. Maritime Policy and Management 24 (2), 183–192.

Qi, X., Song, D.-P., 2012. Minimizing fuel emissions by optimizing vessel schedules in liner shipping with uncertain port times. Transport. Res. Part 48 (4), 863–880.

Reinhardt, L., Plum, C., Pisinger, D., Sigurd, M., Vial, G., 2016. The liner shipping berth scheduling problem with transit times. Transportation Research Part E: Logistics and Transportation Review 86, 116–128.

Santini, A., Plum, C. E., Ropke, S., 2017. A branch-and-price approach to the feeder network design problem. European Journal of Operational Research.

Song, D., Dong, J., 2013. Long-haul liner service route design with ship deployment and empty container repositioning. Transportation Research Part B: Methodological 55, 188 – 211.

Song, D., Li, D., Drake, P., 2015. Multi-objective optimization for planning liner shipping service with uncertain port times. Transportation Research Part E: Logistics and Transportation Review 84, 1–22.

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

Vernimmen, B., Dullaert, W., Engelen, S., 2007. Schedule unreliability in liner shipping: Origins and consequences for the hinterland supply chain. Maritime Economics & Logistics 9 (3), 193–213.

Wang, S., Meng, Q., 2012a. Liner ship route schedule design with sea contingency time and port time uncertainty. Transportation Research Part B 46, 615–633.

Wang, S., Meng, Q., 2012b. Robust schedule design for liner shipping services. Transportation Research Part E: Logistics and Transportation Review 48 (6), 1093–1106.

Wang, S., Wang, X., 2016. A polynomial-time algorithm for sailing speed optimization with containership resource sharing. Transportation Research Part B: Methodological 93, Part A, 394 – 405.

7. A Biased Random-Key Genetic Algorithm for the Liner Shipping Fleet Repositioning Problem

Daniel Müller*

*University of Paderborn
Warburger Straße 100, 33098 Paderborn, Germany
mueller@dsor.de, tierney@dsor.de

As liner carriers adjust their network of cyclical routes throughout the year, vessels must be moved from one route to another in a process called fleet repositioning. Although heuristics such as simulated annealing and reactive tabu search have been used to solve the liner shipping fleet repositioning problem (LSFRP), there is still room for improvement considering the gap to the optimal solution and the computational time. We propose solving the LSFRP with a biased random-key genetic algorithm (BRKGA), as it can more effectively avoid infeasible solutions than the currently available heuristics. The inherent learning mechanism of the algorithm helps to identify solutions with good objective values. We propose four different random-key definitions and perform a computational analysis of these keys on publicly available LSFRP instances. In a comparison with the state-of-the-art algorithm, a hybrid reactive tabu search, we show that the BRKGA is able to find optimal solutions for small and medium sized instances. Our results also show that the BRKGA is not able to compete with the state-of-the-art for larger instances.

Keywords: liner shipping, fleet repositioning, biased random-key, genetic algorithm

7.1. Introduction

Seaborne trade's relevance for the world economy has been growing steadily throughout the last decades, reaching a total amount of transported goods of 10 billion tons per year. As a part of the overall seaborne trade, the transportation of containerized goods also shows a steady growth. In 2015, the number of transported containers per year was almost three times as high as the amount in 2000, reaching a total number of 175 million TEU[1] and a transported volume of 1.7 billion tons UNCTAD (2016). A total of 1.8 million vessels are under operation for the seaborne trade in general and from this about 244,000 vessels are dealing with containerized goods. The largest liner shipping company alone employs about 600 vessels for its trade.

[1]twenty foot equivalent unit

Due to the enormous size of the liner shipping industry and its ongoing growth, the operation of liner shipping networks has become a complex task for liner shipping companies. The operation and coordination of these large amounts of vessels that are distributed over the whole world on a long time scale is very complicated and needs technical support from advanced algorithms and systems (Müller and Tierney (2017)).

Liner shipping companies establish cyclical routes to connect ports in different trade regions. These ports can be located in the same geographical area, e.g. the Mediterranean sea, or they can be dispersed over multiple continents. These routes are called services and are usually operated on a weekly or bi-weekly frequency. Due to this fixed frequency, ports of a service are visited at a fixed time each period. Depending on this frequency and the overall length of the service, the number of needed vessels can be determined. These vessels have a specific schedule according to their time slot in the service rotation.

Liner shipping companies are able to create a large network of connected ports to transport containers by operating several services. Due to economic and seasonal trends, liner carriers add, remove or modify services in their network on a regular basis. In such a modification, vessels are reassigned to other services and have to be repositioned from their current service to a new service. The repositioning of vessels under consideration of the costs for operating and moving the vessels is known as the liner shipping fleet repositioning problem (LSFRP)(Tierney et al. (2014)).

One of the major challenges in the LSFRP is the coordination of the vessels in regard to their sailings. When a vessel is selected for repositioning, it has to start operating on the goal service such that the schedule of the goal service is not altered. In between the start of the repositioning, called the phase-out, and the start of operation at the new service, called the phase-in, the repositioning coordinator has to plan a path that considers the original schedule and the goal schedule.

There are exact approaches as well as heuristic approaches to solve the LSFRP. In other publications, it has been shown that heuristics are more suitable for practical applications as they provide a quick solution to instances of different sizes. The current state-of-the-art algorithm for the LSFRP is a reactive tabu search approach, which is combined with a simulated annealing algorithm (RTS-SA). Due to this hybridization, the authors are able to increase the profits of several public LSFRP instances.

In this paper, we present four different random-key definitions for a biased random-key genetic algorithm (BRKGA) to solve the LSFRP. We discuss the definition of the random keys and the process to decode the keys in order to generate a solution. Our computational experiments with public LSFRP instances show that the BRKGA is able to find optimal solutions of small and medium-sized instances. Unfortunately, the BRKGA has problems to find good solutions for larger instances and therefore is not able to outperform the RTS-SA as the current state-of-the-art algorithm.

The main contributions of this paper can be summarized as follows:

1. A definition for a random-key decoder for the LSFRP
2. Proposition of four different random-key definitions for the LSFRP
3. Computational experiments to compare the BRKGA with the RTS-SA
4. Proposition of the BRKGA combined with hill climbing

This paper is organized as follows: In Section 7.2, we will describe the LSFRP in detail. Followed by Section 7.3 to introduce the general concept of BRKGAs as well as specific details for the BRKGA for the LSFRP. Then, the results of the computational experiments will be presented in Section 7.4. Finally, the paper is concluded in Section 7.5. An outlook on future research is also included in Section 7.5.

7.2. The Liner Shipping Fleet Repositioning Problem

Maritime transportation problems have been classified by Christiansen et al. (2007) into the different planning horizons strategic, tactical and operational. Fleet size and fleet mix decisions as well as ship design or market selection are decisions at a strategic planning horizon. At a tactical level, planners need to decide about the vessel deployment, the routing of vessels and their specific schedules. Furthermore, the tactical level also considers container yard management and stowage planning. Regarding the operational horizon, planners take decisions about vessel speed and the actual routing of the vessels, considering external factors like weather.

Throughout the year, cargo demand is influenced by seasonal variations and general economic trends. In order to survive in the competitive market of the liner shipping industry, liner shipping companies need to adapt to these changes. This is mostly done by adjusting and changing their network of liner services. These changes occur on a strategic to a tactical planning horizon. Changing a liner service, includes adding or removing a service as well as modifying an existing service. When such a change occurs, liner shipping companies have to reassign vessels between services. The process of moving a vessel from a service to another service is called *repositioning*.

The liner shipping fleet repositioning problem (LSFRP) was first mathematically described in Tierney et al. (2012). The model reflects the goal of liner shipping companies to minimize the cost of such a repositioning. For this, revenue generating activities are incorporated into the repositioning plan of a vessel. These activities include the transportation of customer cargo and empty containers. Transporting customer cargo also helps avoiding cargo flow disruptions, which would be very costly to the liner shipping company due to the lost revenue. Other costs are associated with port calls and the bunker consumption of vessels.

Especially the bunker consumption gives liner shipping companies the opportunity to save costs in their operations. Since the fuel consumption of vessels is roughly cubic according to the vessels speed, sailing at less speed can save considerable amounts of money. In an activity called *slow steaming* Meyer et al. (2012), vessels sail below their design speed, reducing the bunker consumption. This approach is a standard procedure, to reduce sailing costs for liner carriers.

Furthermore, vessels can use so-called *sail on service opportunities* (SoS) to generate additional revenue. On their way from their original service to the goal service, vessels can sail on other services as well, replacing another vessel on this service (the *on-service vessel*). While the repositioning vessel is performing its operation on this service, the on-service vessel can be chartered to another company. A SoS can also be used to lay up and maintain the on-service vessel, as the original vessel follows the schedule of the on-service vessel and therefore no cargo flow disruptions on this service are happening.

The repositioning vessel needs to be at the same port as the on-service vessel in order to make use of a SoS. There, all the cargo can be transshipped from the on-service vessel to the repositioning vessel such that the repositioning vessel can transport the cargo. Another option is to let the on-service vessel and repositioning vessel visit the same ports at the same time, which is known as “in tandem” sailing (Tierney et al. (2014)). The on-service vessels stops at every port to unload cargo until it is free and is able to start alternative activities. The repositioning vessels operates as an on-service vessel and loads and unloads cargo where necessary.

The cargo that needs to be transported is defined by its starting port and its destination port. Furthermore, the latest delivery time at the destination port is known. In addition to this specific customer cargo, empty equipment can also be transported. Empty equipment refers to empty containers that are in surplus at some ports, while other ports have a deficit of empty containers. In contrast to normal cargo, empty containers can be transported to any port with a deficit. By transporting empty containers, revenue can be generated as otherwise money would have to be spent to transport these containers with other options.

In this model, containers can have two types: dry and reefer. Dry containers are standard containers, without any further requirements. Reefer containers are used to store cargo that needs to be refrigerated. For this, reefer containers need electric outlets and cooling connections, which are only provided on special places on a container vessel. Each vessel type has a specific capacity of dry and reefer slots for containers, which needs to be considered by the planning as dry and reefer cargo is not interchangeable.

While planning the repositioning process, liner carriers have to consider a specific planning horizon. This horizon is set by the repositioning coordinator and describe the earliest time, the repositioning vessel may leave its original service and the latest possible time the repositioning vessel may start its operations on the goal service. The actual time when the repositioning vessel leaves its original service is called *phase-out*. The time when the vessels arrives at the goal service and starts operating on the schedule of the goal service is called *phase-in*. The phase-out and the phase-in have to be within the planning horizon described before.

Furthermore, the repositioning coordinator has some freedoms in setting the path

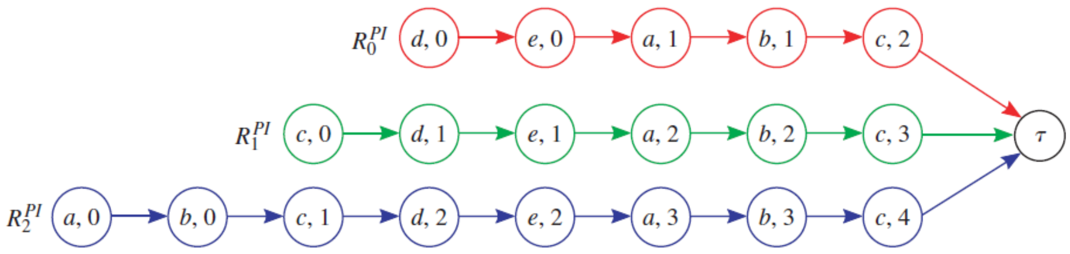


Figure 7.1.: A visualization of the graph structure of the LSFRP as presented by Tierney et al. (2014).

for the repositioning vessel. He can use port calls of the original service and the goal service. As described before, SoS are also an option to find relevant ports with cargo for the repositoning path. In addition to these options, the repositoning coordinator can *omit* or *induce* ports. It can be useful to omit a specific port, i.e. the vessel does not stop at this port, because the timing of the repositioning does not allow for this stop. On the other hand, a repositoning coordinator might decide to induce a port that otherwise would not be visited, because there is some cargo that can be transported. Such a port is also called *flexible* as this port is not included on the original service, the goal service or a SoS and therefore has no fixed time window for a visitation. All ports that are on a service have fixed time windows for a visitation and therefore are called *inflexible*.

The mathematical formulation of the LSFRP is taking the different activities, possible port calls and sailings and creates a graph that describes the possible repositioning plans for a vessel. Each node in this graph represents a port visit and the arcs between the nodes represent the sailings between port visits. Figure 7.1 uses this definition to present an example of three Phase-In options for a single repositioning vessel. In order to solve this problem, Tierney et al. (2014) present a simulated annealing approach. This approach is able to solve large, real-world problems, which is demonstrated by solving public LFSRP instances. A case study extends the results and shows that their algorithm has an advantage over a manual composed repositioning plan. Due to the low runtime of the approach, the authors show that this algorithm can be integrated in a decision support system for repositioning coordinators. A prototype of a decision support system is presented in Müller and Tierney (2017). For the system, the authors also extend the algorithm and the model to increase the flexibility for repositioning coordinators. They present several practical use cases to demonstrate the relevance for planners. In Becker and Tierney (2015), a new state-of-the-art algorithm is presented that combines the simulated annealing with a reactive tabu search algorithm. With this combination, the solutions for some of the previously mentioned public LFSRP instances could be improved.

7.3. A Biased Random-Key Genetic Algorithm for the LSFRP

7.3.1. Overview

Genetic algorithms are based on the evolutionary principle of natural selection and "survival of the fittest". The concepts of individuals, chromosomes, inheritance and mutations are used to find good solutions for optimization problems. Since the first presentation of the concept by Holland (1975), genetic algorithms have been developed to solve complex optimization problems like the vehicle routing problem (VRP) (Baker and Ayechew (2003)) or the job-shop scheduling problem (Pezzella et al. (2008)).

In the "classical" genetic algorithm, a specified number of individuals make up the population of each generation. Every individual has a certain chromosome, which is composed of bits. The structure of the chromosome (e.g. number of bits) is equal in every individual, but the specific value of its' chromosome differs. The chromosome describes a solution to the optimization problem at question, turning the population into a list of different solutions to the underlying problem. By using the objective function of the problem as a fitness function, the individuals and their chromosomes are evaluated. This leads to a ranking of the available solutions in the current generation of individuals. Depending on the fitness evaluation, individuals are selected to build the next generation of individuals. Individuals with a higher fitness value have a higher chance to be selected. In a process called crossover, the genes of two selected individuals at a time are then combined to generate a child individual. There are multiple ways to combine the genes of the two parent individuals. In the end, a child individual is generated, which contains genes from its parents, creating a totally new solution. Furthermore, a mutation process is employed, where single or multiple genes of some child individuals are altered.

The standard approach of genetic algorithms of defining problem specific chromosome encodings makes it necessary that for every problem to be solved, a new definition of the chromosome is necessary. Therefore, Bean (1994) proposed a different approach for encoding the chromosome: random keys. In random-key genetic algorithms (RKGA), instead of using bits, integers or other representations, floating point values in the range of $[0, 1]$ are used for building the chromosomes. With this approach, Bean (1994) aims at eliminating the offspring feasibility problem, which has been a problem for many applications of a GA.

No matter what the underlying problem is about, random keys are used to represent solutions to the problem. In order to evaluate an individual and its random key, a deterministic and problem specific decoder is used. This decoder uses the random-key chromosome to generate a solution from the solution space of the original problem. In GAs, it can occur that the crossover between two parent individuals does not necessarily produce a feasible offspring. By using random keys, the evaluation in the decoder is implemented such that the random keys lead to feasible solutions. By keeping the decoder deterministic, it is ensured that a specific random key will always result in the same solution.

In RKGAs, an elitist strategy is used to compose a new generation of individuals. In an elitist strategy, a certain percentage of the best individuals in a previous generation is directly copied to the next generation. The majority of the new generation is produced by the parametrized uniform crossovers (Bean (1994)) of randomly selected parents. In this crossover procedure, for each gene it is decided, which parent is used as the source of the gene. This decision can be biased towards one of the parents. Instead of mutating chromosomes of individuals, Bean (1994) proposes a process called "immigration". Here, at each generation totally new individuals are generated by creating new random keys.

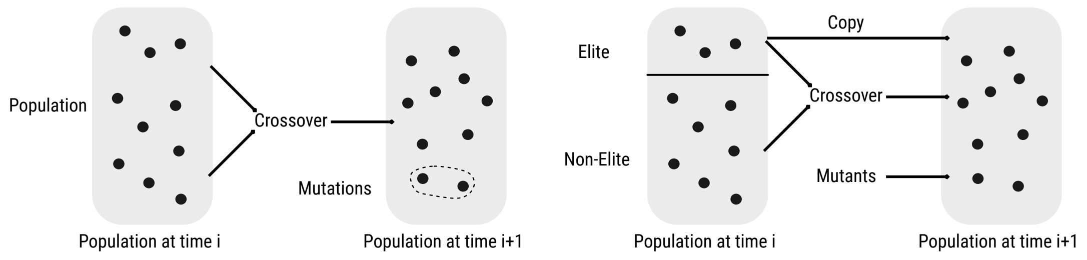
The random-key approach was extended by Gonçalves et al. (2011) by incorporating a different crossover strategy. In the so-called biased random-key genetic algorithm (BRKGA), the population is divided into an elite group, which contains the best individuals of the population. The rest is called non-elite. For the crossover, one parent is selected from the elite group and the other from the non-elite group, guaranteeing that one of both parents contains one of the best solutions of the current population. The selection of the alleles for the new chromosome is done according to a parametrized uniform crossover, which is described by Spears and DeJong (1991). A user-defined parameter determines the probability to select the allele of the elite parent. This parameter can be used to create a bias towards the elite parents by setting the probability above 50%.

Figure 7.2 shows a comparison of the creation of a new generation in a standard GA and the BRKGA. It demonstrates that the BRKGA defines an elite group of individuals, which are directly copied to the next generation. Furthermore, one parent is being selected from the elite group while the other parent is selected from the non-elite group. In contrast to this, the standard GA considers the whole population with equal probability for the selection of the parent individuals. The figure also hints at the difference at mutating the solutions in the next generation. In a standard GA, some individuals are selected and specific alleles are changed, creating a new solution. In BRKGAs, new individuals are randomly created and added to the next population.

7.3.2. Implementation of the BRKGA

The main difference between a standard GA and a BRKGA is the definition of the chromosomes of individuals. In standard GAs, a chromosome usually is very problem specific and gives a direct indication of how the solution looks like. This is not possible in a BRKGA, as the chromosomes are composed of random float values. These values are not directly connected to the problem or a solution of the problem. Therefore a problem specific decoder is needed, which constructs a solution based on the chromosome.

Algorithm 1 presents an overview of how the BRKGA is implemented for our experiments. Lines 1-7 describe the initialization phase of the algorithm. The instance is



(a) Standard GA (b) BRKGA

Figure 7.2.: Visualization of population changes in standard GAs and BRGKAs. Figure adapted from Gonçalves et al. (2011).

accepted as an input to this function as well as a parameter vector, which is described in Table 7.1.

Here, the instance file is parsed (line 1) and the relevant parameters for the algorithm are set (line 2). In the initialization phase, the first generation of individuals is also constructed (lines 4-6). Here, the population is filled with randomly generated individuals, which are then decoded and evaluated.

An important step for the decoder happens in line 2: The nodes of the instance are pre-evaluated. In this step, every node is evaluated regarding two different characteristics. As discussed in Section 7.3.3, this evaluation will be used in the decode function to construct a solution.

Line 7 describes the stopping criteria of the algorithm. There are three different criteria that need to hold in order to proceed (Hottung and Tierney (2016)). The first part states that the maximum number of generations (g^{max}) should not be exceeded, where g is the current generation number. Also, if the amount of generations after the last generation (g^l) with an improvement exceeds the limit (g^{ni}), the algorithm stops. Finally, there is a time limit on the algorithm, which is checked in line 7. These three criteria depend on the specific parameter settings, which are set in line 2.

The main process of the BRKGA is described in the lines 8 to 19, which closely resembles standard GA implementations. Line 8 and 9 demonstrate the use of an elitist strategy. In this strategy, the population is sorted according to their objective value and a specific portion of the best individuals is declared the elite. In the crossover step (line 9), this elite is used to perform the crossover operation with the rest of the population. Due to the biased approach of a BRKGA, the chromosome of the elite parent is preferred against the chromosome of the non-elite parent. In addition to the new individuals that are formed by this crossover operation, additional individuals are created as mutants (line 10). Mutants are completely new individuals, where no parents are used and the whole chromosome is created by randomly selected float values.

Algorithmus 1 : BRKGA for the LSFRP.

```
function BRKGA (instance, g)
    INITIALIZE_ALGORITHM();
    P ← INITIALIZE_POPULATION();
    best ← $-\infty$;
    best_individual ← null;
    foreach $p \in P$ do
        if DECODE(p) > best then
            best ← DECODE(p);
            best_individual ← p;
        end
    end
    while ($g \leqslant g^{max}$) and (($g$ - $g^{l}$) $\leqslant g^{ni}$) and (current_time < time_max) do
        E ← PICK_ELITE(P);
        M ← GENERATE_MUTANTS();
        P' ← E ∪ CROSSOVER(P\E, E) ∪ M;
        foreach $p \in P'$ do
            if DECODE(p) > best then
                best ← DECODE(p);
                best_individual ← p;
                $g^{l} \leftarrow g$;
            end
        end
        P ← P';
    end
    return best_individual;
```

Name	Description
g^{max}	Maximum number of generations
g^{l}	Last generation with an improvement
g^{ni}	Limit of generations with no improvement

Table 7.1.: Description of the parameters for BRKGA function (Algorithm 1).

The population of the next generation is then created, by using the elite group of the current generation, the individuals formed by the crossover step and the newly generated mutants. This population is then decoded and the objective value of each individual is checked whether it is better than the current best solution. If this is the case, the current best individual is stored and the counter for the last iteration with an improvement is set to the current generation number. When all individuals of the

Algorithmus 2 : Decoding an individual

```
function Decode (p)
    paths ← ∅;
    sorted_vessels ← Sort(vessels, p.V);
    foreach vessel in sorted_vessels do
        previous_visit ← null;
        current_visit ← vessel.start_Visitation;
        current_path ← ∅ ∪ current_visit;
        while not at instance.final_Visitation do
            possible_destinations ← Filter(current_visit.outgoing);
            if possible_destinations = ∅ then
                return infeasible;
            else
                sorted_destinations ← Sort(possible_destinations, p.D, p.M,
                  p.R);
                current_visit ← First(sorted_destinations);
                if previous_visit ≠ null and previous_visit.is_Flexible then
                    Set_Speed(previous_visit, current_visit, p.S, vessel);
                end
                if current_visit.is_Flexible then
                    if previous_visit ≠ null then
                        Set_Speed(previous_visit, current_visit, p.S, vessel);
                    end
                end
            end
            current_path ← current_path ∪ current_visit;
            previous_visit ← current_visit;
        end
        paths ← paths ∪ path;
    end
    p.Evaluation ← Evaluate(paths);
    return p.Evaluation;
```

new generation have been checked, the current generation is overwritten with the new generation and the algorithm returns to line 7.

Lines 7 to 19 are performed until at least one of the stopping criteria is met. Finally, the best individual is returned (line 21). If no feasible solution has been found, the algorithm would return no solution at all.

The process of decoding an individual is presented in Algorithm 2. It is identified as "Decode" in Algorithm 1. This algorithm is called with an individual as a parameter, therefore it must be called for every individual. It generates paths for every vessel. These paths are evaluated to determine the objective value of this solution. Once this solution is determined, this specific individual does not need to be evaluated again. The individual contains its chromosome and therefore the used weights in this function. These weights are used as vectors in this definition and describe the weights for vessel selection (V), the weights for direct costs to the next visitation (D), weights for minimal path costs (M), weights for expected revenue (R) and weights for the determination of vessel speeds (S).

The first step in the evaluation of an individual is the sorting of the vessels according to the vessel weights (V). For each vessel, the chromosome contains a weight value and the vessels are sorted in descending order of their weights. For each of these vessels, a path from its phase-out node to the phase-in node needs to be found. The phase-out is known for each vessel and is indicated by the term "start_Visitation" in line 4. As presented by Tierney et al. (2014), the graph of the instance contains a sink, a node without any outgoing connections. As soon as this node is found during the path generation of a vessel, the repositioning path has been found and a path is generated for the remaining vessels (line 6). The specific process of generating the path for a vessel is contained in lines 8 to 29.

Looking at the current visitation (either the starting visitation or a visitation later on the path), possible outgoing edges need to be determined. As each visitation can only be visited by a single vessel, it is important to filter out visitations that are not available for this path (line 9). In order to keep track of visitations that have been used by a vessel, a list of blocked visitations is maintained. In addition, this list contains visitations that are indirectly blocked. Visitations might be indirectly blocked if they have a single outgoing arc, which points to visitation that is used by a vessel.

In case there are possible destinations that can be used for the ongoing path generation, these destinations are sorted according to a weighted sum of direct costs for the next visitation, the minimal path cost to the graph sink and the potential revenue at the next visitation by using the weights from the chromosome (D, M, R) (line 13). After the possible destinations have been sorted, the best visitation of this list is selected (line 14) and added to the current repositioning path (line 24). In case the selected visitation is flexible, the arrival and departure time have to be set as well as the sailing speed on the incoming and outgoing arcs of this node (line 16 and line 20).

After a path for every vessel has been found, the complete repositioning plan is evaluated in order to be able to compare this solution to the solution of other individuals. The evaluation of the repositioning plan considers port call costs, sailing costs as well as fixed costs for the vessels. It corresponds to the objective function of Tierney et al. (2014).

Name	Description
v	Weight for every vessel
a	Weight for every arc of the graph
s	Weight for the determination of the vessel speed on flexible arcs
d	Weight for direct costs to the next visitation
m	Weight for minimal costs for a path to the graph sink
r	Weight for potential revenue of a visitation
o_j	Weight for each outgoing arc j of the starting visitations of the vessels

Table 7.2.: Description of the parameters for the four different BRKGA implementations.

7.3.3. Definition of the random keys

This section contains the definition and implementation details of four random-key definitions for the LSFRP. Table 7.2 gives an overview over the parameters of the random-key decoders. It should be noted that not all the parameters are used by every decoder.

Arc based key definition (ABK)

The path generation for the vessels of an individual are based on selecting relevant nodes along the vessel paths (see Algorithm 2). Therefore, a first simple approach of defining a random key for the LSFRP would be to define weights for each arc in the graph of an instance. In this random key, each vessel gets a weight assigned (v) in order to determine the sequence of the vessels during the path generation. In addition, every arc has its own weight in the random key, described by a. The last weight, s, is used to determine the vessel speed on flexible arcs. During the decoding of an individual, the arc weights are directly used to sort the possible next visitation on the repositioning path of a vessel (Algorithm 2). Therefore, the sort function of line 9 in Algorithm 2 would be modified to accept the vector of weights for every arc.

Visitation based key definition (VBK)

The information used in the random key of Section 7.3.3 to generate a vessel path only refers to the general structure of the graph. By using the weights of each arc to select the next destination in the path, there is no problem specific information used. Making use of further information, like potential costs and revenue might enhance the node selection in path generation.

For this random-key definition, a weight for each vessel (v) is also needed, to create a ranking of the vessels. Furthermore, the random key contains three weight values for each vessel. These weights are used to determine the specific path of each vessel. Weight d is used for direct costs between visitations, weight p is used for the minimal path cost to the graph sink and weight r is used for the potential revenue of a visitation. Again, the weight s is needed to determine the sailing speeds on flexible arcs.

For this random key to work, Algorithm 1 needs to be extended in the initialization phase (lines 1-3) by another method called "Pre_Evaluate". In this method, every node of the instance graph is evaluated in order to determine possible direct costs to the outgoing visitations and to determine the minimal path cost to the graph sink. During the decoding of an individual, the possible revenue of any relevant visitation is calculated on-the-fly. This can also be calculated beforehand in the initialization phase of Algorithm 1. By doing it on the fly, it is possible to consider the current state of the generated path and only take the demands into consideration that are available in this state. In the path generation, these weights and their respective values are then used to compute a weighted sum over the possible visitations. The visitation with the highest value of the weighted sum will be selected for the vessel path (Algorithm 2, lines 9-10).

Visitation based key definition with starting arcs (VBKSA)

This approach is an extension of the random-key definition of Section 7.3.3. Similar to the definition of Section 7.3.3, the chromosome of this approach also contains weights for the ranking of the vessels (v), the direct costs (d), minimal path costs (p), potential revenue (r) and the speed of the vessels (s). The extension of Section 7.3.3 lies within the weight o_j. This key is defined for every outgoing arc of the starting visitations of the vessels. When a path is generated for a vessel and the current visitation equals the starting visitation of the vessel, the weights defined by o_j are used to determine the next visitation. The procedure of selecting the next visitation resembles the approach of Section 7.3.3. After this first sailing of a vessel is determined, the arc weights are not used for the rest of the path. For this, the same procedure as in Section 7.3.3 is used to compute a weighted sum over direct costs, minimal path costs and revenue. In order to integrate this approach into Algorithm 2, line 9 need to be modified such that it uses the sort function with keys for the outgoing nodes from the starting visitation or the keys for the costs and revenue depending on the current visitation.

Visitation based key definition with all arcs (VBKAA)

The final key definition in this paper is a combination of the approaches in Section 7.3.3 and Section 7.3.3. The chromosome of this approach contains keys for the vessel ranking (s), the direct costs (d), the minimal path costs (p), the revenue (r), the vessel speed determination (s) and for every arc of the graph (a). Similar to Section 7.3.3, a weighted sum is computed to determine the next visitation on the generated path. For this, the weights for direct costs, minimal path costs, revenue and the arc weights are used. The visitation with the highest weighted sum value is used as the next visitation on the path. This approach is integrated into Algorithm 2 by considering the weight vector for all arcs in the sort function.

7.4. Computational Evaluation

The computational evaluation is based on inspired public instances that are presented in Tierney et al. (2014). These instances are inspired by real world liner services. An overview of these instances is given in Table 7.3. In total there are 44 instances, with varying numbers of vessels ($|S|$), graph nodes ($|V|$), inflexible and flexible arcs ($|A^i|$ & $|A^f|$), demands ($|M|$) and sail-on-service opportunities ($|SOS|$). Furthermore, the table shows the instance IDs as well as the number of ports with equipment surpluses or demands ($|E|$).

The number of vessels of the instances range from three to eleven. The size of the graph ranges from 36 nodes to 379 nodes and from 150 arcs to 13,705 arcs. Furthermore, the number of demands ranges from 20 to 1,423. Not all instances have ports with equipment surpluses/demands or sail-on-service opportunities. It should be noted that the range of vessel count is not evenly distributed over the instances. There are nine instances with three vessels, but only one instance with ten vessels.

Almost all of these instances have been solved to optimality in the publication of Becker and Tierney (2015). An optimal solution is not known for the instances 35p and 38p. The optimal solutions have been used by Tierney et al. (2014) and Becker and Tierney (2015) to evaluate the performance of their algorithms and will be used for the same purpose in this paper. The goal of this computational evaluation is the comparison of the proposed random-key definitions in Section 7.3.3 with the state-of-the-art algorithm presented by Becker and Tierney (2015). This algorithm will be discussed in the following section. The optimal solutions are the baseline for the comparison.

Each instance was solved 25 times with a time limit for each run of ten minutes. All experiments were performed on Intel Xeon E5-2670 2.6GHz CPUs with 4 GB of RAM. The RTS-SA was started with the same configuration as in Becker and Tierney (2015). For the BRKGA, the population size of each generation is set to 100 individuals, with 10% belonging to the elite and 30% being mutants.

7.4.1. RTS-SA for LSFRP

The current state-of-the-art results on the given instances are achieved by an algorithm presented by Becker and Tierney (2015). They extend the simulated annealing algorithm of Tierney et al. (2014) with a tabu search strategy, resulting in a hybrid approach. When the RTS-SA starts, it first performs an iteration of the simulated annealing. After that, the algorithm alternates between the tabu search and the simulated annealing part until a solution is found.

There are different neighborhoods that are used in this algorithm, some from Tierney et al. (2014), others are added by Becker and Tierney (2015). These neighborhoods

ID	$\lvert S\rvert$	$\lvert V\rvert$	$\lvert A^i\rvert$	$\lvert A^f\rvert$	$\lvert M\rvert$	$\lvert E\rvert$	$\lvert SOS\rvert$
1p	3	36	150	0	28	0	1
2p	3	36	150	0	28	0	2
3p	3	38	151	0	24	0	2
4p	3	42	185	0	20	0	3
5p	3	51	270	0	22	0	3
6p	3	51	270	0	22	0	3
7p	3	54	196	0	46	0	4
8p	3	108	1,185	126	50	6	3
9p	3	108	1,185	126	50	10	3
10p	4	58	449	0	125	0	0
11p	4	62	499	38	125	6	0
12p	4	74	603	0	145	0	2
13p	4	80	632	0	155	0	4
14p	4	80	632	0	155	24	4
15p	5	71	355	0	173	0	0
16p	5	106	420	0	320	0	5
17p	6	102	1,198	0	75	0	0
18p	6	135	1,439	0	87	0	9
19p	6	135	1,439	0	87	33	9
20p	6	142	1,865	0	80	0	4
21p	6	142	1,865	0	80	13	4
22p	6	142	1,865	0	80	37	4
23p	6	153	1,598	159	89	71	9
24p	7	75	482	0	154	0	3
25p	7	77	496	0	156	0	0
26p	7	77	496	0	156	16	0
27p	7	79	571	0	188	0	0
28p	7	90	618	0	189	0	4
29p	7	90	618	0	189	19	4
30p	8	126	1,450	0	265	0	0
31p	8	130	1,362	0	152	0	0
32p	8	144	1,501	0	170	0	3
33p	8	212	2,227	433	179	50	3
34p	9	304	10,577	0	344	0	0
35p	9	357	11,284	35	874	118	4
36p	9	364	11,972	0	1,048	0	4
37p	9	371	11,371	0	1,023	114	7
38p	9	373	11,972	35	1,048	126	4
39p	9	379	11,666	0	1,109	0	7
40p	9	379	11,666	0	1,109	118	7
41p	10	249	8,051	0	375	0	0
42p	11	279	6,596	0	1,423	0	5
43p	11	320	13,058	0	1,013	0	0
44p	11	328	13,705	0	1,108	0	4

Table 7.3.: Properties of the instances tested for this work, from Tierney et al. (2014).

include add, remove and swap operations on the visitations as well as various path completion procedures: random path completion, demand destination completion and demand source completion. In the random path completion, a random vessel with its path and a particular visitation along this path are selected. All subsequent visitations are then removed from the path in order complete it with a random path to the graph sink. In the demand destination and the demand source completion, a random vessel is selected and its path is evaluated whether a demand exists where the origin or the destination nodes are not included in the path. In this case, the demand is not taken yet and both completion strategies try to vary the path such that it includes the origin and destination nodes of this particular demand. Both procedures try to make as little as possible variations on the original vessel path. The last neighborhood of the RTS-SA is an ejection chain, which were introduced by Glover (1991) in order to overcome local optima that are found by neighborhoods producing small changes. In the case of the RTS-SA, the ejection chain is used for node swaps.

ID	Optimal	RTS-SA		ABK		VBK		VBKSA		VBKAA	
1p	-39.83	**-39.83**	**0.00**	**-39.83**	**0.00**	**-39.83**	**0.00**	**-39.83**	**0.00**	**-39.83**	**0.00**
2p	-39.83	**-39.83**	**0.00**	**-39.83**	**0.00**	**-39.83**	**0.00**	**-39.83**	**0.00**	**-39.83**	**0.00**
3p	-61.77	**-61.77**	**0.00**	**-61.77**	**0.00**	**-61.77**	**0.00**	**-61.77**	**0.00**	**-61.77**	**0.00**
4p	-46.62	**-46.62**	**0.00**	**-46.62**	**0.00**	**-46.62**	**0.00**	**-46.62**	**0.00**	**-46.62**	**0.00**
5p	-8.21	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**
6p	-8.21	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**
7p	-11.49	**-11.49**	**0.00**	**-11.49**	**0.00**	**-11.49**	**0.00**	**-11.49**	**0.00**	**-11.49**	**0.00**
8p	-8.21	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**
9p	-8.21	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**	**-8.21**	**0.00**
10p	137.61	**137.61**	**0.00**	135.83	1.29	**137.61**	**0.00**	**137.61**	**0.00**	**137.61**	**0.00**
11p	137.61	**137.61**	**0.00**	-	-	**137.61**	**0.00**	**137.61**	**0.00**	**137.61**	**0.00**
12p	138.55	**138.55**	**0.00**	136.42	1.54	**138.55**	**0.00**	137.61	0.68	138.22	0.24
13p	138.86	**138.55**	**0.00**	132.05	4.91	**138.86**	**0.00**	136.78	1.50	137.92	0.68
14p	138.86	**138.86**	**0.00**	136.78	1.50	138.24	0.45	136.78	1.50	136.78	1.50
15p	-36.59	**-36.59**	**0.00**	-37.30	1.93	-37.30	1.93	-37.30	1.93	-37.30	1.93
16p	-36.59	**-36.59**	**0.00**	-37.30	1.93	-37.30	1.93	-37.30	1.93	-37.30	1.93
17p	-9.36	-9.90	5.72	-14.30	52.75	**-9.36**	**0.05**	-10.72	14.49	-9.50	1.52
18p	5.22	**5.22**	**0.00**	-4.30	182.45	**5.22**	**0.00**	0.22	95.80	**5.22**	**0.00**
19p	5.22	**5.22**	**0.00**	-4.52	186.54	3.77	27.86	-0.84	116.07	**5.22**	**0.00**
20p	-11.85	**-11.85**	**0.00**	-15.11	27.53	-14.41	21.59	-14.25	20.23	-12.96	9.40
21p	-11.85	**-11.85**	**0.00**	-15.18	28.11	-12.60	6.30	-14.41	21.59	-14.41	21.59
22p	-11.85	**-11.85**	**0.00**	-14.41	21.59	-14.35	21.11	-13.61	14.83	-14.41	21.59
23p	5.22	**5.22**	**0.00**	-7.84	250.11	-3.96	175.89	-8.21	257.37	-3.01	157.62
24p	-53.89	**-53.89**	**0.00**	**-53.89**	**0.00**	**-53.89**	**0.00**	**-53.89**	**0.00**	**-53.89**	**0.00**
25p	-53.13	**-53.13**	**0.00**	**-53.13**	**0.00**	**-53.13**	**0.00**	**-53.13**	**0.00**	**-53.13**	**0.00**
26p	-53.13	**-53.13**	**0.00**	**-53.13**	**0.00**	**-53.13**	**0.00**	**-53.13**	**0.00**	**-53.13**	**0.00**
27p	-28.20	**-28.20**	**0.00**	**-28.20**	**0.00**	-28.53	1.17	**-28.20**	**0.00**	**-28.20**	**0.00**
28p	-32.13	**-32.13**	**0.00**	**-32.13**	**0.00**	-32.14	0.04	-32.14	0.04	**-32.13**	**0.00**
29p	-32.13	**-32.13**	**0.00**	**-32.13**	**0.00**	-32.14	0.04	**-32.13**	**0.00**	**-32.13**	**0.00**
30p	5.72	**2.04**	**64.32**	-0.09	101.50	-	-	-	-	-	-
31p	-12.08	**-12.08**	**0.00**	-14.45	19.65	-17.34	43.54	-16.01	32.56	-15.06	24.67
32p	-10.92	**-10.92**	**0.00**	-13.29	21.73	-18.92	73.27	-17.02	55.90	-13.51	23.72
33p	-10.92	**-10.92**	**0.00**	-	-	-	-	-	-	-	-
34p	-2.01	**-4.79**	**138.53**	-78.03	3782.22	-46.93	2234.73	-56.48	2709.75	-36.47	1714.21
35p	-	**133.97**	-	-	-	-	-	-96.35	-	-	-
36p	160.02	**156.77**	**2.03**	-120.86	175.53	-	-	-160.24	200.14	-343.38	314.58
37p	139.31	**134.76**	**3.27**	-84.60	160.73	-193.76	239.09	-76.00	154.56	-147.54	205.91
38p	-	**156.77**	-	-	-	-	-	-98.85	-	-	-
39p	161.53	**155.61**	**3.67**	-90.67	156.13	-128.77	179.72	-86.00	153.24	-117.47	172.72
40p	161.53	**155.61**	**3.67**	-95.86	159.34	-139.50	186.36	-77.79	148.16	-64.39	139.87
41p	-39.60	**-56.95**	**43.81**	-238.48	502.22	-	-	-222.17	461.04	-	-
42p	253.60	**225.49**	**11.08**	-	-	-	-	-	-	-	-
43p	223.98	**175.62**	**21.59**	-95.14	142.48	-	-	-98.46	143.96	-87.83	139.21
44p	254.06	**188.53**	**25.79**	-	-	-	-	-152.72	160.11	-	-

Table 7.4.: Every arc has a weight, no other evaluation, highest weight indicates the path.

7.4.2. RTS-SA vs. BRKGA

Table 7.4 shows the optimal solutions for all used public LSFRP instances and the solutions found by the RTS-SA. Furthermore, every of the four key definitions is included in this table with two columns. The first column of each key definition contains the objective function value, the second column contains the gap to the optimal solution. The best solutions among the four proposed key definitions is printed in bold.

It can be seen that the RTS-SA finds a solution for all instances, even the instances where an optimal solution is not known at the moment. In contrast to this, the BRKGA-ABK with weights for every arc does not find a solution for all instances. Instead there are six instances without a solution of the BRKGA-ABK. With the given key, 15 instances are solved optimally and six more have a gap to the optimal solution of less than 5%. Another group of six instances have a gap value of about 50% or less and the remaining instances have large gaps, the highest gap being 3782.22%.

It should be noted that the instances with the larger gaps tend to be instances with eight or more vessels. Except for one instance (11p), the instances that could not be solved by the BRKGA-ABK also have eight or more vessels. The average gap over all solved instances is 157.47%. This means that the solution of the BRKGA-ABK with weights for every arc is on average 1.5 times lower than the optimal solution.

The results of the BRKGA-VBK can be found in the columns with the heading "VBK". In this key definition, problem-specific information like costs on arcs and possible revenues are used to find the best repositioning. Similar to the results of the BRKGA-ABK, not every instance could be solved by using the random-key definition with direct costs, minimal path costs and expected revenues. In total, there are nine instances that could not be solved, three more as the previous key definition. There are 18 instances that have been solved optimally and three instances, where the gap is less than 1%. In addition, there are three instances with a gap of less then 5%. Four instances have a gap of less than 50% and the remaining instances have larger gaps with the highest being 2234.73%. The average gap over all solved instances is 91.86%.

The columns with the "VBKSA" heading contain the results of the BRKGA-VBKSA definition. This key definition extends the BRKGA-VBK weights only for the arcs that leave the starting visitations of the vessels. With the extended key definition, 15 instances could be solved optimally. Furthermore, there are two instances with a gap of less than 1% and four instances with a gap of less than 5%. There are five instances where the gap is less than 50% and the remaining instances have larger gaps, except for five that could not be solved at all. It should be noted that similar to the previous results, the highest gaps occur at instances with vessel numbers of eight or higher. The average gap over all instances is 122.24%.

The BRKGA-VBKAA definition is the last proposed key definition. The results are presented in the "VBKAA" columns of Table 7.4. The table shows that there are 19 instances that could be solved optimally and seven instances that could not be solved at all. Two instances have a gap of less than 1% and for more instances have a gap of less than 5%. The rest of the instances have either a gap to the optimal solution of less than 50% (five instances) or even larger gaps (seven instances). It should be noted that the highest gap with this key definition is smaller than the highest gap with the other key definitions. On average, the gap to the optimal solution is 79.81%, which is the smallest compared to the previous definitions.

7.4.3. Overview & further analysis

As shown in the previous section, all proposed key definitions for a BRKGA to solve the LSFRP have failed to achieve similar outcomes as the current state-of-the-art algorithm RTS-SA. As presented in Table 7.5, the RTS-SA achieves an average gap over all solved instances of only 7.71%, which is only about a tenth of the gap of the best performing key definition. While the proposed key implementations have problems with instances with nine or more vessels. Furthermore, the RTS-SA is able

Key definition	Solved	Optimal	∅ gap (%)	σ (%)
ABK	38	15	157.47	604.56
VBK	35	18	91.86	372.74
VBKSA	39	15	122.24	430.34
VBKAA	37	19	79.81	281.89
RTS-SA	42	31	7.71	23.95

Table 7.5.: Summary of the results of all four random-key definitions.

Key definition	Solved	Optimal	∅ gap (%)	σ gap (%)
VBKAA	37	19	79.81	281.89
VBKAA with hill climbing	40	27	8.75	23.41
RTS-SA A/R/S	44	9	101.39	170.07
RTS-SA DC	44	13	46.51	100.83
RTS-SA PC	44	30	5.01	11.32
RTS-SA EC	44	28	10.57	27.88
RTS-SA SC	44	31	7.71	23.94

Table 7.6.: Summary of the further analysis of varying configurations of the RTS-SA and the BRKGA-VBKAA.

to solve almost all instances with eight or less vessels optimally.

Therefore, there is no doubt that the RTS-SA has a better performance on the used public instances and should be the preferred choice for decision support systems. Nonetheless, the proposed BRKGA implementations have shown, that for smaller instances they are also able to find optimal solutions.

In order to further evaluate the performance of the BRKGA approach, the best performing key definition (VBKAA) is selected and compared against the performance of the RTS-SA when some of the neighborhoods are excluded. Furthermore, the BRKGA-VBKAA implementation is extended by a hill climbing algorithm that makes use of some of the neighborhoods described in Tierney et al. (2014). These selected neighborhoods include add, remove, demand completion and supply completion. For this extension, the implementation as shown in Algorithm 1 is modified such that the time for the BRKGA is limited to 90% of the total time. The last 10% of the total time available to solve an instance is spent on the hill climbing algorithm. The results of this analysis are presented in Table 7.6[2]. For the different RTS-SA configurations it should be noted that each line contains adds a new type of neighborhood to the configuration. Therefore, RTS-SA DC not only includes the demand completion neighborhood, but also the add, remove and swap neighborhoods.

The table shows that the integration of the hill climbing algorithm greatly improves the overall performance and solution quality. Compared to the BRKGA-VBKAA without hill climbing, three more instances are solved and seven more instances could be solved optimally. The average gap can be reduced to 8.75%. The RTS-SA algorithm also benefits from the various neighborhoods. RTS-SA with the add, remove and swap neighborhood operators only solves nine instances optimally, compared to 31 optimal

[2]DC = demand destination completion, SC = demand supply completion, A/R/S = add/remove/swap, PC = path completion, EC = ejection chain

solutions for the full configuration. Also the average gap decreases when different neighborhoods are included. Interestingly, the smallest average gap can be seen for the results of the RTS-SA PC configuration, but the most optimally solved instances are found in the RTS-SA SC configuration.

When the different RTS-SA configurations are compared to the different VBKAA configurations, it becomes obvious that the VBKAA solutions are able to outperform parts of the RTS-SA configurations. VBKAA without hill climbing achieves more optimal solutions than RTS-SA A/R/S and RTS-SA DC, although it was only able to generally solve 37 instances compared to 44. VBKAA with hill climbing shows a better performance and shows similar values as the RTS-SA SC configuration for average gap and standard deviation in the gap. Only for the number of solved instances as well as the number of optimally solved instances, the VBKAA with hill climbing is worse than any RTS-SA configuration except for RTS-SA A/R/S and RTS-SA DC.

The computational evaluation has shown that by adding these neighborhood operators, the solution quality of the BRKGA is greatly improved. For some instances, the previously not known optimal solution was found in the first iterations of the hill climbing algorithm, showing that the BRKGA solution was “close” to the optimal solution. There are two explanations for this behavior. It is debatable whether the optimal solution would have been found if there would have been more time for the BRKGA to solve the instances. This would mean that the inherent learning mechanism of the BRKGA in its current configuration is too slow for larger instances. For some instances, it was observed that the BRKGA got stuck in local optima, which were hard to leave in order to continue the exploration of the search space. An adjustment of the BRKGA configuration might help in improving the diversification of the solutions. As described by Blum and Roli (2003), a premature convergence to local optima often is a major difficulty in setting up a genetic algorithm.

7.5. Conclusion and future research

In this paper, we presented four different random-key definitions for a BRKGA to solve the LSFRP. It is shown how the BRKGA can be implemented and how the proposed keys are used to decode an individual’s chromosome. This algorithm was chosen to beat the current state-of-the-art: a RTS-SA implementation, which was briefly described in this paper. With an extensive analysis, it has been shown that the BRKGA is not able to outperform the RTS-SA with the proposed key definitions. In an attempt to improve the general BRKGA performance a hill climbing algorithm was added to the solution procedure of the BRKGA with selected neighborhood operators from Tierney et al. (2014). This greatly improved the performance of the BRKGA, but still leaves some gaps to the current state-of-the-art.

As described in the previous section, the BRKGA might show a better performance with a different parameter setting. Future research may use parameter tuning approaches to find the best configuration of the algorithm parameters. Furthermore, other types of neighborhoods operators might be suitable for this algorithm such that they could be included in the BRKGA with hill climbing. Until then, the RTS-SA proposed by Becker and Tierney (2015) is still the best choice when it comes to selecting a heuristic for the LSFRP.

Bibliography

Baker, B. M., Ayechew M. A. (2003). A genetic algorithm for the vehicle routing problem. Computers & Operations Research 30:787–800.

Bean, J. C. (1994). Genetic Algorithms and Random Keys for Sequencing and Optimization. Journal on Computing 6:154–160.

Becker, M., Tierney, K. (2015). A Hybrid Reactive Tabu Search for Liner Shipping Fleet Repositioning. Computational Logistics, 123-138.

Blum, C., Roli, A.: Metaheuristics in Combinatorial Optimization - Overview and Conceptual Comparison. *ACM Computing Surveys*, 35(3):268–308, 2003.

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

Christiansen, M., Fagerholt, K., Nygreen, B., Ronen, D. (2013). Ship routing and scheduling in the new millennium. European Journal of Operational Research, 228(3), 467-483.

Christiansen, M., Fagerholt, K., Ronen, D. (2004). Ship routing and scheduling: Status and perspectives. Transportation science, 38(1), 1-18.

Glover, F. (1991). Multilevel tabu search and embedded search neighborhoods for the traveling salesman problem. Graduate School of Business, University of Colorado.

Goldberg, D. E. (1989). Genetic algorithms in search, optimization and machine learning.

Gonçalves, J. F., Resende, M. G. (2011). Biased random-key genetic algorithms for combinatorial optimization. Journal of Heuristics, 17(5), 487-525.

Holland, J. H. (1975). Adaptation in natural and artificial systems. MIT Press, Cambridge.

Hottung, A., Tierney, K. (2016). A biased random-key genetic algorithm for the container pre-marshalling problem. Computers & Operations Research, 75, 83-102.

Meyer, J., Stahlbock, R., Voß, S. (2012). Slow steaming in container shipping. In: 45th Hawaii International Conference on System Science (HICSS). IEEE, pp. 1306-1314.

Müller, D., Tierney, K. (2017). Decision support and data visualization for liner shipping fleet repositioning. Information Technology and Management, 18(3), 203-221.

Pezzella, F., Morganti, G., Ciaschetti, G. (2008). A genetic algorithm for the flexible job-shop scheduling problem. Computers & Operations Research, 35(10), 3202-3212.

Spears, W. M., DeJong, K. A. (1991). On the virtues of parameterized uniform crossover. Proceedings of the Fourth International Conference on Genetic Algorithms, 230–236.

Stopford, M. (2009). Maritime economics 3e. Routledge.

Tierney, K. (2015). Optimizing liner shipping fleet repositioning plans (Vol. 57). Springer.

Tierney, K., Áskelsdóttir, B., Jensen, R. M., Pisinger, D. (2014). Solving the liner shipping fleet repositioning problem with cargo flows. transportation science, 49(3), 652-674.

Tierney, K., Coles, A. J., Coles, A., Kroer, C., Britt, A. M., Jensen, R. M. (2012). Automated Planning for Liner Shipping Fleet Repositioning. In ICAPS (pp. 279-287).

United Nations Conference on Trade and Development (2017). Review of maritime transport. United Nations, New York.

8. Conclusion

8.1. Summary

This thesis has covered selected topics from the area of liner shipping with a special focus on planning problems surrounding modifications of a liner shipping service network. Chapter 1 and Chapter 2 give an introduction into the topics of liner shipping, decision support systems and selected metaheuristics. Part II contains the research papers that have been produced for this thesis.

The main contributions of the thesis can be summarized as follows:

- Extension of the liner shipping fleet repositioning problem to increase interactivity
- Evaluation of different visualization techniques for liner services
- Development of a prototypical DSS for the liner shipping fleet repositioning problem
- Integration of the fleet deployment problem and the liner shipping cargo allocation problem
- Development of models with varying complexity for the liner shipping single service design problem
- Empirical analysis of vessel travel times with real-world data
- A biased random-key genetic algorithm with hill climbing to solve the liner shipping fleet repositioning problem

In Chapter 1, four research goals have been stated: (1) extending the scope of selected optimization models from liner shipping, (2) developing techniques for better visualization of liner services, (3) analysis of real-world shipping data and (4) proposing a decision support system to integrate developed models and visualization techniques.

As shown in Chapter 4, a reasonable approach to extend the scope of optimization models in liner shipping is focusing on increasing the flexibility and interactivity of optimization models. Such an approach might lead to higher acceptance rates among planners of such a model. Furthermore, Chapter 5 showed that the integration of related decisions into a single model, improves the decision quality by using synergies that otherwise wouldn't be available.

For the second goal, various visualization techniques have been presented in Chapter 4. These techniques are evaluated according to their capabilities to visualize specific characteristics of liner services. Among other, these characteristics include ports and their sequence within the service, vessels, the network structure and the time frame.

An empirical analysis of real-world shipping data has been performed in Chapter 6. This work shows that real-world data can make great contributions to the field of liner shipping. By using published service schedules and spatial data of a selected group of vessels of these services, it was possible to compute a distribution for the lateness of the vessels. This distribution was used to develop the optimization models as well as the simulation model in Chapter 6.

Regarding the last research goal, Chapter 4 proposes a web-based decision support system specifically for the liner shipping fleet repositioning problem. It includes different visualization techniques to display liner services. Furthermore, it integrates the proposed extensions to the optimization model of Chapter 4. The system is developed such that the solution algorithm could be exchanged, making it possible to use the proposed biased random-key genetic algorithm of Chapter 7. Although the planning tasks of Chapter 5 and Chapter 6 are different than the planning tasks of Chapter 4 and Chapter 7, they could be integrated into the proposed system. With minor modifications of the data model and the graphical user interface, the system would be able to present relevant information to the planner for each of the problems. As all chapters discuss problems in liner shipping, they all work with liner services, vessels, ports and time. Therefore, the proposed visualization techniques could be used for any planning task discussed in this thesis.

8.2. Future research

As discussed in the individual chapters, the proposed models and solutions still leave room for improvements and extensions. The following list presents a summary of ideas for future research:

- The performance of the optimization model presented in Chapter 5 can be improved by using a column generation or a heuristic approach.
- The optimization model from Chapter 5 includes assumptions about cargo handling and piloting times that can be relaxed in a stochastic model.
- Additional empirical studies can be executed in order to analyze liner shipping operations and improve the precision of optimization and simulation models in this area.

Some of the presented ideas show that a close collaboration between real-world planners and researchers would be ideal to develop practical optimization models.

Even if such a collaboration is not always possible, there are many options to contribute to the field of liner shipping and hopefully this thesis can be a motivation for such a contribution.