Dissertation

Liner Shipping Network Design

Decision Support and Optimization Methods for Competitive Networks

M.Sc. Stefan Guericke

Schriftliche Arbeit zur Erlangung des akademischen Grades
doctor rerum politicarum (dr. rer. pol.)
im Fach Wirtschaftsinformatik

eingereicht an der
Fakultät für Wirtschaftswissenschaften der
Universität Paderborn

Paderborn, Dezember 2014

Gutachter:
1. Prof. Dr. Leena Suhl
2. Prof. Dr. Wilhelm Dangelmaier

The future has many names:
For the weak, it means the unattainable.
For the fearful, it means the unknown.
For the courageous, it means opportunity.

Victor-Marie Hugo

Acknowledgements

This thesis is the result of the research I conducted as a member of the *International Graduate School of Dynamic Intelligent Systems* and the *Operations Research & Decision Support Lab* at the University of Paderborn. I would like to thank everybody that supported me to complete this dissertation within three years.

Foremost, I would like to thank Prof. Dr. Leena Suhl for sparking my interest on Operations Research during my studies at the University of Paderborn. Furthermore, for providing me the opportunity to work in an enjoyable and productive atmosphere at her chair. I would like to thank my supervisors Prof. Dr. Wilhelm Dangelmaier and Prof. Dr. Gregor Engels for the guidance and the ensuring of a descent research quality.

The colleagues at the DS&OR working group provided a challenging and motivating research atmosphere. In particular, I want to thank Jun.-Prof. Dr. Kevin Tierney for his critical feedback on the work, Christoph Weskamp for always having time for mutual discussions, and my office colleagues Torben Schramme and Marius Merschformann for the pleasant time.

I am deeply thankful for the opportunity of doing the PhD as an *ORCONOMY GmbH* fellow. I would like to thank the managing directors Dr. Ingmar Steinzen, Dr. Jens-Peter Kempkes and Dr. Stefan Bunte for their guidance and the critical discussions. I would like to thank the whole ORCNOMY team for the support during the last three years, and especially Dr. Stephanie Heller for the feedback on the mathematical proof and sharpening my view on mathematical details.

I am indebted to the team of the International Graduate School of Dynamic Intelligent Systems, its head apl. Prof. Dr. Eckhard Steffen, and his assistance Astrid Canisius for the incredible organization, support and the excellent PhD program.

I would like to thank anonymous referees for the feedback, discussions and profound insight into liner shipping network planning.

Credits go as well to all students that participated to the success of this dissertation with critical questioning and inspirational ideas.

I am deeply indebted to my whole family for sharing their interest in this exciting research through discussions and support. In particular, I want to thank my parents who always believed in me, financially safeguarding me during my studies and always supporting me during my endeavors.

Most important, I deeply thank Daniela to share worries, delights and dreams.

Stefan Guericke
Paderborn, December 2014

Contents

1. Introduction

According to the World Trade Organization, the worldwide merchandise trade was estimated worth an 18.3 trillion US$ in 2013[1], which is a worldwide annual increase of 5.3% compared to 2012. The largest proportion of worldwide trade is transported by sea: 80% of the weight and 70% of the value (see UN Conference on Trade and Development (2013) and (Schieck, 2008, p. 177)).

Usually, the shipping industry is distinguished into three different operation modes: industrial, tramp and liner (see Christiansen et al. (2004)). In industrial shipping, enterprises own a fleet or have long-term time charter contracts, which basically make them the ship owner. The enterprises are thereby responsible to manage it profitably and ship their cargo to minimal costs. In tramp shipping, the operator owns or charters a fleet and serves available cargo with a basic contracted amount and tries to maximize the profit by working on the spot market (Christiansen et al. (2004)). Tramp shipping usually transports break bulk, such as coal or grain, and liquids, such as liquefied natural gas or crude oil (Stopford, 2009, p. 55). The third mode of operation is liner shipping and is the scope of this thesis.

Liner shipping has been an important operation mode that emerged at the end of the 19th century with the introduction of steamships (Stopford, 2009, p. 28, 31). Liner operators steam according to a published regular schedule, similar to a bus line, independent of the utilization of the vessels (Christiansen et al. (2004), Schieck (2008)). From a customer's (shipper's) point of view, the advantages are regular transportation opportunities, relatively reliable sailing schedules and predictable transit times (Brooks, 2000, p. 2). Today, liner shipping is mainly connected with the transportation of general cargo in standardized containers, which accounted for 16% of the seaborne trade in tons and 50% of the value in 2013 (see UN Conference on Trade and Development (2013)). The main advantage of the containerization of commodities during the 1950s is the decreased port time due to increased intermodal efficiency (Levinson (2006)).

Beside the twenty foot equivalent unit (TEU), many different specialized container types exist that allow the transportation of break bulk and liquids as well. In general, container vessels are not competitive compared to large scale break bulk vessels for high volumes (Schieck, 2008, p. 209) and are rather used to transport small volumes for specific end customers. Other commodities that are transported by liner operations are vehicles (using roll-on/roll-off vessels) or project cargo, such as sailing boats or other bulky commodities (Schieck, 2008, p. 182,185).

[1] Press release from 2014-04-14 http://www.wto.org/english/news_e/pres14_e/pr721_e.htm

Several mergers and acquisitions in the liner shipping industry resulted in a highly concentrated market with a few, dominating carriers (see Stopford (2009)). The largest 15 liner carriers operate about 79.3% of the worldwide capacity in TEU (see Table 1.1)[2], that is either owned or chartered. The four largest carriers provide about 50% of the worldwide capacity. All of the carriers listed in Table 1.1 operate global networks and serve almost all areas in the world (Ihde, 2001, p. 136).

		Capacity		
Rank	**Operator**	**TEU**	**%**	**∑%**
1	APM-Maersk	2,728,567	15.6%	15.6%
2	Mediterranean Shipping Company	2,481,967	14.2%	29.8%
3	CMA CGM Group	1,578,483	9.0%	38.8%
4	Evergreen Line	886,577	5.1%	43.8%
5	COSCO Container L.	786,585	4.5%	48.3%
6	Hapag-Lloyd	767,703	4.4%	52.7%
7	CSCL	635,625	3.6%	56.3%
8	Hanjin Shipping	593,739	3.4%	59.7%
9	APL	578,772	3.3%	63.0%
10	MOL	569,582	3.3%	66.3%
11	OOCL	500,500	2.9%	69.2%
12	Hamburg Süd Group	498,826	2.8%	72.0%
13	NYK Line	494,458	2.8%	74.8%
14	Yang Ming Marine Transport Corp,	402,786	2.3%	77.1%
15	Hyundai M.M.	377,319	2.2%	79.3%

Table 1.1.: Largest liner operators, July 2014. The percentage is based on the existing worldwide cellular fleet of 4,992 ships with 17,785,515 TEU, (source: `http://www.alphaliner.com/top100`).

The top 15 liner carriers concentrate on nine countries (see Table 1.2). Today, the Chinese carriers COSCO, CSCL, Hanjin and Yang Ming provides the largest capacity. The largest single carrier is Maersk Line in Denmark, followed by the Mediterranean Shipping Company (MSC) in Switzerland and the French CMA CGM Group. The German carriers Hapag-Lloyd and Hamburg Süd Group provide about 7.2% of the worldwide capacity, followed by other Asian operators.

	Capacity	
Country	**TEU**	**%**
China	2,919,235	16.7%
Denmark	2,728,567	15.6%
Switzerland	2,481,967	14.2%
France	1,578,483	9.0%
Germany	1,266,529	7.2%
Japan	1,064,040	6.1%
Taiwan	886,577	5.1%
Singapore	578,772	3.3%
South Korea	377,319	2.2%

Table 1.2.: Operated capacity of the top 15 carriers by country, aggregated data from Alphaliner's top 100 ranking, July 2014, (source: `http://www.alphaliner.com/top100`).

[2]The capacities in Table 1.1 includes owned and chartered vessels

Currently, liner carriers are facing different challenges. The increasing competition due to market regulations and increased vessel sizes lead to decreased freight rates in the last years (Stopford (2009)). Additionally, bunker prices increased by more than 200% in the last five years (see Section 5.5). As a result, liner carriers have very low operating margins[3]: On average, carriers had to deal with a negative margin of -2.6% in the first quarter of 2014 (see Alphaliner Newsletter, Volume 22 (2014)).

These challenges let the liner carriers implement different strategies beside mergers and acquisitions, such as reducing the vessel speed, further increase the vessel size or cooperating with each other. These strategies result in frequent adjustments of the operated network. This is a very complex task that requires many steps, such as assessing the network's feasibility and evaluating the network performance on a monetary basis. The network must also be coordinated with other partners the carrier is cooperating with.

1.1. Research Goals

The research scope of this thesis is to evaluate Operations Research (OR) methods on real-world strategic planning problems of liner shipping carriers, namely the cargo allocation and liner shipping network design problem (LSNDP). The cargo allocation problem determines the cargo routing and service speeds on a given liner network to evaluate it, whereas the network design problem determines the port sequences and capacities. In particular, this thesis has the following four goals:

1. Evaluate large scale real-world liner networks
2. Formalize practical requirements for the LSNDP and generate optimal solutions
3. Optimize the LSNDP for real-world instances in reasonable time
4. Integrate the developed methods into a decision support system

The cargo allocation problem (CAP) allows for the evaluation of a given liner shipping network. In this thesis, the practical requirements for the planning task are analyzed and optimization methods to support manual planning are developed. We refer to manual planning as the process to design liner services without automatically improving the networks. The cargo allocation problem evaluates designed networks on a monetary basis. An important requirement that has to be dealt with are fast solution times in order to get an instantaneous feedback on the impact of the modified network. To the best of the authors knowledge, integrating cargo allocation with several practical aspects, such as speed optimization, vessel drafts,

[3]Ratio of operating income divided by net operating revenues.

empty container repositioning and partner networks has not been done in previous publications.

Based on the CAP, the more complex LSNDP's requirements are analyzed from a practical point of view. Based on these requirements, the research goal is to evaluate the potential of OR methods in the network design. The benefit for the network planners are automated improvements of existing networks and the exploration of non-obvious potential alternatives. The liner shipping network design problem includes all aspects of the integrated cargo allocation problem and extends them by transit time and embargo aspects. This has not been done to the full extent in previous publications.

The integration of the methods into a prototypical planning decision support system (DSS) increases the usability to support network planners and releases the potential of the optimization methods. In the scope of this thesis, challenges such as the graphical presentation are approached.

These goals are derived from the research gap based on the state-of-the-art and are presented in more detail in Section 3.4.

1.2. Outline

This thesis comprises seven chapters and is structured as follows: Chapter 1 corresponds to this introduction which outlines the liner shipping market and introduces the research topic.

In Chapter 2, the liner network planning process is outlined and the liner shipping network design components are presented.

Based on this problem definition, Chapter 3 reviews models and solution approaches for the liner shipping network design and interwoven planning problems. It provides an overview of the state-of-the-art regarding the cargo allocation and network design problem. Based on the research gap and the practical requirements, the goals of this thesis are derived in more detail.

Chapter 4 presents a new approach to solve the integrated liner shipping cargo allocation problem. In particular, the speed optimization and the empty container repositioning is integrated into a single planning problem. In the scope of this thesis, two different mathematical formulations and solution approaches are developed and evaluated on the LINER-LIB benchmark instances[4].

In Chapter 5, different approaches for solving the liner shipping network design problem are described. More specifically, a mixed integer model is presented that can be used to determine optimal liner networks. Next, two heuristic methods are presented that determine good networks in a reasonable amount of time. To further speed up the heuristics, the effects of approximate evaluations (so called surrogates)

[4]The LINER-LIB benchmark suite contains data on ports, demands and vessel types and was published in Brouer et al. (2013).

on the heuristics are analyzed. A sensitivity analysis shows the effects of varied bunker cost on the network design. The concept can be used to analyze under which bunker price changes carriers should adjust their network to cope with the new situation. Finally, the developed methods are applied on a real-world liner network to show the practical applicability.

In Chapter 6, the developed decision support system is presented. An example process integration, functional description and implementation details to increase the usability and acceptance for network planners are presented.

Finally, Chapter 7 concludes this thesis with a summary, a critical assessment of the goals and methods and an outlook on future research topics.

2. Liner Shipping Network Planning

The objective of the liner shipping network planning in the liner industry is managing the network efficiently. The network design problem is a complex strategic planning problem, interwoven with several subproblems. This chapter is organized as follows: First an overview of other liner shipping planning problems is given. Then, the basic liner shipping network design problem, including the network structure, transhipment operations and capacities, is presented. In the remaining sections the reader is introduced into relevant subproblems and extensions of the basic problem that are required to solve practical instances.

2.1. Planning Process Overview

In the liner shipping industry, different planning problems arise that depend on the considered planning horizon. Schmidt and Wilhelm (2000) distinguish the strategic, tactical and operational planning horizon. Strategic decisions are made for a relatively long time period of two to five years and determine the scope of the tactical planning problems with a shorter horizon of 6 to 24 months (Kjeldsen (2011), Schmidt and Wilhelm (2000)). Operational planning problems are typically solved on a daily or weekly basis in the scope of the tactical planning decisions.

Figure 2.1 shows selected planning problems from the liner shipping industry in the different planning horizons. The investment in new vessels and their type are strategic decisions ((Kjeldsen, 2011, p. 130), Meng and Wang (2011b)) because container ships can have a lifespan between 15 and 26 years ((Bendall and Stent, 1987, p. 339), World Shipping Council (2014)). Another long term decision of a liner carrier is the relevant regional focus. Based on the existing fleet and the regions, the network design problem faces the decisions which liner services can be offered to the customers. This liner shipping network design problem is sometimes also called the vessel routing problem (Kjeldsen (2011), Álvarez (2009)). Depending on the publication, the planning horizon can vary from several years to several months (Brouer et al. (2013), Reinhardt et al. (2007) and Reinhardt and Pisinger (2010)).

From the tactical point of view, a specific fleet must be deployed on the services (see for example Álvarez (2009)). Another problem is fleet repositioning that optimizes the transition between two liner networks (Tierney and Jensen (2012) and Tierney et al. (2014)).

Operational planning problems are for example revenue management (Zurheide and Fischer (2012)), environmental routing (Windeck (2013)), vessel schedule recovery (Brouer et al. (2012), Meng et al. (2013)) and stowage planning (see for example

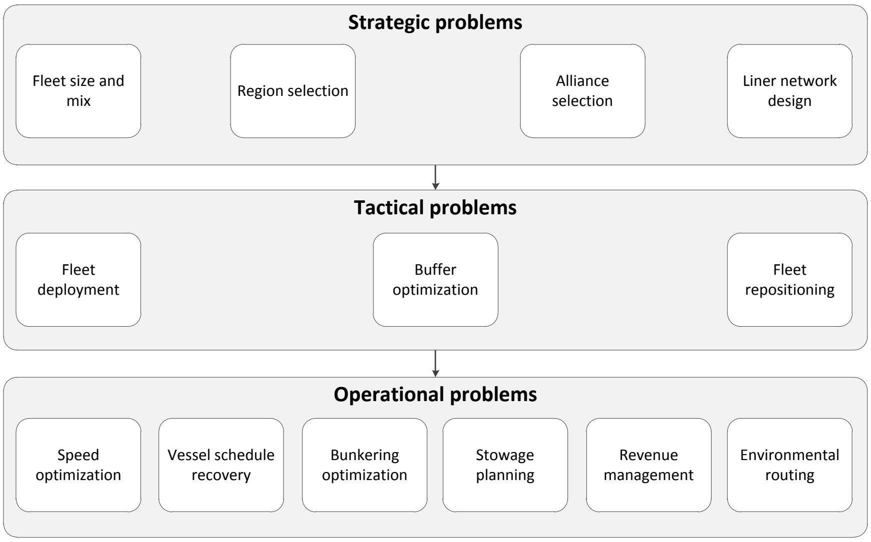


Figure 2.1.: Selected planning problems arising in the (liner) shipping industry in the context of service planning and operating.

Avriel et al. (1998), Imai et al. (2006), Delgado et al. (2012) and Pacino (2013)).

The scope of this thesis is the liner shipping network design problem in a strategic and tactical planning horizon. The problem is defined in more detail in the remainder of this chapter.

2.2. Basic Liner Shipping Network Design Problem

In this section, the basic network design problem is described in detail. The planning task is to define one or more liner services that are offered to the customer. Stopford extends the definition of a liner service (also called sling, string or just service) from Fayle (2005) as:

> ”A liner service is a fleet of ships, with a common ownership or management, which provide a fixed service, at regular intervals, between named ports, and offer transport to any goods in the catchment area served by those ports and ready for transit by their sailing dates. A fixed itinerary, inclusion in a regular service, and the obligation to accept cargo from all comers and to sail, whether filled or not, on the date fixed by a published schedule are what distinguish the liner from the tramp.” (Stopford, 2009, p. 512)

According to this definition, a liner service consists of a served port sequence (port rotation) that performs a round trip, a (single) deployed vessel type and a number of vessels. The connection between two ports is called a *leg* in shipping context. The vessel type determines the volume available on a regular (often weekly) basis, whereas the number of vessels are required to hold the fixed itinerary's frequency subject to the vessel type's minimum and maximum speed. The importance of fixed itineraries is also highlighted by (Brooks, 2000, p. 89) since the shippers and their customers negotiate import and export contracts for a year or more. Thus, the shippers plan with reliable services and do not appreciate carriers changing the service characteristics too frequently (Brooks, 2000, p. 89). Today, the fleet of ships on a single liner services can be operated by several carriers, for example in vessel sharing agreements (see Section 2.6). Furthermore, carriers are not obligated to accept any cargo, but instead are able to select the most profitable.

A liner shipping network contains several liner services that are connected at different ports. The set of all ports is assumed to be given in the scope of this thesis. Parameters involved in the port selection are, for example, the hinterland freight rates (see Rodrigue (2013), (Levinson, 2006, p. 10 and p. 170)), port congestion and charges (see Slack (1985)). The reader is referred to the work of Slack (1985), Malchow and Kanafani (2001) and Lirn et al. (2004) for port evaluation strategies.

At almost all ports, transhipment operations can take place. Transhipment is the transfer of goods from one transportation mode to another and/ or from one liner service to another or from one ship to another. In the scope of this thesis, transhipment is performed between two services at a specific port. An important reason to perform transhipment is freight consolidation which increases the utilization on legs that do not fully utilize a transport carrier (Ballou, 1999, p. 249). Ports where large volumes of containers are consolidated are called *transhipment hubs*. For further transhipment hub benefits, see (Mattfeld, 2006, p. 2). Usually, transhipment operations take place in free-trade zones. Thus, goods can be stored in the zone without being subject to import duties. Typically, ports allow the free storage of transhipment containers for several days. For example, in the port of Hamburg, dry transhipment containers are free of charge for six days (see EUROGATE Container Terminal Hamburg GmbH (2014)). The main drawback of transhipment operations are the increased transit times due to the storage of containers at the container yards and the associated costs (see Section 2.5.2).

Transhipment imposes huge complexity when optimizing liner shipping networks. In this paragraph some of the challenges involved with transhipment operations are presented for the network given in Figure 2.2. The network consists of three liner services, each with its own characteristics. Services typically follow different trades, i.e. connections between markets (see Section 2.4). The network in Figure 2.2 serves trades between Northern Europe, South America and Asia. Tangier is a transhipment hub where cargo is transhiped from on service to another, allowing high utilization for the transport between all geographical regions. Each liner service calls different ports to bundle regional demand. As can be seen in the example service, Brazil has Salvador in the center of its coast line and several other ports in the

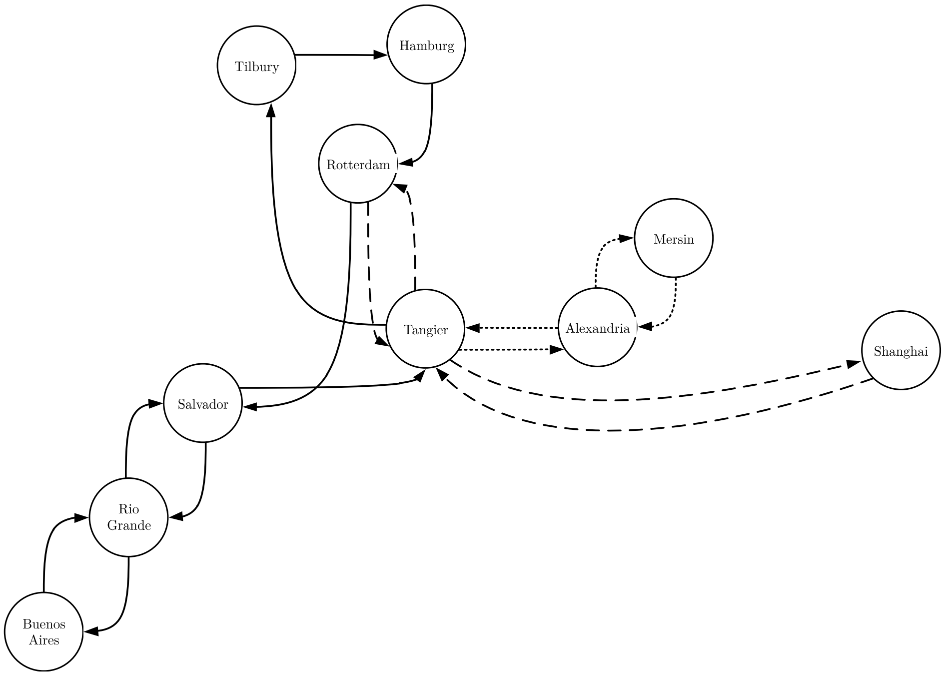


Figure 2.2.: Example liner network with three services.

South. Thus, a large quantity of Brazil's demand is served by Salvador in the North. Since Brazil exports a lot of fresh products, the port is called on the service's North bound to offer low transit times for the perishable goods for Northern Europe and the Mediterranean region. The route from China to Europe calls the ports Shanghai, Tangier and Rotterdam in each direction. On the west bound direction's Tangier call, Chinese cargo with the destination of the Mediterranean region is unloaded and further transported by a feeder service. The cargo from the Mediterranean region can be picked up in Tangier at the return journey back to China. Thereby, the Asia service has enough capacity to transport cargo from Northern Europe to the Mediterranean region. Before leaving Tangier to return to China, the Asia service can pick up cargo from South America leading to higher utilization on the East bound direction with laden containers (see Section 2.8 for trade imbalances). The South America service is also used to transport cargo within Brazil and between Brazil and Argentina. North bound of this service, a connection in Rio Grande and Salvador is used to free capacity South bound and keep the duration between the ports of origin and European destinations short.

Beside the port rotation, the used capacity is also fixed for a liner service. Capacity is provided by the deployed vessel type. Stopford differs seven container ship classes that are extended by (Konings, 2008, p. 115) by ultra large container vessel (ULCV) (see (Stopford, 2009, p. 584)). These classes are given in Table 2.1. However, the

Vessel type	**Vessel type TEU capacity**
Feeder	0-499
Feedermax	500-999
Handy	1000-1999
Sub-Panamax	2000-2999
Panamax	3000-3999
Post-Panamax	4000-5999
Very large box carrier (VLBC)	6000-12000
ULCV	1200-18000

Table 2.1.: Typical container vessel types (see (Stopford, 2009, p. 584) , (Konings, 2008, p. 115)).

first vessel with 18000 TEU maximum capacity was already deployed by Maersk Line (see Maersk Line (2014)).

The vessel types in Table 2.1 can be distinguished in three groups: Feeder and feedermax vessels with relatively low capacity are used on short-haul operations. Short-haul operations are referred to as regional distribution operations for deep-sea services. Handy and Sub-Panamax vessels can be used for short-haul feeder operations but also longer distances where port draft constraints restrict the use of larger vessels, for example in the North Sea (see (Stopford, 2009, p. 582)). Larger ships are used on long-haul trades where they spend up to 80% of their round trip time at sea. Each vessel type has its own characteristics such as speed, bunker consumption or load depending draft. Details on the capacities and the stated TEU are given in Section 2.7.1. Details on the vessels' speeds is given in Section 2.5.1.

Typically a carrier only owns part of its operated fleet. Depending on the carrier's strategy, up to 100% of the used capacity can be chartered (see Alphaliner Newsletter, Volume 1, 2011 (2014)). For purchased vessels, depreciation has to be paid and for time chartered vessels a charter rate has to be paid to the shipowner. This rate can be highly volatile, depending on the current market situation. In the scope of this thesis, chartered and owned capacity is not distinguished and the time charter rate is assumed to be constant. Successive planning steps that deploy either own or chartered vessels to the liner services must ensure further constraints, such as cabotage or manning, that are not in the scope of network design.

Brooks (2000) states that shippers rely on regular services. This regularity is reflected by the liner services port call frequency. Consider the South America service in example Figure 2.2 with seven deployed vessels using a weekly frequency. This leads to a round trip time of $7 \cdot 7 = 49$ days for each vessel. This means that each vessel has a maximum of 49 days to perform the journey. The round trips are performed consecutively, removing any buffer time after a round trip. Delays are thereby potentially propagated to the next round trip. For further details and an example schedule, see Section 2.5.1.

The last basic aspect of a liner shipping network are the used seaways. Although ships can use nearly every route between two ports, major shipping routes were established in the last centuries. Figure 2.3 shows the routes and their usage. The

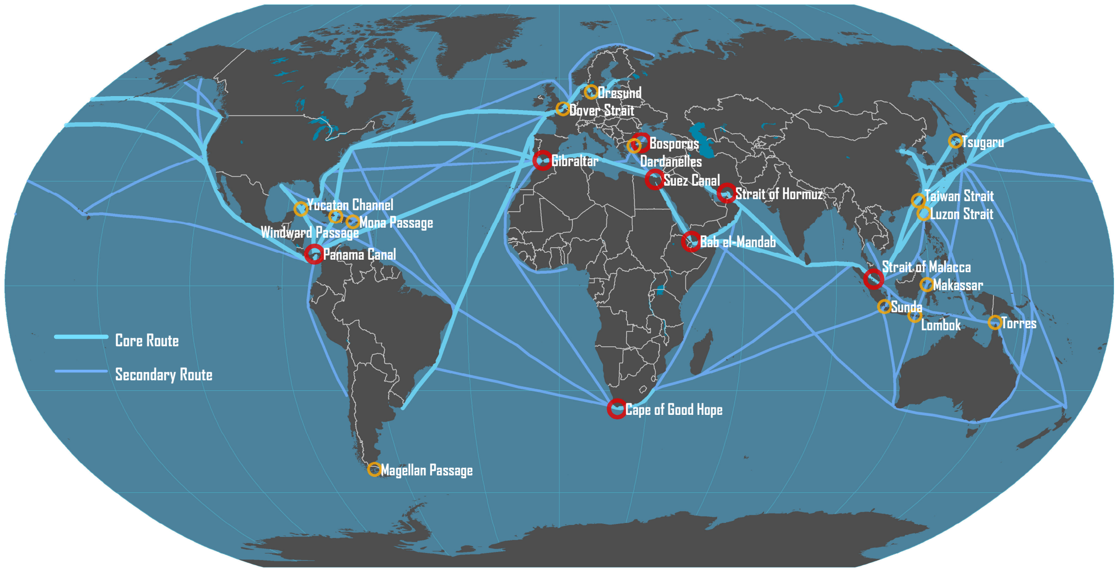


Figure 2.3.: Main global maritime shipping routes, Rodrigue (2013).

route selection and the exact distance between the ports rely on a large number of factors, such as weather, security implications and costs and can differ on each journey. For simplicity, it is assumed that the distance between two ports is constant and externally given. To calculate the distances either distance providers such as www.sea-distances.com (2014), www.sea-rates.com (2014) or the method presented in Appendix E can be used.

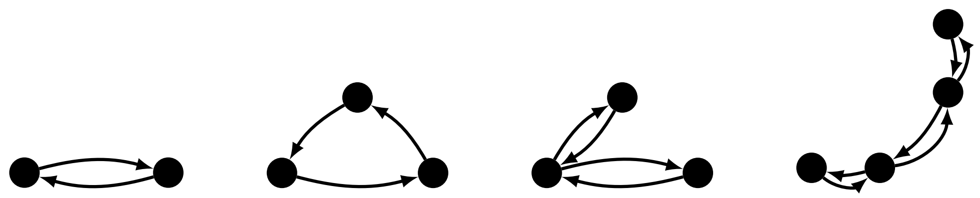
Figure 2.3 shows the high volume trades between northern Europe and Asia and the trade between the North America east coast and Asia. One can see that carriers heavily rely on using canals for their transportation. Although the usage of canals can cost several hundred thousand US$, the alternatives usually lead to much larger distances and much higher bunker costs. The canals with the highest vessel frequencies are today the Suez and the Panama canal. Using the Cape of Good Hope instead of the Suez on a journey between Shanghai, China and Hamburg, Germany results in an additional distance of about 4000 nautical miles (nm) and additional 11.9 days at 14 knots (kn). Assuming a bunker consumption of 43 tons per day with cost per ton of 600 US$, this alternative journey would cost at least 1,500,000 US$, without additional fixed cost such as manning, maintenance or additional vessels to hold the weekly frequency. Thus, there is basically no alternative in using the large international canals on direct connections. Since 2010, the Suez canal has been able to deal with the deadweight of the largest currently deployed ULCVs (see Suez Canal Authority (2014)). The Panama canal's bottleneck are its locks that are currently being expanded, which will allow container vessels of up to 13,000 TEU to pass the locks with the planned opening in 2015 (see (Canal de Panamá, 2014, p. 6)). This Atlantic-Pacific expansion enables routes from Northern Europe to Asia via America and could lead to the merging of existing services as well as the implementation of new trade routes (see Rodrigue (2010)). For a detailed study on the impact of the new Panama locks on the North America freight distribution see Rodrigue (2010).

In the following sections, further details on the network design problem are given. These include the liner service route types, transported cargo, timing aspects, capacities, costs and revenues.

2.3. Route Types and Network Structure

In general, liner services are classified as deep-sea and short-sea (also feeder) services. Deep-sea services serve Transatlantic, Transpacific and Asian trades. Feeder services are usually connected to one or more hubs and pick up or deliver cargo within a smaller geographical region.

Additionally, liner services can be distinguished by their type of port rotation, namely pendulum, cyclic, butterfly and conveyor belt routes (see (Andersen, 2010, p. 28) and (Plum et al., 2013b, p. 2)). Pendulum and cyclic routes are often referred to as simple; butterfly and conveyor belt routes as non-simple routes.



(a) Pendulum. (b) Cyclic. (c) Butterfly. (d) Conveyor belt.

Figure 2.4.: Liner services with different route types.

Pendulum routes in Figure 2.4(a), from a literature point of view, alternate between two ports. Cyclic routes call more than two ports, without calling a port twice per round trip (see Figure 2.4(b)). Butterfly routes in Figure 2.4(c) are cyclic routes that call one port twice. Conveyor belt routes (or routes with multiple butterfly ports) in Figure 2.4(d) call more than two ports and visit more than one port twice per round trip. The advantages of butterfly and conveyor belt routes are the increased capacity on single legs, decreased vessel draft due to increased port calls and potentially improved transit times (see (Plum et al., 2013b, p. 2 and 3)).

In practice, pendulum routes are more often characterized as routes that transport cargo without transhipment and usually call more than two ports. Discussions with liner operators indicated that they rarely, if ever, call the same port more than twice per round trip. This thesis uses the terminology for route types found in literature.

It is assumed that all routes perform a round trip, i.e. that the start equals the end port.

2.4. Transportation of Containerized Cargo

In this section, the structure of the underlying demand, called *cargo flows* in shipping, is explained in more detail. Furthermore, the available container types and sizes are presented.

2.4.1. Demand Structure

Before containerization started in the 1950s, the transported cargo shipped by sea was classified into four segments: Bulk cargo, liquids, and refrigerated cargo. These segments where mainly served by tramp shipping or general cargo liners (see (Stopford, 2009, p. 29)). The post- World War II liners transported a large variety of cargo on the same vessel. These vessels required intensive work by longshoreman to load and unload a vessel. Between 60 - 70% of the costs to transport cargo by sea were accounted to the dock operations (see (Levinson, 2006, 21)), and each port call required several days or weeks. A solution that was introduced in 1956 is the containerization of cargo. This also let the shipping system shift to specialization: homogeneous bulk cargo are transported by bulk carriers or tankers and liquids or automobiles by specialized ships. General cargo is seen as small either loose or unitized cargo that was containerized in the 1950s and is transported by containerships in standardized containers today (see (Stopford, 2009, p. 36)). General cargo can be low value commodities such as scrap metal, lumber or newsprint, medium valued goods such as textile, fruits or vegetables and high value commodities such as consumer electronics and completely or semi knocked down automobiles (see Klug (2010)).

The specific commodity that is transported via containers from an origin to a destination depends on each country's import and export trades. For example, Germany imports the commodity groups coffee, fish meal and fruits from Peru, whereas electrical machinery, apparels and furniture are imported from China (see UN (2014)). The mentioned commodity groups are often transported via containers by sea and specialized break bulk ships (e.g. banana ships), although fruits or high value commodities compete with air freight transportation in case of high employed capital costs or perishable goods.

The majority of cargo transported by carriers is based on medium term contracts. The Federal Maritime Commission (2001) reports that in 2001 about 80% of the cargo was transported under service contracts, whereas Yi (2008) reports up to 95%, depending on the leg and the products. The contracts are usually negotiated once per year between the carrier and the shipper (customer). Concerning the quantities specified in the service contracts, 60% had a minimum commitment of 100 TEUs or less (see Yi (2008)). The committed range was between 1 TEU up to 68,000 TEUs (see Federal Maritime Commission (2001)). Beside the committed freight, carriers also transport spot cargo. Spot cargo is short-term cargo that is typically not known for certain when the liner services are planned. Thus, forecasts can be included in the cargo flows used to plan the services.

In the scope of this thesis, cargo flows are used on a port to port basis without the knowledge of the specific commodity, transported in the container. This is common practice in shipping since it allows the focus on moving goods in standardized units efficiently (see Levinson (2006)).

2.4.2. Container Sizes and Types

For the transportation of commodities, several standardized container types and sizes exist (see International Organization for Standardization (2014)). The possible lengths are 6.06m for a twenty foot equivalent unit (TEU), 12.19m for a forty foot equivalent unit (FEU) and 13.72m for a 45 foot container. Containers are either 2.59m (8.6 ft) or 2.90m (9.6 ft) high, the latter are called high cube containers. For details on the history of container size standardization, see (Levinson, 2006, Chapter 7).

There are a variety of container types that can be used to transport different commodities (see Table 2.2). Fresh or perishable goods require the use of ventilated or refrigerated (for chilled or frozen products) containers. Each liner carrier can provide more commodity specific container interiors, such as for hanging cargo. Between 2002 and 2012, especially reefer containers had a high combined annual growth rate (CAGR) of 3.6% (see Research (2013)) which was partly driven by the exotic fruits' CAGR of 9.1%. In 2004, the worldwide container stock was estimated at 28.5 million TEU, whereof 18% were TEU, 75% FEU, 4% reefer containers and 3% others (such as open top) (see (Stopford, 2009, p. 510)). The revenue and weight varies per container type and is considered in this work. Considering the weight is important to assure the correct vessel draft (see Section 2.7).

Name	Purpose
General purpose container	For any general cargo
Open top container	Over high cargo or crane loading from top side
Flat container	Heavy or oversize cargo and project cargo
Ventilated container	Air circulating container, e.g. for coffee and fruits
Refrigerated (reefer) container	Cooling or refrigerating container for example for meat
Tank container	Transportation of chemicals, alcohols, fruit juices etc.

Table 2.2.: Selection of frequently used container types.

Especially perishable commodities transported in ventilated and reefer containers often require a specific transit limited time between the origin and destination.

These durations are described in the next section.

2.5. Timing Aspects

Liner services are designed with a fixed round trip time that introduces several requirements on the port and sea duration. In this section, the effects on the network planning and the transportation of cargo subject to transit times are presented.

2.5.1. Port Calls and their Impact on the Vessels' Speed

Historically, the majority of the deep sea services have been offered on a weekly basis (see for example Agarwal and Ergun (2008), Yan et al. (2009), Stopford (2009) and Song and Dong (2013)). Although liner shippers sometimes offer bimonthly or biweekly port departures (see Reinhardt and Pisinger (2010)), variable frequencies are rarely considered in network design (see Brouer et al. (2013) for a weekly and biweekly formulations).

Depending on the frequency, a number of vessels has to be deployed on the service to respect the service frequency. This results in a maximum port call duration to perform a round trip for each vessel.

Figure 2.5 shows a deployed service that have to be served weekly. The number of vessels per service must be within the range resulting from the minimum and maximum speed and the service distance. The service's total distance is 6876 nm. Assuming deploying a vessel type with a minimum speed of 13 Knots (kn) and 24 kn maximum speed, this would lead to $6876nm/13kn = 22.03$ days and $6876nm/24kn = 11.94$ days respectively, per round trip. Thus, $\lceil 22.03/7 \rceil = 4$ vessels using minimum or $\lceil 11.94.03/7 \rceil = 2$ vessels using maximum speed are required to hold the weekly round trip time. The same capacity is offered per week either way because weekly round trips are assumed. The exact number of vessels is determined by the port durations (see Figure 2.6). It shows a service calling Hamburg,

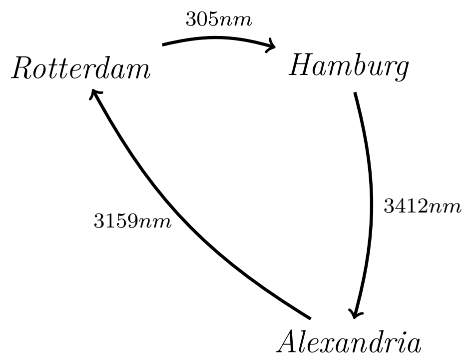


Figure 2.5.: Round trip of example service with approximate leg distances.

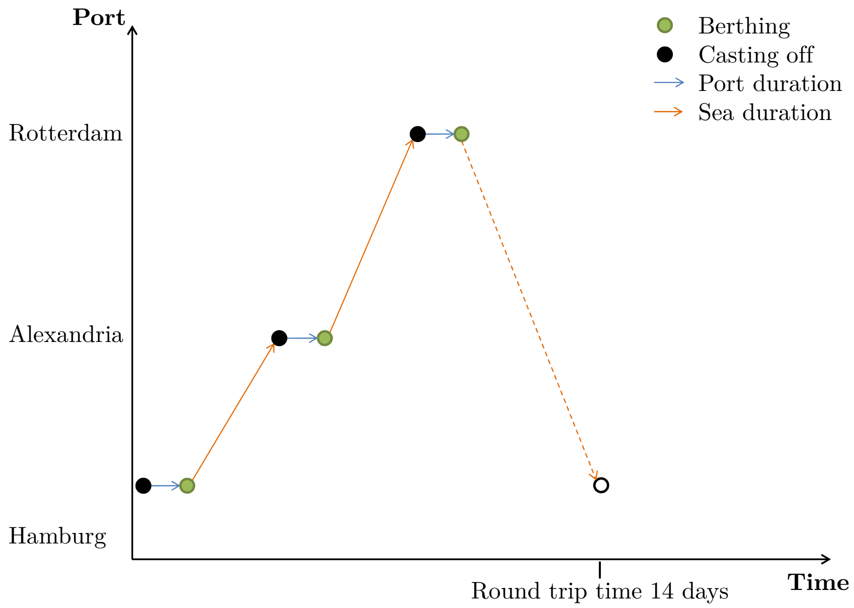


Figure 2.6.: Duration of example service with two vessels serving the ports on a weekly basis.

Alexandria and Rotterdam with two vessels and a weekly frequency.

A vessel's round trip time is divided into the duration at sea and the duration at ports. Increasing the port duration (by more container moves) decreases the time at sea because the service's distance stays constant. Therefore the speed must be increased and this can lead to a capacity shortage when the maximum speed is reached. A trade-off between increased bunker cost and time charter cost occurs when more vessels are deployed to decrease the speed. Decreasing the speed lower than 20 kn is called *slow steaming* in the shipping industry (see for example Rodrigue (2013)). However, slow or even super slow steaming strategies can have a negative impact on the transit time (see (Notteboom and Vernimmen, 2009, p. 335) and Section 2.5.2). In practice, the port duration is determined by the following port processes:

1. Berthing
2. Loading/unloading preparation
3. Cargo loading/unloading
4. Bunkering (i.e., fueling the vessel)

5. Port buffer to compensate uncertainties (such as port congestion etc.)

6. Casting off

The processes berthing and casting off are performed by the port's tug boats and typically take between one and several hours (for example six to seven hours through the Elbe river to reach some of the container terminals in the port of Hamburg, Germany). In the scope of this thesis, all processes except loading and unloading preparation and bunkering are considered. The duration of loading and unloading cargo at the ports is determined by the moved cargo. Other process' durations are given by external parameters that can be used to respect further port processes. There is ongoing work in the research community that optimizes the additional buffer in each port (see for example the work of Wang and Meng (2011), Wang and Meng (2012a) and Wang and Meng (2012d)).

By using the port processes presented above, a proforma schedule can be generated. *Proforma schedules* define the arrival and departure time at each port within a service and provide information on the duration at sea. Successive planning methods can adjust the port call durations, the buffer time and the leg specific speed to meet the service synchronization. Synchronization is an important task when transhipping perishable goods, adjusting the services with partners and handling tide and container terminal berth windows.

2.5.2. Transit Times in the Maritime Context

Transit times are a major criterion for the service quality of a liner carrier (see Notteboom (2006), Panayides and Song (2013) and Wang and Meng (2014)) and highly influence the structure of the overall network. In general, *transit times* define the maximum duration between two ports. Transit time can be seen from different perspectives: From an ocean carrier's point of view that offers schedules on a regular basis, transit time can be defined "as the number of sailing days on a port-to-port basis" (Notteboom (2006)). From a broader logistics perspective, transit time is the door-to-door duration in days, including intermodal transportation systems. Transit times are especially important for perishable commodities but also for the customers' demand planning that is based on determined replenishment lead times (see (Busch and Dangelmaier, 2002, p. 198)). Three factors determine the transit time between ports:

1. Port sequence

2. Vessel speed and the duration at sea

3. Port call durations

The port call sequencing is crucial for the transit time: Ports that are called last within a region and first in another region provide the lowest transit times. This is especially important on the intercontinental trade-bundling services (see Notteboom (2006)). The speed of the vessels determine the duration to steam the leg between two ports and play another important role. Beside the days at sea, also the port call duration highly influences the transit time. Today, liner carriers make increasing use of hub-and-spoke systems that connect major deep-sea services with short-sea services (see (Hsu and Hsieh, 2005, p. 209) and (Imai et al., 2009, p. 756)). Because transhipment operations at these ports also impact the transit time, the duration of transhipment operations must be considered as well.

In the scope of this thesis, transit times are considered on a port-to-port basis due to the unknown commodity groups that are transported. The transit times are regarded as a strategic or tactical decision of the carrier to create competitive liner networks.

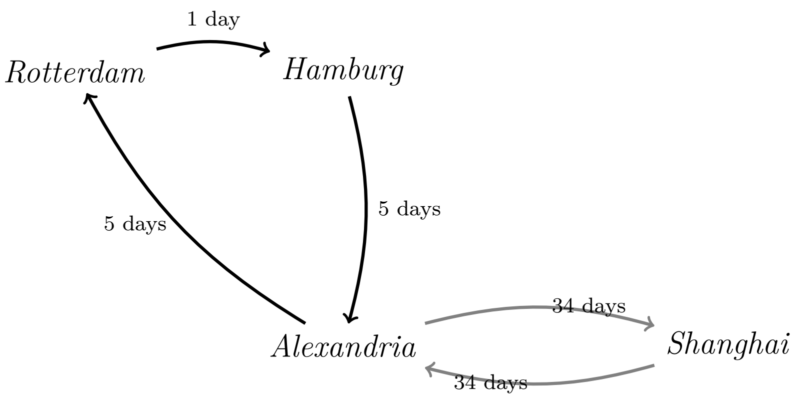


Figure 2.7.: Example transit time between Rotterdam and Shanghai using service A (black) and B (gray).

To get an insight into how transit times are considered in this thesis, see Figure 2.7. For simplicity, assume a fixed port duration of 24 hours per port call. Service A calls Rotterdam, Hamburg and Alexandria with two vessels whereas service B calls Alexandria and Shanghai with 10 vessels. The leg durations already consider the distance and required speed. The transit time between Rotterdam and Shanghai is the path in the network with the minimum duration: From Rotterdam to Hamburg and Alexandria on service A, transhipping to service B and then from Alexandria to Shanghai. The duration is approximated as $1 \cdot 4$ days at the ports and $1 + 5 + 34$ days at sea, which is a total of 44 days. In the scope of this thesis, a constant transhipment duration is added to include the duration containers are stored in the container yards during transhipment.

The example shows two important aspects: First, a badly created network can highly decrease the transit time performance. This can be partly compensated for by increasing the speed of the incident services, if possible. Otherwise, the network structure must be changed to respect the transit times. In the example in Figure 2.7, service B could call Rotterdam in its port rotation to reduce the transit time between Rotterdam and Shanghai.

Second, the transit time is only relevant if a network calls ports that are constrained by a maximum transit time. If either one port is not called, no link between the two ports exist or no cargo is transported between two ports, transit times become irrelevant for this connection. However, as soon as both ports are called, a link exists and cargo is transported, transit times must be respected.

2.6. Cooperative Agreements

Operating services on a regular basis leads to relatively inflexible port calls. Seasonal demands, volatile cash flow due to volatile freight rates and large costs associated with new vessels force carriers to search for opportunities to manage these challenges. One way to tackle the difficulties is to cooperate with other carriers and is described in this section.

2.6.1. Liner Conferences, Consortium Agreements and Global Alliances

The liner conference system, developed in the mid-1870s, is the industry's oldest and, in the past, most important kind of cartel-type agreement (see (Stopford, 2009, p. 556), (Basedow et al., 2012, p. 6)). The membership of a liner conference could be either closed or open subject to government agency monitoring (see (Wong and Bamford, 2012, p. 3)). The common objective of the conference members was to "fix and agree on schedules, in order to rationalize the capacity and the frequency of services offered to their customers, as well as the tariffs that are publicly available" (Basedow et al., 2012, p. 6).

Starting from October 2008, all carriers operating in the EU are subject to regulation 4056/86 that removes the block exemption for the liner shipping industry from the 1958's Treaty of Rome on antitrust. In short, carriers are not allowed to operate in price or capacity fixing conferences when offering routes into or out of Europe. Shipping conferences outside of Europe are not affected by the EU regulation but other anti-trust laws, such as the American Ocean Shipping Reform Act that took effect in May, 1999. Today, most of the main trade lanes are subject to anti-trust laws (see for example Shipping Watch (2014)).

As a result of increasing anti-trust popularity in the 1990s , the carriers switched to other forms or cooperation, such as consortia, alliances and mergers.

Consortia, or consortium agreements, offer joint services by two or more carriers on the same shipping route (see EU Article 1 of the Commission Regulation 870/95). On these services, cross-slot charters are used to reserve capacity for each member of the consortium. Subject to the anti-trust law, each member sets pricing and

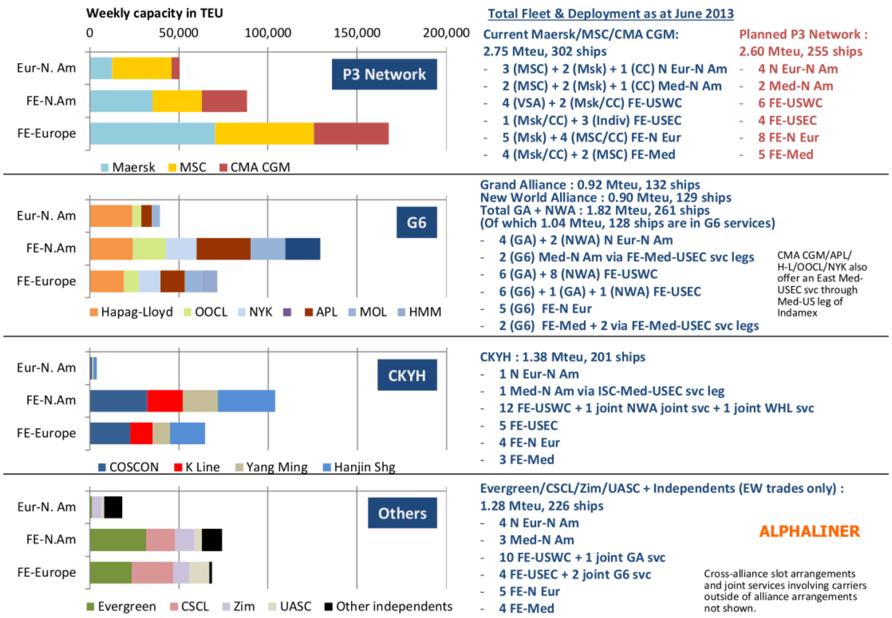


Figure 2.8.: East-West Carrier Alliances 2013, from Alphaliner Newsletter, Volume 26, 2013 (2014). The implementation of the P3 alliance was rejected by Chinese regulators in July, 2014 (see for example List (2014)).

transportation conditions with shippers individually (see (Basedow et al., 2012, p. 7)).

"A global alliance is a sort of consortium whose geographic scope is not a single trade, but is instead worldwide" ((Basedow et al., 2012, p. 7)) or on certain major trades (see (Panayides and Wiedmer, 2011, p. 2)). Global alliances effects on horizontal integration (which refers to vessel employment and utilization) have been stated as a key change in the last 20 year in the liner shipping industry (see (Lun et al., 2010, p. 63)). The main advantages of alliances are the increased geographical scope, possibility to perform vessel planning and coordination on a global scale, risk and investment sharing, economies of scale, entry in new markets, increasing service frequencies and combining purchasing power (see (Midoro and Pitto, 2000, p. 33)). Similar to consortium agreements, global alliances can contain specific collaborative agreements such as vessel sharing or slot sharing agreements (explained in the next section). In vessel sharing agreements partners cooperate by optimizing their container-vessel assignment to improve the utilization on specific legs or synchronizing the vessel departures (see (Panayides and Wiedmer, 2011, p. 2)). In Figure 2.8, global alliances such as the Grand Alliance and CKYH for the East-West trades are shown.

Cooperation plays an important role when designing the networks for each carrier individually. Cooperation is not only relevant for the main carriers but also for the small ones (see (Lu et al., 2010, p. 325)). The following sections provide both the legal and economic overview on how cooperation is performed and how it can be considered in the liner shipping network design.

2.6.2. Slot Sharing Agreements

An important form of cooperation is a slot sharing agreement. Within a consortium or strategic alliance, partners share free capacity to maximize their vessel utilization. Some authors claim that operational partnerships will become even more important (see (Sys, 2009, p. 265)). A carrier can sell slots on an operated service, purchase slots from a partner or do both with a slot swap.

Service	Operator	Deployed Vessel	Allocation @ 14 tons EB/SB	Allocation @ 14 tons WB/NB	Reefer Allocation per leg	Reefer Surcharges	Slot Rate
Service 1	Carrier 1	4000	3000	3500	150	-	-
Service 2	Carrier 2	2000	500	300	100	150	800
Service 3	Carrier 3	2000	400	-	20	150	300

Table 2.3.: Example slot sharing agreement between three carriers.

Table 2.3 shows an example agreement between three carriers within a given trade. Carrier 1 operates a service with a 4000 TEU vessel type. In the East and South bound direction of its service, 1000 slots are available can be sold out. Of these, carrier 2 purchases 500 and carrier 3 400 slots. On the West and North bound direction, carrier 1 has more own cargo to transport and thus decreases its sold slots. Carriers 2 and 3 call ports that are not served by carrier 1 and thus slots are purchased at a specific rate. Carrier 1 has committed by contracts to transport cargo for its partners but benefits from its increased network. If carrier 1 decides to change its network, at least the already called ports have to be called due to the contracts. However, due to the already mentioned anti-trust laws, the carriers are not allowed to exchange their specific cargo flow information. Instead, it is assumed that a carrier that wants to optimize the services (here carrier 1) can only approximate the partners' cargo flows between the previously called ports. When he chooses to optimize the liner services he tries to decrease its slot costs, leading to an imbalance between the slot rates (slot price) and the actually implemented slot costs on his own service.

Slot sharing can be agreed on any subset of the legs within a liner service. In Table 2.3, the East, West, South and North bound directions represent these subsets.

2.7. Liner Service Capacities

In this section the liner service capacities are described in detail. Because vessels (of a specific type) are assigned to the liner services, the capacity is referred to as the liner service capacity. However, the service and vessel type capacities result from specific vessels whose tonnage is subject to several physical constraints that also apply to a vessel type. The assignment of a vessel type and a number of vessels to a service is called a *vessel deployment*.

2.7.1. Container Vessel Capacities

The amount of containers on a vessel is limited. Usually, a vessel's capacity is given by its TEU tonnage (see (Schönknecht, 2009, p. 5)). This capacity is limited by the container size and type and assumes a net weight of 14 tons per container. In practice, a TEU container weighs between 2250 kg (empty) and 24000 kg (fully loaded). Similar ranges hold for forty foot and high cube containers. Stowing a container vessel with forty foot containers reduces the transportable containers by approximately 50%, because their length is double compared to a TEU. A vessel might not only transport dry containers but also reefer containers. These containers need electricity and thus special container plugs to cool or freeze cargo within the container.

For example, the container vessel "CHARLOTTE MAERSK" is currently operated on Asia – Europe trades and has a total nominal container capacity of 10,457 TEU and 700 reefer plug slots. Furthermore it has a maximum deadweight of 105,000 tons. It can transport 6,600 containers that weighs 14 tons each. The large amount of reefer container plugs can also be used to stow dry containers.

Summarizing, a container vessel has the three capacity types: Dry slots, reefer container plugs and maximum deadweight. These capacities must already be considered in strategic and tactical planning methods because every type can limit the amount of containers on a vessel.

From an operational point of view, the capacities must be considered in much more detail regarding the container content, minimum stowage distances between containers and the vessel stability. The reader is referred to the work of Avriel et al. (1998) and Imai et al. (2006) for details on container vessel stowage planning.

2.7.2. Port Depth and Vessels' Deadweight Scale

The ports called within the services often have a maximum depth allowing vessels to enter the ports. The ports' depths depend on the tide levels, the geographical position (salt water percentage) and the infrastructure offered by the port at the berths. The depths can vary between 8 meters in small feeder ports up to 18 meters in deep sea ports such as the Jade Weser port in Wilhelmshaven or 17.5 meter in Shanghai (see World Port Source (2014)). These allow large vessels to enter the ports independent of the tide levels that can influence the ports' depth by several meters. Tide levels become important when scheduling the liner service and reserving berth windows at the container terminals. In the scope of the network design, tide levels

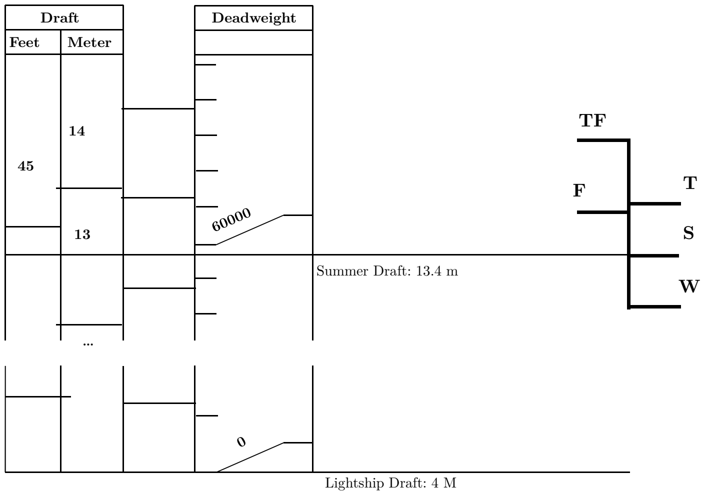


Figure 2.9.: Example deadweight scale for a Panamax vessel.

are assumed to be constant and a maximum depth is used for each port.

A vessel has a lightweight displacement (based on the weight completely unloaded, considering only the hull, machinery and installed equipment). This displacement results in the vessel's *lightship draft*. In practice, the draft of a vessel is subject to the deadweight of the vessel. Depending on the loaded cargo, the vessel's draft increases according to the vessel's *deadweight scale*, shown in Figure 2.9.

According to the deadweight scale, the example vessel has a draft of 4 meters. When the vessel deadweight is nearly 60000 tons deadweight (tdw), the vessel draft increases to 13.4 meters. For a specific vessel operating on a liner service, the maximum deadweight and draft are subject to the geographical region and the time of year according to the plimsoll mark. In the summer region (S) the maximum draft is 13.4 meter, leading to a maximum deadweight of approximately 60000 tdw. The zones are defined according to waves and winds: In the summer region (S), not more than 10% and 8 Beaufort (34 kn) wind, in the tropical region (T) not more than 1% or 8 Beaufort winds are allowed and further constraints regarding storms exist.

For the maximum vessel types' draft in the scope of this thesis, the summer load line is used as an upper bound because it is the most restrictive. Example deadweight scales from an ocean carrier's vessel types indicate that the scales can be linearized with relatively small errors. To calculate the draft it is assumed that the draft only depends on the loaded cargo and the whole deadweight can be used to transport cargo. Since the specific ports for bunkering are unknown in the planning horizon, no constraints on the bunker amount are imposed. For details on how the displacement and the draft of a vessel is calculated, see (Barrass, 2006, p. 383) and

Ayre (1885).

Depending on the draft, a vessel cannot call all ports fully loaded. This is a key problem for the increasingly larger vessels with larger draft. Ports must be called in such a way that the vessels' draft fits the port's depth (see (Baird, 1996, p. 148) and (Midoro et al., 2005, p. 98)). This requires the ocean carriers to carefully design the liner services in such a way that draft constrained ports within a service are called after ports with a larger import ratio to unload enough cargo from the vessels to pass the draft - depth constraint (see (Stopford, 2009, p. 354)).

2.8. Empty Container Repositioning

In Section 2.4.1, the trade between China and Germany is described. The example indicates that trade volumes between countries often differ from each other, leading to *trade imbalances*. This has been a major problem since the beginning of container shipping (see Levinson (2006), Shintani et al. (2007)). Figure 2.10 shows the calculation of trade imbalances on the main shipping trades according to Fan et al. (2012). The figure shows a trade imbalance of about 40% between Asia and the US and South America, meaning the cargo volume is 40% higher from Asia to America compared to the trade from America to Asia. Similar, the trade from Asia to Europe is about 60% higher than the eastbound trade in 2009. The exact calculation of the trade imbalance between two countries or regions differs in the scientific community (see Xu et al. (2010), Fan et al. (2012)).

Ocean carriers often own or lease a stock of containers to brand containers for marketing purposes (see (Notteboom and Rodrigue, 2008, p. 168)) and container pools or the gray box concept are still not accepted in practice (see (Notteboom and Rodrigue, 2008, p. 168) and Notteboom and Rodrigue (2007)). From the liner shipping point of view, the effect of trade imbalances are stocks of empty containers at different ports. Figure 2.11 shows the overall empty container repositioning flow. In the example, customer A wants to transport one container to customer B. He rents a container from the carrier that transports the empty container from either

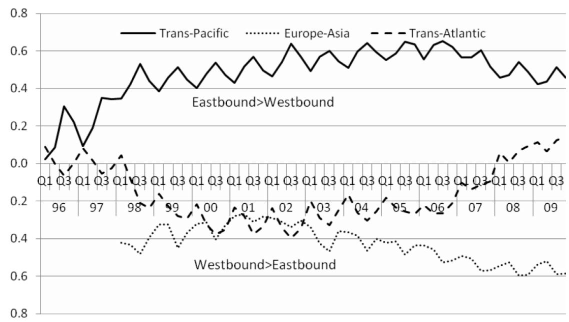


Figure 2.10.: Trade imbalances on the main trades, (Fan et al., 2012, p. 249).

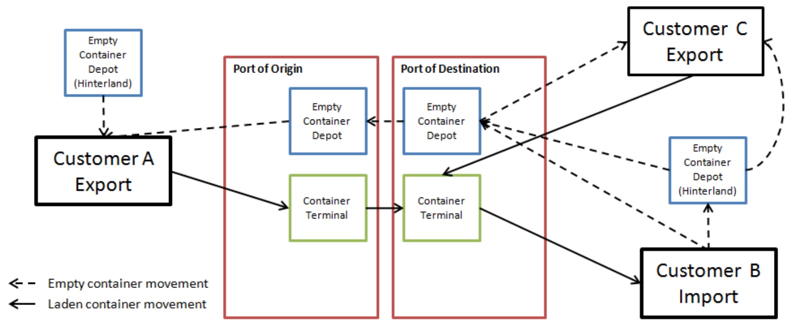


Figure 2.11.: Empty container repositioning using depots (on the basis of (Lai et al., 1995, p. 689)).

a hinterland or port depot to the customer. The customer stows the container with goods and moves the container to the port of origin's container terminal. There, the container is transported by sea to the port of destination (transhipment ports can be used in between as well). Trucks, trains or both transport the container to the final customer where the cargo is unloaded. Afterwards, the empty container at customer B must be transported to an empty container depot in a port or hinterland. From a more operational point of view, the container can be transported directly to customer C for a new export.

This example shows that a lot of aspects are involved in the empty container repositioning. Additional aspects such as depot optimization, container leasing or container safety stocks have to be considered in the operational planning process as well. For planning models focusing on the empty container depot optimization, see (Flapper et al., 2005, p. 65 ff), Boile et al. (2008) and Mittal (2008), for empty container safety stock and flow optimization see Feng and Chang (2008) and Erera et al. (2009).

The overall empty container repositioning costs are estimated at 27% of the total world fleet running cost (Song et al., 2005, p. 15). Thus, it is important to include the empty container repositioning in the network design. From a strategic point of view, specific customers and hinterland depots are often unknown because they are determined by the network design. The hinterland repositioning costs (such as intermodal transportation cost and safety stocks) are considered by a constant cost factor per transported cargo flow unit. The main challenges arising with empty containers are the increased vessel utilization and the opportunity cost by unavailable slots to transport cargo flows. In the scope of this thesis, each transported cargo flow requires an empty container of the cargo flow's container type and size at its port of origin and provides an empty container at its port of destination. Thus, a repositioning (or balancing) of the empty containers must be performed. Note that empty containers mainly use the capacity type of dry slots, whereas the deadweight utilization is relatively low.

2.9. Costs and Revenues

Operating a liner network results in different costs and revenues gained from the cargo's shipper. Stopford defines building-blocks of liner shipping costs (Stopford, 2009, p. 539 ff):

1. Ship costs per day
2. Port charges (excluding cargo handling)
3. Container and container handling costs
4. Administration costs

According to (Stopford, 2009, p. 539 ff), ship costs per day include the operating costs (OPEX), capital costs for the time charter (or vessel depreciation) and bunker cost. Port charges are both, per TEU and per call. Container costs and container handling costs include leasing, maintenance, repair, terminal, refrigeration, transhipment, hinterland transportation and cargo claim cost. Of these costs, transhipment and hinterland transportation are the largest cost factors. Administration costs are calculated as cost per employee involved in running the enterprise.

In Figure 2.12, the costs per service voyage for different vessel types are given. The vessel cost ratio decreases with the size of the vessel, whereas the container handling and onward hinterland transportation costs increase with the vessel size. This can be explained by increased port handling and transhipment costs because larger vessels generally lead to more transhipment operations due to reduced port calls. The economy of scale through large vessels decreases the total voyage related costs per TEU from 1,299 US$ to 1,023 US$ for a 4300 TEU ship down to 902 US$ per ship, which is a total decrease of 69.44%.

In the scope of the network design, the following costs were identified with network planners of a global liner carrier:

1. Bunker cost per day at sea in the planning horizon
2. Vessel capital cost in the planning period
3. Port cost per call per vessel type
4. Container handling costs (transhipment costs, loading and unloading cost at the port of origin and destination and equipment cost per container type including hinterland repositioning, depreciation and leasing cost)
5. Slot purchasing cost at partner services

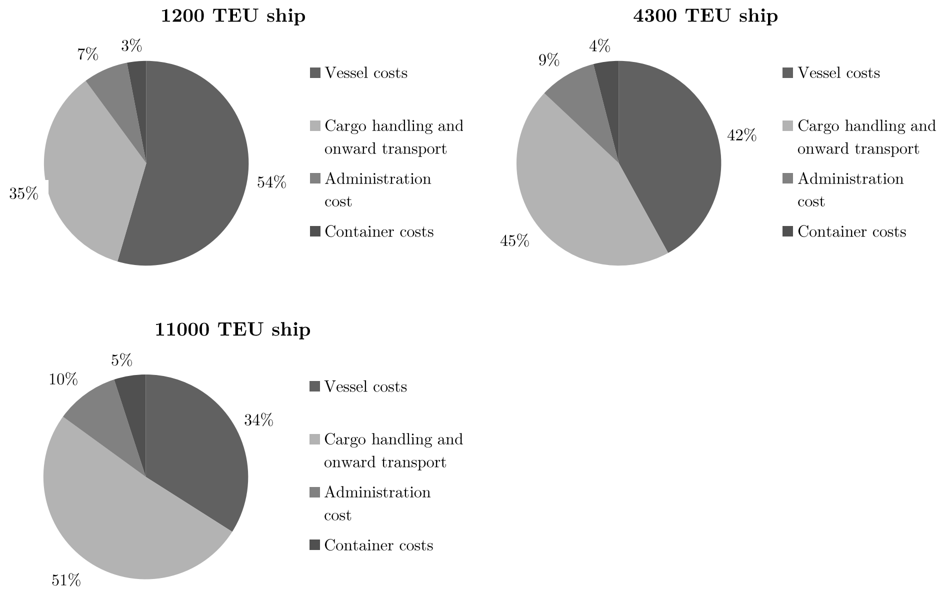


Figure 2.12.: Voyage costs per vessel type (see (Stopford, 2009, p. 542)).

In Section 2.8, empty container repositioning costs were described in detail. In the strategic planning, hinterland and safety stock costs are not included. Rather, for each served laden container a fixed equipment depreciation must be paid. Furthermore, for each loading and unloading operation of an empty container at a port, fixed cost must be paid.

In general, the largest cost components are, depending on the liner service, the capital, bunker, transhipment and slot purchasing costs.

The capital cost can be either time chartering costs or costs for running own vessels (beside others, this includes depreciation, insurance, maintenance and manning costs), depending on whether the carrier has bought the vessel or charters the vessel from the charter market. Note that both costs are unified for the remainder of this thesis as charter cost. For example, the capital costs for a 11,000 TEU vessel is about 46,301 US$ per day. This is calculated with a depreciation period of 20 years with an interest rate of 8% p.a. (see (Stopford, 2009, p. 540)).

In the literature (see Álvarez (2009), Brouer et al. (2013)), the vessels' bunker cost are often calculated approximately based on the cubic bunker consumption function proposed by (Stopford, 2009, p. 234): $F(S) = F^* \left(\frac{S}{S^*}\right)^a$, where a depends on the vessel's engine and is about 2 for steam engines and 3 for diesel engines.

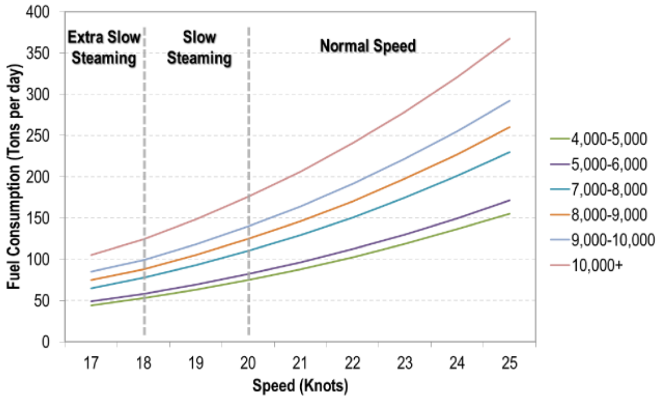


Figure 2.13.: Bunker consumption in tons per day for different vessel types, (Notteboom and Cariou (2009)).

Notteboom and Cariou (2009) also consider the age and the power of the vessel and find the bunker consumption subject to the speed in knots presented in Figure 2.13.

Figure 2.13 shows that the consumption highly varies with the speed and differs from the formula in Stopford (2009): a 5000-6000 TEU vessel's bunker cost per day, assuming bunker cost per ton of 600 US$, are 88,768 US$ at 21 kn using the data given in Notteboom and Cariou (2009) and 79,048 US$ using Stopford's formula. Wang and Meng (2012c) analyze the cubic approximation in more detail and conclude that the cubic approximation should only be used when no historical data is available. In Figure 2.14 both, the approximation from Stopford (2009) and a polynomial of order three is fitted to the existing historical data for different speeds (also known as bunker profiles). Below 17 knots, the cubic approximation becomes clearly worse than the polynomial approximation using a linear least squares fitting (see for example Bevington and Robinson (1969)).

Notteboom and Vernimmen (2009) mention the large bunker cost impact on the liner services and thus the liner network design problem. The (super) slow steaming strategy became more relevant in the last years. This thesis uses polynomial bunker consumption in the real-world test instances and cubic bunker consumption curves in the artificial instances. Thus, the approximation errors in the real-world instances can be reduced. Based on the bunker consumption, the bunker cost are calculated by multiplying the consumption per day at sea in the planning horizon for each vessel and the bunker costs per ton, currently about 658 US$ per metric ton in Busan and 585 US$ in Rotterdam (see Bunker Index (2013)).

The costs introduced above decrease the profit of a carrier. Within the network design problem in a strategic or tactical planning horizon, an ocean carrier seeks to maximize its profit (see Álvarez (2009), Agarwal and Ergun (2008) and Brouer et al. (2013)). A carrier gets revenue from the shipper for transporting cargo of a specific type from its origin to its destination port using one or more liner services. The

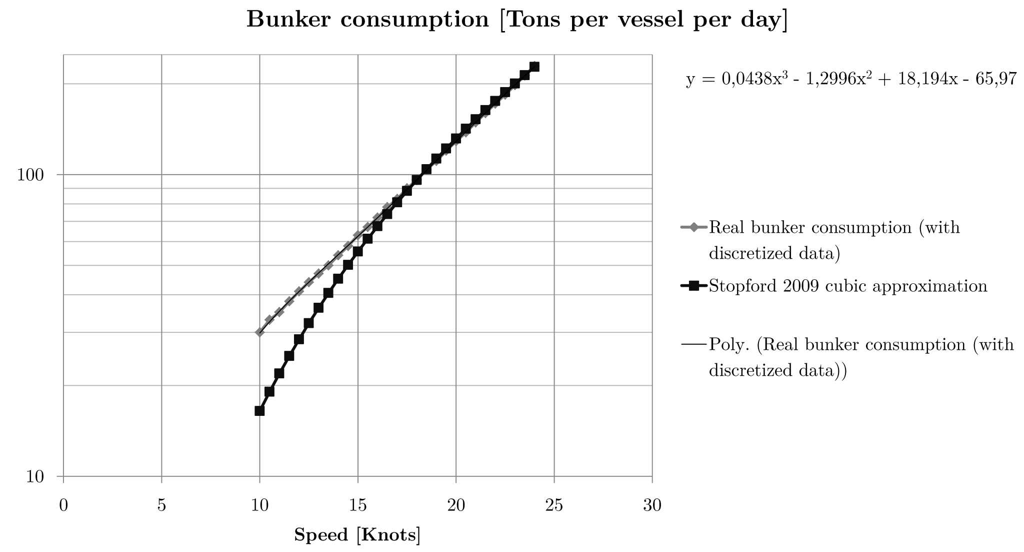


Figure 2.14.: Example bunker profile for a 4300 TEU vessel. The y-axis is scaled logarithmically to the basis of 10.

carrier specific revenue per cargo flow depends on the trade region, the deployed capacity (supply), the container type and other additional offered services, such as intermodal transportation. The cargo can be either contracted with shippers in a medium or long term or obtained in a short term through the spot market. Ocean carriers tend to prefer having contracts with a negotiated freight rate to have a higher certainty in their planning processes. The revenue often include surcharges such as the bunker adjustment factor (BAF), the peak-season surcharge or the currency adjustment factor (CAF), allowing the carrier to compensate for some of the uncertainties.

Using the revenue and costs, the earnings before interest and taxes (EBIT) of a network can be calculated. In the scope of this thesis, EBIT is referred to as profit.

2.10. Bunker Cost Uncertainty in the Tactical Planning Horizon

In the strategic and tactical planning horizon, several uncertainties exist that can influence the network structure, whereas cargo volumes and freight rates as the largest uncertainties, see (Buxton and Stephenson, 2001, p. 298). Stopford (2009)

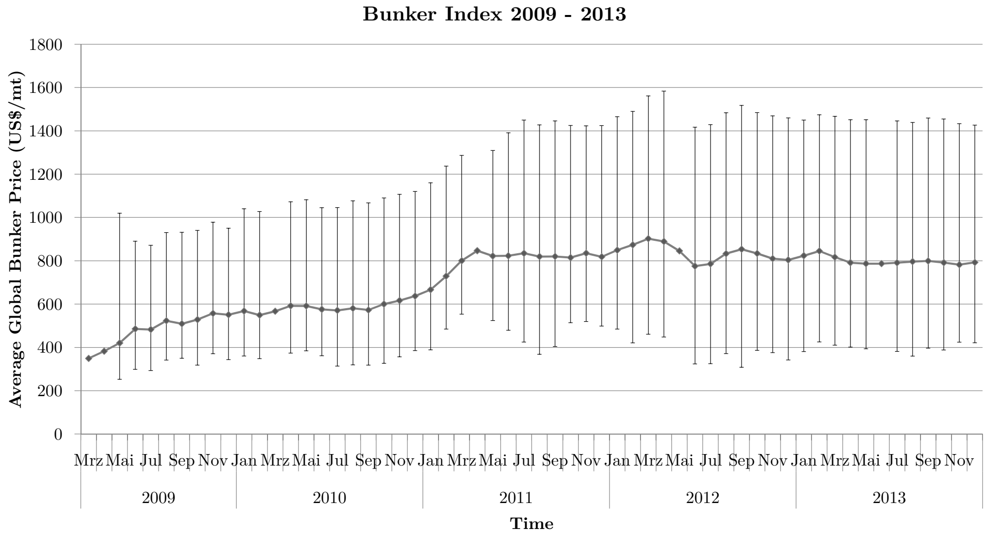


Figure 2.15.: Bunker Index, aggregated data from Bunker Index (2014).

state the dominant bunker costs for liner shipping networks.

As indicated in the previous section, bunker costs are a large factor of the overall costs and play an important role when designing networks. Figure 2.15 shows the average prices over time and the deviation from the average in different ports. The data is given by the Bunker Index (BIX) that collects and aggregates bunker prices. Currently, the data set contains 157 port prices. The average global bunker price per ton increased from 350 US$ per ton to 800 US$ per ton in the last four years, giving it an increasingly important role when designing networks.

As a result, carriers implement slow or even super slow steaming strategies to decrease bunker consumption. When liner services are implemented, a planning problem is to decide where and how much to bunker on each port within a round trip.

Figure 2.16 shows the average price changes on a daily basis. The price is highly volatile and increased by more than 30% between 2010 and 2011. However, due to computational challenges involved with the liner network design problem, this thesis assumes a constant bunker price. To deal with the volatile bunker cost in the industry, sensitivity analysis should be performed to identify the bunker price impact on the network structure.

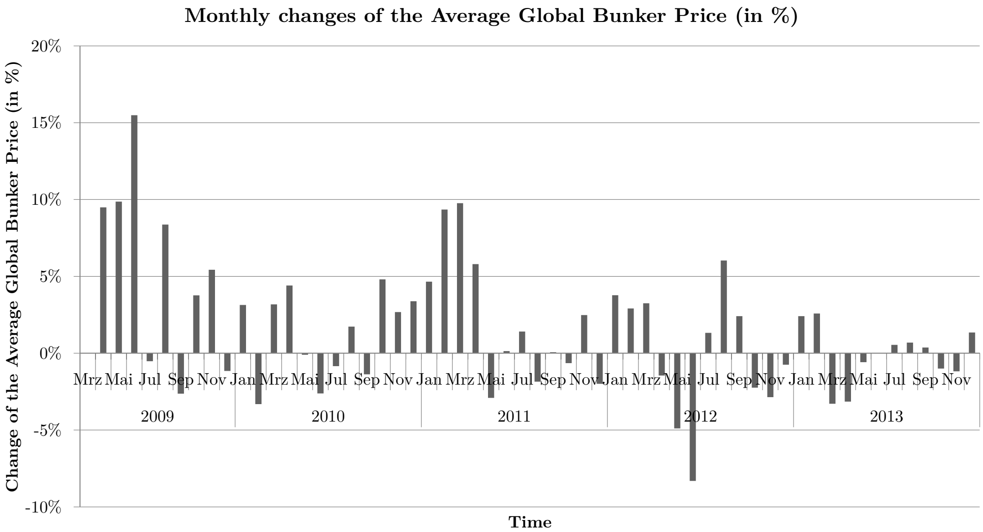


Figure 2.16.: Monthly BIX changes, data from Bunker Index (2014).

2.11. Summary

In this chapter, the liner shipping network design problem, its objective and its practical constraints are introduced. The problem's objective in the scope of this thesis is to maximize the profit in the planning horizon. The revenue for transporting containers should be maximized under consideration of bunker, handling, container, port and charter cost. Furthermore, cost for using slots on partner services are imposed. The objective is subject to the following requirements:

1. Ensure a valid container flow in the network
2. Perform a balancing of empty containers
3. Consider container vessel capacities
4. Respect the port depth with load dependent vessel drafts
5. Ensure transit times between ports

It has been shown that the network design in liner shipping is a complex planning problem with many practical requirements. In the next chapter, the state-of-the-art regarding Operations Research methods in liner shipping is analyzed and the research goals are derived from the research gap.

3. State-of-the-Art and Research Opportunities

This chapter introduces selected optimization techniques for solving combinatorial optimization problems. Related planning problems, such as the vehicle routing problem, the pickup and delivery problem and the min-cost flow problem, are presented. Based on these fundamental planning problems, the state-of-the-art in strategic maritime liner research is given. In particular, publications on the liner shipping network design, the cargo allocation, empty container repositioning and speed optimization are presented.

3.1. Selected Optimization Techniques

In this section, a selected overview of the state-of-the-art methods to solve combinatorial optimization problems is given. The introduced methods are selected to cover the solution approaches developed in this thesis. First, techniques to solve optimization problems to optimality and afterwards heuristic approaches are presented.

3.1.1. Linear and Mixed Integer Programming

Linear Programming has been used since before World War II to solve linear optimization problems (see (Suhl and Mellouli, 2013, p. 9)). A (compact) linear optimization problem is defined as

$$z_{LP} = max\{cx : Ax \leq b, x \in \mathbb{R}^n_+\} \tag{3.1}$$

with an $m \times n$ matrix A, $1 \times n$ vector c and $m \times 1$ vector b (see (Nemhauser and Wolsey, 1999, p. 27)). x is a vector of dimension n and also called a vector of decision variables. The function cx that is either minimized or maximized is called the objective function. The inequalities defined as $Ax \leq b$ are called constraints. For a detailed introduction on linear programming, see (Suhl and Mellouli, 2013, p. 33ff).

Problem (3.1) can be rewritten in another notation that is more common (see Problem (3.2)). The vector of objective coefficients c is represented by a parameter c_j for each variable x_j of the optimization problem. The right hand side b is defined as the parameter b_i for each constraint. The optimization uses a parameter a_{ij} for each row i and column j instead of the matrix a.

$$
\begin{aligned}
(P)\quad z_{LP} = max \sum_{j=1}^{n} c_j x_j & & (3.2)\\
s.t & & \\
\sum_{j=1}^{n} a_{ij} x_j \leq b_i & \qquad \forall i = 1, 2, \cdots, m & (3.3)\\
x_j \geq 0 & \qquad \forall j = 1, 2, \cdots, n & (3.4)
\end{aligned}
$$

Efficient solution approaches to solve large scale liner programs exist, such as the simplex method or the interior point method (see (Suhl and Mellouli, 2013, p. 46), (Chvátal, 1983, p. 13) for the simplex method, (Chvátal, 1983, p. 97) for the revised simplex method, (Nemhauser and Wolsey, 1999, p. 37) for the dual simplex method and (Chvátal, 1983, p. 443) for the interior point method).

Integer and mixed integer programming extend linear programming (3.1) with integer variables. This introduces computational complexity for solving the problem. The program (3.5) is called an integer program (IP) because it only consists of integer variables.

$$(IP) \qquad z_{IP} = max\{cy : Ay \leq b, y \in \mathbb{Z}^n_+\} \tag{3.5}$$

In comparison, (3.6) defines a mixed integer program where y is a n-dimensional vector of integer and x a p-dimensional vector of continuous variables.

$$(MIP) \qquad z_{MIP} = max\{cx + hy : Ax + Gy \leq b, x \in \mathbb{R}^p_+, y \in \mathbb{Z}^n_+\} \tag{3.6}$$

Mixed integer programs (MIP) allow much broader model formulations because logical relationships, fixed cost etc. can be expressed using the integer (or binary) y variables. Note that a MIP also enforces a linear combination of variables in the objective function as well as the constraints. When relaxing the integer requirement of a MIP, the resulting model is called the LP relaxation.

A drawback of MIPs is that the efficient solution methods to solve LPs are not applicable to solve MIPs. The activity of variables in the optimal solution(s) of the simplex method can be fractional whereas they are forced to be integer in order to be feasible for the MIP (see (Suhl and Mellouli, 2013, p. 10)). Therefore, different general algorithms such as Branch & Bound (B&B) and Branch & Cut (B&C) were developed. B&B basically decomposes the problem into smaller problems. Starting with the root node, B&B *branches* (or divides) the original problem into smaller problems by choosing appropriate values for currently fractional integer variables.

Each resulting problem is solved using the LP relaxation and again separated into smaller problems and so on. This approach results in a complete enumeration of the search space and is impractical for large scale problems. *Bounding* is used to prune nodes in the search tree that cannot contain better solutions as the currently best found solution (incumbent). For details on the bounding step of B&B see (Wolsey, 2000, p. 94). For further information on B&B, see (Nemhauser and Wolsey, 1999, p. 355).

In most of the state-of-the-art MIP solvers such as Cplex (see IBM CPLEX (2014)), XPress (see Frontline Solvers (2014)) or Gurobi (see Gurobi (2014)), a B&C is implemented that additionally uses preprocessing methods and heuristics to improve the runtime. B&B will be used as an acronym for using a MIP solver as solution method throughout this chapter.

3.1.2. Delayed Column Generation

"Column generation (CG) is nowadays a prominent method to cope with a huge number of variables" (Lübbecke and Desrosiers, 2005, p. 1007). Among the first suggesting *delayed column generation* to solve large linear programs were Gilmore and Gomory (1961).

In some problems, the number of possible columns (variables) in problem (3.1) is too large to be completely enumerated. The problem containing all columns is also called the *master problem*. In some cases, it can be reasonable to work on a subset of columns, instead of the problem containing all columns. The master problem becomes a *restricted master problem* (*RMP*) when only a subset of the decision variables are used:

$$(RMP) \qquad z_{RMP} = max\{\tilde{c}\tilde{x} : \tilde{A}\tilde{x} \leq \tilde{b}, \tilde{x} \in X\} \tag{3.7}$$

The main idea is to successively solve the *RMP*, retrieve information from the optimal solution and use it to extend the *RMP* by columns that potentially improve the objective value, through a subproblem called the *pricing problem* (subproblem). The information relevant to the pricing problem are the dual values of the *RMP* constraints. These represent the impact of changes of the constraints' right hand side on the objective value. The overall column generation solution process is outlined in Algorithm 1.

The *RMP* is initialized with a set of (artificial) columns X^0 that is successively extended. After the *RMP* is solved, the (relevant) duals are stored for the pricing problem (subproblem) $PRICING(\pi_i)$ to generate new columns that can improve the objective value. Each new column is added to the *RMP* if it has positive reduced cost (for a maximization problems). According to the duality theory, this can but need not improve the objective value of the *RMP*. Typically, the pricing problem is an optimization problem itself, such as a shortest path problem.

Algorithm 1: Delayed Column Generation Procedure

1 Create initial set of columns X^0
2 Solve the RMP and store duals π_i incident to row i
3 Solve the pricing problem $PRICING(\pi_i)$. Terminate if no columns with positive reduced costs found
4 Add columns to RMP and goto line 2

The efficiency of the column generation method relies on efficient solution methods for the pricing problem and (relatively) small restricted master problems. If the master problem is a linear program without integer variables, column generation converges to the global optimality of the master problem. For a detailed introduction, survey papers and implementation details on column generation see Chvátal (1983), Lübbecke and Desrosiers (2005) and Desaulniers et al. (2005).

3.1.3. Metaheuristics

Metaheuristics aim to find approximate solutions in a reasonable time. They do not guarantee optimality as in the previously described methods. “A metaheuristic is formally defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, learning strategies are used to structure information in order to find efficiently near-optimal solution” (Osman and Laporte, 1996, p. 513 and 514). Well known metaheuristics include simulated annealing (SA), genetic algorithms (GA), evolutionary algorithms (EA) and, increasingly, variable neighborhood search strategies (see Mladenović and Hansen (1997)). Genetic algorithms and variable neighborhood search methods are presented in the following paragraphs.

Genetic Algorithm A genetic algorithm (GA) was first used by Holland (1975) to solve combinatorial optimization problems. Genetic algorithms are based on a set (population) of individuals that is successively improved. Darwins survival of the fitness principle is used to optimize the whole population according to a fitness function. Glover and Kochenberger (2003) formulate a general version of a genetic algorithm as shown in Algorithm 2. The very basic version of a genetic algorithm works as follows: Each individual is associated with chromosomes that represent their properties (used to represent a solution for the problem at hand). At the beginning of the algorithm, an initial population is created. Afterwards, the algorithm is executed until the termination criteria, such as a maximum runtime or iterations without a population improvement, is reached. In each iteration, a new individual (child, offspring) is created from existing individuals (parents) using their chromosomes.

Algorithm 2: Genetic Algorithm (see (Gendreau and Potvin, 2010, p. 113))

```
Choose an initial population of chromosomes while termination condition not
satisfied do
    repeat
        if crossover condition satisfied then
            select parent chromosomes
            choose crossover parameters
            perform crossover
        end
        if mutation condition satisfied then
            choose mutation points
            perform mutation
        end
        evaluate fitness of offspring
    until sufficient offspring created;
    select new population
end
```

As in the evolution process described by Darwin, mutation can randomly change chromosomes of an individual. This is repeated until a predetermined number of offspring are created. Finally, individuals from the previous population and offspring are selected for the next generation based on their fitness (objective function).

Several extensions to this classical approach exist, such as different mutation or parent selection strategies. One particular extension is the use of local search procedures on the individuals. In this case, the genetic algorithm is also called a hybrid genetic algorithm (see for example Gonçalves et al. (2005)) or memetic algorithm (see for example Moscato et al. (1989)).

For the theoretical background and extensions of the GA see Michalewicz and Fogel (2004), Glover and Kochenberger (2003) and Holland (1975). In the remainder of this thesis, we will speak of evolutionary algorithms (EA) instead of genetic algorithms due to the missing chromosome structure (see Kruse et al. (2013)).

Variable Neighborhood Search Variable neighborhood search is a metaheuristics proposed by Mladenović and Hansen (1997). “The basic idea is a systematic change of neighborhood within a local search” ((Glover and Kochenberger, 2003, p. 145)). Mladenović and Hansen (1997) present different approaches for the variable neighborhood concept: Variable neighborhood descent (VND) and basic variable neighborhood search (VNS).

The VND process is shown in Algorithm 3. To summarize, the algorithm searches all neighborhoods of the initial solution x and selects the best one. It is a de-

Algorithm 3: Variable neighborhood descent (see (Gendreau and Potvin, 2010, p. 64))

Select set of neighborhood structures $N_k^{'}$ for $k = 1, \cdots, k_{max}^{'}$
Find initial solution x
repeat
    $k = 1$
    repeat
        Find best neighbor $x' \in N_k'(x)$ of x
        if $f(x') > f(x)$ **then**
            $x = x'$
        else
            $k = k + 1$
        end
    until $k = k_{max}^{'}$;
until *no objective improvement*;

terministic local search that should find the local optima for solution x with respect to all neighbors. Whereas most local search heuristics use one neighborhood structure, VND and VNS uses k'_{max} different neighborhood structures, denoted $N'_k, k = 1, \cdots, k'_{max}$. Algorithm 3 starts with the first structure and selects the best neighbor in this structure as long as a better solution x' is still found. Otherwise, the next neighborhood structure is chosen. Typically the changes applied to the solution i increases with increasing k using a nested structure.

Algorithm 4 outlines the basic variable neighborhood search. A random solution x' is created from the kth neighborhood structure of the current best solution x. Afterwards, a local search is performed on this random solution. The result x'' should be a local optimum of x' in the current neighborhood structure. If the local optimum x'' is better than the current best solution x, the solution is applied and the first neighborhood structure is used again. Otherwise, the next higher neighborhood structure is explored.

Similar to other metaheuristics, many extensions and problem specific implementations exist. Mladenović and Hansen (1997) also propose reduced VNS and variable neighborhood decomposition search. Furthermore (Glover and Kochenberger, 2003, p. 150) introduce several hybrid approaches and present applications on the traveling salesman problem, the vehicle routing problem and others (see (Glover and Kochenberger, 2003, p. 152ff), Hemmelmayr et al. (2009a) and Parragh et al. (2010)).

For further metaheuristics the reader is referred to Reeves (1993), Michalewicz and Fogel (2004) and Gendreau and Potvin (2010).

Algorithm 4: Basic variable neighborhood search (see (Gendreau and Potvin, 2010, p. 64 and 67))

```
Select set of neighborhood structures N'_k for k = 1, ··· , k'_max
Find initial solution x
repeat
    k = 1 repeat
        x' = Shake(N_k(x))
        x'' = LocalSearch(x')
        if f(x'') > f(x) then
            x = x''
            k = 1
        else
            k = k + 1
        end
    until k = k'_max;
until stopping condition met;
```

3.1.4. Fitness Approximation in Metaheuristics

Throughout this thesis it will be shown that the main challenge of solving the liner shipping network design problem is to solve the time consuming cargo allocation problem as part of the metaheuristics. To tackle this, an approximation method for the real fitness is developed. A function that determines the fitness heuristically is also known as a *fitness approximation* or a *surrogate* (see Jin et al. (2003), Goh and Tenne (2010) and Jin (2011)). In general, surrogates are intensively used in problems that use a time consuming fitness function. In particular, mechanical optimization problems where a simulation is used to evaluate a solution are often solved with surrogates (see (Davarynejad et al., 2012, p. 245)).

Jin (2005) provides a comprehensive survey of surrogates in evolutionary algorithms. Fitness approximations have been mainly applied to:

- Structural optimization problems, such as aerodynamics or fluid dynamics
- Solution approaches for optimization problems that require the user to evaluate solutions
- Problems in noisy environments
- Fitness landscape approximation problems

Booker et al. (1999) present a general framework for the use of surrogates in optimization approaches and evaluate it in a helicopter rotor blade design problem.

Sano and Kita (2000) use a history of previous fitness evaluations and samples for the evaluation of a new individual. They use the maximum likelihood method to evaluate a new individual approximately within a genetic algorithm that is applied to noisy non-linear functions. Davarynejad et al. (2012) develop a method to adaptively change the fitness granularity. They keep a pool of individuals with exactly computed fitness functions. If a new individual is similar to an individual of that pool according to some features, the pool's exact fitness is used instead of the crude estimate. Otherwise, the individual is added to the pool. Beside others, they apply a genetic algorithm to an airplane wing design and a piezoelectric actuator design problem[1].

Takagi and Iba (2005) provide an overview of interactive evolutionary computation that integrates the user into the optimization process. They provide example applications such as industrial design, musical melody production, hearing aid and others. Another application of interactive optimization is the work of Chou et al. (2012). They develop districting plans for the city of Philadelphia with a special focus of the fairness of the solutions.

Fitness approximations are also used in noisy environments. Papapanagiotou et al. (2013) solve the orienteering problem with stochastic travel and service times. Based on a set of scored vertices, the orienteering problem's goal is to determine a length limited path (not necessarily consisting of all vertices) that maximizes a score value (see Vansteenwegen et al. (2011)). Papapanagiotou et al. (2013) present a monte carlo simulation based objective function approximation to handle the uncertainties in the problem. Papapanagiotou et al. (2014) extend this method by a combination of a monte carlo simulation and an analytical approximation. Weyland et al. (2013a) develop a metaheuristic framework for stochastic combinatorial optimization problems based on graphics processing units (GPU). The class of stochastic optimization problems have in common that they can be approximately evaluated using monte carlo sampling. In the framework of Weyland et al. (2013a), this is performed in parallel on the GPUs. They apply their framework on the probabilistic traveling salesman problem with deadlines and show significant improvements by efficiently utilizing the GPUs. Weyland et al. (2013b) extend the work of Weyland et al. (2013a) by improving their GPU algorithms.

Ratle (1998) present a method to speed up evolutionary optimization methods by approximating the fitness landscape to quickly evaluate some of the individuals. They apply their methods to test cases introduced by Keane (1994).

Beside the four areas that Jin (2005) mention, fitness approximation methods are also used in graph theory. For example, Kim and Moon (2014) solve the linear

[1] A piezoelectric actuator is a type of motor that transforms electric signals to mechanical output (such as motion, pressure etc.) and uses piezoelectric materials, which induce voltage under elastic deformation.

ordering problem using a genetic algorithm. The exact fitness function is given by the sum of the weights of the used edges. Kim and Moon (2014) propose an approximation of the fitness by using only a subset of all edges' weights. They gradually increase the number of edges during the evolutionary process.

Several problems occur when approximating the real (optimal, exact) fitness, whereas the largest is misguiding the metaheuristics to a *false optimum* according to the surrogate (see Jin et al. (2000)). For a survey on the application of surrogates in evolutionary algorithms (EA), not only restricted to fitness approximations but also EA operators, see Jin (2011).

Although mainly used in EAs, fitness approximations are also applied to other metaheuristics. Singh et al. (2010) use surrogates in simulated annealing to solve multi-objective test problem. Audet et al. (2008) extend the variable neighborhood search metaheuristic by surrogates and apply the method to an engineering problem in the chemical industry.

For a comprehensive survey on fitness approximations see Jin (2005). For further approaches to approximate the fitness function, such as artificial neuronal networks or fitness estimations using clustering algorithms, see (Jourdan et al., 2006, p. 59ff).

Summing up, the surrogate applications focus mainly on complex engineering problems. Only a few attempts have been made to apply the concept of fitness approximations to discrete optimization problems. To the best of the author's knowledge, surrogates have not been used for solving vehicle routing or maritime problems yet.

3.2. Related Combinatorial Optimization Problems

In this section, related combinatorial optimization problems to the liner shipping network design problem are presented. They provide a basic understanding of the models developed in this thesis.

3.2.1. Vehicle Routing Problems

Vehicle routing and pickup and delivery problems arose from the classical traveling salesman problem (TSP) and its multiple vehicle extensions (see (Bektas, 2006, S. 212 - 213)). The planning task is to create a distance minimal round trip for a salesman who wants to visit n cities exactly once. Miller et al. (1960) formulated the problem as a mixed integer linear optimization problem. For further formulations and solution approaches for the TSP see Bellmore and Nemhauser (1968), Lawler et al. (1985), Dorigo and Gambardella (1997) or Basu (2012).

The vehicle routing problem (VRP) and extends the TSP by vehicle capacities and dates back to the publication of Dantzig and Ramser (1959). The following vehicle flow formulation (3.8) - (3.14) is based on (Toth and Vigo, 2002, p. 12).

$$
\begin{aligned}
&VRP \\
&min \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij} && (3.8) \\
&s.t. \\
&\quad \forall j \in V \backslash \{0\} : \sum_{i \in V} x_{ij} = 1 && (3.9) \\
&\quad \forall i \in V \backslash \{0\} : \sum_{j \in V} x_{ij} = 1 && (3.10) \\
&\quad \sum_{i \in V} x_{i0} = K && (3.11) \\
&\quad \sum_{j \in V} x_{0j} = K && (3.12) \\
&\quad \forall S \subseteq V \backslash \{0\}, S \neq \emptyset : \sum_{i \notin S} \sum_{j \in S} x_{ij} \geq r(S) && (3.13) \\
&\quad \forall i, j \in V : x_{ij} \in \{0, 1\} && (3.14)
\end{aligned}
$$

V is set of customers, K the number of vehicles. The formulation minimizes the total distance traveled by the vehicles. Furthermore, the vehicles' round trips start and end in a depot (node 0). Constraints (3.13) encodes the subtour elimination constraint and the capacities of each vehicle. This basic formulation, also denoted as the capacitated vehicle routing problem, is extended in many different ways such as multiple depots (see Renaud et al. (1996)), split deliveries (see Archetti et al. (2006)), backhauls (see Goetschalckx and Jacobs-Blecha (1989) and Toth and Vigo (1997)) and time windows (VRPTW, see (Toth and Vigo, 2002, p. 6) and Kallehauge et al. (2005)). An important subclass of vehicle routing problems with pickup and delivery operations is described in the following section.

3.2.2. Pickup and Delivery Problems

The large family of pickup and delivery problems (PDP) has been extensively studied in literature. The main difference to vehicle routing problems is the transportation of demand between two or more specific nodes. Typically, two dimensions are distinguished in PDPs: commodity number and either traveling salesman or vehicle routing constraints (see Table (3.1)).

According to (Cordeau et al., 2004, p. 431), three types of commodities exist in the literature: one-commodity (single-commodity), two-commodity and multi-commodity (n-commodity) pickup and delivery problems. In single-commodity

PDPs a single type of goods is either picked up at or delivered to a node. In two-commodity PDPs each node can be used to pickup or deliver commodities. This is a variant of the VRP with backhauls, where all deliveries must be performed before any pickup. In the multi-commodity (n-commodity) PDP, each commodity has a single pickup and a single delivery node. All three types of PDPs can be either connected with the traveling salesman problem (PDTSP) or the vehicle routing problem (VRPPD), respectively. The difference is that PDTSP need not consider capacities whereas VRPPD consider capacities on a per vehicle basis. Note that nowadays several PDTSP formulations consider per arc capacities.

Commodity type	**TSP**	**VRP**
One-Commodity PDP	1-PDTSP	VRPPD
Two-Commodity PDP	2-PDTSP	2-VRPPD
Multi-Commodity PDP	m-PDTSP	m-VRPPD

Table 3.1.: Classification of pickup and delivery problems.

For the single-commodity PDTSP, also called 1-PDTSP, see Hernández-Pérez and Salazar-González (2003) and Hernández-Pérez and Salazar-González (2004). For the multi-commodity PDTSP (m-PDTSP), see Hernández-Pérez and Salazar-González (2009) and Plum et al. (2012) which also use arc capacities. The n-commodity pickup and delivery requests can be also found in passenger transportation where they are called dial-a-ride problems (DARP), see for example Jaw et al. (1986). For the liner shipping industry, the 2-PDTSP and m-PDTSP are of special interest. The 2-PDTSP represent the empty container repositioning problem, where demand origins at one or more nodes and must be delivered to other one or more nodes. The m-PDTSP represents the integrated routing of laden containers because each cargo flow is transported between exactly one origin and one destination node. The m-PDTSP is described as a mixed integer programming formulation in the following paragraphs according to Hernández-Pérez and Salazar-González (2009). The objective is to find a cost minimal Hamiltonian path such that all the commodities are collected and delivered while holding the capacity constraints of the vehicle. A Hamiltonian path is a closed path in a graph that visits every node exactly once.

Let $G = (V, A)$ be a complete directed graph with n nodes $V = \{0, 1, \cdots, n, n{+}1\}$ and the arcs $A \subseteq V \times V$. The depot is at nodes 0 and $n + 1$. Let K be the set of commodities, each associated with a weight q_k, an origin s_k and destination d_k. The vehicle capacity per edge is Q. For any subset $S \subset V$, let $\delta^+(S) = \{(i, j) \in A : i \in S, j \neq S\}$ and $\delta^-(S) = \{(i, j) \in A : i \neq S, j \in S\}$. $\delta^+(S)$ stores the arcs whose origin i lies but the destination j lies not in the set S and vice versa for $\delta^-(S)$. c_a are the cost or distance for arc a. x_a is a $0/1$ variable indicating whether arc a is used on the Hamiltonian path. f_a^k is the flow of commodity k on arc a.

$$min \sum_{a \in A} c_a x_a \tag{3.15}$$

$s.t.$

$$\forall i \in V \backslash \{0\} : \sum_{a \in \delta^-(\{i\})} x_a = 1 \tag{3.16}$$

$$\forall i \in V \backslash \{n+1\} : \sum_{a \in \delta^+(\{i\})} x_a = 1 \tag{3.17}$$

$$\forall S \subset V, 0 \neq S : \sum_{a \in \delta^-(S)} x_a \geq 1 \tag{3.18}$$

$$\forall S \subset V, n+1 \neq S : \sum_{a \in \delta^+(S)} x_a \geq 1 \tag{3.19}$$

$$\forall i \in V, k \in K : \sum_{a \in \delta^+(\{i\})} f_a^k - \sum_{a \in \delta^-(\{i\})} f_a^k = \begin{cases} q_k, if\ i = s_k \\ -q_k, if\ i = d_k \\ 0, else \end{cases} \tag{3.20}$$

$$\forall a \in A : \sum_{k \in K} f_a^k \leq Q x_a \tag{3.21}$$

$$\forall a \in A : x_a \in \{0, 1\} \tag{3.22}$$

$$\forall a \in A, k \in K : 0 \leq f_a^k \tag{3.23}$$

The constraints (3.16) - (3.19) assure a simple cycle starting and ending at the depot $0 = n+1$. Constraints (3.20) ensure the pickup $q_k \geq 0$ and delivery $q_k \leq 0$ for all commodities. The capacity of the vehicle on all arcs a is defined in constraints (3.21). Constraints (3.22) define the integrality of the x_a variables and (3.23) the nonnegativity of the f_a^k variables. For further information and subtour elimination constraints see Hernández-Pérez and Salazar-González (2009).

There exist several extensions for the PDTSP, such as the work of Şahin et al. (2012) that extend the basic problem by multiple vehicles and split loads, transhipment extension such as described in Shang and Cuff (1996) and time windows such as in Mitrovic-Minic and Laporte (2006).

3.2.3. Min-Cost Flow Problems

A generalization of the classic transportation problem introduced by Orden (1956) is the class of min-cost flow problems (MCFP). Problem (3.24) - (3.26) defines the MCFP based on (Suhl and Mellouli, 2013, p. 192). The network is a directed graph $G = (V, E)$ with vertices N and edges E. The cost of transporting one unit on edge

(i, j) is c_{ij} and b_i is the demand at vertex i. b_i can be positive, meaning it offers supply, or negative when is has a demand. A transhipment vertex is indicated by $b_i = 0$.

$$min \sum_{(i,j)\in E} c_{ij}x_{ij} \tag{3.24}$$

$$s.t.$$

$$\forall i \in V : \sum_{(i,j)\in E} x_{ij} - \sum_{(j,i)\in E} x_{ji} = b_i \tag{3.25}$$

$$\forall (i, j) \in E : l_{ij} \leq x_{ij} \leq u_{ij} \tag{3.26}$$

The objective (3.24) is to minimize the transportation costs. Constraints (3.25) balance the flow in the network and (3.26) imposes lower and upper bound on the flow variables. This basic formulation uses a single commodity. A simple extension leads to the class of multi-commodity flow problems, formulated in (3.27) - (3.29), see for example (Fleischer and Skutella, 2006, p. 36). The main differences are the flow variables per commodity $k \in K$. This leads to commodity specific arc cost c_{ijk} and demands/supplies b_{ik} per node.

$$min \sum_{(i,j)\in E,k\in K} c_{ijk}x_{ijk} \tag{3.27}$$

$$s.t.$$

$$\forall i \in V, k \in K : \sum_{(i,j)\in E} x_{ijk} - \sum_{(j,i)\in E} x_{jik} = b_{ik} \tag{3.28}$$

$$\forall (i, j) \in E, k \in K : l_{ijk} \leq x_{ijk} \leq u_{ijk} \tag{3.29}$$

In this formulation, each flow variable x is extended by a arc-commodity pair x_{ijk}. The objective again is to minimize the commodity specific arc costs and balance the supply and demand.

To summarize, several related planning problems are presented in this section. Although these provide a basis, the problems arising in the maritime industry require major extensions that lead to a relatively new maritime research stream. The specific differences are outlined in Section 3.4.

3.3. Liner Shipping Network Planning

In this section, maritime liner shipping planning problems are presented. In particular, work in the field of liner shipping network design (LSNDP), cargo allocation (CAP) and empty container repositioning (ECRP) as well as speed optimization (SO) is analyzed regarding the cargo allocation and network design requirements presented in Chapter 2. Afterwards, the research gap is presented and the goals of this thesis derived.

One of the earliest publications specific to the maritime industry is the work of Appelgren (1969) and Appelgren (1971) to solve the ship scheduling problem for the tramp industry. He is followed by many publications in industrial, tramp and liner shipping. The reader is referred to the following comprehensive survey papers on routing and scheduling in the maritime industry: Ronen (1986), Ronen (1993), Christiansen et al. (2004), Kjeldsen (2011), Christiansen et al. (2012), Tran and Haasis (2013) and Meng et al. (2013).

3.3.1. Liner Shipping Network Design Problem

The liner shipping network design problem solves the complex planning task to create liner services and deploy capacity to the services. Depending on the publication, the capacity is either regarded as specific vessels or the vessel type. The LSNDP is also called vessel deployment or vessel routing problem in the liner shipping context. Most of the publications incorporate routing of laden containers into the network design to calculate the transhipment costs. In Section 3.3.2, publications relevant to this subproblem are discussed. Table 3.2 gives a comprehensive overview on the LSNDP publications and their features relevant to this thesis. Some of the works solve the similar hub-and-spoke network design problem where hub ports or complete liner services are predetermined. Due to the similarity they are also considered in Table 3.2.

The first column specifies the objective function of the publication that can be either cost minimization or profit maximization. Cost minimization usually requires the transport of all defined cargo flows. This can be unsuitable from a strategic or tactical point of view since it can lead to low utilized services to fulfill the cargo transportation. The second column specifies whether the repositioning of empty containers is integrated into the network design decisions. The third column specifies how the capacity constraints are integrated into the models, either per vessel, per vessel type or per leg. The fourth column defines whether transhipment and transhipment costs are considered. The next column defines whether the models determine the routes or are externally given. The sixth column marks literature that considers transit times in the network design. Columns seven to nine indicate whether the speed is variable and if it is subject to a port duration based on the moved containers and pilotage. Column ten and eleven indicate if the vessel

Publication	Objective (max/min)	Empty Container Repositioning	Capacity (1,V,VT)	Cargo transhipment (-,T,TC)	Routes predetermined/variable (P,V)	Transit times per port	Route types (C, B, CB)	Vessel speed fixed/variable (F,V)	Deadweight scales	Partner networks (SC=Slot Chartering)	Solution method	Largest instance's ports/cargo flow count
Pape (1980)	min	-	-	-	P	-	all	-	-	-	H	19/-
Rana and Vickson (1988)	max	-	1	-	V	-	C	F	-	-	L,B	20/-
Rana and Vickson (1991)	max	-	V	-	V	-	C	F	-	-	L,SG	20/-
Cho and Perakis (1996)	max	-	V	-	P	-	C	F	-	-	BB	-
Powell and Perkins (1997)	min	-	V	-	P	-	C	F	-	-	BB	-
Fagerholt (2004b)	min	-	V	-	P	-	C	F	-	-	H	40/-
Reinhardt et al. (2007)	min	-	V	TC	V	-	C,B	F	-	-	BC	5/5
Shintani et al. (2007)	max	X	VT	TC	V	-	C	V	-	-	GA	20/-
Agarwal and Ergun (2008)	max	-	VT	T	V	-	C	F	-	-	CG,B	20/114
Álvarez (2009)	max	-	VT	TC	V	-	C	F	-	-	TS	120/14k
Fagerholt et al. (2009a)	min	-	V	-	V	-	C	V	X	-	H	-
Chen and Zeng (2010)	max	X	V	-	V	-	C	F	-	-	GA	10/90
Agarwal and Ergun (2010)	max	-	VT	T	V	-	C	F	-	X	CG,B	10/27
Reinhardt and Pisinger (2010)	min	-	V	TC	V	-	C,B	F	-	-	BC	15/9
Andersen (2010)	min	-	V	TC	V	-	C	F	-	-	LNS	16/325
Gelareh and Pisinger (2011)	max	-	VT	TC	V	-	o	F	-	-	B	-
Meng and Wang (2011a)	min	X	VT	TC	P	-	C	F	-	-	BB	46/600
Wang and Meng (2012b)	min	-	VT	TC	P	-	C	V	-	-	BB	46/652
Brouer and Desaulniers (2012)	max	-	VT	TC	V	-	C,B	F	-	-	H	-
Kjeldsen (2012)	min	-	V	TC	V	-	C	V	-	-	CG	25/50
Plum et al. (2012)	min	-	VT	-	V	-	C	F	-	-	CG	16/16
Brouer et al. (2013)	max	-	VT	TC	V	-	C,B	V	-	-	TS,CG	110/4k
Plum et al. (2013b)	max	-	VT	TC	V	-	all	F	-	-	CG	20/37
Gelareh et al. (2013)	max	-	VT	TC	V	-	o	F	-	-	B,H	50/-
Mulder and Dekker (2013)	max	-	VT	TC	V	-	C	F	-	-	GA	58/-
Wang (2013)	min	X	VT	TC	P	-	C	F	-	SC	BB	20/-
Polat (2013)	min	-	VT	-	V	-	C	F	-	-	VNS	26/-
Song and Dong (2013)	min	X	VT	o	V	-	all	V	-	-	H	8/-
Plum et al. (2013a)	min	-	VT	-	V	-	C	F	-	-	BP	25/-
Guericke and Suhl (2013)	max	X	VT	TC	V	x	all	V	X	SC	GA	38/1.8k
Wang and Meng (2014)	max	-	VT	TC	V	o	C	F	-	-	CG	12/-

Table 3.2.: Literature on the liner shipping network design problem
(X) feature considered
(-) feature not considered
(o) feature partly considered
Capacity: 1=Single vessel, V=Vessels, VT=Vessel types
Transhipment: T=Transhipment possible, TC=Transhipment with cost
Route types: C=Pendulum and cyclic, B=Butterfly, all=Conveyor belt
Solution methods: L=Lagrangian Relaxation, CG=Column Generation, BB=Branch & Bound, BC=Branch & Cut, BP = Branch & Price, B = Benders decomposition, SG=Subgradient method, TS=Tabu Search, GA=Genetic Algorithm, LNS = Large Neighborhood Search, H = Heuristic

type's deadweight scale and partner networks are considered. The last two columns give information on the solution approach and the solved instance sizes (where applicable). Note that some authors do not provide information on the number of ports or cargo flows that lead to the main complexity in the LSNDP. Unconsidered or unmentioned features in a publication are indicated with a "-".

In the remainder of this section, publications on the liner shipping network design are described based on Table 3.2 in detail.

Publications without Transhipment and Single Service Planning

Pape (1980) is an early work on the fleet deployment problem. He assigns specific vessels to predetermined port sequences. The cargo flows are approximated by frequency the vessels call the ports. Port durations only depend on the size of the vessel. The publications of Rana and Vickson (1988) and Rana and Vickson (1991) are the first that solve the liner shipping network design problem. They formulate a non-linear model and solve it with a lagrangian relaxation using the subgradient method. Cho and Perakis (1996), Powell and Perkins (1997) and Fagerholt (2004a) extend the model by further aspects. The main downside of the publications are the lack of cargo transhipment operations that are nowadays heavily used in liner shipping (see (Álvarez, 2009, p. 188), (Kjeldsen, 2011, p. 143)). These models are not applicable to most of today's carriers. Chen and Zeng (2010) and Song and Dong (2013) focus on empty container repositioning and thereby do not include container transhipment in their work. Plum et al. (2012) propose a column generation approach for the multi-commodity one-to-one pickup and delivery problem with maximum path durations but do not include transhipment between paths. Polat (2013) solves the feeder service network design problem but does not consider transhipment costs because the cargo has to be transported directly. Plum et al. (2013a) formulate an exact model to solve the single liner shipping service design problem without transhipments between services. Gelareh et al. (2013) solve the single string planning problem on a real-world instance using a heuristic. They also do not consider transhipment for the existing services.

Publications without Transhipment Costs

Agarwal and Ergun (2008) introduce the ship scheduling and network design problem in the liner shipping context. They are the first who include transhipment in the network design problem but do not associate any costs with these operations. They consider weekly round trips in their model formulation and propose three different solution approaches: A greedy heuristic, a column generation method and a two-phase benders decomposition based algorithm. The methods are evaluated on randomly generated instances. Agarwal and Ergun (2010) extend the model of Agarwal and Ergun (2008) to support the joint optimization of networks of different carriers. They incorporate a game theory approach with the model developed in Agarwal and Ergun (2008) to find an optimal solution for several carriers.

Publications with Transhipment and associated Costs

The first publication that includes transhipment costs in the network design problem is the working paper of Reinhardt et al. (2007) and the publication Reinhardt and Pisinger (2010). Beside simple routes they include butterfly routes in their model and solve it using a customized Branch and Cut method. They model inter-service transhipment ports by counting the service legs using integer variables and pay attention for the *center* (hub) port. They solve instances up to 15 ports and nine demands to optimality. Álvarez (2009) also includes transhipment costs and suggests a solution approach using a tabu search and a multi-commodity flow model to evaluate a solution. Álvarez solves large scale instances up to 120 ports and, although not mentioned, probably thousands of cargo flows. He shows that a combination of metaheuristics joined with a multi-commodity flow problem (MCFP) can work for large instances. He models the MCFP as an arc flow formulation without empty container repositioning, deadweight scales, transit times or partners. The transhipment constraints from Álvarez (2009) are also used in the cargo allocation problem developed in this thesis. The work of Álvarez (2009) provides the basic solution approach for the work of Brouer et al. (2013).

In 2011 and 2012 several authors tackle subproblems of the liner shipping network design problem. Gelareh and Pisinger (2011) solve the fleet deployment, network design and hub-location problem using a decomposition method. They allow transhipment at hub ports but do not require the feeder networks to operate on round trips or allow transhipment between these services. Meng and Wang (2011a) include the empty container repositioning problem in the network design problem but require a manager to predetermine profitable services. Wang and Meng (2012b) solve the liner ship fleet deployment problem and include transhipment operations. Again, they assume externally given liner services. Brouer and Desaulniers (2012) introduce a mathheuristic for the LSNDP, combining greedy approaches with integer programming. They report promising preliminary tests but do not report numerical results. Repositioning of empty containers, variable speed and transit times are not included in the work of Brouer and Desaulniers (2012).

In 2013, several publications are released supporting not only pendulum, circle and butterfly but also conveyor belt routes (see Plum et al. (2013b), Song and Dong (2013) and Guericke and Suhl (2013)). Plum et al. (2013b) present a mixed integer program for the liner shipping network design problem with transhipment and conveyor belt routes. They introduce several additional constraints to tighten the LP relaxation of their model. Thereby, they are able to evaluate their model on the Baltic and WAF instances of the LINER-LIB 2012 (see Brouer et al. (2013)). Song and Dong (2013) introduce a three stage optimization method to solve the service route design problem. Repositioning of empty containers and conveyor belt route types of a single long-haul service is included in their model. To the best of the author's knowledge, Guericke and Suhl (2013) are the first that consider variable speeds, transit times and empty container repositioning in the LSNDP. They present an evolutionary algorithm approach where the fitness function was evaluated using a multi-commodity flow problem with an arc flow formulation based on Álvarez (2009).

Further, more recent approaches to solve the network design problem are Mulder and Dekker (2013) and Brouer et al. (2013). Mulder and Dekker (2013) propose a genetic algorithm and use an arc flow formulation similar to Álvarez (2009) to evaluate the fitness of the individuals. They introduce port clusters to reduce the size of the cargo allocation problem from Álvarez (2009) to speed up the evaluation of networks. These clusters are converted to feeder services during their solution approach. Mulder and Dekker (2013) do not consider empty container repositioning, transit times and variable speeds. Brouer et al. (2013) introduce the LINER-LIB 2012, which contains benchmark instances for the LSNDP. It contains seven instances based on Maersk Line's network and publicly available data. These instances also include data for future work, such as transit times per cargo flow. The LINER-LIB 2012 is also used in the scope of this thesis. Brouer et al. (2013) extend the solution approach from Álvarez (2009) and develop a column generation based method. They do not consider transit times and empty container repositioning. The port duration is assumed to be constant. They conclude their work with the importance of including transit time and slow steaming strategies in the LSNDP.

Several authors include the transit times in their work. The network design problem is extended by deadlines (LSNDPD) by Wang and Meng (2014). They formulate the LSNDPD as a non-linear non-convex program and propose a column generation based heuristic to solve the problem. The method is applied to a liner shipping network with 12 ports and 73 cargo flows on the Asia-Europe trade. The main drawback is that Wang and Meng (2014) do not allow transhipment between the liner services.

3.3.2. Cargo Allocation and Empty Container Repositioning Problems

Liner shipping network design problems that consider cargo transhipment must deal with the container routing because it determines a large proportion of the operating costs. The routing of containers must be considered in strategic, tactical and operational planning problems, each with specific requirements. In this section, the cargo allocation (cargo routing) problem (CAP) is reviewed from a strategic point of view. The repositioning of empty containers can be incorporated with the CAP problem because the balancing is interconnected with serving the cargo flows (see Chapter 2).

Therefore, this chapter not only reviews CAP publications, but also specific net-

Publication	Objective (max/min)	Empty container repositioning	Capacity constraints (VT,RG)	Cargo transhipment including transhipment cost(-,T,TC)	Transit times per port	Route types Cyclic/Butterfly/Conveyor belt (C, B, CB)	Vessel speed fixed/variable (F,V)	Port duration/ pilotage	Deadweight scales	Solution method	Largest instance's port/cargo flow number
Álvarez (2009)	max	-	VT	TC	-	C	F	-	-	TS	120/-
Reinhardt and Pisinger (2010)	min	-	VT	TC	-	C,B	F	-	-	BC	15/9
Kjeldsen (2012)	min	-	VT	TC	-	C	V	-	-	CG	25/50
Mulder and Dekker (2013)	max	-	VT	TC	-	C	F	-	-	GA	58/-
Crainic et al. (1993)	min	X	-	-	-	C	F	-	-	BB	-
Shen and Khoong (1995)	min	X	VT	-	-	C	F	-	-	BB	-
Song and Carter (2009)	min	X	VT	T	-	C	F	-	-	BB	-
Di Francesco et al. (2009)	min	X	VT	TC	-	C	F	-	-	BB	-
Song and Dong (2011)	min	X	VT	-	-	C	F	-	-	Sim	6/-
Epstein et al. (2012)	max	X	VT	T	-	C	F	-	-	BB	-
Brouer et al. (2011)	max	X	VT	TC	-	C	F	-	-	CG	234/16.3k
Wang et al. (2013b)	max	X	VT	-	X	C	V	X	-	BB	8/16
Wang et al. (2013c)	min	-	-	TC	X	C,B	F	-	-	BB	166/-
Wang (2014)	min	-	VT	TC	-	C	F	-	-	BB	40/458
Mhaky and Lee (2014)	max	-	VT	TC	X	CB	F	-	-	CG	33/300
Guericke and Tierney (2014)	max	X	RG	TC	-	CB	V	X	-	BB	39/365

Table 3.3.: Literature on the CAP within the LSNDP and the ECRP
(X) feature considered
(-) feature not considered
Capacity: VT=Vessel types, RG=Resource groups
Solution methods: CG=Column Generation, BB=Branch & Bound, BC=Branch & Cut, TS=Tabu Search, GA=Genetic Algorithm, Sim=Simulation

work design problems and empty container repositioning problems that route laden containers as well. Table 3.3 provides an overview on these publications. For presentational purposes, first network design problems that incorporate cargo allocation are presented. Afterwards, empty container repositioning problems (ECRP) and specific cargo allocation problems are introduced. For each publication, features relevant to the cargo allocation described in Chapter 2 are given.

Cargo Allocation within Liner Network Design

Álvarez (2009), Reinhardt and Pisinger (2010), Kjeldsen (2012) and Mulder and Dekker (2013) provide multi-commodity flow formulations within the network design to determine the transhipment costs. The publications have in common that they use decision variables within the flow balance constraints to determine the loaded and unloaded cargo at each port in each service. This modeling approach is also used in the arc flow formulation in Chapter 4. None of these publications integrate empty container repositioning or variable speeds.

Empty Container Repositioning

Crainic et al. (1993) formulate an operational empty container repositioning model and analyze the container inventories under stochastic demand and supply by using stochastic programming models. They do not consider the routing of laden containers but rather a fixed demand at each port. A decision support system to solve the multiperiod empty container repositioning problem is introduced by Shen and Khoong (1995). They include leasing-in and leasing-off decisions in their model. Song and Carter (2009) analyze the scale of empty containers of three major shipping routes and assess strategies to handle empty containers. They conclude that a strategy that coordinates the empty containers at routes and share empty containers among carriers reduced the repositioning costs by 12 – 18% in 2003 to 2007. They show that up to 28% of the container movements were the movements of empty containers and indicate a trend that this figure might increase in future. Di Francesco et al. (2009) use a multi-scenario approach to support the uncertain decision making of empty container repositioning. They solve their formulation using system dynamics. The ECRP with flexible destination ports is studied by Song and Dong (2011). In their formulation, laden containers were routed and empty containers can be spontaneously unloaded when needed. Epstein et al. (2012) develop a decision support system for the South American carrier CSAV. They use a multi-commodity, multi-period flow model and an inventory model to formulate this problem. They incorporate safety stock levels with the empty container problem but do not consider variable vessel speeds.

For further operational empty container repositioning problems the reader is referred to the work of Shintani et al. (2007), Saeidi et al. (2013) and Liu et al. (2010).

Plain Cargo Allocation

In this paragraph cargo allocation problems that explicitly model container movement is presented. Brouer et al. (2011) introduce the cargo allocation problem with empty container repositioning to evaluate networks in a strategic planning horizon.

They used time periods for the repositioning and suggest a column generation approach to solve this problem. With the help of a relatively small restricted master problem and shortest path pricing problems they solve large scale instances with up to 9 time periods, 234 ports and more than 16,000 cargo flows in less than one hour. Their solution method is promising for solving large scale instances of the cargo allocation problem in this thesis because time periods are not considered in the network design context. This has the ability to decrease the runtime. The work of Wang et al. (2013b) schedules container ships under consideration of transit time sensitive cargo flows and variable speeds. In their work, the demand quantity depends on the transit times offered by the carrier. However, they do not include container transhipments and empty container repositioning in their non-linear non-convex formulation.

Wang et al. (2013c) uses a link-based multi-commodity flow formulation to solve the container routing problem for one cargo flow under consideration of transit times and transhipment operations. The vessels' speed is assumed to be constant and no capacity constraint is imposed to the legs. They solved their integer programming model on 166 world wide ports using IBM CPLEX (2014). Wang (2014) proposes a novel hybrid-link-based container routing model and considers butterfly routes in mid-sized networks. They prove a unimodularity property of their integer programming model and could therefore solve large instances.

Mhaky and Lee (2014) include transit times in their cargo allocation problem. They define different service levels for their network that result in average speeds for the network. On each network, the cargo is routed with the speed optimization performed in a preprocessing step. They solve their cargo allocation problem using a column generation approach. No empty container repositioning or speed adjustment per leg is considered in their work.

Guericke and Tierney (2014) propose a cargo allocation model with speed optimization, empty container repositioning and cargo flow depended transit times. Their model is solved with Gurobi and indicates large runtimes, especially for the medium sized LINER-LIB networks. They do not consider deadweight scales or partner networks in the model.

3.3.3. Speed Optimization

Bunker (fuel) costs are a huge factor when moving vessels. A lot of research has been published on optimizing vessels' speed, both for the tramp and liner shipping industry. Table 3.4 gives an overview of literature on speed optimization problems. Note that these publications are related to both liner and tramp shipping. Speed optimization models from tramp shipping often incorporate time windows which make them also applicable to liner shipping by fixing the end of the voyage according to the round trip time. Table 3.4 shows the publications' features: bunker consumption function, port dependent bunker prices, uncertainty in bunker prices and port durations, the

Publication	Bunker consumption function (cbc, cubic see Ronen (1982)/) (quad, quadratic/other)	Port dependent bunker costs	Uncertain bunker prices	Port duration deterministic/ stochastic/cargo dependent (D,S,C)	Solution method	Largest instance's port count
Ronen (1982)	cbc	-	-	D	-	-
Fagerholt et al. (2009b)	cbc	X	-	D	NLP	16
Norstad et al. (2010)	cbc	-	-	D	Alg	-
Gatica and Miranda (2011)	cbc	X	-	D	BB	-
Meyer et al. (2012)	other	X	-	D	Form	-
Kim and Kim (2012)	cbc	X	-	-	L,H	28
Qi and Song (2012)	cbc	-	-	S	H	8
Yao et al. (2012)	cbc	X	-	D	BB	15
Wang and Meng (2012c)	cbc	-	-	C	BB	87
Vilhelmsen et al. (2013)	other	X	-	D	BB	38
Hvattum et al. (2013)	quad	-	-	D	Alg	-
Norlund and Gribkovskaia (2013)	cbc	-	-	D	Alg	10
Norstad et al. (2013)	other	X	-	D	BB	-
Kim (2013)	cbc	X	-	-	L,H	14
Sheng et al. (2013)	cbc	X	X	D	BB	15

Table 3.4.: Literature on the speed optimization
(X) feature considered
(-) feature not considered
Solution methods: BB=Branch & Bound, L=Lagrangian relaxation approach, H=Heuristic, NLP=Nonlinear programming solver, Form=Analytical formula, Alg=Exact algorithm

solution method and the instance size. Because speed optimization models typically do not incorporate cargo routing decisions, the cargo flows are omitted in Table 3.4. As can be seen in Table 3.4, the instances deal with smaller instances since it can be reasonable to decompose the problem per liner service. According to the schedule, the vessel can be jointly loaded, unloaded and bunkered and thus can be seen independent from other services.

One of the first publications in the area of speed optimization is Ronen (1982) who provide a cubic consumption formula of the vessel's speed. The optimal speed is calculated algebraically. Although several authors (see Wang and Meng (2012c) and Psaraftis and Kontovas (2013)) claim that the cubic consumption function leads to bad approximations for low speeds and several vessel types, the cubic approximation is often used in literature (see Table 3.4). Until the 2009, the attention on speed optimization was relatively low. This might be related to the relatively low bunker cost as indicated in Figure 2.15 in Section 2.10.

From an operational point of view, speed optimization must determine the speed of the vessels on each voyage leg and the ports where to bunker. Section 2.10 shows that the bunker price highly depends on the port. Therefore, most of the publications have port dependent bunker prices (see for example Fagerholt et al. (2009b), Yao et al. (2012), Kim and Kim (2012) and Kim (2013)). A non-linear speed optimization model considering time windows is presented by Fagerholt et al. (2009b). They optimize the vessels' speed on a per leg basis using an exact solution strategy. Yao et al. (2012) propose an integrated model that determines the speed per leg and the bunkering amount at each port of a liner service's round trip. They assume a deterministic port duration independent from the actual routed cargo. Their non-linear model's discretization using a piecewise linear function is solved using IBM CPLEX (2014). Similar work is done by Kim and Kim (2012). Beside the speed and bunkering port, they consider greenhouse gas emissions because of the imposed carbon taxes. They develop a heuristic to solve a practical case study on one transpacific liner service. Kim (2013) develops a lagrangian heuristic for the problem introduced by Kim and Kim (2012).

An integrated ship routing and scheduling with speed optimization for the tramp shipping industry is introduced by Norstad et al. (2010). The consideration of leg dependent cubic speeds leads to a non-linear convex optimization problem that is solved using a multi-start local search heuristic. A similar work is published by Gatica and Miranda (2011). They discretize the pickup and delivery time windows to enable practical constraints. Vilhelmsen et al. (2013) propose a column generation method to solve the tramp shipping routing and scheduling with bunker optimization. The application of the introduced tramp shipping routing models to the liner industry is limited because tramp ships do not operate on services. A supply vessel optimization model is presented in Norlund and Gribkovskaia (2013). Similar to tramp vessels they do not operate on round trip services.

Meyer et al. (2012) propose an analytical operational speed optimization model considering much more vessel specific parameters, such as fuel oil and lubricating oil consumption as well as waiting time at the ports. They conclude that the optimal vessel speed mainly depends on the freight rates and bunker prices, and that slow steaming is a very good economical vessel operating mode. Additionally, they conclude that the often applied cubic bunker consumption function is not appropriate to reflect real-world consumption curves.

A large challenge in liner shipping are uncertain port durations (see Notteboom (2006)). Therefore, Qi and Song (2012) include uncertain port times when optimizing vessel schedules. They solve this problem using a simulation-based method. They do not consider the routing of laden or empty containers. Recently, not only deterministic bunker prices, but also uncertain prices have been considered. Sheng et al. (2013) consider uncertainty in bunker prices and developed a multistage stochastic program in a rolling horizon to determine the port where to bunker. They apply their method to different liner services. An exact algorithm to determine the optimal speed is given in Hvattum et al. (2013). They assume given time windows for the fixed port calls but do not consider port durations by cargo flow routing.

Additional survey papers on tactical and operational speed optimization models are presented in Wang et al. (2013a) and Psaraftis and Kontovas (2013).

3.4. Research Gap and Opportunities

In this section the state-of-the-art regarding the cargo allocation and the liner shipping network design problem is summarized and the research gap is derived according to the problem definition in Chapter 2. Based on the research gap, the goals of this thesis are defined.

In the previous sections, related planning problems for the requirements specified in Chapter 2 are presented. Although vehicle routing and pickup and delivery problems share common properties with the liner shipping network design problem, such as round trips, capacities and origin-destination demands, the following main differences exist (for details on these aspects, see Chapter 2):

1. A liner service can start at every node (no fixed depot exists).

2. A port must not be served but can be served several times by one or more vessels.

3. The vessels' speed is a fundamental aspect in liner shipping because the cubic fuel consumption function imposes large costs.

4. Container transhipment between vessels is common practice and can be done at every port (more or less efficiently).

5. Empty containers must be repositioned in the network.

6. No nightly buffer between each round trip exists.

Due to these differences, the existing planning problems cannot be easily adapted to the LSNDP. Thus, researchers developed specialized solution methods. The state-of-the-art analysis has shown that lots of quantitative work has been performed to design and optimize single liner shipping services and whole networks. Notteboom states the key decisions for liner network planners as: Service frequency, fleet size, vessel size mix and number of port calls (see (Notteboom, 2006, p. 20)). These key decisions are already broadly supported in quantitative models in the literature. The recent results of Plum et al. (2013b) show that small instances can be solved close to optimality, whereas middle-sized instances with more than 100 ports can be solved efficiently using heuristics (e.g. Álvarez (2009) and Brouer et al. (2013)).

Beside these fundamentals aspects, Chapter 2 introduces further practical requirements for liner shipping networks such as empty container repositioning, transit times, slow steaming strategies and partner integration. The importance of these aspects is also highlighted by Brouer et al. (2011), Notteboom (2006), Gelareh et al. (2010), Brouer et al. (2013) and Wang and Meng (2014). They occur on different planning levels in liner shipping because they are connected with the cargo allocation in the network. Section 3.3 shows that a research gap still exists between the cargo allocation problem found in literature and the real-world requirements for this problem. More specifically, integrating the aspects of empty containers repositioning, speed optimization, deadweight scales, transhipment operations and liner carrier cooperation into the cargo allocation problem is still an open research topic.

The more difficult LSNDP not only relies on the cargo allocation but also determines the underlying liner shipping network structure. The state-of-the-art analysis in the previous sections shows that some of the aspects (such as transit times or empty container repositioning) are already partly integrated into subproblems of the network design. Especially the work of Wang et al. (2013c) and Wang et al. (2013b) are first steps towards the transit time consideration in liner shipping network design. Álvarez (2009) and Brouer et al. (2013) present heuristic solution methods to solve large scale instances of the liner shipping network design. However, the state-of-the-art analysis also indicate that no work considers the aspects transhipment, empty container repositioning, transit times, slow steaming strategies and partners in an integrated manner yet.

3.5. Goals of this Thesis

Based on the research gap presented in the preceding section, the goals of this thesis are as follows:

1. Evaluate large scale real-world liner networks to simplify the computer supported manual planning process.

2. Formalize missing practical requirements of liner networks and generate optimal solutions.

3. Develop optimization methods to automatically optimize medium-sized liner networks in reasonable computational time.

4. Integrate the methods into a prototypical decision support system.

In the following paragraphs, each goal is presented in detail.

Evaluate large scale real-world liner networks

Evaluating network changes is a complex task that includes determining cargo allocation, services' speeds and empty container balancing on predetermined networks. The challenge is the routing of thousands of cargo flows on arbitrarily capacitated services. This routing determines the duration in the ports and thereby the overall speed of the services' vessels. Solving this problem to optimality is difficult, if not impossible, if performed manually. The first goal of this thesis is to develop an optimal solution approach to allocate containers to services' legs to support the manual planning. With the help of the model, carriers can evaluate global impacts of regional changes to liner service and the performance of the overall network. The model extends the state-of-the-art by integrating the following aspects:

1. Integrated service speed optimization
2. Detailed consideration of service capacities
3. Deadweight dependent vessel drafts
4. Partner networks
5. Empty container repositioning

The cargo allocation should be integrated with the speed optimization to respect the interdependency between duration at the ports and the need to increase the speed. So far, capacities are considered on a per slot basis. In real-world networks, not only the (dry) slots limit the amount of transportable containers but also the reefer container plugs and the maximum deadweight. The load dependent draft of a vessel is another aspect that has not been integrated into the cargo allocation problem before. This physical constraint is highly important to ensure the feasibility and efficiency of a given network. Partner liner services should be integrated into the cargo allocation on a slot charter basis because it represents the most common form of cooperation. Finally, the consideration of empty container repositioning is important to assure capacity on the vessels.

The cargo allocation problem is able to support the *manual planning process* of liner network planners. This process consists of repetitively evaluating adjusted networks for optimization. The results of the cargo allocation provides information on the leg's utilization, cargo routing, services' speed and vessel draft. These information can be used to manually change the network structure. The second application of the method is the quantitative evaluation of networks within optimization algorithms. For both applications a short computational runtime is essential: On the one hand, users expect to get the result for this subproblem quickly, on the other hand, optimization algorithms rely on evaluating hundreds or thousands of networks. To achieve this goal, the cargo allocation problem should be solved within *a few* *seconds*, even for global scale liner networks. Thus, planners and heuristics can get an almost immediate feedback on the network changes. The methods to solve the cargo allocation problem are presented in Chapter 4.

Formalize real-world requirements for the LSNDP and generate optimal solutions

Based on the research gap and the requirements in Chapter 2, the state-of-the-art liner shipping network design problem is extended by the following major aspects:

1. Transit times
2. Embargo constraints
3. All cargo allocation aspects, namely:
   - Speed optimization
   - Service capacities
   - Deadweight dependent vessel drafts
   - Partner networks
   - Empty container repositioning

It has been shown in Chapter 2 that transit times are a major requirement for practical networks. Thus, solution approaches developed in this thesis respect transit times. Because this thesis aims to solve the strategic network design, transit times are based on a port-to-port basis to ensure that the network structure is capable to hold the maximum durations required by shippers.

Real-world liner services are often subject to cabotage constraints due to political reasons. Cabotage means that specific ports must not be served together within one service. Instead, the ports can be called via different services which implies transhipment operations.

Considering transit times in the network optimization introduces a trade-off between serving cargo (and respecting the maximum duration) and the cubic bunker consumption. This trade-off should be integrated in the solution approaches by enabling a per leg adjustment of the speed. The speed optimization should already be considered in the cargo allocation. Further aspects from the subproblem of cargo allocation are service capacities, deadweight scales, partner networks and empty container repositioning. These must be integrated into the liner shipping network design as well. Discussions with network planners from global carriers showed that considering these aspects in the liner network planning is essential to create practically applicable networks. The extensions are formalized as a mixed integer program. The mathematical model to solve the liner shipping network design problem is presented in Section 5.1.

Optimize the LSNDP for real-world instances in reasonable time

Brouer et al. (2013) show that the liner shipping network design problem is NP-hard and thus difficult to solve. Plum et al. (2013b) present nearly optimal solutions for small instances of the basic LSNDP without transit times, partners and slow steaming. Thus, we expect the integrated LSNDP not to be solvable to optimality, even for small instances in a reasonable amount of time and computational resources. To be able to solve practical instances, a suitable solution method must be developed. The runtime is not as constrained as in the cargo allocation problem due to its strategic or tactical planning horizon. Discussions with liner network planners and runtimes presented in literature indicate that *several hours* seem to be a reasonable upper bound for finding good solutions. With the help of this method, medium-sized instances with several thousand cargo flows, approximately 40 ports and the aspects presented above can be optimized. The solution approach to solve the liner shipping network design is presented in Section 5.2 and 5.3.

Integrate the developed Methods into a Decision Support System

The approaches to solve the cargo allocation and liner shipping network design problem heavily rely on the technical background of the users. To allow the applicability of the methods for network planners, the methods are integrated into an interactive decision support system (DSS) to support the manual and iterative planning process of liner network planners. This requires the modeling of the planning process in practice and a suitable mapping of the DSS according to this process. The usability of the methods is improved by providing a graphical user interface for the cargo allocation and network design methods.

The DSS is a proof of concept, answering the question of how to use Operations Research methods for the liner network planning problems occurring in practice. Minor questions are how to display liner service legs on world maps and define a responsive user interface with little maintenance overhead. The decision support system is presented in Chapter 6.

The remainder of this thesis is structured as follows according to the goals: Chapter 4 presents the concept and numerical results for the cargo allocation problem, Chapter 5 exact and heuristic methods to automatically optimize liner shipping networks in artificial and real-world networks. Chapter 6 presents the decision support system to provide the developed mathematical methods to liner network planners.

4. Evaluating Networks - The Integrated Cargo Allocation Problem

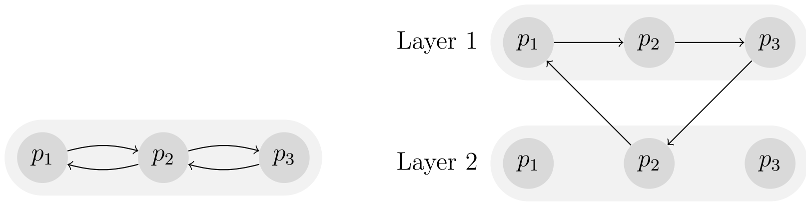
In this chapter, the integrated cargo allocation problem (CAP) for a given liner shipping network is described in detail. The objective is to obtain an optimal container allocation with maximized profit. The presented model extends the state-of-the-art regarding cargo allocation problems by integrating the following aspects into a single planning problem (see Chapter 2):

1. Speed optimization
2. Load dependent port durations
3. Load dependent vessel drafts
4. Multiple capacity types
5. Empty container repositioning
6. Support for all route types

First, a new approach to distinguish different calls of the same port and the mathematical notation for all models used in this thesis is presented. Next, two formulations of the same problem, namely an arc flow and a path flow formulation, are presented. Although the models incorporate a linearization of the non-linear bunker consumption function using integer variables, it is proved that the models can be solved to optimality using the linear relaxation. Finally, numerical results for the optimal and approximate solution approaches are shown.

4.1. Distinguishing Port Calls in Liner Services

Chapter 2 presents different route types used in real-world liner networks. The butterfly and conveyor belt routes call identical ports two times per round trip. Thus, cargo can be transshipped within the same service or between different services at different port calls. The calls of the same port must be distinguishable for a liner service to correctly calculate the transhipment costs. From the literature point of view, the port calls could be distinguished by marking the butterfly port (see Reinhardt and Pisinger (2010) and Brouer et al. (2013)), which allows butterfly, but not conveyor belt routes. A second approach is to separate the service legs (see Plum



(a) One layer representation.
(b) Two layer representation.

Figure 4.1.: Layered service representation of a butterfly rotation to correctly handle transhipments at port p_2.

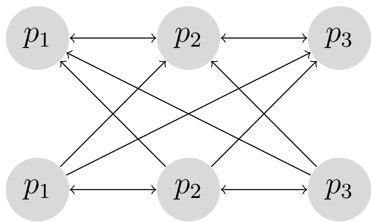


Figure 4.2.: All edges required to create arbitrary valid route types for three ports.

et al. (2013b)) to allow all route types. Plum et al. (2013b) use binary variables to distinguish the port call.

In the scope of this thesis another formulation is introduced that is based on different layers that enables all route types and correctly account for the transhipment costs. The graph of service legs and ports for service s is denoted as $G_s = (E_s, N_s)$, with edges E_s and nodes N_s. Ports are represented by nodes and legs by edges. To distinguish the calls of a same port p within one service the network is extended by layers $\mathbb{L}_s$. The set of nodes for a specific service s is then a set of tuples (p, l) with a port $p \in N_s$ and a layer $l \in \mathbb{L}_s$, $P_s \subseteq \{(p, l) : p \in E_s, l \in \mathbb{L}_s\}$. L_s is a set of tuples with layered legs that define a valid round trip. A round trip requires the edges to be connected, thus there must exist exactly one incoming and one outgoing edge for all layered ports in P_s, $\forall (p, l) \in P_s : |\{(i, p, l', l) : (i, p, l', l) \in L_s\}| = 1$ and $|\{(p, j, l, l') : (p, j, l, l') \in L_s\}| = 1$. Figure 4.1 shows a simple service with the port rotation $p_1 \to p_2 \to p_3 \to p_2 \to p_1$, allowing the port call differentiation of p_2 with a layered port rotation of $(p_1, 1) \to (p_2, 1) \to (p_3, 1) \to (p_2, 2) \to (p_1, 1)$. The layered leg structure for this service is $L_s = ((p_1, p_2, 1, 1), (p_2, p_3, 1, 1), (p_3, p_2, 1, 2), (p_2, p_1, 2, 1))$.

As can be seen in Figure 4.1, the service uses a second layer to distinguish the two calls of port p_2. Figure 4.2 shows all possible edges for the three ports to allow conveyor belt routes.

In Appendix B, an algorithm to create layered service legs based on a sequence of non-layered ports is presented. The algorithm has a runtime of $\mathcal{O}(|L_s|)$ for a service s and is thereby fast enough to transform large scale liner networks.

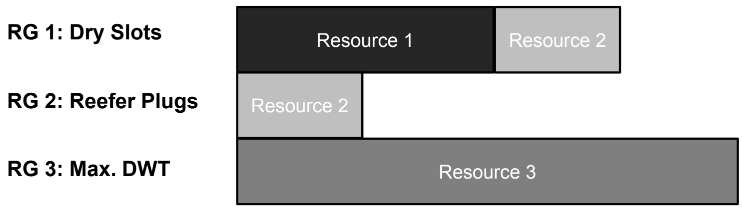


Figure 4.3.: Resource groups for a vessel type.

4.2. Common Notation

This thesis uses a common notation for the mathematical models which is presented in this section. Further model specific sets and parameters are described where needed.

Network Structure

A liner shipping network is represented by a set of liner services S. Liner services can be either operated by the carrier itself or by a partner. They are elements of the set of operated S^O or partner services S^P, respectively, with $S = S^O \cup S^P$ and $S^O \backslash S^P = \emptyset$. Each service s is represented by the tuple (VT_s, VC_s, L_s) where $VT_s \in VT$ is the service's vessel type, $VC_s \in \mathbb{N}$, $VC_s > 0$ the number of vessels deployed and $L_s \subset P_s \times P_s$ the set of connected layered edges (see previous section), called legs in the shipping industry. Let $L = P \times P$ be the set of service independent legs where P is the set of ports. $P_s \subseteq P \times \mathbb{L}_s$ is the set of layered ports called by service s. Let $C = E \cup N$ define the set of all transportable containers, either empty containers of type E or cargo flows N. Cargo flows are distinguished by cargo flows of the carrier N^O and partner cargo flows N^P. The set of all cargo flows consists of $N = N^O \cup N^P, N^O \cap N^P = \emptyset$. Whereas the transport of cargo flows from the operated carrier is optional, typically a fixed percentage of partner cargo flows must be transported due to contracts. The parameter $\theta \in [0, 1]$ indicates the percentage of partner cargo flows that must be transported.

The distance between port i and j for leg $(i, j) \in L$ is denoted as $l_{i,j}$ and given in nautical miles (nm). For simplicity, let l_s be the overall distance of service s, calculated as $l_s = \sum_{(i,j,l,l') \in L_s} l_{i,j}$.

Liner Service Capacities

As described in Chapter 2, a container vessel has different capacity types. We refer to each capacity type as a *resource group* and introduce a general concept that is shown in Figure 4.3 for container vessels.

Each of the resource groups contains one or more resources that utilize the specific resource group. Dry containers can be placed in both dry slots or reefer plugs. However, this reduces the plugs available for reefer containers that need a cooling. The resource groups for the example vessel "CHARLOTTE MAERSK" are defined as: Capacity of resource group $RG1$: $DrySlots$ are 9,757 (for resource 1, i.e. dry slots) plus 700 for resource group $RG2$: $ReeferPlugs$, together 10,457. Resource group $RG2$ is utilized by the resource 2 only, and has a capacity of 700. Reefer containers are utilizing both resource group $RG1$ and $RG2$, reducing the overall capacity available. The third resource group $RG3$: $Max.DWT$, the maximum deadweight, is the upper bound for all transported container payload. A forty foot dry container utilizes resource 1 by factor 2 and resource 3 by its specific weight. Beside the three resource groups presented, further types can be introduced for real operations. For example, length or commodity group capacities can be defined (due to security reasons). For tactical planning, the three resource groups offer sufficient detail (based on discussions with liner network planners). Note that the resource group concept can also be used for further container types and for mixed cargo vessel capacities, such as roll-on roll-off container vessels.

The resource group concept is formalized as follows to allow simple capacity modifications: Let RG be the set of resource groups, R the set of resources such as $weight$ and $R_{rg} \subseteq R$ the set of resources utilized by resource group r. In partner services, each leg can have a specific capacity. To formalize the leg dependent capacities, a set of segments $SG_s \subseteq L_s, s \in S^P$ is defined. A segment is a list of legs with specific resource capacities. Let the resource groups incident to segment $sg \in SG_s$ of partner service s, be denoted as RG_{sg}.

Cargo Flows

A cargo flow $n \in N, n = (o, d, q^{Max}, e, u, r)$ is defined as a container quantity with a maximum of $q_n^{Max} \in \mathbb{R}_+$ containers in the planning horizon between an origin and destination port $o_n \in P$, $d_n \in P$. A cargo flow has a specific equipment type $e_n \in E$ and has an associated resource utilization vector u. Let $u_{r,c}, r \in R, c \in C$ define the utilization of resource r of container $c, c \in C$. The revenue gained for the transport of one unit of cargo flow n is denoted as r_n. All monetary parameters and variables are in US$.

Costs and Revenues

The fixed port call cost when vessel type vt calls port p is defined as $\phi^{PC}_{p,vt}$ and ϕ^{D}_{vt} is the time charter or vessel depreciation cost for one vessel of type vt over the whole planning horizon. ϕ^{CH}_p is the container handling cost at port p at the cargo flow's origin or destination. ϕ^{TS}_p is the transhipment cost at port p. ϕ^{C}_e is an additional container cost for leasing, depreciation and hinterland repositioning for one unit of a cargo flow of type e. Let $bc(vt, k)$ be the bunker consumption per day at sea in metric tons (1000 kilograms) of a vessel of type vt that steams with k knots. Each metric ton (mt) costs ϕ^T US$, so the bunker costs per day is given by $bc(vt, k) \cdot \phi^T$. The slot cost on a partner service $s \in S^P$'s leg (i, j, l, l') for container type e is given by $\phi^S_{s,i,j,l,l',e}$.

Timing Aspects

Let τ denote the planning horizon length in days. Let $\tau^E_{p,vt}$ denote the duration in days to move (either load or unload) one container in port p with vessel type vt. This value can depend on the specific vessel type, because larger vessels are usually served by more cranes. Let f_s define the frequency of service s in days. The frequency determines how often a port is visited by an arbitrary vessel of service s. Note that throughout this thesis a constant value of 7 days is used. τ^{RT}_s is the number of round trips a vessel can perform on service s in the planning horizon τ. τ^{Add}_p is the additional time for vessels visiting port p in hours per port call. τ^{Add}_p contains the pilotage and additional buffer in days that can be used to reserve bunkering time or schedule delays.

Vessel Types

Each vessel type vt has a minimum and maximum speed in knots, k^{Min}_{vt} and k^{Max}_{vt}. Furthermore, a vessel type has a capacity of $C_{rg,vt}$ for a specific resource group. Let $C^O_{rg,vt}$ define the capacity per leg of resource group rg for vessel type vt of operated services and $C^P_{s,rg,i,j,l,l'}$ the capacity for leg segments for partner services $s \in S^P$. To simplify the capacity usage (for operated and partner services) in the mathematical models, the following parameter is defined:

$$C_{s,rg,i,j,l,l'} := \begin{cases} C^O_{rg,vt} \cdot VC_s \cdot \tau^{RT}_s, & \textit{if } s \in S^O \\ C^P_{s,rg,i,j,l,l'} \cdot \frac{\tau}{7}, & \textit{if } s \in S^P \\ 0, & \textit{else} \end{cases} \tag{4.1}$$

Each vessel type vt has a deadweight scale that defines the draft of a vessel subject to its deadweight load. Let dws^S_{vt} define the slope of the linearization and dws^I_{vt} the intercept of the y-axis. D^{Max}_p defines the maximum depth in meters of port p. It is assumed that the deployed vessel type VT_s for all services s respect the compatibility of the vessel type's lightship draft and the port depths.

4.3. Arc-flow Formulation for the Cargo Allocation Problem

The mathematical model for the integrated cargo allocation problem as a non-linear mixed-integer program is presented. It is modeled as a flow formulation with flow quantity decision variables for each services' legs for each equipment type and cargo flow. The model is based on the work of Álvarez (2009), but extended by empty container repositioning, speed optimization, deadweight scales and port dependent durations and partner services. The model is also presented in Guericke and Suhl (2013).

4.3.1. Mathematical Model

The arc flow formulation uses the following additional parameter:

$\mathbb{P} \geq 0$	Penalty costs for unserved partner cargo
$in_{s,p,l} \in L_s$	Stores the incoming leg for port p at layer l
$out_{s,p,l} \in L_s$	Stores the outgoing leg from port p at layer l

The arc flow formulation uses the following decision variables:

$\alpha_n \in \mathbb{R}_+$	Served quantity of cargo flow n, $\alpha_n \leq q_n^{Max}$ must hold
$x_{s,c,i,j,l,l'} \in \mathbb{R}_+$	Flow quantity of cargo flows or empty containers $c \in C$ on service s leg $(i,j,l,l') \in L_s$
$l_{s,c,p,l}, u_{s,c,p,l} \in \mathbb{R}_+$	Loaded and unloaded quantity of cargo flows and empty containers ($c \in C$) from/to liner service s at port $(p,l) \in P_s$
$k_s \in \mathbb{R}_+$	Speed in knots of all vessels deployed on service s
$\tau_s^B \in \mathbb{R}_+$	Uniformly distributed buffer in days in the planning horizon at all ports of service s
$\rho^P \in \mathbb{R}_+$	Slack variable for unserved contractually agreed partner cargo
$\rho_s^K \in \mathbb{R}_+$	Slack variable for vessel speeds above maximum speed

The non-linear cargo allocation problem presented in (4.2) – (4.18) is referred to as *CAParc*.

$$CAParc$$

$$max = \sum_{n \in N} (r_n - \phi_{e_n}^C)\alpha_n \tag{4.2}$$

$$- \sum_{s \in S^O} \left(\sum_{p \in P_s} (\phi_{p,VT_s}^{PC} \frac{\tau}{f_s}) - \phi_{VT_s}^D VC_s \right) \tag{4.3}$$

$$- \sum_{s \in S,(p,l) \in P_s} \sum_{n \in N: p=o_n \vee p=d_n} \phi_p^{CH}(u_{s,n,p,l} + l_{s,n,p,l}) \tag{4.4}$$

$$- \sum_{s \in S,(p,l) \in P_s} \sum_{n \in N: p \neq o_n \wedge p \neq d_n} \phi_p^{TS}(u_{s,n,p,l} + l_{s,n,p,l}) \tag{4.5}$$

$$- \sum_{s \in S^O, e \in E} \sum_{(p,l) \in P_s} \phi_p^{TS} \cdot (u_{s,e,p,l} + l_{s,e,p,l}) \tag{4.6}$$

$$- \sum_{s \in S^P, c \in C} \sum_{sg=(p,j,l,l') \in SG_s} \sum_{rg \in RG_{sg}} \sum_{r \in rg} \phi^S_{s,p,e_c} u_{r,c} l_{s,c,p,l} \tag{4.7}$$

$$- \phi^T \sum_{s \in S^O} (\tau VC_s$$

$$- \sum_{(p,l) \in P_s, c \in C} \left(\tau^E_{p,VT_s} (u_{s,c,p,l} + l_{s,c,p,l}) + \frac{\tau}{f_s} (\tau_p^{Add} + \tau_s^B) \right)) bc(k_s) \tag{4.8}$$

$$+ \mathbb{P} \left(\rho^P + \sum_{s \in S^O} \rho_s^K \right) \tag{4.9}$$

The objective of model *CAParc* consists of seven different terms: revenue for each transported cargo flow minus the container depreciation cost (term (4.2)). The fixed port call and vessel time charter costs for the predetermined network's services (term (4.3)) are added for reference purposes only and can be removed from the model. Next, container handling at the origin and destination ports for each cargo flow (term (4.4)), transhipment costs for transhipping cargo flows (term (4.5)) and empty containers (term (4.6)) from one service to another are imposed. Finally, partner slot cost in term (4.7) are subtracted. The bunker cost for all services for the time at sea during the planning horizon is calculated in term (4.8). If not enough partner cargo can be transported in the given network, the last term (4.9) adds penalty costs to the objective. Additionally, the service speed must be in the interval of minimum and maximum speed, if possible.

The constraints are divided into flow balancing and transhipment amounts for laden and empty containers, capacity constraints and speed constraints.

$$x_{s,c,in_{s,p,l}} + l_{s,c,p,l} = x_{s,c,out_{s,p,l}} + u_{s,c,p,l} \qquad \forall s \in S, c \in C, (p,l) \in P_s \tag{4.10}$$

$$\sum_{s \in S, l \in \mathbb{L}_s} l_{s,n,p,l} = \sum_{s \in S, l \in \mathbb{L}_s} u_{s,n,p,l} + \begin{cases} \alpha_n, & \mathit{if}\ p = o_n \\ -\alpha_n, & \mathit{if}\ p = d_n \\ 0, & \mathit{else} \end{cases} \qquad \forall p \in P, n \in N \tag{4.11}$$

$$\sum_{s\in S, l\in \mathbb{L}_s} l_{s,e,p,l} = \sum_{s\in S, l\in \mathbb{L}_s} u_{s,e,p,l} + \begin{cases} -\sum_{\substack{n\in N:\\ e_n=e}} \alpha_n, & if\ p = o_n \\ \sum_{\substack{n\in N:\\ e_n=e}} \alpha_n, & if\ p = d_n \\ 0, & else \end{cases} \quad \forall p \in P, e \in E \tag{4.12}$$

$$\sum_{\substack{r\in rg,\\ c\in C}} u_{r,c} x_{s,c,i,j,l,l'} \leq C_{s,rg,i,j,l,l'} \qquad \forall s \in S, (i,j,l,l') \in L_s, rg \in RG \tag{4.13}$$

$$\sum_{c\in C,(i,p,l',l)\in L_s} u_{weight,c}\, dws^S_{VT_s} x_{s,c,i,p,l',l} + dws^I_{VT_s} \leq D^{Max}_p \frac{\tau}{f_s} \qquad \forall s \in S^O, (p,l) \in P_s \tag{4.14}$$

$$\sum_{c\in C,(p,j,l,l')\in L_s} u_{weight,c}\, dws^S_{VT_s} x_{s,c,p,j,l,l'} + dws^I_{VT_s} \leq D^{Max}_p \frac{\tau}{f_s} \qquad \forall s \in S^O, (p,l) \in P_s \tag{4.15}$$

$$k_s = \frac{\sum_{(i,j,l,l')\in L_s} l_{i,j}}{f_s VC_s - \sum_{(p,l)\in P_s, c\in C} \left(\tau^E_{p,VT_s} \frac{1}{\tau/f_s} (u_{s,c,p,l} + l_{s,c,p,l}) + \tau^{Add}_p + \tau^B_s \right)} \qquad \forall s \in S^O \tag{4.16}$$

$$k^{Max}_{VT_s} \leq k_s \leq k^{Max}_{VT_s} + \rho^K_s \qquad \forall s \in S^O \tag{4.17}$$

$$\sum_{n\in N^P} \alpha_n \geq \theta \cdot \sum_{n\in N^P} q^{Max}_n - \rho^P \tag{4.18}$$

Constraints (4.10) ensure the flow balance at each service port. Containers on a service's vessel are either continued to be transported or unloaded at a port. The incoming leg to port (p,l) of service s is obtained by the left hand side. The right side determines the outgoing flow of port (p,l). The unloaded containers either arrive at their destination or must be picked up by another service's vessel. Constraints (4.11) and (4.12) ensure that unloaded containers are picked up again later. The explicit modeling of unloaded and laden containers is required to associate transhipment costs in the objective function. Furthermore, these constraints provide demand and supply at the ports.

Constraints (4.13) limit the capacity for each service s and resource group rg. Constraints (4.14) and (4.15) ensure that the draft of a vessel, determined by its deadweight, does not exceed the depth of the port. Both, when a service enters and leaves the port the deadweight scale constraints must be considered. Constraints (4.16) set the required speed for each service operated by the carrier. The speed is calculated by taking the distance of the service divided by the duration at sea available to perform the round trip. Each service is operated by a fixed frequency f, leading to a round trip time of f times the number of deployed vessels VC_s. The time at sea can be calculated by subtracting the duration at all service's ports that is determined by fixed parameters (such as duration for pilotage and additional buffer) and by the unloaded and loaded volume multiplied by the duration to move one container.

Constraints (4.18) ensure that a fixed percentage of partner cargo is transported. Otherwise the missed amount is penalized in the objective function by activating variable ρ^P. Due to the speed calculation in constraints (4.16), the presented model is a mixed integer non-linear program (MINLP) and could be solved using commercial or open source non-linear solvers, such as BARON (2014) and Couenne (2014).

4.3.2. Bunker Cost Discretization

The MINLP *CAParc* is expected to be hard to solve (see for example (Byrd et al., 2006, p. 35)). Thus, the non-linear constraints are linearized using the L01 method described in Padberg (2000). The key idea is not to calculate the bunker cost based on the speed of the services but rather based on the overall port duration per service as sea. This duration can be calculated with a linear expression. The duration is mapped to a piecewise port call duration linearization for which the resulting bunker costs can be calculated in advance. Based on the overall duration, the service speeds result from the remaining duration to travel the total service distance. The speed times the duration is used to determine the overall bunker cost for the time at sea in the planning horizon. The linearization approach is described in this section.

The denominator in constraints (4.16) is linearized and inserted into the objective function *CAParc* to get a function that determines the bunker cost based on the overall port duration per service. The duration is discretized into $i \in \mathbb{D}$ intervals. The variables $z_{s,0}, z_{s,1} \cdots z_{s,\mathbb{D}}$ and $y^L_{s,1}, \cdots y^L_{s,\mathbb{D}}$ per service $s \in S^O$ are introduced according to a discretization approach (see Padberg (2000)). The approximated function has $\mathbb{D}$ support points where it is piecewise linearized per interval. The z-variables represent the overall port duration for all vessels and for all round trips in the planning horizon for a specific service s. The optimization problem *CAParc* is then extended by the following constraints (see Padberg (2000)):

$$z_{s,0} \leq a_{s,1} - a_{s,0} \qquad \forall s \in S^O \tag{4.19}$$
$$z_{s,i} \geq (a_{s,i} - a_{s,i-1}) \cdot y_{s,i} \qquad \forall 1 \leq i \leq \mathbb{D} - 1, s \in S^O \tag{4.20}$$
$$z_{s,i+1} \leq (a_{s,i+1} - a_{s,i}) \cdot y_{s,i} \qquad \forall 1 \leq i \leq \mathbb{D} - 1, s \in S^O \tag{4.21}$$
$$z_{s,i} \geq 0 \qquad \forall 0 \leq i \leq \mathbb{D}, s \in S^O \tag{4.22}$$
$$y_{s,i} \in \{0, 1\} \qquad \forall 1 \leq i \leq \mathbb{D}, s \in S^O \tag{4.23}$$

The constraints (4.19) – (4.23) successively activate $y_{s,i}$ until the required port duration is reached. The duration, when using the discretization, is the sum of $z_{s,i}$ variables and the x-offset $a_{s,0}$. The total port duration of service s at the support point i is $a_{s,i}$.

$$PCD_s = a_{s,0} + \sum_{i=1}^{\mathbb{D}} z_{s,i} \tag{4.24}$$

The bunker cost for service s is calculated as follows and must be added to the objective function of the cargo allocation problem:

$$\phi_s^B = b_{s,0} + \sum_{i=1}^{\mathbb{D}} \frac{b_{s,i} - b_{s,i-1}}{a_{s,i} - a_{s,i-1}} z_{s,i} \tag{4.25}$$

Using the L01 approach, the non-linear bunker cost are linearized. With increasing port call duration, the speed for all vessels must be increased to hold the required frequency f_s. The port call duration is limited by the maximum speed of a vessel. The $b_{s,i}$ parameters in equation 4.25 are precalculated function values of the nonlinear bunker cost function for durations in the interval $[a_{i-1}, a_i]$.

Using a linearization of the bunker cost overestimates the real costs. With an increased number of support points $\mathbb{D}$, the bunker cost decreases because the accuracy of the *real* bunker consumption increases. Using artificial and real-world instances, 20 uniformly distributed support points seemed to be reasonable for a strategic planning horizon.

To let the model determine the bunker cost correctly the port duration term (4.24) must equal the port call duration denominator in constraints (4.16):

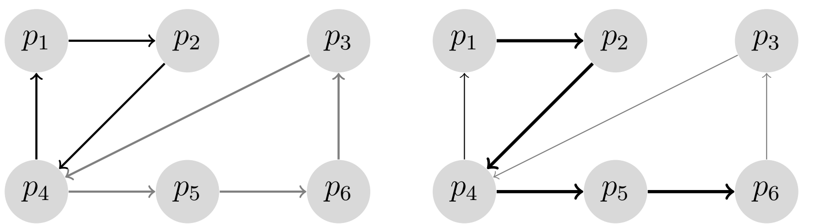
$$PCD_s = a_{s,0} + \sum_{i=1}^{k} z_{s,i} = \sum_{\substack{p \in P_s,\\ l \in \mathbb{L}_s, c \in C}} (u_{s,c,p,l} + l_{s,c,p,l}) \cdot \frac{1}{tp_p} \cdot \frac{1}{24} + \frac{\tau_p^{Add}}{24} \cdot \tau_s^{RT} \cdot VC_s + \tau_s^B$$

With the help of the discretization, the original mixed integer non-linear program *CAParc* is transformed into a mixed integer program (MIP) that can be solved with commercial MIP solvers such as IBM CPLEX (2014) and Gurobi (2014). Note that the transformed model is just an approximation of the original *CAParc* model because the cubic bunker cost function is discretized. In the numerical results of this chapter the approximation quality is analyzed.

4.4. Path-Flow Formulation for the Cargo Allocation Problem

The mathematical model *CAParc*, described in the previous section, is an arc flow formulation. It contains decision variables that store the container flows for all legs, services and commodities. The drawback of this formulation is the large number of decision variables and constraints that can lead to long solution times. One method that has been shown to work well in large scale CAP instances by Brouer et al. (2011), is a delayed column generation approach. An adaption of this concept for the integrated cargo allocation problem is presented in this section. With the help of the column generation approach we aim to improve the runtimes to determine an optimal container allocation. The reformulated model solves the same problem as described above, including speed optimization and empty container repositioning with respect to the capacity constraints.

With the help of Figure 4.4, the basic idea of the path flow formulation is shown. Two services are given in this example, service one (black) and service two (gray) (see Figure 4.4(a)).



(a) Liner network containing two services, the back and the gray.

(b) Container path between port p_1 and p_6 incident to two services.

Figure 4.4.: Container path for a cargo flow c_1 incident to two services.

Cargo flow c_1 should be routed from $p_1 \to p_6$, leading to the bold arcs in Figure 4.4(b). In the arc flow formulation, the variables x_{s_1,c_1,p_1,p_2}, x_{s_1,c_1,p_2,p_4}, x_{s_2,c_1,p_4,p_5} and x_{s_2,c_1,p_5,p_6} would have the value 100 when transporting 100 containers of cargo flow c_1. The path flow formulation for the routing in Figure 4.4(b) defines a container path $o = (s_1, p_1), (s_1, p_2), (s_1, p_4), (s_2, p_4), (s_2, p_5), (s_2, p_6)$ and one decision variable λ_o. For this path, λ_o is set to 100 if 100 containers of commodity c_1 are transported on this path. Furthermore, a profit can be associated with the container path o, i.e. the container and handling costs at port p_1 and p_6 and transhipment costs at port p_4, because the commodity is transhiped from service s_1 to s_2. Details on the network structure to correctly obtain the costs associated with a container path used in the delayed column generation are presented in Section 4.4.2.

4.4.1. Restricted Master Problem

Chapter 3.1.2 introduces the delayed column generation method. This method iteratively solves a restricted master problem (RMP), denoted *CAPrmp*. The RMP uses container paths to reduce the number of decision variables and constraints.

The path flow formulation requires the following sets in addition to the ones presented in Section 4.2:

O	Set of container paths in the restricted master problem
SL_o	Set of service legs used by container path o. $SL_o \subseteq \{(s,i,j,l,l')\|s \in S, (i,j,l,l') \in L_s\}$

The container paths per cargo flow use the following parameters:

p_o	Profit for transporting one container on container path $o \in O$
n_o	Cargo flow of container path $o \in O$, $n_o \in N$
$\tau^P_{o,s}$	Overall port duration in days for service s when one container is transported on path o
τ^{Min}_s	Minimum port call duration of service s $\tau^{Min}_s = \left(7 \cdot VC_s - \min(7 \cdot VC_s, \frac{l_s}{k^{Min}_{VT_s} \cdot 24})\right) \cdot \frac{\tau}{7}$
τ^{Max}_s	Maximum port call duration of service s $\tau^{Max}_s = \left(7 \cdot VC_s - \min(7 \cdot VC_s, \frac{l_s}{k^{Max}_{VT_s} \cdot 24})\right) \cdot \frac{\tau}{7}$
$in_{s,p,l} \in L_s$	Stores the incoming leg for port p at layer l
$out_{s,p,l} \in L_s$	Stores the outgoing leg from port p at layer l

The path flow formulation uses the following decision variables:

$\lambda_o \in \mathbb{R}^+_0$	Amount of containers of cargo flow n_o routed on the container path o. The variables are not constrained to be integer since we allow the transportation of container fractions which should be sufficient in the considered long planning horizons
$x_{s,e,i,j,l,l'} \in \mathbb{R}^+_0$	Flow quantity of empty containers $e \in E$ on leg $(i,j,l,l') \in L_s$ of service s
$l^E_{s,e,p,l}, u^E_{s,e,p,l} \in \mathbb{R}^+_0$	Loaded and unloaded quantity of empty containers $(e \in E)$ from/to liner service s at port $(p,l) \in P_s$
$\tau^B_s \in \mathbb{R}^+_0$	Additional buffer used for service s

$\rho^P \in \mathbb{R}_0^+$	Penalty cost for contractually agreed unserved partner cargo
$\rho_s^K \in \mathbb{R}_0^+$	Penalty cost for steaming above maximum speed

The model below uses the bunker cost linearization approach presented in the previous section.

$$CAPrmp$$

$$max \sum_{o \in O} p_o \lambda_o \tag{4.26}$$

$$- \sum_{s \in S^O} \left(\sum_{(p,l) \in P_s} \phi^{PC}_{p,VT_s} \frac{\mathcal{T}}{f_s} - \phi^{D}_{VT_s} \cdot VC_s \right) \tag{4.27}$$

$$- \sum_{s \in S,} \sum_{(p,l) \in P_s, e \in E} (l^E_{s,p,l,e} + u^E_{s,p,l,e}) \phi^{TS}_p \tag{4.28}$$

$$- \sum_{s \in S^P, e \in E} \sum_{sg=(p,j,l,l') \in SG_s} \sum_{rg \in RG_{sg}} \sum_{r \in rg} \phi^S_{s,p,e} u_{r,c} l^E_{s,c,p,l} \tag{4.29}$$

$$- \sum_{s \in S^O} \left(b_{s,a_0} + \sum_{i \in \mathbb{D}: i \geq 1} \frac{b_{s,a_i} - b_{s,a_{i-1}}}{a_i - a_{i-1}} z_{s,i} \right) \tag{4.30}$$

$$+ \mathbb{P} \left(\rho^P + \sum_{s \in S^O} \rho^K_s \right) \tag{4.31}$$

Term (4.26) provides the profit of transporting one container on path $o \in O$. The profit is calculated by taking the revenue and subtracting the container depreciation, handling and transhipment costs. The container path profit is calculated in the pricing subproblem (see Section 4.4.2). The term (4.27) subtracts the fixed port and charter costs. The term (4.28) of the objective calculates the empty container transhipment costs. Note that the empty container balancing is done in the restricted master problem because it depends on the cargo flow volumes of the container paths. Term (4.29) defines the slot cost for the empty containers on the partner services. The term (4.30) calculates the bunker costs for the whole planning horizon according to the previously introduced linearization. It is based on the duration for transporting empty and laden containers. Term (4.31) penalizes unserved partner cargo and vessel speeds above the type's maximum speed.

$$\sum_{o \in O: n_o = n} \lambda_o \leq q^{Max}_n \qquad \forall n \in N \tag{4.32}$$

$$\sum_{r \in rg} \left(\sum_{e \in E} u_{e,r} \cdot x^E_{s,i,j,l,l',e} + \sum_{o \in O:(s,i,j,l,l') \in SL_o} u_{n_o,r} \lambda_o \right) \leq C_{s,rg,i,j,l,l'} \qquad \forall s \in S, (i,j,l,l') \in L_s, rg \in RG_s \tag{4.33}$$

$$\left(\sum_{e \in E} u_{e,weight} x^E_{s,i,j,l,l',e} + \sum_{o \in O:(s,i,j,l,l') \in SL_o} u_{n_o,weight} \lambda_o \right) dws^S_{VT_s} + dws^I_{VT_s} \leq D^{Max}_j \frac{\tau}{f_s} \qquad \forall s \in S^O, (i,j,l,l') \in L_s \tag{4.34}$$

$$\left(\sum_{e \in E} u_{e,weight} x^E_{s,i,j,l,l',e} + \sum_{o \in O:(s,i,j,l,l') \in SL_o} u_{n_o,weight} \lambda_o \right) dws^S_{VT_s} + dws^I_{VT_s} \leq D^{Max}_i \frac{\tau}{f_s} \qquad \forall s \in S^O, (i,j,l,l') \in L_s \tag{4.35}$$

$$x^E_{s,in_{s,p,l},e} + l^E_{s,p,l,e} = x^E_{s,out_{s,p,l},e} + u^E_{s,p,l,e} \qquad \forall s \in S, (p,l) \in P_s, e \in E \tag{4.36}$$

$$\sum_{s \in S,(p,l) \in P_s} l^E_{s,p,l,e} = \sum_{s \in S,(p,l) \in P_s} u^E_{s,p,l,e} + \sum_{o \in O: e_{n_o} = e \wedge d_{n_o} = p} \lambda_o - \sum_{o \in O: e_{n_o} = e \wedge o_{n_o} = p} \lambda_o \qquad \forall p \in P, e \in E \tag{4.37}$$

$$\sum_{(p,l) \in P_s} \tau^{Add}_p \frac{\tau}{f_s} + \sum_{(p,l) \in P_s, e \in E} \left(\tau^E_{p,VT_s} (u^E_{s,p,l,e} + l^E_{s,p,l,e}) + \tau^B_{s,p,l} \right) + \sum_{o \in O:(s,i,j,l,l') \in SL_o} \tau^P_{o,s} \lambda_o = \sum_{i \in \mathbb{D}: i \geq 1} z_{s,i} + \rho^K_s \qquad \forall s \in S^O \tag{4.38}$$

$$z_{s,i} \geq (a_i + a_{i-1}) y_{s,i} \qquad \forall s \in S^O, i \in \mathbb{D} : i \geq 1 \tag{4.39}$$

$$z_{s,i+1} \leq (a_{i+1} - a_i) y_{s,i} \qquad \forall s \in S^O, i \in \mathbb{D} : i \geq 1 \wedge i < |\mathbb{D}| - 1 \tag{4.40}$$

$$\tau_s^{Min} \leq \sum_{i \in \mathbb{D}} z_{s,i} \leq \tau_s^{Max} \qquad \forall s \in S^O \tag{4.41}$$

$$\sum_{n \in N^P, o \in O: n_o = n} \lambda_o \geq \theta \sum_{n \in N^P} q_n^{Max} - \rho^P \tag{4.42}$$

Constraints (4.32) limit the maximum amount of transported volume for each cargo flow. Constraints (4.33) set the maximum transported quantity of empty and laden containers for all services, layered legs and utilized resource groups. Constraints (4.34) and (4.35) limit the cargo volumes that enter or leave a service's port by the vessel type's deadweight scale. Constraints (4.36) balance the empty container flow within the RMP. The empty container transhipment and empty container demand as well as supply are calculated in (4.37). Constraints (4.38) calculate the overall port duration per service by activating the $z_{s,i}$ variables. For these variables, constraints (4.39) and (4.40) specify the variable activation range, according to the formulation in Padberg (2000). Constraints (4.41) set the minimum and maximum port call duration for the whole planning horizon according to vessel speeds. τ_s^{Min} is the minimum port duration according to the maximum speed and is calculated in the parameter list. Respectively, τ_s^{Max}, is the maximum duration when steaming at minimum speed. $\tau_s^{Max} < 0$ means that not enough vessels are deployed on the service to perform the round trip with maximum speed. In this case the model makes use of the penalty variable ρ_s^K to artificially increase the speed.

Constraint (4.42) ensures a minimum served amount of partner cargo, if possible. Otherwise, the penalty variable ρ^P is activated.

4.4.2. Network Structure for determining Container Paths

Generating new container paths, i.e. extending the set of containers paths O, is done in the pricing (sub)problem. The pricing problem is solved for each cargo flow independently. The input for the cargo allocation problem is a fixed liner shipping network, such as the example in Figure 4.5 with two services. The objective is to find a path from an origin to a destination on which to route each cargo flow. On that path, different costs can occur, in particular transhipment to other services, transhipment within a service (due to butterfly or conveyor belt routes), container handling and slot purchasing cost. To make use of efficient solution methods to determine container paths in the pricing problem, the network structure makes use of the layers presented in Section 4.1 and artificial connections between services and ports.

Figure 4.6 shows the corresponding network to the original liner service network in Figure 4.5. The pricing network structure is similar to the one presented by (Wang et al., 2013c, p. 3) but is extended by multiple source and target ports. Thus, the network structure can be created for all pricing problems and column generation

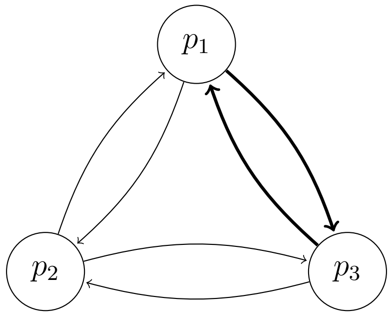


Figure 4.5.: Flat network structure with two liner services as input for the pricing network 4.6.

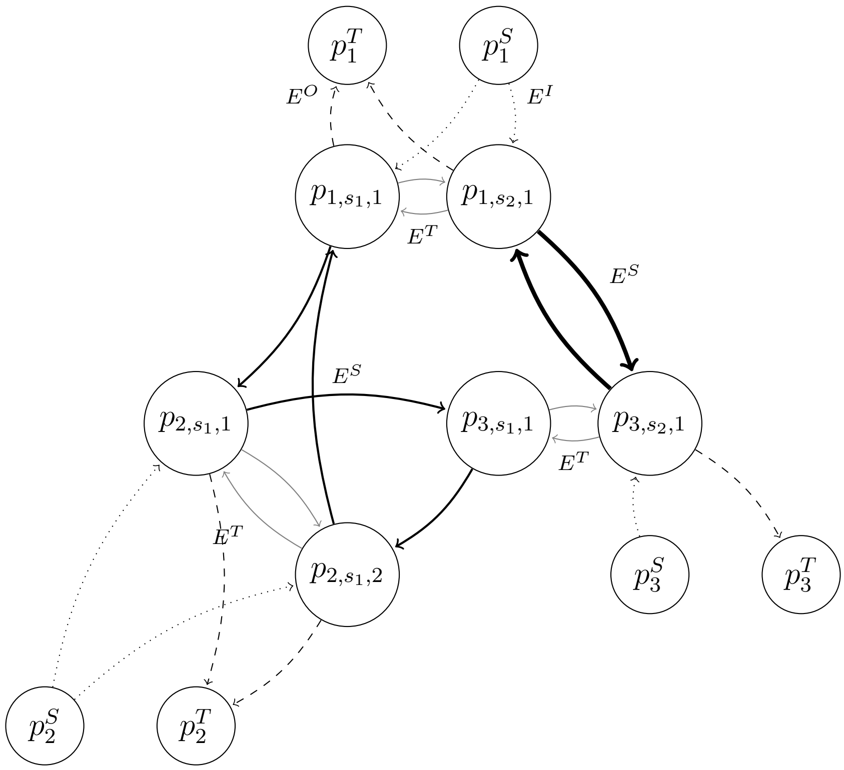


Figure 4.6.: Network structure for the pricing problem for the example in Figure 4.5 for one cargo flow n. The black edges are service legs, the remaining edges are artificial edges for the pricing problem.

Dual	Description	Constraint	LB	UB
π_n	Duals for serving a cargo flow n	(4.32)	0	∞
$\delta_{s,i,j,l,l',rg}$	Duals for the capacity constraint	(4.33)	0	∞
$\gamma^I_{s,i,j,l,l'}$	Duals for the ingoing DWTS constraint	(4.34)	0	∞
$\gamma^O_{s,i,j,l,l'}$	Duals for the outgoing DWTS constraint	(4.35)	0	∞
$\phi^{EC}_{p,e}$	Duals for the empty container balancing	(4.37)	$-\infty$	∞
π^{PC}_s	Duals for the port call duration constraint	(4.38)	$-\infty$	∞
π^{PCF}	Duals for the unserved partner cargo flows	(4.42)	$-\infty$	0

Table 4.6.: Dual variables and ranges in the cargo allocation path flow formulation.

iterations at once. To account for different cargo flows and CG iterations, only the edge costs must be updated.

For each port i with a cargo flow destination and for each cargo flow origin, an artificial node p_i^T and p_i^S is introduced respectively. For each port that is called by a service, a service specific node $p_{i,s,l}$ is introduced and connected to the source port with edge $e \in E^I$ and to the destination port with edge $e \in E^O$. The port also depends on the layer on which the port is defined in the service to distinguish several calls of the same port within a service round trip. The ports $p_{2,s_1,1}$ and $p_{2,s_1,2}$ are an example for the butterfly service in Figure 4.5.

Two or more services can call the same port. Thus, transhipments between services can be performed at that port (see, for example, port p_1 and p_3 in network 4.6). Liner service ports that are called by another service are connected by a transhipment edge $e \in E^T$. If a service calls a port more than once per round trip, a transhipment edge is introduced between the different port layers (see for example p_2 of service s_1 in Figure 4.6). Beside transhipment and cargo handling edges, an edge $e \in E^S$ per service's leg is introduced. Each edge of the pricing problem has a coefficient that is described in the remainder of this section.

The restricted master problem *CAPrmp* defines the following dual variables that are part of the edge costs. Duals are "shadow" costs resulting from increasing the capacity of the corresponding constraint by one (marginal) unit. Table 4.6 shows the dual variables from the restricted master problem that are relevant to determine a new container path $o \in O$ for a cargo flow n.

The duals can be used to calculate the reduced costs, i.e., the condition for a container path to possibly lead to an objective improvement. Reduced costs express the benefit for activating a column with one marginal unit, respecting the interdependency of other variables. The equation for the reduced cost can also be derived from transforming model *CAPrmp* to its dual formulation, determining the constraints that are incident to a cargo flow n and summing up the negated left hand sides of the constraints.

$$
\begin{aligned}
\bar{c_o} &= p_o - \pi_{n_o} \\
&- \sum_{(s,i,j,l,l')\in SL_o} \left(\sum_{rg\in RG} \sum_{r\in rg} u_{n_o,r}\delta_{s,i,j,l,l',rg} + u_{n_o,weight} dws^S_{VT_s} (\gamma^I_{s,i,j,l,l'} + \gamma^O_{s,i,j,l,l'}) \right) \\
&+ \phi^{EC}_{o_n,e_n} - \phi^{EC}_{d_n,e_n} - \sum_{s\in S^I_o} \tau^P_{o,s}\pi^{PC}_s - \begin{cases} \pi^{PCF}, & \textit{if } n \in N^P \\ 0, & \textit{else} \end{cases}
\end{aligned} \tag{4.43}
$$

The reduced cost for a container path o serving cargo flow n_o is given in equation (4.43). It assumes a routing on the legs $o = ((s_1, i_1, j_1, l_1, l'_1), \cdots (s_n, i_n, j_n, l_n, l'_n))$, transhipments at ports P^{TS}_o with $P^{TS}_o = \{((s_m, i_m, j_m, l_m, l'_m), (s_k, i_k, j_k, l_k, l'_k)) \in o : k = m + 1 \wedge s_m \neq s_k\}$. Container path o uses the services $S^I_o \subseteq S$.

$$
\tau^P_{o,s} = \sum_{(s,i,j,l,l')\in P^{TS}_o} \tau^E_{i,VT_s} + \tau^E_{o_{n_o},VT_s} + \tau^E_{d_{n_o},VT_s} \tag{4.44}
$$

Equation (4.43) uses the profit p_o of container path o by taking the revenue for transporting one unit of cargo flow n on the path, subtracting the container handling cost at the origin and destination, the transhipment costs and slot purchasing costs. The profit as follows:

$$
p_o(n) = r_n - c^{CH}_{o_n} - c^{CH}_{d_n} - \sum_{((s_1,i),(s_2,i))\in P^{TS}_o} \left(\phi^{TS}_i + \phi^{TS}_i\right) - \sum_{p\in P^{TS}_o} \phi^S(s,n,p) \tag{4.45}
$$

For each edge in the pricing network (see Figure 4.6), the unit cost per cargo flow n is defined as shown in equations (4.46) - (4.50) where the slot cost are given separately in equation (4.51) for simplicity.

$$
\begin{aligned}
E^I(p,(s,p,l)) &:= c^{CH}_p + \tau^E_{p,s} \cdot \pi^{PC}_s + \begin{cases} +\phi^{EC}_{p,e_n}, & \textit{if } p = o_n, \\ 0, & \textit{else} \end{cases} \\
&\quad + \phi^{SC}(s,n,p) && (4.46) \\
E^O((s,p,l),p) &:= c^{CH}_p + \tau^E_{p,s} \cdot \pi^{PC}_s + \begin{cases} -\phi^{EC}_{p,e_n}, & \textit{if } p = d_n, \\ 0, & \textit{else} \end{cases} && (4.47) \\
E^S(s,i,j,l,l') &:= \sum_{r\in rg} (u_{n,r} \cdot \delta_{s,i,j,l,l',rg}) +
\end{aligned}
$$

$$dws^S_{VT_s} \cdot u_{n,weight} \cdot (\gamma^I_{s,i,j,l,l'} + \gamma^O_{s,i,j,l,l'}) \tag{4.48}$$

$$E^T(s_1, s_2, i, l_1, l_2) := 2\phi^{TS}_i + \tau^E_{p,s} \cdot (\pi^{PC}_{s_1} + \pi^{PC}_{s_2}) + \phi^{SC}(s_2, n, o(l_2)) \tag{4.49}$$

$$E^{IS}(s, p, l, l') := 2\phi^{TS}_{p,s} \tag{4.50}$$

$$\phi^S(s, n, p) := \begin{cases} 0, & \textit{if } s \neq S^P \\ \sum_{rg \in RG, r \in rg \cap R_n} \phi^{SC}_{sg_p, r}, & \textit{else} \end{cases} \tag{4.51}$$

Equation (4.46) calculates the edge costs for the cargo flow origin. These costs occur for any unit of the transported cargo flow. It consists of the terms for the container handling costs, the reduced costs for the port call (indirectly for the additional speed), the additional cost for transporting empty containers to the port of origin and the slot costs if the service is operated by a partner. Respectively, (4.47) calculates the reduced costs at the port of destination for one unit of cargo flow n. Term (4.48) calculates the reduced costs for transporting one unit on leg i, j on service s. The term consists of the indirect costs associated with the vessel deadweight scale constraints, which are mainly costs by decreasing the transportation volume of other cargo flows. Term (4.49) calculates the transhipment cost between different services by pricing the transhipment costs, the costs for empty container repositioning and if necessary the costs for loading on a partner service. Finally, equation (4.50) calculates the costs associated with the transhipment within a liner service.

Algorithm 5 shows an overview of the delayed column generation method. The *RMP* is initialized and the pricing network *net* for the liner shipping network *NW* is instantiated. Afterwards, the initial container paths for all cargo flows are created and added to the *RMP*. Since no dual information is available yet, the containers are routed on their shortest (i.e. cheapest) path. Furthermore, information of non-routeable cargo flows (for example when a port is not served by the network) is stored. The *RMP* is solved using a commercial linear programming (LP) solver and the solution, as well as duals, is stored. Afterwards, the algorithm tries to find improving container paths for all routeable cargo flows and adds them to the model if they have positive reduced cost.

The method $CreatePath$ in line 9 of Algorithm 5 first updates the edge cost which depend on the current cargo flow according to equations (4.46) - (4.50). Afterwards a shortest path algorithm is executed with the source node $p^S_{o_n}$ and the target $p^T_{d_n}$. Note that the edge costs leaving the source node can be negative. This can be explained by the fact that transporting one unit of a specific cargo flow can reduce the number of empty containers required to balance the network. Thus, the duals can be interpreted as opportunity costs for reducing the amount of empty containers.

They include the overall effects of the empty repositioning on the restricted master problem's objective. Because source nodes have only outgoing edges and the target nodes only incoming, no negative cost cycles can occur. Thereby, shortest path algorithms that cannot handle negative cost cycles can also be used, such as the Bellman Ford or Dijkstra algorithm.

Algorithm 5: Overview of the delayed column generation for the integrated cargo allocation problem.

Input: A liner shipping network NW
Output: Profit of the network NW

$m \leftarrow RMP(NW)$;
$net \leftarrow CreatePricingNetwork()$;
$O \leftarrow InitialColumns(net, NW)$;
$m.Add(O)$;
while $O \neq \emptyset$ **do**
    $solution \leftarrow LPSolve(m)$;
    $O \leftarrow \emptyset$;
    forall the $n \in N$ **do**
        $o \leftarrow m.CreatePath(solution, net, NW)$;
        if $\bar{c}_o(n) > 0$ **then**
            $O \leftarrow O \cup \{o\}$;
        end
    end
end
return $(objective, solution)$;

In general, the column generation solution approach does not lead to integer solutions and a more advanced Branch & Price method must be used. In the next section, we mathematically prove that relaxing the bunker cost linearization integer variables still provides optimal integer solutions for the cargo allocation problem. With this result, the linear relaxation of the restricted master problem provides the optimal solution for the mixed integer program and efficient solution methods, such as the dual simplex, can be used.

4.5. Relaxing the Integrality Constraints for the Bunker Cost

Padberg's approximation model for separable non-linear functions used in the arc- and path-flow formulations relies on binary $z_{s,i}$ variables (see (Padberg, 2000, p. 2)). This results in a mixed binary program instead of a pure linear program. In this section we prove that a linear relaxation of the $z_{s,i} \in \{0,1\}$ variables to be $0 \leq z_{s,i} \leq 1$ is possible in the scope of the specific assumptions for the cargo allocation problem. For simplicity, only one service is used, leading to y_i and z_i variables. The case for several services can be performed in a similar way because the service discretizations are mutually independent.

The common function approximated in the arc- and path-flow formulation is the bunker cost for a given overall port call duration in the whole planning horizon (see equation 4.52). Details on the bunker cost are described in the previous sections.

$$\phi(x) = \phi^T \cdot bc \left(\frac{l}{24(f \cdot VC - \frac{x}{\frac{VC}{RT}})} \right) \cdot (\tau \cdot VC - x),\ x \geq 0 \tag{4.52}$$

The bunker consumption function bc is a function of speed to the power of a constant factor a. Stopford suggests $a = 2$ for steam engines and $a = 3$ for diesel engines (see (Stopford, 2009, p. 234)). It is shown in Chapter 2 that practical bunker profiles are often approximated using cubic polynomial functions.

To be able to relax the integer constraint on the z-variables, the following assumptions must hold:

Assumption 1 $\phi(x)$ is a bunker cost function such that an increase in the port call duration cannot lower the function value. This means that increasing the speed must always increase the total bunker cost, and the time saved at sea by higher speed cannot compensate higher fuel consumption.

Assumption 2 The x-axis discretization of $\phi(x)$ is uniformly distributed with k intervals with fixed width:
$a_i - a_{i-1} = a_{i+1} - a_i = c > 0,\ \forall\ 1 \leq i \leq k$

Assumption 3 For the discretization of $\phi(x)$ using model 1 from Padberg (2000) the following strict monotonicity must hold:

$$b_0 < \frac{b_1 - b_0}{c} < \cdots < \frac{b_k - b_{k-1}}{c}$$

where b_i is the function value at a_i, i.e. $\phi(a_i) = b_i$ with $b_i > b_{i-1}$.

Theorem In the optimal solution of a linear program (LP) relaxation that approximates $\phi(x)$ as cost function, an integer solution with less or equal cost for all relaxed z_i variables can be found in constant time.

Proof: Let $z^*, y^* \in \mathbb{R}^k$ be the optimal solution for the approximated $\phi(x)$ function. z^* is partitioned into:

$$\begin{aligned} Z^I &= \{z_i^* : 1 \leq i \leq k, z_i^* = 1\} \\ Z^F &= \{z_i^* : 1 \leq i \leq k, 0 < z_i^* < 1\} \\ Z^Z &= \{z_i^* : 1 \leq i \leq k, z_i^* = 0\} \end{aligned}$$

The relevant constraints in model 1 in (Padberg, 2000, p. 2) for the z and y variables are:

$$y_1 \leq c \tag{4.53}$$
$$c \cdot z_i \leq y_i \leq c \cdot z_{i-1} \;\; \forall 1 \leq i \leq k-1 \tag{4.54}$$
$$0 \leq y_k \leq c \cdot z_{k-1} \tag{4.55}$$

By constraints (4.54) and (4.55) it follows that $c \cdot z_i \leq c \cdot z_{i-1}$ and $z_i \leq z_{i-1}$. Elements in Z^F are connected as $Z^F = \{z_i^* : n \leq i \leq p\}$ for $1 \leq n, p \leq k$. By the partition it follows that $z_i^* = 1 \; \forall i < n$ and $z_i^* = 0 \; \forall i > p$.

We prove the theorem by constructing an alternative solution for the LP with integral z variables, show that it is feasible (*Step one*) and that the costs are not larger than those of the optimal solution (*Step two*). Define z', y' as follows:

$$z_i' = \begin{cases} 1, & i < n+a \\ 0, & else \end{cases} \tag{4.56}$$
$$y_i' = \begin{cases} c, & i < n+a \\ \sum_{i=n}^{p+1} y_i^* - c(a-1), & i = n+a \\ 0, & else \end{cases} \tag{4.57}$$

with

$$a := \left\lceil \sum_{i=n}^{p+1} \frac{y_i^*}{c} \right\rceil \tag{4.58}$$

In Figure 4.7 the variable values for y^*, y' are illustrated. Each interval's maximum value is c by assumption two. In the optimal solution, the z^* variables might be activated fractionally, leading to y^* below the upper bound c. The new constructed solution y' stacks y^* variables until the interval capacity c is reached. Thus, $n+a \leq p$.

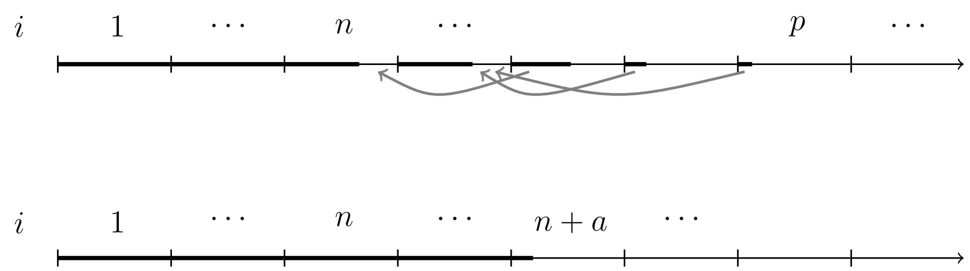


Figure 4.7.: Graphical illustration of the variable values in the optimal solution y^* (top) and the new constructed solution y' (bottom).

We show that for the y' variables the following holds:

$$y'_i \geq 0 \ \ \forall 1 \leq i \leq k \tag{4.59}$$

To show that $y'_{n+a} \geq 0$, it suffices to show that $y'_{n+a} \geq 0$, because $c \geq 0$ by assumption two:

$$\begin{aligned}
& y'_{n+a} \geq 0 \\
& \Leftrightarrow \sum_{i=n}^{p+1} y^*_i - c(a-1) \geq 0 \Leftrightarrow \sum_{i=n}^{p+1} y^*_i + c \geq ca \Leftrightarrow \sum_{i=n}^{p+1} \frac{y^*_i}{c} + 1 \geq a \\
& \Leftrightarrow \sum_{i=n}^{p+1} \frac{y^*_i}{c} + 1 \geq \left\lceil \sum_{i=n}^{p+1} \frac{y^*_i}{c} \right\rceil \Leftrightarrow \left\lceil \sum_{i=n}^{p+1} \frac{y^*_i}{c} \right\rceil - \sum_{i=n}^{p+1} \frac{y^*_i}{c} \leq 1
\end{aligned}$$

This is the definition of the ceiling function and true by definition.

Step one is to show that z', y' is a valid solution for the LP. We divide the constraint validation for z', y' into three cases, namely $1 \leq i < n+a$, $i > n+a$ and $i = n+a$.

For all $1 \leq i < n+a$,

$$c \cdot z'_i \leq y'_i \leq c \cdot z_{i-1} \Leftrightarrow c \leq c \leq c$$

and for all $i > n+a$,

$$c \cdot z'_i \leq y'_i \leq c \cdot z'_{i-1} \Leftrightarrow 0 \leq 0 \leq 0$$

is always true. Let $i = n+a$. The lower bound

$$c \cdot z'_{n+a} \leq y'_{n+a} \Leftrightarrow c \cdot 0 \leq y'_{n+a}$$

holds because all $y'_i \geq 0$ (see 4.59). For the upper bound on y'_{n+a},

$$y'_{n+a} \leq c \cdot z'_{n+a-1} \tag{4.60}$$

must hold because:

$$\begin{aligned}
& y'_{n+a} \leq c \cdot z'_{n+a-1} \overset{Def. z'}{\Leftrightarrow} y'_{n+a} \leq c \Leftrightarrow \sum_{i=n}^{p+1} y^*_i - c(a-1) \leq c \\
& \Leftrightarrow \sum_{i=n}^{p+1} y^*_i \leq ca \Leftrightarrow \sum_{i=n}^{p+1} \frac{y^*_i}{c} \leq a \Leftrightarrow \sum_{i=n}^{p+1} \frac{y^*_i}{c} \leq \left\lceil \sum_{i=n}^{p+1} \frac{y^*_i}{c} \right\rceil
\end{aligned} \tag{4.61}$$

which is always true for the ceiling function. Therefore, all cases for *step one* are proved and z', y' is a valid solution for the LP.

Step two is to show that $\phi(y') \leq \phi(y^*)$. We show that the increased cost of the changes to the y' variables for $n \leq i \leq n+a$ is less or equal to the cost reduction for $n+a+1 \leq i \leq p+1$ of the original solution y^*. The following inequality must be valid:

$$\begin{aligned}\sum_{i=n}^{n+a-1} \frac{b_i - b_{i-1}}{c}(y'_i - y^*_i) + \frac{b_{n+a} - b_{n+a-1}}{c}(y'_{n+a} - y^*_{n+a}) \\ \leq \sum_{i=n+a+1}^{p+1} \frac{b_i - b_{i-1}}{c}(y^*_i - y'_i)\end{aligned} \tag{4.62}$$

The left hand side sums the increased cost due to the increased y' variables. The right hand side sums the decreased y^* variable values. The additional cost must be less or equal to the cost reduction. We prove this by contradiction:

$$\begin{aligned}&\sum_{i=n}^{n+a-1} \frac{b_i - b_{i-1}}{c}(\underbrace{y'_i}_{=c} - y^*_i) + \frac{b_{n+a} - b_{n+a-1}}{c}(y'_{n+a} - y^*_{n+a}) \\ &\quad > \sum_{i=n+a+1}^{p+1} \frac{b_i - b_{i-1}}{c}(y^*_i - \underbrace{y'_i}_{=0}) \\ &\Leftrightarrow \sum_{i=n}^{n+a-1} \frac{b_i - b_{i-1}}{c}(c - y^*_i) + \frac{b_{n+a} - b_{n+a-1}}{c}(y'_{n+a} - y^*_{n+a}) > \sum_{i=n+a+1}^{p+1} \frac{b_i - b_{i-1}}{c} y^*_i \\ &\Leftrightarrow \sum_{i=n}^{n+a-1} (b_i - b_{i-1}) + \frac{b_{n+a} - b_{n+a-1}}{c} y'_{n+a} > \sum_{i=n}^{p+1} \frac{b_i - b_{i-1}}{c} y^*_i \\ &\Leftrightarrow^{1} \sum_{i=n}^{n+a-1} (b_i - b_{i-1}) + \frac{b_{n+a} - b_{n+a-1}}{c} y'_{n+a} > \sum_{i=n}^{p+1} (b_i - b_{i-1}) \\ &\Leftrightarrow^{2} (b_{n+a-1} - b_{n-1}) + \frac{b_{n+a} - b_{n+a-1}}{c} y'_{n+a} > (b_{p+1} - b_{n-1}) \\ &\Leftrightarrow^{3} \; b_{n+a-1} + (b_{n+a} - b_{n+a-1}) \frac{y'_{n+a}}{c} > b_{p+1} \\ &\Leftrightarrow b_{n+a-1} > b_{p+1}\end{aligned}$$

[1] The upper bound for y^*_i is $y^*_i \leq c \Leftrightarrow \frac{y^*_i}{c} \leq 1$
[2] Telescoping series, $+b_{n-1}$
[3] The lower bound for y'_{n+a} is 0

This is a contradiction, because $n + a \leq p + 1$ and $n + a - 1 < p + 1$ and by assumption 3, $b_{n+a-1} < b_{p+1}$ holds.

Thus, we have proved that the inequality (4.62) holds and therefore the constructed integer solution is not worse than the LP optimal solution. The constructed integer solution cannot be better than the optimal LP solution (LP relaxation) because the mixed integer solution is at least as good as the LP solution.

To construct a valid integer solution from the fractional z values in the optimal solution, the definitions for z' and y' can be used.

4.6. Numerical Results for the Integrated Cargo Allocation Problem

In this section, numerical results for the models and solution methods developed in this chapter are presented. The models are implemented in *C#* with Microsoft's *.NET* framework 4.5. Gurobi 5.6.0 with the dual simplex method is used to solve the arc flow cargo allocation formulation and the restricted master problem in the path flow formulation. The tests are performed on a machine with Intel Core2 Quad CPUs with 2.83 GHz, 8 GB RAM and Windows 7 64 Bit. The implementation uses the .NET library Quickgraph 3.0[4] to solve the shortest path subproblems. Quickgraph is inspired by the Boost Graph Library[5] and provides a wide range of different graph algorithms such as Dijkstra's algorithm (Dijkstra (1959)) that is used to solve the pricing problems of the cargo allocation problem. The library uses an accelerated version of the basic Dijkstra algorithm using Fibonacci heaps (see (Cormen et al., 2001, p. 476ff)).

This section is organized as follows: First, the problem instances and the reference network generation are introduced. Afterwards, numerical results for the arc-flow and path-flow formulation are presented and discussed. For the arc-flow formulation, we present several key performance indicators that can be used to evaluate the networks and their practical usefulness. Note that the path-flow formulation uses the same data and would lead to identical results. Finally, the effects of solving the path-flow formulation heuristically are shown.

4.6.1. Problem Instances

The problem instances used throughout this thesis are described in more detail. The cargo allocation problem requires problem instance data, such as ports and cargo flows, but also existing liner networks. These liner networks are created for all instances of the LINER-LIB 2012 benchmark suite that provides the data (see `http://www.linerlib.org/` and Brouer et al. (2013)).

[4]The source code and Nuget package is available at `http://quickgraph.codeplex.com/`.
[5]See `http://www.boost.org/libs/graph/`

LINER-LIB Benchmark Instances

Brouer et al. (2013) present a set of seven benchmark instances for the liner shipping network design problem. The benchmark suite contains a set of ports with geographical position (longitude and latitude), a set or legs with distances, vessel types for each instance as well as available vessel types and demands (cargo flows) for each instance. The instance's ports are limited to ports where cargo origins or destinates. This is a simplification because ports without cargo can still act as transhipment hubs. However, in practice it is unlikely that a port is called without leading to any profit. The cargo flows in the LINER-LIB have a deterministic weekly quantity, a revenue for transporting one unit and a maximum transit time. As described in Chapter 2, in the scope of this thesis the transit time is seen on a port-to-port basis (see Notteboom (2006)). Of the set of transit times, only those are used that can be served with the vessel types' maximum speed on a direct trip.

In Table 4.7 an overview of the different instance sizes is given. Instances *Baltic* and *WAF* (West Africa) are feeder networks, meaning all cargo either origins or destinates at a single port (transhipment hub). Thus, no regional cargo flows exist between the ports except the transhipment hub. This hub is based on the transhipment port used by Maersk Line. The number of ports, demands and legs in these feeder networks are relatively small. The second group of networks *Mediterranean* and *Pacific* have a similar amount of ports and legs. However, they differ clearly by the number of cargo flows and transit times because they cover different geographical regions.

The last group of networks are *WorldSmall*, *EuropeAsia* and *WorldLarge* with many ports, cargo flows, legs and vessel types. Thus, they are expected to be difficult to solve. In the scope of this thesis, parameters are evaluated for different groups of networks, with a focus on the small and medium sized networks.

Instance	**Ports**	**Cargo Flows**	**Transit Times**	**Legs**	**Vessel Types**	**Equipment Types**
Baltic	12	22	22	132	2	1
WAF	20	37	33	380	2	1
Mediterranean	39	365	365	1,482	3	1
Pacific	45	722	719	1,980	4	1
WorldSmall	47	1,764	1,748	2,162	6	1
EuropeAsia	114	4,000	3,996	12,882	6	1
WorldLarge	201	9,622	9,599	40,200	6	1

Table 4.7.: LINER-LIB 2012 instance information (see Brouer et al. (2013)).

An important detail for the numerical results shown in the next sections is the demand distribution. Figure 4.8 shows the percentage of cargo flows for a percentage of different measures in the WAF instance. This is either the cumulative sum of quantity times revenue ($cum.Q * R$, i.e. the maximum total revenue per week), the cumulative quantity ($cum.Q$) and the revenue per cargo flow ($cum.R$). Figure 4.8 illustrates that the revenue is distributed nearly uniformly, i.e. most of the cargo flows have similar revenues. On the other hand, the quantity and possible total revenue is not uniformly distributed. For example, 20 percent of all cargo flows leads to approximately 60% of the possible total revenue per week. This information is used to evaluate the influence of limited cargo flows on the networks profit.

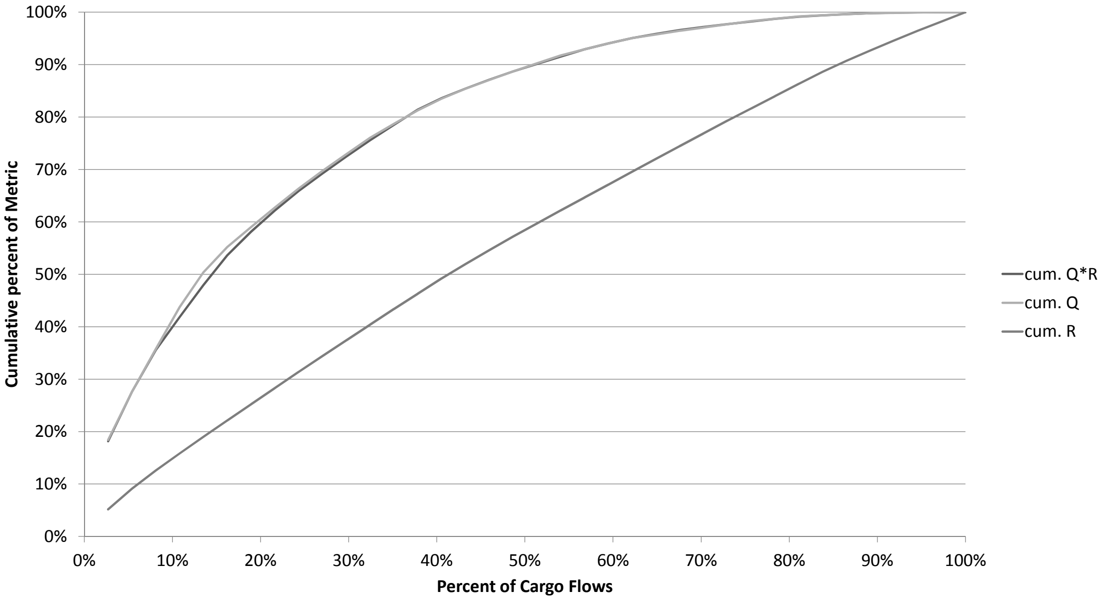


Figure 4.8.: Distribution of the cumulative cargo flow quantity ($cum.Q$), revenue ($cum.R$) and quantity * revenue ($cum.Q * R$) for the LINER-LIB WAF instance.

Note that all instances show a similar result for the quantity and maximum weekly revenue distribution, and this is plotted in the figures in Appendix C.1. Especially the Mediterranean (see Figure C.2) instance indicates a tendency to have a non-uniform revenue distribution, probably due to several cargo flows that actually originate in other regions and were simplified in the scope of the benchmark suite. The Pacific, WorldSmall, EuropeAsia and WorldLarge instances (see Figures C.3ff) indicate that the cargo flow quantity is distributed on fewer cargo flows. This observation is used in the following chapters to create new networks.

For further details on the sets and parameters provided by the LINER-LIB, see Brouer et al. (2013).

Reference Networks for the Cargo Allocation Problem

The cargo allocation problem determines the optimal profit for given networks. Therefore, networks for the LINER-LIB instances presented above must be constructed. For each problem instance, random networks are generated with a construction heuristic described in Section 5.2.3. To respect the stochastic fluctuations of the runtime, 20 networks are created per instance.

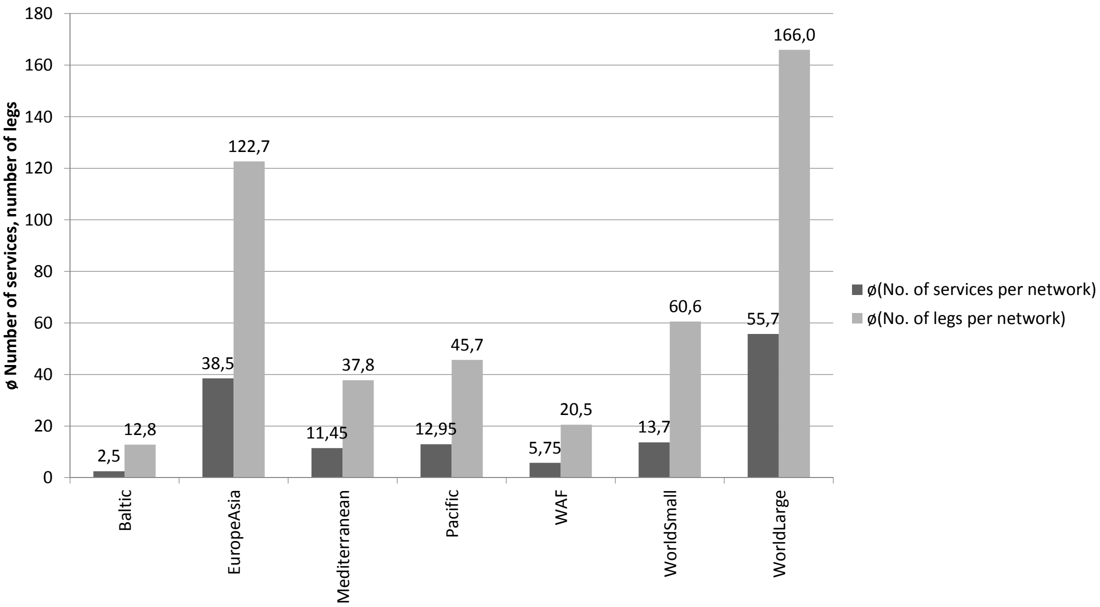


Figure 4.9.: Average sum of services and legs for random networks per LINER-LIB instance.

The method creates relatively small networks which might not be representative of evaluate larger networks. Thus, the heuristic is extended to fill up the randomly created networks with pendulum services at randomly served ports to extend transhipment possibilities. The number of services for a random network s of LINER-LIB instance i is picked uniformly from the interval $[|Ports_i|/7, |Ports_i|/2]$. Figure 4.9 shows the average sum of the services and legs for all random networks per LINER-LIB instance. On average, the number of services are similar to those presented in the solutions in (Brouer et al., 2013, p. 26 and 27).

4.6.2. Arc-Flow Formulation

First, the results for the arc flow formulation of the cargo allocation problem are presented. The model is solved with the commercial LP solver Gurobi 5.6.0 with the default parameters with the exception that the dual simplex LP solution method is used. The dual simplex has been evaluated to be faster compared to interior point methods. Recall that the objective is to allocate cargo flows on one or more services to get one or more paths from the origin to destination. This results in durations at each service's port which determines the required speed to perform a round trip.

In Figure 4.10 the results for all seven instances of the LINER-LIB and 20 randomly generated networks per instance are presented. On the horizontal axis the instance name and the number of services per network is given to have a simplified measure of the network size. On the vertical axis, the logarithmic (base 10) average runtime in seconds is shown. One can observe a tendency that larger networks for a specific instance lead to higher runtimes. However, the complexity of a network is also determined by other factors such as the transhipment possibilities, the number of legs and the tightness of the service capacities.

The results in Figure 4.10 show a runtime of less than 0.1 seconds for the small Baltic and WAF instance. Gurobi needs between one to six seconds to solve the medium sized Mediterranean and Pacific instances to optimality. The WorldSmall instance is more difficult to solve, leading to runtimes of up to six minutes for the largest instance with 22 services and 30 seconds on average. Note that there are two times as many cargo flows in this instance compared to the Pacific instance. The large EuropeAsia and WorldLarge instances have runtimes of up to 23 and 50 minutes for the largest networks and about two minutes on average.

To get an overview about the utilization of the services, Figure 4.11 shows the best average utilization per service and the percent of served cargo flows in the optimal solution. One can observe that most of the networks have highly utilized services, probably those that follow the large trades (because the construction heuristic focuses on these cargo flows). Furthermore, one can observe that only a fraction of the cargo flow is actually served, either because the networks do not provide a connection between the origin and destinations or because the available container paths are not cost efficient enough to transport the cargo flow and gaining the revenue. Optimizing networks can lead to an increase of the served cargo flows and the utilization, especially on the long legs that connect different geographical regions.

With the help of Figure 4.12, the port durations and the average service speed is evaluated. On the vertical axis of Figure 4.12, the average port duration in hours and the average speed in knots is given. A constant total pilotage duration of 3 hours and no constant buffer per port call is assumed. Additionally, we have used a relatively high container move rate of 50 containers/hours. Two important observations can be made from Figure 4.12: First, the average port call duration varies strongly between different networks (and within ports of a service). Especially large transhipment hubs are called more than 30 hours. One should note that the cargo allocation model formulation can compensate networks where services have too many vessels deployed. In this case, the model adds a buffer to the services' ports. This can

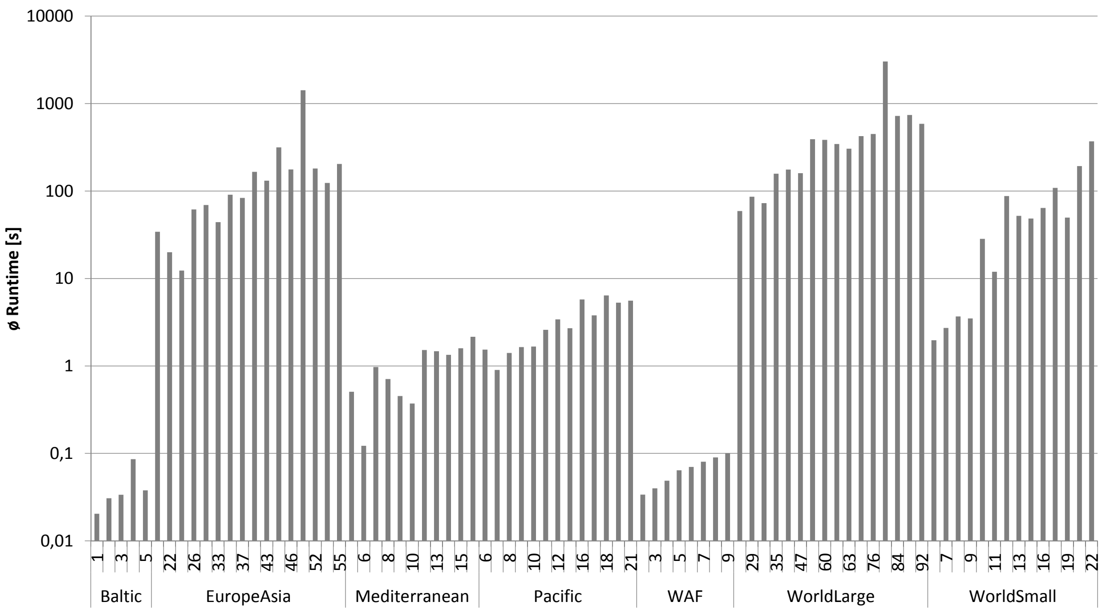


Figure 4.10.: Runtime results for solving the cargo allocation to optimality using Gurobi 5.6 with the arc-flow formulation.

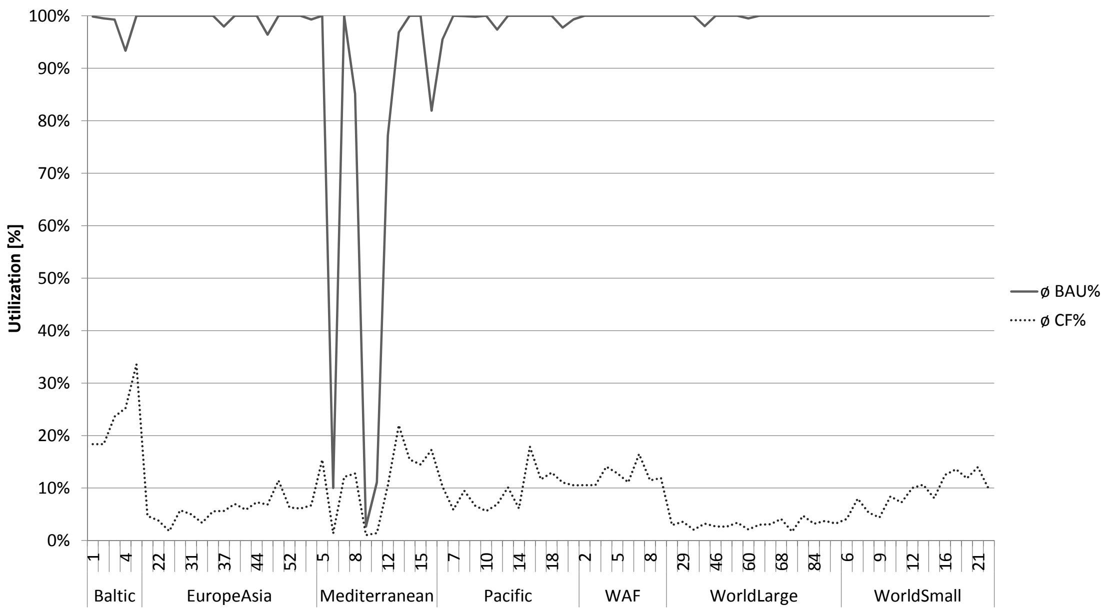


Figure 4.11.: Best average utilization per network ($BAU\%$) and average served cargo flow percentage ($CF\%$).

Figure 4.12.: Average speed in knots and duration per port call in hours.

increase the port call duration in some instances. Second, due to the large bunker cost, the optimal solution tries to steam as slow as possible in the offered services. All instances provide vessel types with a minimum speed of 10 knots, leading to slow steaming strategies for the given networks. In the large instances, the average speed can be higher because some vessel types (e.g. Panamax vessels) have a minimum speed of 12 knots.

The results indicate the importance of including detailed load dependent port durations and speed optimization in the cargo allocation.

4.6.3. Path-Flow Formulation

The results for the arc-flow formulation already indicate promising runtimes to solve large scale instances of the cargo allocation problem. The model formulation can thereby be used to support the manual network planning process by allowing the evaluation on a network to optimality.

One goal of this thesis is to solve the more complex liner shipping network design problem (see Section 3.4) that integrates the cargo allocation problem. Thereby, a more sophisticated delayed column generation (CG) method has been proposed that models the problem as a path-flow formulation. For each cargo flow, a container path is generated. The results for the CG are presented in the succeeding sections and use the same networks as for the arc-flow formulation results. The restricted master

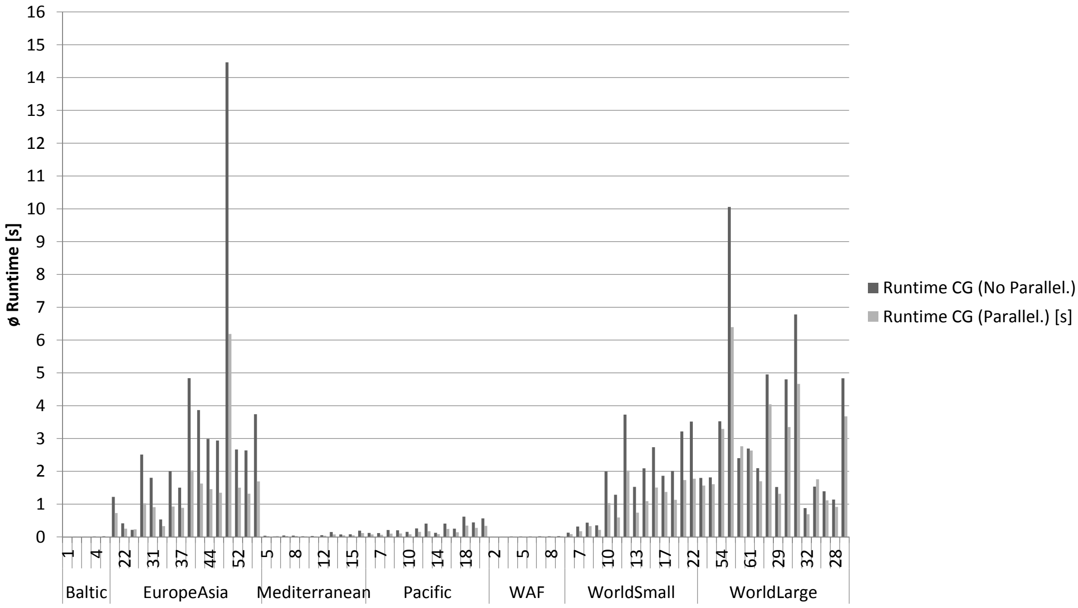


Figure 4.13.: Average runtime in seconds of the parallel and non parallel implementation for networks of different sizes for the LINER-LIB instances.

problem is solved with Gurobi 5.6.0 and the pricing problem with the Quickgraph implementation of the Dijkstra algorithm.

Figure 4.13 shows the runtimes for the CG method on different LINER-LIB instances and different networks. On the horizontal axis the instance and the number of services are shown. On the vertical axis the average runtime in seconds of all networks of the specific instance and network size is given. All instances are solved to optimality. Solving multiple shortest-path pricing subproblems has been parallelized to further accelerate the solution process. The parallelization uses different physical processors or cores on modern computers to solve the subproblem for different cargo flows simultaneously. The computer used to solve the cargo allocation problem has four cores that leads to four parallel shortest path pricing problems. Using the standard threading classes of the .NET framework, the solution time could be decreased, especially on the medium and large instances (see Figure 4.13). On average, the parallel implementation is about 60 % faster compared to the non-parallel version.

The cargo allocation problem for the small instances' random networks could be solved in a fraction of a second to optimality. The instance WorldSmall, EuropeAsia and WorldLarge took several seconds to be solved. Figure 4.13 highlights that there does not exist a direct correlation between the network size and the CG runtime. This can be explained by the randomness of the networks: If the networks are highly interconnected and a lot of transhipment possibilities exist, it is expected that more paths between the cargo flow's origin and destination will be created in the column generation process. However, a slight tendency that larger networks lead to larger runtimes can be observed.

To get a better insight into the column generation method, Table 4.8 shows different metrics for container paths resulting from the solution. Recall that a container path is the routing of a cargo flow from its origin to its destination using legs from one or more services with transhipment operations performed at the service transitions. If the shortest (cheapest) path is capacitated, the CG method would create an equally or more expensive alternative container path. The restricted master problem decides if paths should be used to route the cargo flow. For each instance, 20 different networks are solved to optimality and the minimum, average and maximum number of activated container paths in the optimal solution per demand is stored. Table 4.8 highlights the minimum, average and maximum number of used container paths for all networks N_i of instance i. Furthermore, the average standard deviation σ_n^C is given in the last column.

	Number of container paths			
LINER-LIB Instance	min	average	max	standard deviation
Baltic	1	1.017	2	0.048
EuropeAsia	1	1.010	3	0.078
Mediterranean	1	1.002	2	0.012
Pacific	1	1.014	3	0.096
WAF	1	1.008	2	0.027
WorldLarge	1	1.007	3	0.057
WorldSmall	1	1.019	3	0.130

Table 4.8.: Minimum, average and maximum number of container paths as well as the standard deviation in the optimal solution for different problem instances and networks.

In Table 4.8, two important observations can be made: First, all instances indicate an average number of container paths per cargo flow of nearly one. Supported by the low standard deviation, this means that most of the cargo is served on one single path from the origin to the destination. This information is used for the cargo allocation heuristics presented in Section 4.6.5.

4.6.4. Comparison of Numerical Results

The comparison of the arc and path-flow formulation of the integrated cargo allocation problem is given in Figure 4.14. The figure shows the average runtime in seconds per instance (logarithmically scaled) for both formulations as well as the ratio between the runtimes. Values larger than one indicate a benefit of the column generation solution method, for example a ratio of 100 indicates that the column generation's runtime to solve the cargo allocation is 100 times lower (better) compared to solve the arc flow formulation with Gurobi 5.6.0's dual simplex method.

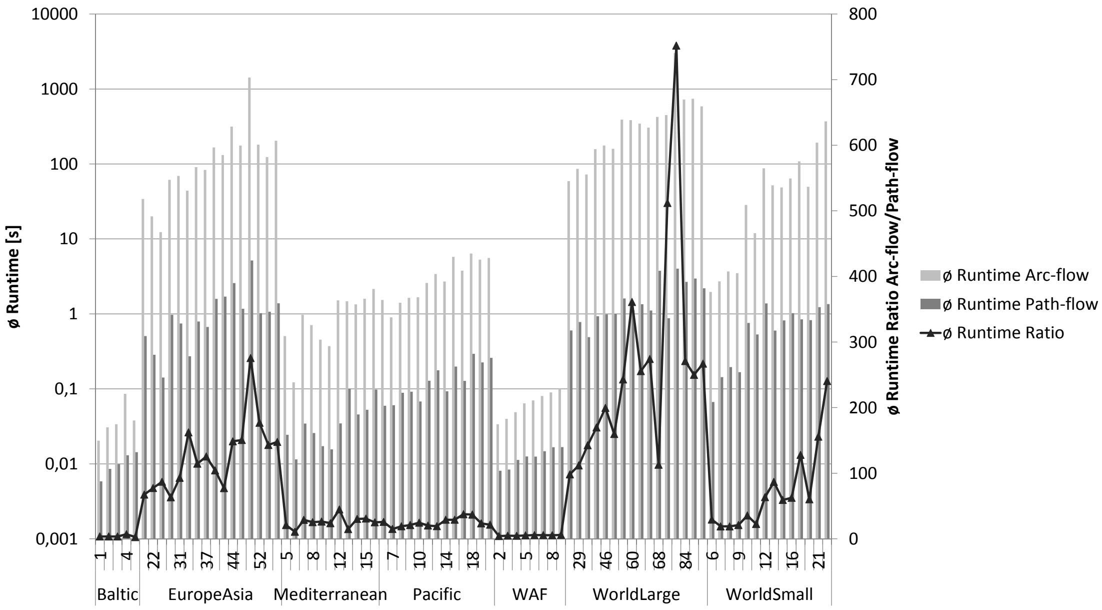


Figure 4.14.: Comparison of the arc-flow formulation runtimes and the path-flow formulation runtimes.

Figure 4.14 shows the superiority of the column generation method, especially on the medium and large instances. The ratios vary between a factor of 20 and 240 for the medium instances. Similar results can be seen for the large EuropeAsia and WorldLarge instances. Here the minimum speedup is about 70 for the smallest networks and up to several hundred times faster runtimes compared to the arc-flow formulation in the largest networks.

The results clearly show that column generation is a suitable method to solve large scale instances of the integrated cargo allocation problem in less than 10 seconds. This allows planners to evaluate the effects of local service changes on a global basis. It must be noted that the results are based on random networks that are created by construction heuristics and should be improved during an optimization. Therefore, it is expected that the initial networks are unprofitable. One network property that can influence the column generation approach is the deployed service capacity. In case of too many vessels, cargo can be allocated on its cheapest path without rejecting the path and creating alternative paths during the solution method. In case of very tight capacities, it is expected that several paths should be used for the cargo flow, leading to increased runtimes. This dependency has to be observed during the automatic optimization runs in Chapter 5.

4.6.5. Choosing an appropriate Approximation Level

In the previous sections, the column generation method has been solved to optimality within several seconds. To further speed up the methods to solve the cargo allocation problem hundreds and thousands of times within metaheuristics, further ways to decrease the runtime must be explored.

The drawback is that this usually comes along with decreased accuracy of the resulting solution. Throughout the next sections, the following definition of an optimality gap is used (whereas z^* is the optimal solution of the cargo allocation problem and z_i is any, not necessarily optimal, solution):

$$gap(z_i) = \frac{|z^* - z_i|}{z^*}$$

The gap can be larger than one (100%) if both objective values are negative. To calculate the optimal solution z^* to compare the heuristics below, the column generation is solved to optimality each time. Note that due to operating system activities, relatively small deviations in the runtime ratios of two optimal solutions can occur.

There are mainly three different approaches to solve the cargo allocation with the CG heuristically:

1. Limiting the amount of cargo flows
2. Varying the bunker cost discretization accurateness
3. Limiting the number of CG iterations

The effects of these heuristic approaches on the gap and the runtime improvement are analyzed in the following sections. For simplicity, the results are shown for the instances WAF, Mediterranean, Pacific and EuropeAsia. Results for the remaining instances can be found in Appendix C.2.

Limiting the Amount of Cargo Flows

Liner shipping networks are designed to follow trade patterns (see Notteboom and Rodrigue (2008)), in particular the underlying cargo flows. According to discussions with a global liner carrier the networks are furthermore aligned to high volume trades. Thus it is expected that transporting a fraction of the available cargo flows reduces the runtime and the profit only slightly.

In the scope of this thesis, four different strategies to select the most *promising* cargo flows are analyzed:

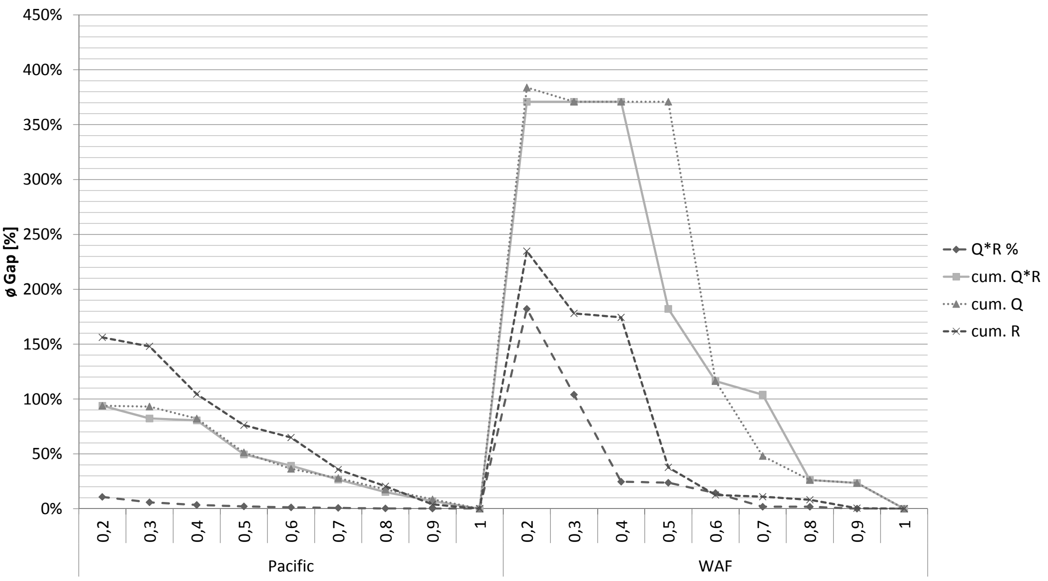


Figure 4.15.: Gap in percent for varied cargo flow percentages and different cargo flow selection strategies (for the Pacific and WAF instance).

1. Use p percent of the cargo flows, ordered descending by quantity * revenue ($Q * R\%$).
2. Use the cargo flows that provide p percent of the maximum total revenue ($cum.Q * R$).
3. Use the cargo flows that provide p percent of the total quantity ($cum.Q$).
4. Use the cargo flows that provide p percent of the revenue ($cum.R$).

The first strategy selects the largest number of cargo flows of all strategies, because it is independent of the cumulative quantity or revenue. The second strategy select much less cargo flows because the quantity * revenue is not uniformly distributed in any of the analyzed instances (see Appendix C.1). The same applies to the third strategy that also uses the cumulative method, but with the quantity instead. The last strategy is a mix of the first and the succeeding two because the revenue is distributed nearly uniformly in half of the instances (see Appendix C.1). The abbreviations of the strategies are used in Figures 4.15 - 4.17.

In Figures 4.15 and 4.16 the resulting average gap for the instances Pacific, WAF and EuropeAsia and Mediterranean as well as different cargo flow percentage p and selection strategies is shown. The results are averaged over 20 networks per instance. For most of the strategies, the instances shown in Figure 4.15 are highly sensitive

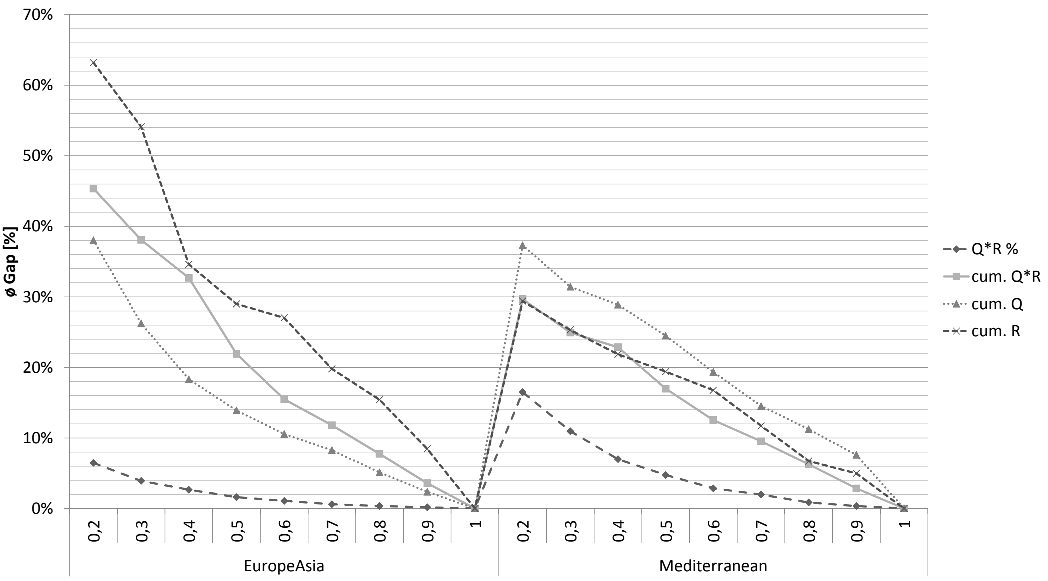


Figure 4.16.: Gap in percent for varied cargo flow percentages and different cargo flow selection strategies (for the EuropeAsia and Mediterranean instance).

to decreased cargo flow amount. Especially the WAF networks give very high gaps of 100% and more when the cargo is limited to 70% or less. The Pacific networks are slightly more robust, but also result in high gaps with 50% of the cargo flows. In the WAF networks, a few cargo flows lead to a relatively small percentage of the total revenue, compared to other instances (see Figure 4.8). Limiting the amount as shown in Figure 4.15 leads to few cargo flows and thereby to low profit and large gaps. One can observe that the $Q*R\%$ strategy works superior to most of the other strategies for most of the networks because the number of cargo flows is relatively high, even with a high approximation.

The results for the EuropeAsia and Mediterranean instance, presented in Figure 4.16, show a slightly different picture. First, the maximum gap is less than half of the previous instances, although only 20% of the cargo flows are available. Furthermore, the cumulative strategies have a gap of less than 10% with a cargo flow amount of 70% or more. Although they work better in these instances, the $Q*R\%$ strategy performs again much better with gaps less than 8% with a cargo flow amount of 40% or more.

To sum up, the $Q*R\%$ strategy works best with all instances up to an approximation of 60% for the small and medium sized instances and 30% for the large instance. The *cum.Q* strategy works well for approximations up to 70% for the large scale instance as well, the $cum.Q*R$ up to 70% for the medium sized and the *cum.R*

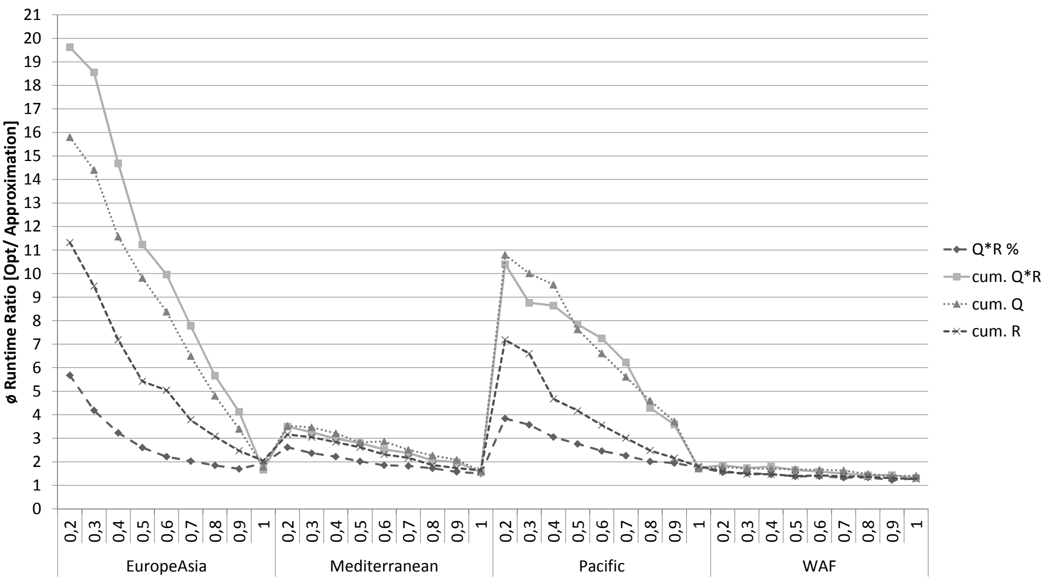


Figure 4.17.: Runtime improvement when solving Column Generation heuristically by limiting the cargo flow amount according to different strategies.

strategy for the small instance up to an approximation of 60%. The disadvantage of the gap goes along with a speedup of the runtime.

In Figure 4.17 the average ratio of the optimal to the approximation runtime for all instances are shown. Limiting the cargo flows of the large EuropeAsia instance to 30%, the $Q * R\%$ strategy improves the runtime by a factor of four, whereas the $cum.Q$ strategy improved the runtime by factor 6.5 for approximations of 70%. The $Q * R\%$ strategy for approximations up to 60% for the medium and small instances lead to a runtime improvement by a maximum factor of 2.5 for the Pacific instance. Due to the very small solution times of less than a tenth of a second (see Figure 4.13), the WAF networks average speedup is very small. The $cum.Q * R$ strategy for approximations not less than 70% for the medium sized instances reduced the runtime by a factor of 2.5 (Mediterranean) to 6 (Pacific). The $cum.Q * R$ strategy for the WAF instance did not lead to any clear speed up.

To conclude, routing not less than 70% of the cargo flows using the $cum.Q$ selection strategy can improve the runtime by a factor of 6 and more in the large instance. An alternative with a much lower gap and a runtime improvement by factor 4 is to use no less than 30% of the cargo flows with the $Q * R\%$ strategy. Routing no less than 70% of the cargo flows using the $cum.Q * R$ selection strategy for the small and medium instances can improve the runtime by a factor of 2 and more.

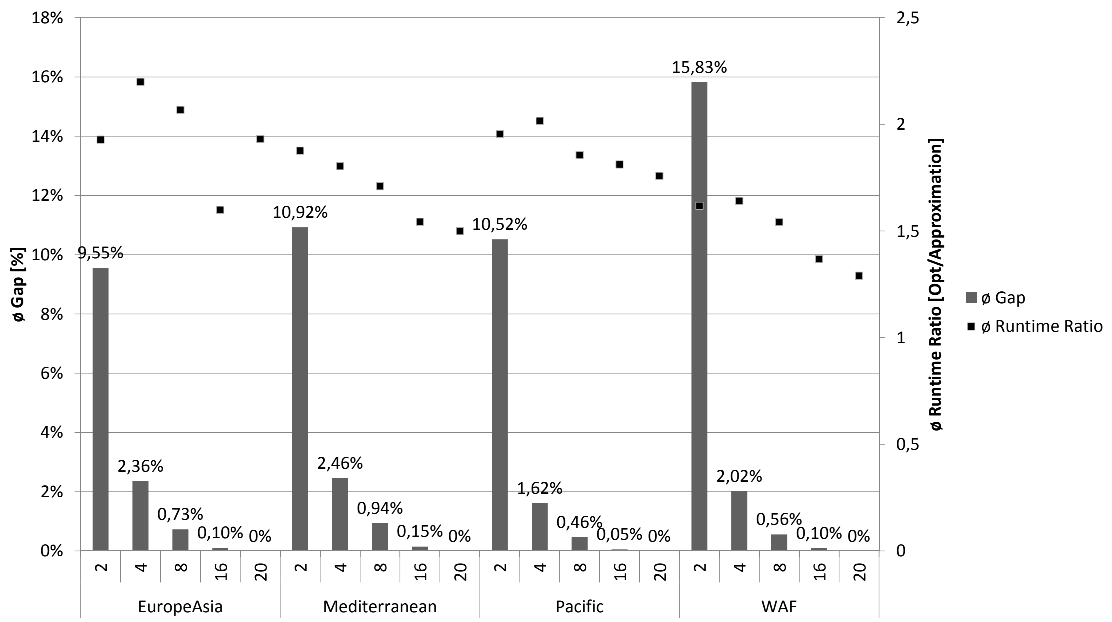


Figure 4.18.: Average effects of the different bunker cost linearization support points on the objective function and the runtime.

Varying Bunker Cost Discretization's Supporting Points

Another possibility to heursitically solve the cargo allocation problem is to decrease the accuracy of the bunker cost function. Originally, this function is a polynomial of roughly degree three, but we linearize it using discrete intervals (see Chapter 4). In theory, infinity many discrete intervals must be used to get the actual result of the function. This approach is not applicable when solving linear programs where a finite set of decision variables must be used. Figure 4.18 shows the dependency of the cargo allocation problem on the number of support points (discretization intervals).

On the horizontal axis the LINER-LIB instance and the number of support points (maximum of 20 points) is given. On the left vertical axis of Figure 4.18 the average gap in percent of all networks relative to the formulation that uses 20 support points is given with the bars. On the right vertical axis the runtime ratio between the optimal path flow and the approximation is shown as points in the diagram. The results indicate that all instances show a gap of up to 15.83% when approximating the total bunker consumption using a linear function with two support points (a single line with constant slope). This can be explained by the fact that a single line always overestimates the real bunker costs. If the number of support points increases by a factor of two, the gap is reduced in average by a factor of five. This

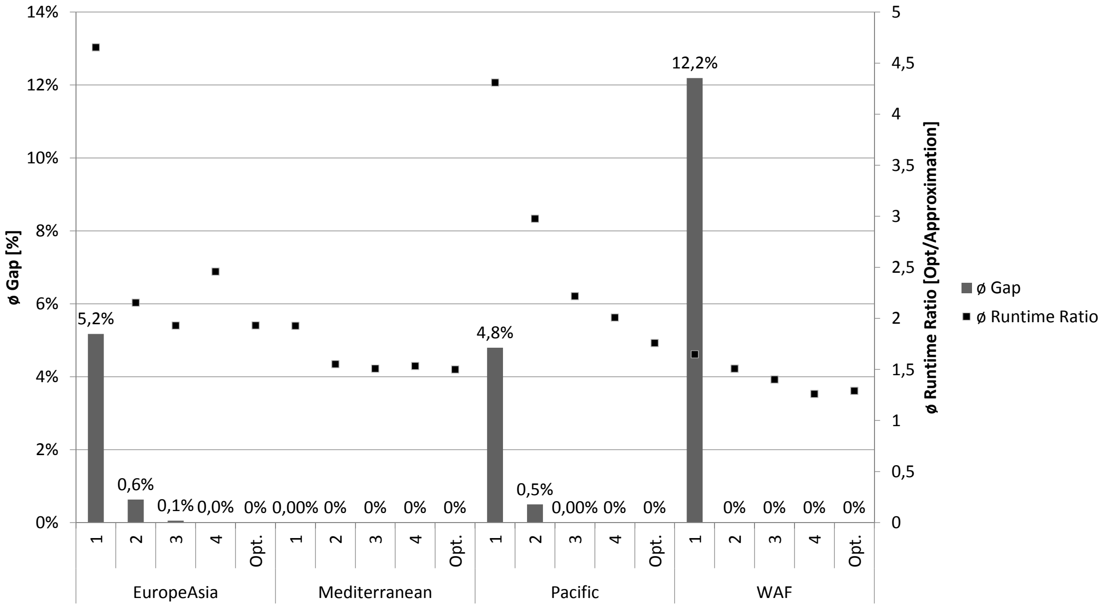


Figure 4.19.: Gap and runtime improvement when terminating the column generation solution approach prematurely.

leads to a very small gap when using close to 20 support points. In the scope of this strategic planning problem 20 support points is a reasonable approximation. When extensions of this problem are used for operational allocation problems, the bunker cost can be linearized more accurately. Due to tighter formulations in the restricted master problem, the results in Figure 4.18 indicate a slight runtime ratio decrease when a linear discretization is used.

To conclude the cargo allocation approximation with bunker cost, using between 4 to 20 support points leads to reasonable small gaps for all instances and a runtime improvement factor if 0.5 to 1. Compared to reducing the cargo flows, the runtime improvement when decreasing the bunker cost accurateness has a much lower benefit.

Limiting the number of Column Generation Iterations

Another heuristic approach for the column generation method is to terminate the solution process prematurely. For example, aborting the process after the first iteration leads to shortest (cheapest) container paths in the restricted master problem. This method should give the largest runtime improvement, but can lead to the largest gaps especially if the networks are highly capacitated.

Figure 4.19 shows the maximum number of iterations performed in the column generation solution approach per instance. The last column *Opt.* on the horizontal axis indicates that the cargo allocation has been solved to optimality, if necessary with more than 4 maximum iterations. On the left vertical axis the average gap to the optimal solution (for 20 networks per instance), on the right vertical axis the runtime ratio is shown as points in the diagram. The figure illustrates that all instances except the Mediterranean, which is solved after the first iteration to optimality, have an average gap of 4.8 to 12.2% after the first iteration. The gap decreases by a factor of approximately 10 (for some instances even faster) in each additional column generation iteration. The runtime ratio improvement increases with less iterations. Compared to the optimal solution, the speed up factor is about four to five in the medium and large instances (that are not solved to optimality after the first iteration). The ratio is expected to decrease with more complex and capacitated networks because more container paths would be created. All instances except the EuropeAsia network are solved to optimality after the second iteration (the third iteration must be used to verify that the solution is optimal).

Summing up, the results indicate a large runtime benefit of factor two to five when limiting the amount of CG iterations to one and a relatively small gap of 0 to 12%.

4.7. Comparison and Interpretation of Results

With the help of the path flow formulation and a column generation approach, the optimal cargo allocation on large scale liner networks can be determined within a few seconds. This is a speed up of factor 50 to 300 compared to the arc flow formulation solved with Gurobi 5.6.0 using the dual simplex method. The state-of-the-art is extended by integrating empty container repositioning, speed optimization, load dependent vessel drafts and capacity types into the cargo allocation.

The exact results can be approximated by limiting the amount of cargo flows, the bunker cost discretization accuracy and the maximum number of iterations in the column generation solution process. Limiting the cargo flow amount and the CG iterations seem to be the most promising approximation. This leads to low gaps and speedups of four to eight (while keeping relatively low gaps) and further 2 to 4.5 by the maximum number of CG iterations. Using eight support points for the bunker cost linearization still provides runtime speed ups of factor 1.5 to 2.

With the help of the developed optimization methods, liner shipping networks can be quickly evaluated and the manual planning process of liner network planners is supported. The next chapter presents optimization methods to automatically improve liner networks.

5. Improving Networks - The Liner Shipping Network Design Problem

The second and third goals of this thesis are to develop optimization methods for liner shipping networks. The assumption of fixed services as in the cargo allocation problem is relaxed in this problem. The decision is to choose one or more liner services, each consisting of a port rotation (port sequence), a vessel type and a number of deployed vessels. The liner services must be chosen in such a way that the profit, according to the cargo allocation, is maximized (see Chapter 4). The problem formulation in this thesis extends the state-of-the-art by additionally incorporating transit times and embargo constraints in the liner shipping network design problem.

For simplicity, weekly service frequencies are assumed throughout this chapter. Furthermore, cabotage regions and other vessel specific constraints that would require specific vessels are not considered. These assumptions must be relaxed in successive planning problems such as the fleet deployment.

A mixed integer formulation for the network design problem is presented. With this model, optimal networks can be created for very small instances and the gap to the optimal solution be calculated. Afterwards, two metaheuristics for the network design problem are introduced. Therefore, first a decomposition concept that is used for the construction and improvement heuristics is presented. After providing numerical results for the metaheuristics, the impact of approximated fitness functions (surrogates) on the metaheuristic convergence is analyzed. A sensitivity analysis on uncertain bunker cost gives an outlook whether uncertainty should be included in future solution approaches.

Finally, the developed metaheuristics are evaluated on a real-world liner shipping network.

5.1. Mixed Integer Formulation

The liner shipping network design problem with transit times, partner services, deadweight scales and speed optimization is presented as a mixed integer program (MIP). Non-linear constraints are linearized using the L01 approach from Padberg (2000) (used for the cargo allocation as well). The main difficulty when formulating the problem as a MIP are the speed constraints and variable number of vessels. The speed of the vessels deployed on a service is given by dividing the service distance by the duration at sea. Both terms are decision variables which means the fraction must be linearized. Therefore, several sets must be introduced that allow the linear formulation of the integrated speed optimization. After presenting the mathematical model, numerical results for the small LINER-LIB instances are presented.

5.1.1. Mathematical Model

The liner shipping network design problem can either be based on a set of services or vessels that are deployed on services (see e.g. Reinhardt and Pisinger (2010)). In the scope of this thesis, the model is based on services because the number of services is usually much lower compared to the vessels owned or operated by a liner carrier. The model uses a maximum number of services in the given problem instance, referred to as service candidates. An upper bound for the maximum number of vessels per service, a discrete set of possible speeds and a number of port calls (see Chapter 4) must be provided to linearize the non-linear bunker consumption constraints.

The model presented below incorporates the cargo allocation problem (CAP), because it determines the service speed and the cargo allocation that defines the bunker and container handling cost. Thus, several aspects are similar to the CAP. In the following tables, the additional sets, parameters and variables required for the mixed integer model are presented in addition to the sets and parameters from Section 4.2.

Sets

K	Set of discrete speeds in knots
S	Set of all services, $S = S^O \cup S^P$
S^O	Services (candidates) that can be operated by the carrier
S^P	Set of all partner services
$\mathbb{L}$	Set of network layers
L_s^L	Legs used by partner service $s \in S^P$
$L^L = \mathbb{L} \times L$	set of all layered legs that can be used to create new services
VC	Set of deployable vessels per operated service
T	Set of transit-time requirements, $T \subseteq P \times P$
$P^L = \mathbb{L} \times P$	Set of layered ports
$\bar{P}$	Set of port sets that cannot be called together on one service due to embargo constraints

Parameters

f	Fixed frequency in days of all operated services, $f = 7$
γ	Fixed percentage of all partner cargo that must be transported
θ_t	Maximum transit-time duration in days for transit-time requirement $t = (i, j) \in T$
$bc_{v,k}$	Bunker consumption in metric tons per day for a vessel of type $v \in VT$ steaming at k knots

$\mathbb{M}^F$	Number of containers that can be transported in the planning horizon
$\mathbb{M}^{PC}$	Maximum number of calls of the same port within one service round trip
$\mathbb{M}^P$	Maximum number of port calls per service
$\mathbb{M}^K_k$	Maximum voyage duration in days between two ports
$\mathbb{M}^{Sea}_k$	Upper bound for the maximum duration at sea per service
$\mathbb{M}^{TrMax}_{i,j}$	Upper bound for the maximum duration in days to steam leg (i,j)

Decision variables

$\alpha_n \in \mathbb{R}^+_0$	Served quantity of cargo flow $n \in N$, $\alpha_n \leq q^{Max}_n$
$x^L_{s,n,i,j,l,l'} \in \mathbb{R}^+_0$	Transported quantity of cargo flow $n \in N$ on leg (i,j,l,l') of service $s \in S$
$l^L_{s,n,p,l} \in \mathbb{R}^+_0$	Loaded quantity of cargo flow $n \in N$ on service $s \in S$ at port (p,l)
$u^L_{s,n,p,l} \in \mathbb{R}^+_0$	Unloaded quantity of cargo flow $n \in N$ from service $s \in S$ at port (p,l)
$x^E_{s,e,i,j,l,l'} \in \mathbb{R}^+_0$	Transported quantity of empty containers of type $e \in E$ on leg (i,j,l,l') of service $s \in S$
$l^E_{s,e,p,l} \in \mathbb{R}^+_0$	Loaded quantity of empty containers of type $e \in E$ on service $s \in S$ at port (p,l)
$u^E_{s,e,p,l} \in \mathbb{R}^+_0$	Unloaded quantity of empty containers of type $e \in E$ from service $s \in S$ at port (p,l)
$y^S_s \in \{0,1\}$	Indicates whether service $s \in S^O$ is activated
$y^L_{s,i,j,l,l'} \in \{0,1\}$	Indicates whether service $s \in S^O$ uses leg (i,j,l,l')
$y^P_{s,i,l,v} \in \{0,1\}$	Indicates whether service $s \in S^O$ calls port $(i,l) \in P^L$ with vessel type $v \in VT$
$y^{VT}_{s,v} \in \{0,1\}$	Indicates whether service $s \in S^O$ uses vessel type $v \in VT$
$w_{s,v,vc} \in \{0,1\}$	Indicates whether service $s \in S^O$ uses $vc \in VC$ many vessels of type $v \in VT$
$b_{s,v,vc,k} \in \mathbb{R}^+_0$	Bunker consumption of service $s \in S^O$ in the planning horizon if $vc \in VC$ many vessels of type $v \in VT$ steam at k kts
$\tau^S_{s,k} \in \mathbb{R}^+_0$	Duration at sea in days in the planning horizon when all vessels of service $s \in S^O$ steam $k \in K$ knots on all legs
$y^K_{s,k} \in \{0,1\}$	Variable indicating whether all vessels deployed on service $s \in S^O$ steam at $k \in K$ knots
$\tau^B_{s,p,l} \in \mathbb{R}^+_0$	Additional buffer time in days in the whole planning horizon at port (p,l) of service $s \in S^O$
$\delta^I_{s,p,l,v} \in \mathbb{R}^+_0$	Incoming draft of vessel type $v \in VT$ at service's $s \in S^O$ port call $(p,l) \in P^L$

$\delta^O_{s,p,l,v} \in \mathbb{R}^+_0$	Outgoing draft of vessel type $v \in VT$ at service's $s \in S^O$ port call $(p,l) \in P^L$
$y^T_{s,t,i,j,l,l'} \in \{0,1\}$	Variable indicating whether service $s \in S$ leg $(i,j,l,l') \in L^L$ is used to fulfill the transit time $t \in T$
$\tau^T_{s,t,i,j,l,l'} \in \mathbb{R}^+_0$	Duration in days to steam leg $(i,j,l,l') \in L^L$ at service $s \in S$ in days of transit time requirement $t \in T$
$\rho^P \in \mathbb{R}^+_0$	Slack variable for not transported partner cargo flows that must be transported

The objective is to maximize the carrier's profit before interest and taxes (EBIT) in the whole planning horizon:

$$max \;\; profit = \sum_{n \in N} \left(r_n - \phi^C_{e_n}\right) \cdot \alpha_n \tag{5.1}$$

$$- \sum_{s \in S^O} \sum_{v \in VT} \left(\sum_{p \in P^L} (\phi^{PC}_{p,v} \frac{\tau}{f} y^P_{s,p,l,v}) + \sum_{vc \in VC} (\phi^D_v \; vc \; w_{s,v,vc}) \right) \tag{5.2}$$

$$- \sum_{s \in S^O} \sum_{v \in VT, vc \in VC} \sum_{k \in K} (\phi^T b_{s,v,vc,k}) \tag{5.3}$$

$$- \sum_{s \in S^O} \left(\sum_{(p,l) \in P^L} \sum_{n \in N} \left(\begin{cases} \phi^{CH}_p, \; if \; p = o_n \vee p = d_n \\ \phi^{TS}_p, \; else \end{cases} \right) (l^L_{s,n,p,l} + u^L_{s,n,p,l}) \right) \tag{5.4}$$

$$- \sum_{s \in S^O} \left(\sum_{(p,l) \in P^L} \phi^{TS}_p \sum_{e \in E} (l^E_{s,e,p,l} + u^E_{s,e,p,l}) \right) \tag{5.5}$$

$$- \sum_{s \in S^P} \sum_{sg=(i,j,l,l') \in SG_s} \sum_{rg \in RG_{sg}} \sum_{r \in rg}$$

$$\left(\sum_{n \in N} \phi^S_{s,i,e_n} u_{n,r} l^L_{s,n,i,l} + \sum_{e \in E} \phi^S_{s,i,e} u_{e,r} l^E_{s,e,i,l} \right) \tag{5.6}$$

$$- \mathbb{P} \rho^P \tag{5.7}$$

Term (5.1) determines the revenue for transporting the cargo flows minus the container depreciation. The revenue is gained for transporting both, own and partner cargo. The second term (5.2) subtracts the port call cost for all used services and the vessel depreciation in the planning horizon. The third term (5.3) calculates the bunker cost for the deployed vessels in the whole planning horizon. In the fourth term (5.4) the container handling cost at the port of origin and destination as well as the transhipment costs at each port and service are calculated for the laden containers (cargo flows) and for empty containers in term (5.5). Term (5.6) calculates the slot cost when using partner services. The slot cost are modeled by activating the service specific leg cost when loading a container onto the partner service's vessels. Finally, term (5.7) imposes penalty cost if partner cargo, on which the partners have contractually agreed, is not transported.

The objective is subject to several constraints that are presented in the remainder of this section. They are distinguished by flow balance and transhipment, draft and capacity, liner service, round trip and port duration, transit time and business constraints.

Flow Balance and Transhipment Constraints

The following constraints ensure the flow within the network and explicitly respect transhipment operations to correctly associate the costs for the objective function.

$$\sum_{(i,p,l',l)\in L^L} x^L_{s,n,i,p,l',l} + l^L_{s,n,p,l} = \sum_{(p,j,l,l')\in L^L} x^L_{s,n,p,j,l,l'} + u^L_{s,n,p,l} \qquad \forall s \in S, n \in N, (p,l) \in P^L \tag{5.8}$$

$$\sum_{(i,p,l',l)\in L^L} x^E_{s,e,i,p,l',l} + l^E_{s,e,p,l} = \sum_{(p,j,l,l')\in L^L} x^E_{s,e,p,j,l,l'} + u^E_{s,e,p,l} \qquad \forall s \in S, e \in E, (p,l) \in P^L \tag{5.9}$$

Constraints (5.8) and (5.9) ensure the flow balance for services and ports. Incoming containers must be either transported to the next called port of the service or unloaded. Outgoing containers are either transported to the port by the service or loaded at the port from another service (transhipment operation). The explicit modeling of loading and unloading decisions is required to associate transhipment costs with these operations.

$$\sum_{l\in\mathbb{L},s\in S} l^L_{s,n,p,l} = \sum_{l\in\mathbb{L},s\in S} u^L_{s,n,p,l} + \begin{cases} \alpha_n, & \textit{if } p = o_n \\ -\alpha_n, & \textit{if } p = d_n \\ 0, & \textit{else} \end{cases} \qquad \forall n \in N, p \in P \tag{5.10}$$

Containers unloaded from a service at a port must be loaded by another service at the same port. An exception is made when a laden container arrives at its port of destination d_n and is removed from the network (see constraints (5.10)). Containers can also enter the network at their port of origin o_n.

$$\sum_{l \in \mathbb{L}, s \in S} l^E_{s,e,p,l} = \sum_{l \in \mathbb{L}, s \in S} u^E_{s,e,p,l} - \sum_{n \in N: o_n = p} \alpha_n + \sum_{n \in N: d_n = p} \alpha_n \qquad \forall e \in E, p \in P \tag{5.11}$$

Similar constraints ensure that all transported laden containers are balanced with empty containers (see constraints (5.11)). The sum of loaded empty containers of type e at port p must equal the sum of unloaded empty containers (due to transhipment) and the difference of unloaded and loaded laden containers that can be used to serve other cargo flows with empty containers.

Vessel Draft and Capacity Constraints

The next group of constraints ensure the vessel type - port compatibility and vessel capacity for all operated liner services.

$$d_v \cdot y^P_{s,p,l,v} \leq D^{Max}_p \qquad \forall s \in S^O, (p,l) \in P^L, v \in VT \tag{5.12}$$

Constraints (5.12) ensure that the lightship[1] draft of vessel type v used on service s is less than the maximum depth at port p. This is a basic requirement to deploy vessel type v to service s.

Beside this lightship draft load dependent drafts are considered as well in the next constraints.

$$\sum_{(i,p,l',l) \in L^L} \left(\sum_{n \in N} u_{n,weight} x^L_{s,n,i,p,l',l} + \sum_{e \in E} u_{e,weight} x^E_{s,e,i,p,l',l} \right) dws^S_v + dws^I_v = \delta^I_{s,p,l,v} \qquad \forall s \in S^O, (p,l) \in P^L, v \in VT \tag{5.13}$$

$$\sum_{(p,j,l,l') \in L^L} \left(\sum_{n \in N} u_{n,weight} x^L_{s,n,i,p,l',l} + \sum_{e \in E} u_{e,weight} x^E_{s,e,i,p,l',l} \right) dws^S_v + dws^I_v = \delta^O_{s,p,l,v} \qquad \forall s \in S^O, (p,l) \in P^L, v \in VT \tag{5.14}$$

[1] Lightship draft refers to a vessel's draft without any payload.

Constraints (5.13) and (5.14) set the load dependent auxiliary draft variables for all vessels used on service s. The draft is calculated by the linearization of the deadweight scale (DWS), provided for all vessel types. The constraint uses the slope dws_v^S and the intersection dws_v^I for the selected vessel type v of the service (see Chapter 4) to calculate a draft of service s at port p, l for vessel type v. Note that the draft is set for all cargo transported to and from the port in the planning horizon. The draft auxiliary variable is set independent of the vessel type that is actually selected by the model.

$$\delta^I_{s,p,l,v} \leq D_p^{Max} \frac{\mathcal{T}}{f} + \mathbb{M}_v^{DWS}(1 - y^P_{s,p,l,v}) \quad \forall s \in S^O, (p,l) \in P^L, v \in VT \tag{5.15}$$

$$\delta^O_{s,p,l,v} \leq D_p^{Max} \frac{\mathcal{T}}{f} + \mathbb{M}_v^{DWS}(1 - y^P_{s,p,l,v}) \quad \forall s \in S^O, (p,l) \in P^L, v \in VT \tag{5.16}$$

The load dependent vessel type - port compatibility is ensured by constraints (5.15) for incoming and by constraints (5.16) for outgoing cargo to/ from port p. Note that the deadweight scale constraints are imposed on operated services only, because the actual load of the partner services are unknown. Both constraints deactivate the draft upper bound when vessel type v and port p, l is not been selected for service s. Otherwise, the maximum depth D_p^{Max} must hold at port p. We multiply the depth with the number of calls in the planning horizon to account for the vessel draft in the planning horizon.

$$\sum_{n \in N, r \in rg} u_{n,r} x^L_{s,n,i,j,l,l'} + \sum_{e \in E, r \in rg} u_{e,r} x^E_{s,e,i,j,l,l'} \leq$$

$$\sum_{v \in VT} C_{v,rg} \frac{\mathcal{T}}{f} y^{VT}_{s,v} \qquad \begin{array}{r} \forall s \in S^O, \\ (i,j,l,l') \in L^L, \\ rg \in RG \end{array} \tag{5.17}$$

Constraints (5.17) limits the capacity for each resource group rg and used service leg (i, j, l, l') for the vessels on service s operated by the carrier at hand in the planning horizon. Laden and empty containers utilize different resources r, such as the container weight, according to the coefficients $u_{n,r}$ and $u_{e,r}$. The capacity constraint is imposed for the selected vessel type $y^{VT}_{s,v}$ in service s.

Liner Service Constraints

The next set of constraints ensure that the port rotation of each service performs a round trip. These constraints are similar to the vehicle routing problem with the exception that no predetermined depots exist.

$$\sum_{(i,p,l',l)\in L^L} y^L_{s,i,p,l',l} = \sum_{(p,j,l,l')\in L^L} y^L_{s,p,j,l,l'} \qquad \forall s \in S^O, (p,l) \in P^L \tag{5.18}$$

$$\sum_{(i,p,l',l)\in L^L} y^L_{s,i,p,l',l} \leq 1 \qquad \forall s \in S^O, (p,l) \in P^L \tag{5.19}$$

$$\sum_{(p,j,l,l')\in L^L} y^L_{s,p,j,l,l'} \leq 1 \qquad \forall s \in S^O, (p,l) \in P^L \tag{5.20}$$

$$\sum_{n\in N} x^L_{s,n,i,j,l,l'} + \sum_{e\in E} x^E_{s,e,i,j,l,l'} \leq \mathbb{M}^F y^L_{s,i,j,l,l'} \qquad \forall s \in S^O, (i,j,l,l') \in L^L \tag{5.21}$$

$$\sum_{(i,j,l,l')\in L^L : l'\neq l} y^L_{s,i,j,l,l'} \leq 1 \qquad \forall s \in S^O, l \in \mathbb{L} \tag{5.22}$$

$$\sum_{(i,j,l,l')\in L^L} y^L_{s,i,j,l,l'} \leq 1 \qquad \forall s \in S^O, (i,j) \in L \tag{5.23}$$

$$\sum_{(i,p,l,l')\in L^L} y^L_{s,p,j,l,l'} \leq \mathbb{M}^{PC} \qquad \forall s \in S^O, p \in P \tag{5.24}$$

$$\sum_{(i,j,l',l)\in L^L} y^L_{s,i,j,l,l'} \geq 2y^S_s \qquad \forall s \in S^O \tag{5.25}$$

Constraints (5.18) ensure the connectivity at each service's port. If the port p is called by operated service s, an incoming and outgoing leg must be activated. Constraints (5.19) and (5.20) ensure that a port on a specific layer is not visited more than once. Constraints (5.21) enable the routing of containers on the service's leg if it is activated. The upper bound $\mathbb{M}^F$ can be set to the number of containers that can be transported in the planning horizon, i.e. the sum of q^{Max}_n variables. The bounds can be improved if any information about the maximum service length is known, such as the largest vessel type's slot capacity times the weeks in the planning horizon. Constraints (5.22) state that not more than one leg connecting two layers per service can exist. This is required to avoid cycling between different network layers. Constraints (5.23) ensure that a specific leg between port i and j is not used more than once per service which is commonly found in practical networks. The layer constraints (5.24) ensure that a port is not called more than $\mathbb{M}^{PC}$ many times per service round trip. Although theoretically not limited, $\mathbb{M}^{PC}$ is set to two in practical networks. Constraints (5.25) ensure that if a service is activated, at least two legs for a pendulum service are used.

Liner Service Round Trip Constraints

The model now can create port rotations that allow complex route types. The next step is to select exactly one vessel type per service and deploy vessels such that the service can perform a weekly round trip.

$$\sum_{(p,l)\in P^L} y^P_{s,p,l,v} \leq \mathbb{M}^P y^{VT}_{s,v} \qquad \forall s \in S^O, v \in VT \tag{5.26}$$

$$\sum_{v\in VT} y^P_{s,p,l,v} = \sum_{(i,p,l',l)\in L^L} y^L_{s,i,p,l',l} \qquad \forall s \in S^O, (p,l) \in P^L \tag{5.27}$$

$$\sum_{v\in VT} y^{VT}_{s,v} \leq 1 \qquad \forall s \in S^O \tag{5.28}$$

$$\sum_{vc\in VC} w_{s,v,vc} = y^{VT}_{s,v} \qquad \forall s \in S^O, v \in VT \tag{5.29}$$

$$\sum_{vc\in VC} w_{s,v,vc} = y^S_s \qquad \forall s \in S^O, v \in VT \tag{5.30}$$

Constraints (5.26) ensure that the vessel type variable $y^{VT}_{s,v}$ is activated if a port is called by a service with vessel type v. $\mathbb{M}^P$ limits the maximum number of port calls per service. Practical network services indicate that a value of 30 for the maximum number of port calls per service is a reasonable upper bound. Long services using large vessels are still limited by the panama canal's locks, see Chapter 2. However, the offering of long round-the-world services is performed in practice, although several disadvantages are recognized (see (Stopford, 2009, p. 528)). Constraints (5.27) state that if a port is visited by a vessel in service s, one outgoing leg must exist. Constraints (5.28) limit the number of vessel types per service to one. To allow more than one vessel type per service, the value can be increased. Due to decreased time charter rates for medium sized panamax vessels, this could be an interesting option in the future. After the vessel type is selected, a number of vessels of this type must be determined. This is formalized in constraints (5.29), by forcing $w_{s,v}$ be exactly one predetermined number vc of vessels per service. If service s is used, at least one vessel must be deployed (see constraints (5.30)).

The next two groups of constraints deal with the timing to deploy enough services to ensure weekly round trips. This is the major mode of operation in liner services and highly relevant when designing practical networks.

$$\sum_{k\in K} y^K_{s,k} = y^S_s \qquad \forall s \in S^O \tag{5.31}$$

$$\tau^S_{s,k} \leq \mathbb{M}^{Sea} y^K_{s,k} \qquad \forall s \in S^O, k \in K \tag{5.32}$$

$$\tau^S_{s,k} \geq \sum_{(i,j,l,l') \in L^L} l_{i,j} \cdot y^L_{s,i,j,l,l'} \frac{1}{k24} \frac{\tau}{f} - \mathbb{M}^K_k (1 - y^K_{s,k}) \qquad \forall s \in S^O, k \in K \tag{5.33}$$

$$\sum_{v \in VT} \sum_{vc \in VC} vc\ f\ w_{s,v,vc} = \frac{1}{\tau/f} \left(\sum_{k \in K} \tau^S_{s,k} + \sum_{(p,l) \in P^L} \tau^P_{s,p,l} \right) \qquad \forall s \in S^O \tag{5.34}$$

For each activated service, exactly one average speed must be selected (see constraints (5.31)). Constraints (5.32) ensure no duration at sea is set if the speed is not activated. $\mathbb{M}^{Sea}_k$ is an upper bound for the total maximum duration at sea. This value is set to $\tau_{vc \in VC}^{max} vc$ and determines the duration when the vessel do not serve any cargo at the ports.

The selected service speed leads to the duration at sea in days, calculated by constraints (5.33). The duration is the distance of each activated leg divided by the selected speed. To simplify the handling of the days at sea for different vessels in the planning horizon, the value is multiplied by the number of weeks in the planning horizon τ/f. This equals the number of vessels times the number of round trips by the assumption of weekly port calls. The term at the right hand side of constraints (5.33) disables the constraints if the speed is not selected. $\mathbb{M}^K_k$ can be set to the maximum voyage duration in days between two ports, for example by taking the maximum leg distance divided by the minimum speed of all vessel types.

Constraints (5.34) ensure that enough vessels are deployed to perform weekly port calls. The round trip on the left hand side is calculated by taking the number of deployed vessels vc times the constant frequency f (if vc vessels have been selected for service s). The round trip time for each operated service must equal the duration at sea $\tau^S_{s,k}$ plus the duration in ports $\tau^P_{s,p,l}$. To determine the duration per round trip, the variables are divided by the number of weeks in the planning horizon. In case of non-predetermined frequencies, f would be a decision variable for each service and would introduce a high degree of further complexity to the model.

Port Duration and Bunker Cost Constraints

Before ensuring the transit times are held, the port call duration and resulting bunker cost are calculated.

$$\begin{aligned} \tau^P_{s,p,l} = \tau^E_p & \left(\sum_{n \in N} \left(l^L_{s,n,p,l} + u^L_{s,n,p,l} \right) + \sum_{e \in E} \left(l^E_{s,e,p,l} + u^E_{s,e,p,l} \right) \right) \\ & + \tau^{Add}_p \frac{\tau}{f} y^P_{s,p,l} + \tau^B_{s,p,l} \qquad \forall s \in S^O, (p,l) \in P^L \end{aligned} \tag{5.35}$$

$$b_{s,v,vc,k} \geq \tau^S_{s,k} bc_{v,k} - \mathbb{M}^{Sea}_k (1 - y^S_{s,k}) - \mathbb{M}^{Sea}_k (1 - w_{s,v,vc}) \qquad \begin{array}{r} \forall s \in S^O, \\ v \in VT, \\ vc \in VC, \\ k \in K \end{array} \qquad (5.36)$$

Constraints (5.35) set the overall port call duration in days at each port based on the duration in days to move one container at port p, τ^E_p. The time to load and unload laden and empty container is considered. The load and unload time is zero if the port p, l is not called by service s because no containers can be transported. The parameters τ^{Add}_p add the constant pilotage and strategic buffer in days per port call and must be multiplied by the number of all port calls in the planning horizon. This additional duration is only imposed if service s calls port p on layer l. We use the model buffer variable $\tau^B_{s,p,l}$ per service and port call to increase the flexibility of the model for holding the round trip time. The model can decide to stay longer at certain ports but increase the service speed for holding the transit times.

Constraints (5.36) determine the activity of the bunker consumption helper variable $b_{s,v,vc,k}$ that stores the bunker consumption of service s in the planning horizon when steaming with vc vessels of type v at k knots . The variable is used in the objective to get the associated bunker cost. The constraint is deactivated by the second and third term in case the speed, vessel type or vessel count is not selected for the service.

Transit Time Constraints

The next constraints ensure the transit time between two ports is enforced if cargo is transported. The idea is to determine a path between two ports and determine the overall path duration by the sum of all used service legs.

$$\begin{aligned} \tau^T_{t,s,i,j,l,l'} \geq & \sum_{k \in K} \left(l_{i,j} y^L_{s,i,j,l,l'} \frac{1}{k24} - \frac{l_{i,j}}{k24} (1 - y^K_{s,k}) \right) \\ & + \frac{1}{\tau/f} \frac{1}{2} \left(\tau^P_{s,i,l} + \tau^P_{s,j,l'} \right) - \mathbb{M}^{TrMax}_{i,j} \cdot y^T_{t,s,i,j,l,l'} \qquad \begin{array}{r} \forall t \in T, \\ s \in S, \\ (i,j,l,l') \in L^L \end{array} \end{aligned} \qquad (5.37)$$

In constraints (5.37), the transit time specific's duration $\tau^T_{t,s,i,j,l,l'}$ in days on leg (i, j, l, l') of service s is calculated. The duration consists of the duration at sea (first term of the right hand side) plus the time at port i ($\tau^P_{s,i,l}$) and port j ($\tau^P_{s,j,l'}$). The port durations must be transformed to weekly durations (number of weeks in the planning horizon) because they store the overall duration in the planning horizon.

At a port, the time when a specific container is unloaded or loaded is usually unknown in the strategic planning horizon. It is assumed that it takes on average half of the total port duration $\tau^P_{s,i,l}$. For partner services $s \in S^P$, a constant duration (such as 24 hours) and an average design speed can be assumed because the actual speed is often not known for certain. $\mathbb{M}^{TrMax}_{i,j}$ is an upper bound for the maximum duration to steam leg (i, j). For example, the duration when steaming with the minimum speed and the maximum port duration can be used as an upper bound.

The following constraints ensure that if one unit of laden containers is transported between ports o and d, the transit time between the ports must hold.

$$\tau^T_{t,s,i,j,l,l'} \leq \mathbb{M}^{TrMax}_{i,j} \cdot y^T_{t,s,i,j,l,l'} \qquad \forall t \in T, s \in S, (i,j,l,l') \in L^L \tag{5.38}$$
$$\tau^T_{t,s,i,j,l,l'} \leq \mathbb{M}^{TrMax}_{i,j} \cdot y^L_{s,i,j,l,l'} \qquad \forall t \in T, s \in S, (i,j,l,l') \in L^L \tag{5.39}$$

Constraints (5.38) limit the duration of service s leg (i, j, l, l') if it should not be considered for the transit time. Similar, the duration on the leg can only be used if it is served by service s (see constraints 5.39).

$$\sum_{(i,p,l',l) \in L^L, s \in S} y^T_{t,s,i,p,l',l} = \sum_{(p,j,l,l',s \in S) \in L^L} y^T_{t,s,p,j,l,l'} \qquad \begin{array}{r} \forall t \in T, (p,l) \in P^L, \\ p \neq o_t \wedge p \neq d_t \end{array} \tag{5.40}$$

$$\sum_{(p,j,l,l') \in L^L, s \in S} y^T_{t,s,p,j,l,l'} \leq 1 \qquad \forall t \in T, (p,l) \in P^L, p = o_t \tag{5.41}$$
$$\sum_{(i,p,l',l) \in L^L, s \in S} y^T_{t,s,i,p,l',l} \leq 1 \qquad \forall t \in T, (p,l) \in P^L, p = d_t \tag{5.42}$$

The transit time duration is modeled by finding a path from o_t to d_t. Each activated path variable must be part of a path (see constraints (5.40)) that does not contain more than one incoming leg (see constraints 5.41) and outgoing leg (see constraints (5.42)).

$$\sum_{s \in S} \sum_{(i,j,l,l') \in L^L} \tau^T_{t,s,i,j,l,l'} \leq \theta_t \qquad \forall t \in T \tag{5.43}$$
$$\sum_{n \in N: o_n = o_t \wedge d_n = d_t} \alpha_n \leq \mathbb{M}^F \cdot \sum_{s \in S, (o_t,j,l,l) \in L^L} y^T_{t,s,o_t,j,l,l'} \qquad \forall t \in T \tag{5.44}$$

Constraints (5.43) state that the duration of all activated legs on the path must not be larger than the transit time duration θ_t in days. Finally, constraints (5.44) ensure that the transit time between ports o and d holds when any cargo is transported between o and d.

Business Constraints

The following constraints define further business rules that were identified through discussions with liner carriers (see Chapter 2).

$$\sum_{n \in N^P} \alpha_n \geq \theta \cdot \sum_{n \in N^P} q_n^{Max} - \rho^P \tag{5.45}$$

Constraints (5.45) ensure that a minimum required percentage of partner cargo is served. If this is not possible, the penalty variable ρ^P is activated.

$$\begin{aligned} &\sum_{n \in N^O, r \in R_n} u_{n,r} \cdot x^L_{s,n,i,j,l,l'} \\ &+ \sum_{e \in E, r \in R_e} u_{e,r} \cdot x^E_{s,e,i,j,l,l'} \leq C^P_{s,i,j,l,l',rg} \qquad \forall s \in S^P, (i,j,l,l') \in L_s, rg \in RG \end{aligned} \tag{5.46}$$

Similar to the capacity constraint of operated services, constraints (5.46) ensure the maximum partner capacity per leg for different resource groups. Network planners are able to set contractually defined slot capacities for partner services on a per leg basis.

$$\sum_{p \in \bar{p}} \left(\sum_{l \in \mathbb{L}, v \in VT} y^P_{s,p,l,v} \right) \leq 1 \qquad \forall s \in S^O, \bar{p} \in \bar{P} \tag{5.47}$$

Constraints (5.47) avoid ports on the same service that are subject to embargo. Ports that are forbidden to be called together on a service are sets $\bar{p} \in \bar{P}$ containing two or more ports. The constraint ensures that out of each set only one port p is called per service.

To include partner services in the network, the service, leg, vessel type and vessel count variables are fixed in the model. The objective does not impose any costs for the partner services in S^P (except the slot cost) because the partner is in charge of paying these. The draft constraints are not relevant for partners because the actual tonnage is unknown.

5.1.2. Subtour Elimination Constraints

The liner shipping network design formulation in the previous Section 5.1 allows subtours within a given service that are not allowed. Figure 5.1 illustrates an invalid service with two subtours that are disconnected from each other. Note that this configuration would be valid if the subtours were selected as individual services. The removal of subtours must be assured for the final solution of the mixed integer model.

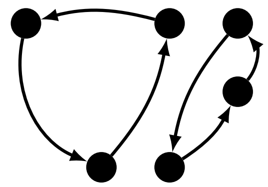


Figure 5.1.: Invalid port rotation for one service due to existing subtours.

Several well-known formulations to avoid subtours exist in literature in the context of the traveling salesman and vehicle routing problem (see (Suhl and Mellouli, 2013, p. 244)). N denotes the set of all nodes (ports and layers), S the set of all subsets of N. Three possibilities are:

$$\sum_{i \in S, j \notin S} x_{ij} \geq 1, \quad \forall\, \emptyset \subset S \subseteq N - 1 \tag{5.48}$$

$$\sum_{i \in R, j \in R} x_{ij} \leq |R| - 1, \quad \forall\, \emptyset \subset R \subseteq \{2, 3, \cdots, |N|\} \tag{5.49}$$

$$z_i - z_j + n \cdot x_{ij} \leq n - 1, \quad \forall\, i, j := 2, \cdots, n \; (i \neq j) \tag{5.50}$$

Constraints (5.48) ensure that all nodes in each subset S is connected with every other node that is not in the subset. This works by enforcing an activated edge x_{ij} between these subsets. Variant (5.49) avoid cycles within each subset R of $\{2, 3, \cdots, |N|\}$. Each subset with $|R|$ many nodes must not contain more than $|R| - 1$ edges. Otherwise, the tour contains cycles. Constraints (5.50) introduce an sequencing of the nodes to disable subtours that do not contain the starting node which is assumed to be node 1. The formulation uses additional variables.

One can observe that constraints (5.48) and (5.50) are not applicable to the liner shipping network design problem because not all ports must be served, however, they can be served several times. The subtour elimination constraints proposed for the liner shipping network design problem is based on constraints (5.49) and shown in constraints (5.51). The optimal solution of the mixed integer program is denoted as x^* and S^* a set of activated service candidates, whereas each service $s \in S^*$ uses a set of legs L_s^* and (layered) ports P_s^*.

$$\sum_{(i,j,l,l') \in L*_s} y_{s,i,p,l',l} \leq |P_s^*| - 1 \qquad \forall s \in \bar{S}^* \tag{5.51}$$

The subtour elimination constraints (5.51) are added for all services that contain subtours $\bar{S}^*$ by checking each service $s \in S^*$. If the service contains subtours this means that too many edges are used to serve the ports. We therefore force the number of activated legs to be equal to the number of activated ports minus one. It does not cut feasible solutions because the subtours can still be served by different services, each not containing subtours.

The subtour elimination constraints can be added to the model before solving it. However, enumerating all subsets of activated edges containing subtours (legs) leads to an exponential number of constraints. To reduce this number, their creation can be delayed. One way is to solve the mixed integer program to optimality, check for subtours, exclude them from the model and resolve it. The drawback of this approach is twofold: First, solving a model can be very time consuming and solving it several times might not be applicable. Second, the gap to the optimal solution provided by the MIP solver is not guaranteed to be the *real gap* because constraints are missing.

To circumvent these problems, the subtour elimination constraints are inserted within the solution process of the MIP solver. Most of the available solvers provide callbacks to insert custom application logic at certain events such as when incumbents (new best integer solutions) are found. In the scope of this thesis, the callback method of Gurobi (2014) is used to add the subtour elimination constraint when a subtour in an incumbent solution is found. These *lazy constraints* are then propagated in the solver's threads that solve the B&B tree. The usage of lazy constraints requires specific solver parameters. For example, dual information from the presolver cannot be used anymore because the dual bounds are not valid when constraints are missing in the initial model.

The mixed integer problem combined with the subtour elimination constraints can be solved using commercial and non-commercial MIP solvers. In the next section, runtime results for solving the model with Gurobi are presented.

5.1.3. Numerical Results for the Mixed Integer Program

The previously described mixed integer program is applied to the LINER-LIB benchmark instances[2], already presented for the cargo allocation problem (see Chapter 4). The numerical results in the remainder of this chapter are performed on identical computers with the following configuration: Intel Core 2 Quad CPU 2.83 GHz, 8

[2]The LINER-LIB instances are developed by Brouer et al. (2013).

GB RAM and Windows 7 64 Bit. The model is implemented in C# using .NET framework 4.5 and compiled as a 64 Bit application. Gurobi 5.6.0 64 Bit is used as MIP-solver. To get information on the solution progress, such as the current lower and upper bound as well as the activities, callbacks are registered in Gurobi.

Table 5.4 shows the model sizes for the Baltic and WAF instance for different sets. Recall that the Baltic instance contains 12 ports, 132 legs, 22 cargo flows and 22 transit times, whereas the WAF contains 20 ports, 37 cargo flows, 33 transit times. *Rows* indicate the number of constraints in the model, *Cols.* the number of columns, i.e. variables. *Bin.* indicate the number of binary variables. *NZ* presents the number of non-zero elements in the problem matrix.

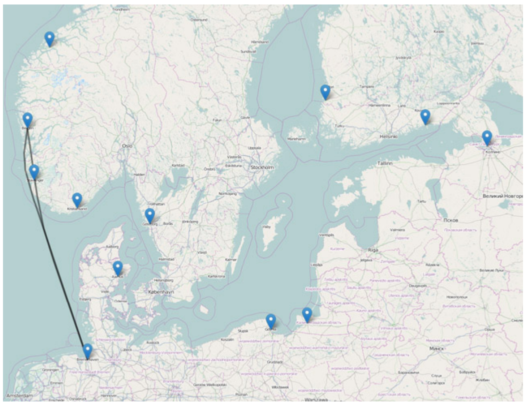
	Maximum number of						
Instance	**Services**	**Vessels**	**Layers**	**Rows**	**Cols.**	**Bin.**	**NZ**
Baltic	1	1	1	6,261	8,711	2,829	3,1406
	1	1	2	25,836	37,074	12,148	137,966
	1	2	1	6,772	9,456	3,064	3,4548
	2	1	1	13,352	19,196	6,088	71,387
	2	2	2	51,074	74,450	24,401	283,701
WAF	1	1	1	13,884	25,712	6,252	73,207
	1	1	2	82,039	144,661	38,921	479,836
	1	2	1	28,091	43,042	13,167	348,674
	2	1	1	34,492	57,793	15,647	194,046
	2	2	2	252,427	392,596	122,807	1,517,454

Table 5.4.: Resulting model size using different input sets (including transit times).

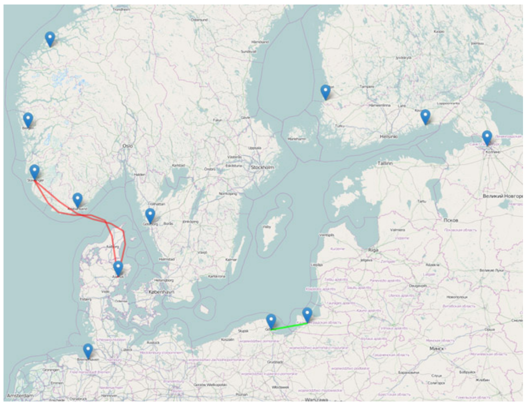
The table indicates that especially the number of layers ($\mathbb{L}$) and the maximum number of services (S^O) influence the problem size and thus the duration to solve the instance to optimality. The WAF MIP with not more than 2 services, vessels and layers, already contains $122,807$ binary variables.

In Figure 5.2(a) to 5.2(d) example incumbents during the solution process of the Baltic instance are shown graphically by extracting the variable activities. It can be seen that the solver first identifies networks with unprofitable pendulum services. Recall that all cargo flows in the Baltic instance originate or arrive at Bremerhaven, Germany, because a feeder network should be created. As a result, incumbent two (see Figure 5.2(b)) does not serve any cargo flows. Due to the shorter distances the objective value is improved compared to Figure 5.2(a). The third incumbent (see Figure 5.2(c)) is profitable and contains one pendulum and two circle services serving Bremerhaven. The fourth incumbent further increases the profit and extends the service length. The service calling Ålesund (Norway) and Kotka (Finland) is a butterfly route and calls Bremerhaven two times during the round trip.

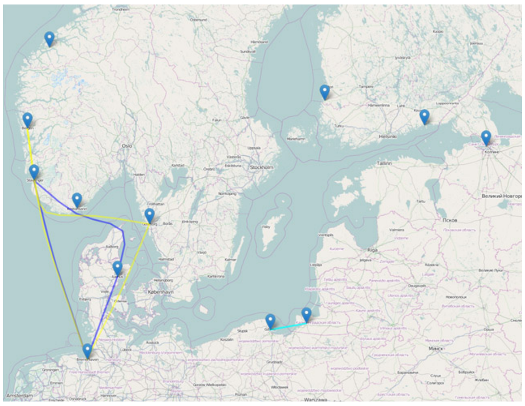
The mixed integer program relies mainly on three different sets: The maximum number of services, the maximum number of vessels per service and the maximum



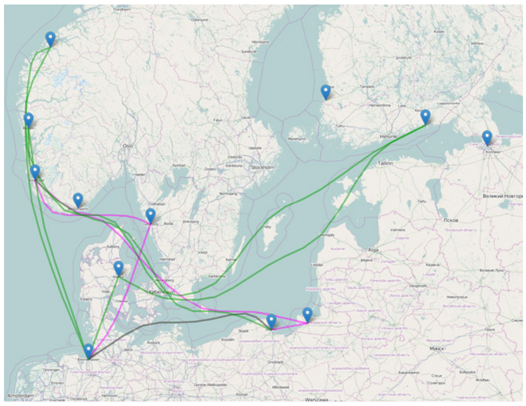
(a) First incumbent. Profit is -9.53 million US$.



(b) Second incumbent. Profit is -6.16 million US$.



(c) Third incumbent. Profit is 5.80 million US$.



(d) Fourth incumbent. Profit is 6.81 million US$.

Figure 5.2.: Example incumbents for the Baltic instance using not more than three services, two layers and two vessels per service.

number of layers. Using more than one layer allows the model to create complex route types. In Figure 5.3, the solution progress for a maximum of two vessels per service and simple route types are shown. The maximum number of services that the model can deploy is shown as different data sets indicated by *Max S*. The maximum runtime of Gurobi is limited to 12 hours for the Baltic instance. Except when limiting the model to create only one service, no instance could be solved to optimality. In particular, a remaining gap of more than 10% can be observed. In the largest model with up to four services, the first incumbent solution was found after 2.5 minutes.

When enabling complex route types using two layers, the gap is closed even slower (see Figure 5.4) and an optimal network could be determined using only one service. Another problem is the increased duration until Gurobi finds the first incumbent

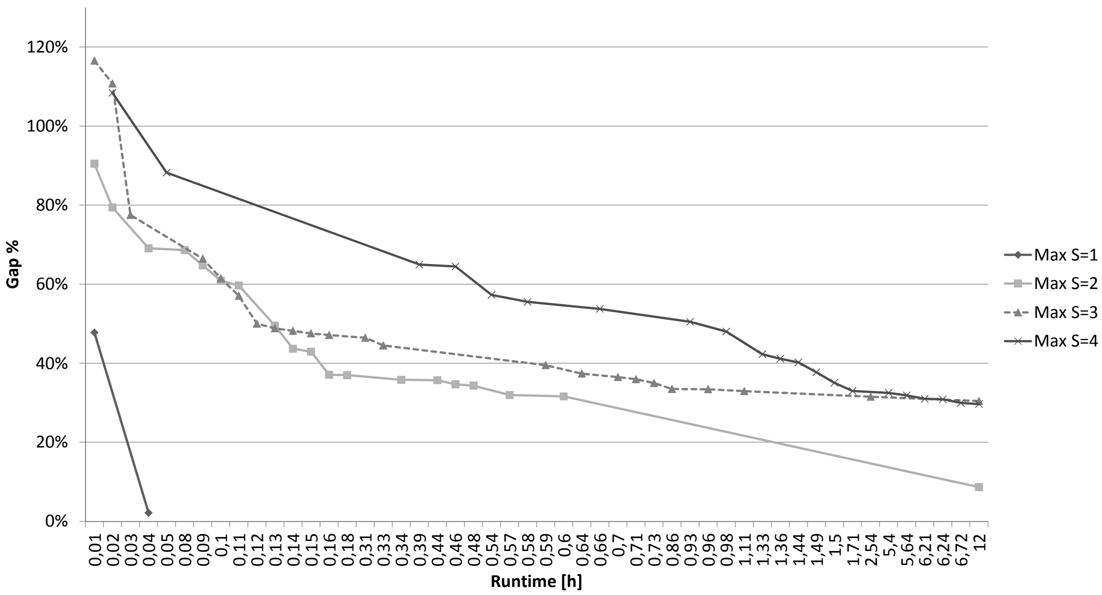


Figure 5.3.: Gap in the Baltic instance per service upper bound allowing one layer and a maximum of two vessels per service. The model is limited to deploy a maximum of 1 to 4 services (*MaxS*).

solution. For the largest model with four services, it takes about 30 minutes to find the first integer solution.

Although the instances could not be solved to optimality, information can be inferred from the non-optimal bounds. In Figure 5.5 the best incumbent (best lower bound) and the best upper bound (LP relaxation) for different data sets are shown. On the x-axis the number of vessels (1,..,4) and the maximum services (1,..,2) are shown. In Figure 5.5(a), the network is limited to use one layer, in Figure 5.5(b) two layers are allowed. The figures indicate, that solving the Baltic instance with complex route type increases the complexity of the problem as the remaining gap after 12 hours gets larger. The best solution found in the one layer case is 23.9 million US$, in the two layer case 20.5 million US$ with a maximum of 4 and 2 vessels per service respectively. This information provides a lower bound for heuristics. A clear assessment of upper bounds is not possible. The results indicate that using a maximum amount of four vessels per service and deploy not more than two services cannot lead to a profit larger than 35 million US$. However, Figure 5.4 states a remaining gap of more than 40%, so the actual upper bound can be around 21 million US$ (in the two service case). Detailed numerical results without considering transit times are given in Appendix D.1. Although the gap is smaller on average, most of the instances could not be solved to optimality as well.

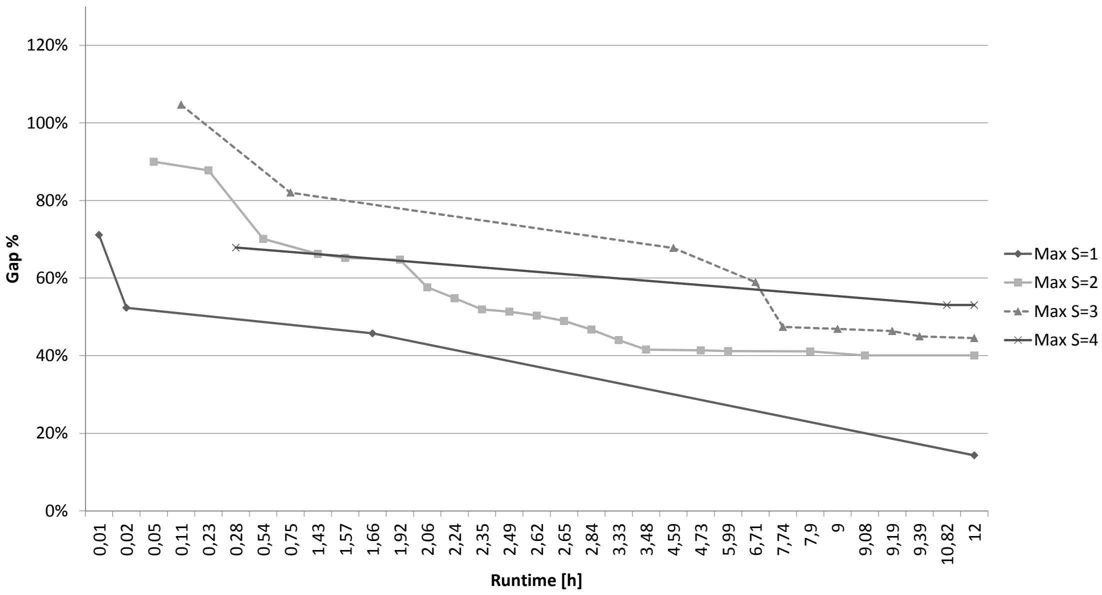
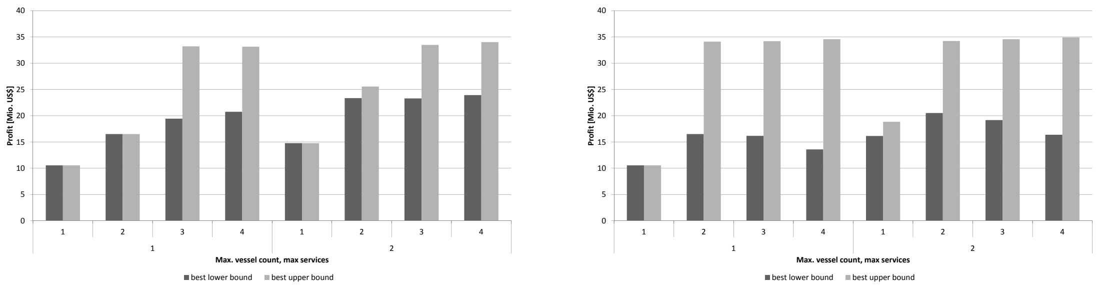


Figure 5.4.: Gap in the Baltic instance per service upper bound allowing two layers and a maximum of two vessels per service.



(a) One layer allowed.

(b) Two layers (extended route types) allowed.

Figure 5.5.: Best bounds found for the Baltic LINER-LIB instance within 12 hours.

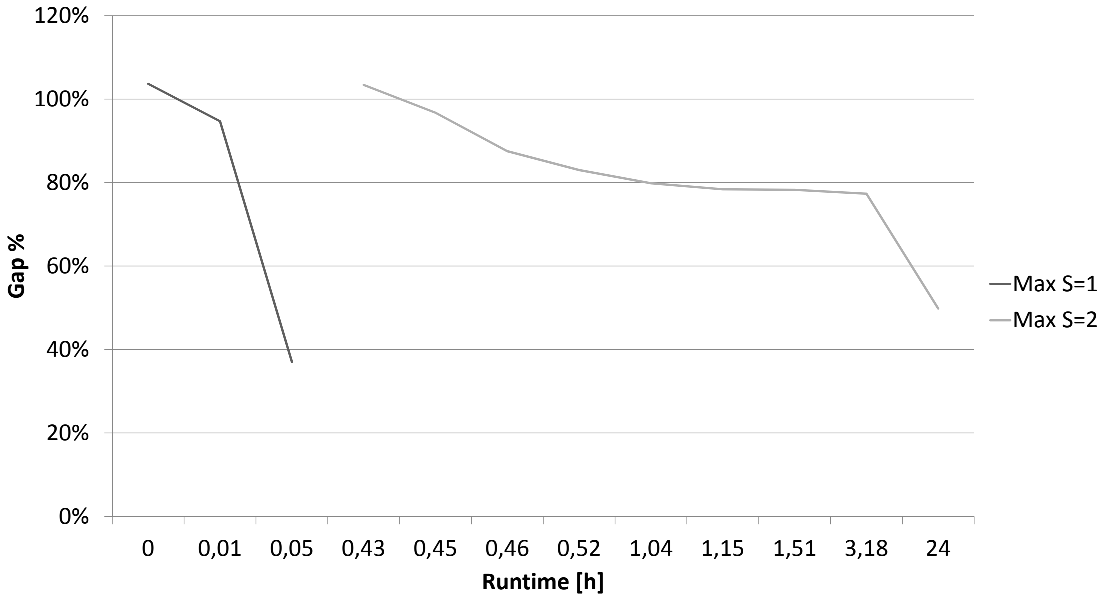


Figure 5.6.: Gap in the WAF instance per service upper bound allowing one layer and a maximum of two vessels per service.

Figures 5.6 and 5.7 present the gap convergence and the resulting bounds for the WAF LINER-LIB instance. To get results for the mixed integer model, we limit the amount of layers to one and increase the maximum runtime to 24 hours. Figure 5.6 indicates that the duration to find a first integer solution increase to nearly 45 minutes when two services are allowed. The resulting gap for the data sets are still more than 35%.

Figure 5.7 shows the lower and upper bound, found after a runtime of 24 hours. The minimum number of vessels has been increased to two, because no profitable network with services deploying not more than one vessel were found. This is a result of the feeder network structure in the WAF instance where all cargo flows have their origin or destination in Algeciras, Southern Spain. The distance to the closest profitable port requires at least two vessels. The best network found has an objective of 72.6 million US$. This is used as a lower bound for the heuristics as well. The drawback is that the upper bound seems to be even more unreasonable compared to the Baltic instance. The MIP solver reported a remaining gap of 65% after 24 hours. Additionally, even three or four services with complex route types might be beneficial for the WAF instance. However, within the runtime no integer solution could be found.

The results from the mixed integer model show that the problem is hard in the sense of runtime. This can be explained by four model properties:

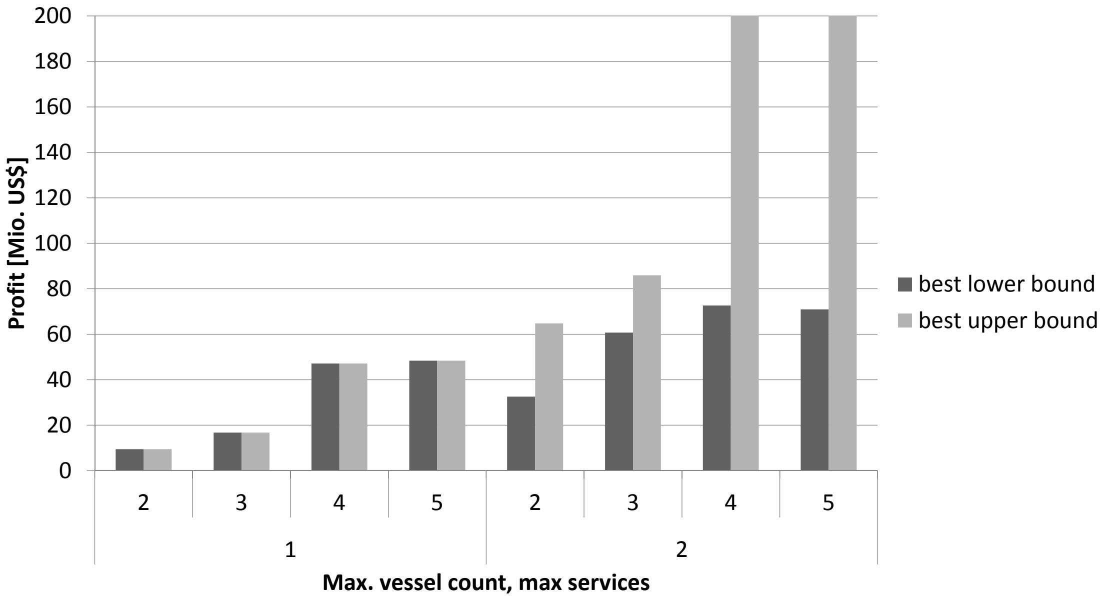


Figure 5.7.: Best bounds found for the WAF LINER-LIB instance within 24 hours.

1. The gap of the LP-relaxation solved at each Branch & Bound node is relatively large to the integer solutions: Relaxing the integer variables does not approximate the convex hull in a reasonable way due to the interdependency between the constraints using binary variables. Thus, the constraints do not provide a tight formulation regarding integer solutions.

2. A lot of inter-dependencies between the variables exist that lead to invalid solutions. This results in enumerating many solutions individually which might prevent efficient sub-tree pruning in the Branch & Bound process.

3. The model contains many binary variables to discretize the bunker cost, define the liner services' port rotations and ensure the transit times. Furthermore, several Big-M constraints to model the logic associated with the linearization are used. This can be disadvantageous for the runtime as well.

4. The model faces symmetry problems, because the port rotations and services can be varied but still lead to identical solutions. Modeling the problem using services (instead of specific vessels) and enabling MIP solver symmetry checks reduce but does not eliminate the problem.

Our analysis for two of the LINER-LIB instances confirmed the results presented in Plum et al. (2013b) about the complexity of this problem. The introduction of the highly relevant transit times makes the problem even harder and results in large gaps compared to previous publications. Although the instance sizes could be increased in the last years (compare Reinhardt and Pisinger (2010)), the liner shipping network design problem still provides a computational challenge for state-of-the-art mixed integer solvers.

We conclude that the second goal of this thesis, to formalize real-world requirements of the liner shipping network design problem and solve small instances to optimality, has been reached. However, real-world instances with more than 10 ports require metaheuristics to obtain good integer solutions. The next sections present two metaheuristics for the liner shipping network design problem and assess the methods on the LINER-LIB instances and a real-world liner network.

5.2. Metaheuristics

The numerical results of the exact mixed integer program in the previous section indicate that only very small instances can be solved to optimality. Álvarez (2009), Mulder and Dekker (2013) and Brouer et al. (2013) show the successful application of metaheuristics to the liner shipping network design problem. For solving larger instances with respect to the practical requirements from Chapter 2, this section introduces two metaheuristics: An hybrid evolutionary algorithm and a variable neighborhood search.

In the scope of related vehicle routing problems many metaheuristics are evaluated in the literature. Among others, genetic algorithms and variable neighborhood search algorithms are shown to find good solutions (see for example Baker and Ayechew (2003), Tasan and Gen (2012) and Nazif and Lee (2012) for genetic algorithms on the VRP and Mladenović and Hansen (1997), Bräysy (2003) and Kytöjoki et al. (2007a) for variable neighborhood search algorithms on the VRP). Due to the successful use in vehicle routing and in liner shipping network design (see for example Mulder and Dekker (2013) for a genetic algorithm for the LSNDP), we have selected two promising metaheuristics to solve the network design problem: An evolutionary algorithm and a variable neighborhood search. We extend the classic evolutionary algorithm by local search algorithms from the variable neighborhood search (see Mladenović and Hansen (1997)) to support the search space exploration. Thus, it is referred to a hybrid evolutionary algorithm. The second algorithm that has become popular and successful in vehicle routing is variable neighborhood search. These algorithms are presented in the remainder of this section.

To cope with the different aspects in liner network design, first a decomposition approach for both metaheuristics is presented. Afterwards, the fitness calculation approach and numerical results are shown. A new approach to approximate the objective function is presented. It shows promising results on the medium-sized instances.

5.2.1. Decomposition Approach for the Metaheuristics

Chapter 2 presents the aspects of the liner network design problem and Chapter 3 the state-of-the-art solution approaches for the subproblems of the network design. The subproblems considered in this thesis are:

1. Determine liner services (port rotation, vessel type and number of vessels)
2. Assure transit time and embargo constraints
3. Allocate cargo on the network to determine transhipment and bunker costs
4. Ensure a repositioning of empty containers
5. Respect the vessel draft subject to its deadweight
6. Consider capacity types

Table 5.5 shows the subproblems that must be covered by the cargo allocation and the network design problem. Except the determination of liner services and the transit time, each aspect from the network design is also relevant for the cargo allocation and has been integrated into a column generation solution approach (see Chapter 4). Subproblems 3 – 6 are therefore already solved with the previously presented method.

Subproblem	**CAP**	**LSNDP**
Determine liner services (port rotation, vessel type and number of vessels)		✓
Assure transit time and embargo constraints		✓
Allocate cargo on the network to determine transhipment and bunker costs	✓	✓
Ensure repositioning of empty containers	✓	✓
Respect the vessel draft in respect to its deadweight	✓	✓
Consider capacity types	✓	✓

Table 5.5.: Subproblems of the cargo allocation presented in Chapter 4 and liner shipping network design problem.

The previous Section 5.1 shows that the subproblems one and two are the main challenges of the network design problem. Therefore, in the scope of this thesis, a decomposition approach of the liner shipping network design problem is used (see Figure 5.8).

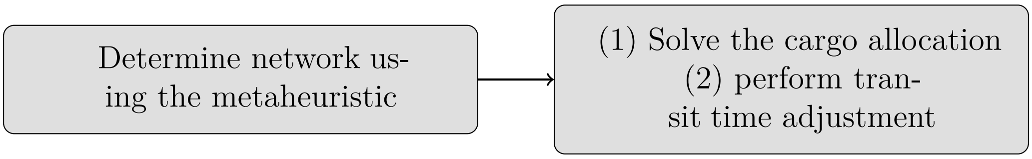


Figure 5.8.: Decomposition of the overall liner shipping network design problem.

A metaheuristic is used to select and modify networks (i.e. the port rotation, vessel type and number of vessels) using implementation specific operators.

Then, one or more networks (depending on the metaheuristic) are evaluated in step (1) by solving the cargo allocation problem. Recall that the solution of this optimization problem provides the duration at ports and at sea. Therefore, the second part of the evaluation is to evaluate embargo and transit time constraints (see step (2) in Figure 5.8). Service speeds can be increased to hold the transit times but also lead to increased bunker prices. Under some circumstances, the resulting schedule is not valid anymore. Therefore, a reparation of the schedule and, if necessary, an invalid solution must be penalized.

The right hand side of the decomposition in Figure 5.8 is referred to as the *fitness* of a solution for the further use in metaheuristics. The following sections show the fitness calculation in more detail.

5.2.2. Determine the Fitness of a Solution

The fitness (i.e. the objective function value) of a solution (i.e. a liner shipping network) is determined as follows:

1. Solve the cargo allocation problem
2. Try to increase the speed on incident legs to hold the transit times
3. Penalize transit times that cannot be held in the network
4. Penalize embargo constraints

The cargo allocation problem solution provides durations at the ports and at sea. If a transit time between two port does not hold, the speed is increased on a per leg basis (if possible). Afterwards, if necessary, the solution is penalized if the speed is not sufficient to hold the requirements. Finally, the solution validates whether embargo constraints are invalid.

To explain the implications of transit times on the cargo allocation, see Figure 5.9. Two services are shown, one with the port rotation $p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow p_1$ and another with $p_2 \rightarrow p_4 \rightarrow p_2$. After solving the cargo allocation problem (see Chapter 4), the services' vessels remain at a port for a determined duration $d^P_{i,l,s}$. Additionally, the cargo allocation model might have introduced an artificial port buffer $b^P_{i,l,s}$ to

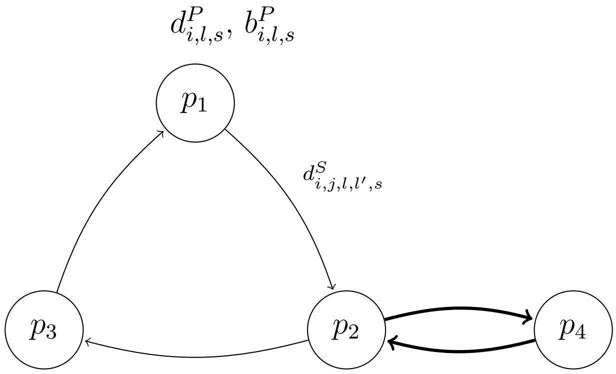


Figure 5.9.: Durations resulting from the cargo allocation problem, for example given at one port and one leg.

compensate for badly designed networks with too many deployed vessels that would steam below its minimum speed. To hold the constant round trip time of each service, the CAP sets the vessel's leg independent speed k_s and thereby the duration at sea to $d^S_{i,j,l,l',s}$.

The solution from the cargo allocation provides the duration at all ports and all legs, for example given for one port and leg in Figure 5.9. If cargo is routed from its origin i to its destination j, the transit time $\theta_{i,j}$ between the origin and destination must hold. Given all durations by solving the cargo allocation problem, the minimum duration to route a cargo is calculated by taking the shortest path from i to j on the given network. In Figure 5.9, the shortest path for cargo between p_1 and p_4 is the path $p_1 \rightarrow p_2 \rightarrow p_4$. The durations are fixed, because the transit time is not incorporated into the cargo allocation problem. If the duration is larger than the required transit time $\theta_{i,j}$ between port i and j different actions can be performed:

1. Directly impose a penalty cost for the difference between the required and the actual transit time

2. Set the cargo flow volume between port i and j to zero because the constraint does not hold

3. Adjust the average speed of the overall service

4. Adjust the speed on the legs incident to the shortest path between port i and port j

Each action has advantages and disadvantages. Directly imposing penalty cost would speed up the fitness function in terms of computation time but make it more difficult for the metaheuristics to find feasible solutions because the CAP is independent from the transit times. This means, that evolved networks might not match all transit times. Another disadvantage is the problem of actually finding valid solutions. Tests indicate that the networks are systematically reduced, and afterwards expanded again until a valid network is found. The second method requires the adjustment of the durations at the ports, legs and buffers and is not considered in the scope of this thesis.

The third method can simplify the solution finding process. Once the CAP identifies a profit optimal network, the overall speed can be increased on the whole service or single legs. In other words, the flexibility of the resulting cargo allocation can be used to increase the overall networks speed. The main drawback of this approach is the resulting invalid schedule timing: If the average (or leg specific) speed of a service is increased and thereby the duration at sea decreased, its schedule becomes invalid because the service's round trip time stays constant. In case of adjusting the whole service speed, an additional artificial buffer must be introduced at the service's ports. This can lead to transit time violations because the time at the ports increase, making it again difficult to create valid solutions.

In this thesis, the fourth method is used to deal with the transit time constraints which relaxes the leg independent speed per service assumption. This allows reparation methods to use both, unaffected legs' sea duration and port durations to fix the services' schedules. This method is visualized in Figure 5.10 for the example presented above. The problem instance might have imposed a transit time constraint between port p_1 and p_4 of $\theta_{1,4} = 5$ days. In the example, the only route between port p_1 and p_4 is the path $p_1 \rightarrow p_2 \rightarrow p_4$, with the incident services s_1 and s_2. For simplicity of the example, we assume that the duration of the path is six days using service independent speeds $k_{s_1} = 10kts$, $k_{s_2} = 12kts$ resulting from the CAP. The duration can be decreased by increasing the speed on leg p_1, p_2 of service s_1 and leg p_2, p_4 of service s_2 until the transit time is met or the maximum speed is reached. For simplicity, we have set the speed to $k_{p_1,p_2,1,1,s_1} = k_{p_2,p_4,1,1,s_2} = 15kts$ so that the duration is 5 days now.

Unfortunately, the round trip durations of the overall service are not met anymore when modifying the legs' speeds. A repair procedure must either decrease the speed on the remaining legs or increase the buffer time at port p_3 or p_4. Due to the cubic bunker cost, decreasing the speed down to the minimum is preferable, then, the new bunker costs are applied to the solution. Finally, if the transit time could not be reached due to the maximum speed constraints, penalty costs are imposed on the fitness function.

Algorithm 6 formalizes the method described above. The functions *incSpeed* and *repairSchedule* are described in more detail in the next sections. The algorithm is initialized by setting the leg independent speed to each leg. All transit times are evaluated where at least one container is routed from the transit time's origin to the destination. The reason is that in case a network does not offer a connection between the origin and destination or no cargo flows are served, there is no need for

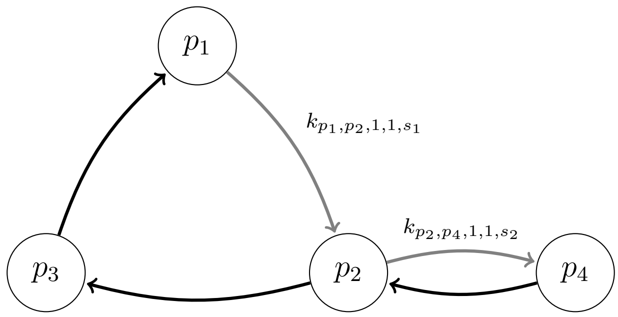


Figure 5.10.: Incident service legs for a transit time requirement between port p_1 and p_4 are marked gray. Service legs that have to be adjusted are marked black.

Algorithm 6: Speed adjustment process to hold transit times.

Input: A liner shipping network NW and its cargo allocation problem solution, specifically the port durations due to pilotage, unloaded and loaded cargo flows and fixed strategic port buffer $d^P_{i,l,s}$, the artificial buffer time $b^P_{i,l,s}$ and the speeds per service k_s

Output: Network evaluation that holds the transit time if possible

$k_{i,j,l,l',s} = k_s \; \forall s \in S^O, (i,j,l,l') \in L_s$;
forall the $(i,j) \in T : \exists n \in N : o_n = i \wedge d_n = j \wedge \alpha_n > 0$ **do**
    $p_{i,j} = dijkstra(i, j, d^P_{i,l,s}, b^P_{i,l,s}, k_{i,j,l,l',s})$;
    if $p_{i,j} = nil$ **then**
        continue;
    end
    $d_{i,j} = duration(p_{i,j})$;
    if $d_{i,j} > \theta_{i,j}$ **then**
        $k_{i,j,l,l',s} = incSpeed(d_{i,j} - \theta_{i,j}, k_{i,j,l,l',s}, p_{i,j})$;
    end
end
$k_{i,j,l,l',s}, b^P_{i,l,s} = repairSchedule(d^P_{i,l,s}, b^P_{i,l,s}, k_{i,j,l,l',s})$;
$applyCosts(k_{i,j,l,l',s})$

the carrier to hold the transit times. Afterwards, the shortest path subject to the sea, port and transhipment durations is calculated and stored in the variable $p_{i,j}$. For the whole path, $d_{i,j}$ stores the duration in days. If the duration cannot be held, the speed of all service legs on the path p is increased if possible (see next section). Finally, the algorithm tries to repair the schedule and applies the new bunker cost to the solution.

Increase Vessels' Speed

Increasing the vessels' speed to meet transit time requirements can impose large bunker costs. Due to the cubic consumption function, it is advantageous to first increase the speed on the legs with lower speed. In Figure 5.11, the example legs incident to the shortest path for a transit time are given. In the first row, the speeds equal the results from the cargo allocation problem and are identical over all service legs. We try to respect the transit time by starting to increase the speed uniformly on the legs with the lowest speeds (those of service s_2 in the figure).

When the speed is increased further, other service leg speeds equal the ones just increased (see Figure 5.11, row two). Then the legs of service s_1 and s_2 are increased until the next higher service speed with 14 kn is reached. As indicated in the last row, all but the first service's vessel type's maximum speed is 14 kn. Thus the speed cannot be increased any further. The algorithm terminates if either all incident service legs are at their maximum speed upper bound or the transit time requirement holds for the new durations.

Repair the Proforma Schedule

Algorithm 6 increases the speed per leg, possibly making the proforma schedule invalid. The following linear optimization model repairs all schedules of a network. The main idea is to slow down the vessels on the legs that are not incident to any transit times. The remaining gap to the service's round trip time (RTT) is filled by reducing artificial port buffers (if available) from the cargo allocation solution.

Sets

S^O	Services operated by the carrier.
P_s	Ports visited by service s.
L_s	Legs used in service s.

Parameters

$d^P_{i,l,s}$	CAP durations in days due to cargo flows, strategic port buffer and pilotage at port i of service s
$b^P_{i,l,s}$	CAP buffer duration in days at port i of service s

$d^{Min}_{i,j,l,l',s}$	Leg duration in days when steaming with adjusted speed according to algorithm 6. $d^{SO}_{i,j,l,l',s} = \frac{l_{i,j}}{max\{k^{Max}_{VT_s}, k_{i,j,l,l',s}\} \cdot 24}$
$d^{Max}_{i,j,l,l',s}$	Maximum duration in days when steaming with minimum speed. $d^{Max}_{i,j,l,l',s} = \frac{l_{i,j}}{max\{k^{Min}_{VT_s}, k_{i,j,l,l',s}\} \cdot 24}$

Variables

$\tau^B_{i,l,s} \in \mathbb{R}^+_0$	Artificial port buffer in days at port i's lth visit in service s.
$\tau^S_{i,j,l,l',s} \in \mathbb{R}^+_0$	Duration at sea on service s leg $(i,j,l,l') \in L_s$ in days.
$\tau^P_s \in \mathbb{R}^+_0$	Penalty duration for service s if the round trip cannot be held

FIXSDLE

$$max \sum_{s \in S^O} \left(\sum_{(i,j,l,l') \in L_s} \tau^S_{i,j,l,l',s} - \tau^P_s \right) \tag{5.52}$$

$s.t.$

$$RTT_s = \sum_{(i,l,s) \in P_s} \left(d^P_{i,l,s} + \tau^B_{i,l,s} \right) + \sum_{(i,j,l,l',s) \in L_s} \tau^S_{i,j,l,l',s} + \tau^P_s \quad \forall s \in S^O \tag{5.53}$$

$$d^{Min}_{i,j,l,l',s} \leq \tau^S_{i,j,l,l',s} \leq d^{Max}_{i,j,l,l',s} \quad \forall s \in S^O, (i,j,l,l') \in L_s \tag{5.54}$$

$$0 \leq \tau^B_{i,l,s} \leq b^P_{i,l,s} \quad \forall s \in S^O, (i,l) \in P_s \tag{5.55}$$

Objective (5.52) maximizes the durations at each leg, i.e. steam as slow as possible. Constraints (5.53) ensure that after solving the model, all of the network's services are valid, if possible. Constraints (5.54) set the bounds for the duration at sea. The duration's lower bound is the duration after adjusting the speeds. If a lower duration (higher speed) would be possible, the transit times can be violated again. Note that the lower bound originates from the speed selected by the cargo allocation problem that allows speeds larger than the maximum speed for badly designed networks. The upper bound is the duration when steaming at minimum speed. The bounds on the port buffer in constraints (5.55) are either zero or the artificial buffer that has been introduced by the cargo allocation.

Model *FIXSDLE* can be solved within fractions of a second. Even in large networks, the upper bound on the rows are the number of services (150 - 200 for the world's largest carrier's network). Assuming 15 legs per service, the model would have up to 3000 columns (continuous variables).

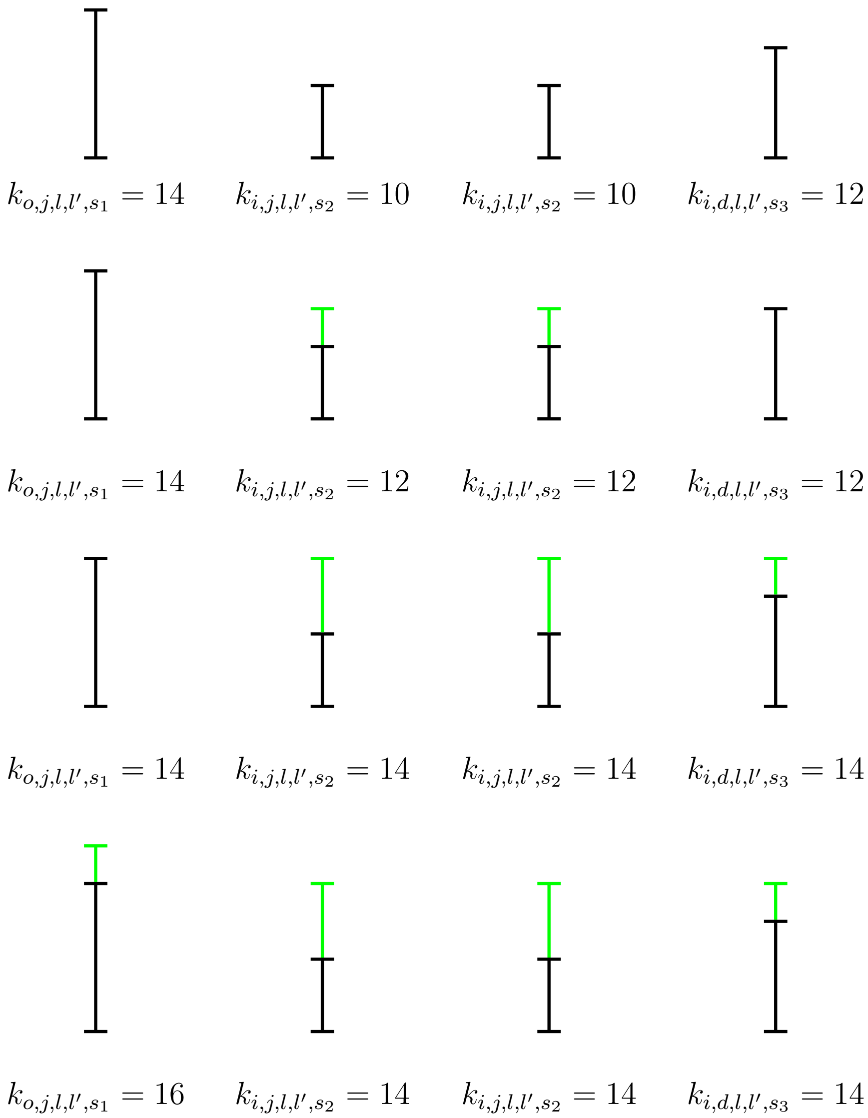


Figure 5.11.: First row: Initial, leg independent speeds (in kn) per service after solving the cargo allocation problem for four legs incident to the shortest path for transit time $\theta_{o,d}$. Succeeding rows: Increasing leg dependent speeds per service until transit times are met or maximum speed is reached.

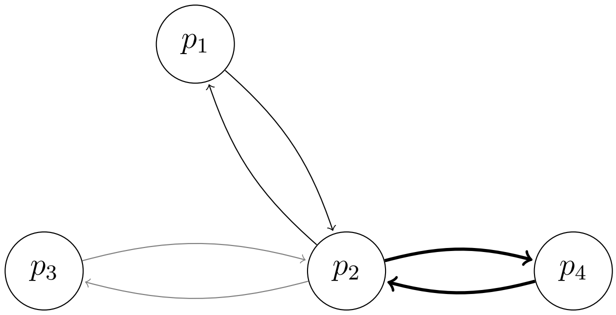


Figure 5.12.: Random network that uses pendulum services.

The presented approach is used to calculate the fitness for solutions (liner networks) within metaheuristics. Additionally, it can be used to evaluate the network changes of liner network planners, for example by providing information on invalid transit times.

5.2.3. Construction Heuristics

In this section, two construction heuristics to create initial networks are presented: A pendulum service heuristic and a clustering heuristic. To the best of our knowledge, these heuristics have not been presented in literature. The main challenge of these construction heuristics is to provide solutions as good as possible in a reasonable amount of time. For liner shipping networks, different aspects describe the performance of initial solutions: First, the geographical coverage should be large enough such that all trade regions are already served. Furthermore, the solutions should be valid regarding the port drafts and transit times, and, finally, they should lead to a reasonable profit.

Pendulum Service Heuristic

A simple method to create liner networks is to create services with pendulum routes that contain only two ports. Because liner services are usually aligned with cargo flows, services use legs where cargo flows exist. The cargo flows can be either taken randomly or based on their possible revenue. After the alternating ports for a new service are identified, a vessel type whose lightship draft holds the depth of the ports is selected. The number of deployed vessels are selected based on the design speed and the pilotage, buffer and constant port call duration of 24 hours per port. In Figure 5.12, an example initial network with three services on four ports is shown.

The advantage of this construction heuristic is that networks can be created very quickly and a large variety of ports are called, which leads to increased geographical coverage.

However, the solution approach has several fundamental disadvantages:

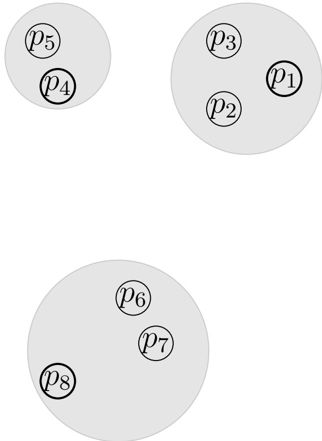


Figure 5.13.: Clustering with three regions used for construction heuristic. The reference ports in each region are marked bold.

- The networks impose large fixed costs for the vessels
- Beneficial transhipment operations to extend the cargo flow coverage are only performed at random ports
- Invalid networks are possible due to high transit times when serving cargo along different services

The construction heuristic presented in the next section covers the disadvantages of high fixed costs, improves transhipment operations and reduces the risk of invalid networks due to the transit times.

Clustering Heuristic

The clustering heuristic basically works in two steps: First, an initial network that serves the largest ports within determined regions is created. Second, the created services are extended locally with additional ports. The heuristic is outlined in Algorithm 7.

Figure 5.13 shows the result of the first heuristic's step of clustering the ports, see line 1 in Algorithm 7. A parameter of the heuristic is to select the number of regions (*cc*) in the problem instance. The k-means clustering algorithm (Hartigan and Wong (1979)) is used in this thesis to cluster ports into regions using the L2 distance metric. For each cluster (or region), the port with the largest depth is selected as a representative. This port is used to ensure that large vessels can enter the region. Afterwards, a random list of region combinations is used to create initial pendulum services (see Algorithm 7, line 9). A bound for the required vessels is determined by dividing the weekly cargo flows by the average vessel capacity (its

Algorithm 7: Clustering heuristic to create an initial liner shipping network design.

Input: Problem instance data, ports P, demands N, cluster count cc, service count sc

Output: A liner shipping network design, ND

1 $C \leftarrow KMeans(P, cc)$;
2 $R \leftarrow \emptyset$;
3 $TSP \leftarrow \emptyset$;
4 **for** $i = 0; i < |C|; i++$ **do**
5   **for** $j = 0; j < |C|; j++$ **do**
6         $R \leftarrow R \cup \{(GetRepresentative(C[i]), GetRepresentative(C[j]))\}$
        $TSP \leftarrow TSP \cup \{GetRepresentative(C[i]), GetRepresentative(C[j])\}$;
7   **end**
8 **end**
9 $ND \leftarrow CreateInitialServices(R, sc)$;
10 $f = Evaluate(ND)$;
11 **for** $p \in sort(P \setminus TSP)$ **do**
12     $ND' \leftarrow AddPort(ND, p)$;
13     $f' = Evaluate(ND')$;
14   **if** $f' > f$ **then**
15         $ND \leftarrow ND'$;
16         $f = f$;
17   **end**
18 **end**
19 **return** ND;

total nominal slots). Until this upper bound is reached, pendulum services between two random regions are created. These services provide the basic network design.

In the second step of the heuristic in line 10, the network is evaluated by a cargo allocation approximation described in Chapter 4. Afterwards, the algorithm tries to insert all unserved ports into the closest service with method *AddPort*. If the network's fitness improves, the port is inserted and the next port is selected. Finally, the best network found is returned. Considering the fitness usually leads to networks that hold the transit time because invalid networks have a negative fitness.

The presented construction heuristics are used to provide a starting solution (or solutions) for the metaheuristics in the next section.

5.2.4. Improvement Heuristics

In the scope of the liner shipping network design problem, several metaheuristics have already been used successfully (see Chapter 3). This section proposes a hybrid evolutionary algorithm that combines the default approach with local search methods. Then, a novel variable neighborhood search heuristic for the liner shipping network design is presented.

Hybrid Evolutionary Algorithm

The general concept of a genetic algorithm, introduced by Holland (1992) is described in Chapter 3. In the scope of this thesis, the algorithm is extended to cope with the liner shipping network design problem. The extensions are:

- Problem specific crossover operators
- Mutation operators
- Local search method

The algorithm is based on a population of solutions (individuals). The population evolves according to a fitness function. The algorithmic approach presented in Section 3.1.3 is extended by custom crossover selection strategies and operators, elitism, population variation operators and local search operators. These adjustments to the basic version are required to deal with the complex problem of optimizing liner networks. The hybrid evolutionary algorithm is described in Algorithm 8.

Each individual of a population is represented as an object in the memory. The reason not to choose the classical chromosome-representation used in genetic algorithms is the disadvantage of handling and validation. An individual is a liner network design for a specific problem instance basically with a set of liner services. Each liner service defines an ordered set of legs between ports, has a vessel type attached and deploys a specific number of vessels. Each object has several properties, such as a liner service's route type or utilization. The properties can be used to simplify the development of reporting and validation components. For each individual, a method is called that evaluates the fitness function according to the approach presented in Section 5.2.1 and determines how good a solution is compared to others. The fitness is returned as a real value that can be negative.

In the scope of this thesis, two crossover selection strategies are evaluated (see line 7 of Algorithm 8): A binary tournament (BT) and roulette (wheel) selection (RS). Binary tournament selects two parents for the crossover by selecting the best of four randomly chosen parents. Roulette selection selects parents by selecting them according to their unified fitness distribution. For more information on tournament selection see Srinivas and Patnaik (1994) and Beasley and Chu (1996) for roulette selection. Both are used to create an offspring based on two relatively good parents.

The crossover method decides how to combine the parents to the offspring, in analogy to the child's chromosomes based on its parents. The crossover method is a core component of an evolutionary algorithm, because it must ensure diversity of

Algorithm 8: Hybrid Evolutionary Algorithm for the liner shipping network design problem.

Initialize population P_0 using the Pendulum and Clustering heuristic;
Initialize evaluations $E = Evaluate(P_0)$;
Initialize iteration $i = 0$;
while *runtime* $<=$ *MaxRuntime* **do**
    Initialize offspring $O \leftarrow \emptyset$;
    for $i < (|P_i|/2)$ **do**
        $(A, B) = SelectParents(P_i)$;
        $O \leftarrow O \cup \{\ Crossover(A, B)\ \}$;
    end
    for $o \in O$ **do**
        for $m \in M$ **do**
            if *mutation condition satisfied* **then**
                $o \leftarrow Mutate(o, m)$;
            end
        end
        $E \leftarrow E \cup \{EvaluateFitness(o)\})$;
    end
    $P_{i+1} \leftarrow SelectPopulation(E, P_i, O)$;
    $i \leftarrow i + 1$;
end
return *BestNetwork*

solutions and hold common properties. In the scope of this thesis, three operators are implemented and tested:

1. Cycle crossover (CY) (for an application of the vehicle routing problem, see for example Michalewicz and Fogel (2004)),
2. Informed cycle crossover (IC),
3. Clustering crossover (CL).

The cycle operator creates offspring by counting the number of services in each parent and randomly selects services from each parent. Thus, the structure of the parents is kept in the next population. The informed cycle operator can additionally extract ports from single services and join them with the best position of a random service in the offspring. The third operator identifies the served regions of the two parents, loops through a random list of region combinations and selects a random set of services between these regions. With a low probability, a service can be skipped in this process, otherwise the network would be expanded too quickly.

After the offspring are created, random mutations are applied to the solutions (see line 13 in Algorithm 8) with a specific probability. The operators used in this thesis are:

1. Delete port
2. Delete service
3. Insert port sequence
4. Move port sequence between services
5. Insert pendulum service

Results indicate that very small changes in the solutions can lead to large changes in the fitness. To reduce the negative impacts of modified networks, local search strategies are applied to solutions. Strategies that only rely on distance measures, such as a 2-opt operator (see Croes (1958)), have only worked well on small instances because they do not consider the transit times. Therefore, a variable neighborhood descent (VND) has been implemented to improve the networks after the mutation. The VND is applied to a percentage of the population's individuals and is described in detail in the next section.

The succeeding population is a subset of the parents and the created offspring. To limit the size of the population, one has to select a subset of all individuals that continue to the next population (see line 18 of Algorithm 8). In the scope of this thesis, the best *ELM%* of individuals are selected that have distinct fitness values. This strategy is called *elitism*. To allow other individuals to survive this selection process and evolve further to diversify the population, the remaining population is tried to be filled with random individuals with distinct fitness. Taking the distinct fitness into account allows to diverse the population. One should not select too many similar individuals because this can lead to local minima. After a fixed number of iterations, random individuals (created by the construction heuristics) are injected into the population to diversify it and avoid premature local optima.

Variable Neighborhood Search

Variable neighborhood search (VNS) algorithms have been successfully used for solving vehicle routing problems in recent years (see for example Hansen and Mladenović (2003)).

Summarizing, VNS starts with an initial network and randomly modifies (called shaking in VNS terms) it to generate a new neighborhood solution (see Algorithm 9). Then, the best network within a specific range of possible changes is searched. The process of modifying and performing a local search is repeated until a stopping

Algorithm 9: Variable Neighborhood Search for the liner shipping network design problem using a metropolis criterion.

1 Select set of neighborhood structures $N_k^{'}$ for $k = 1, \cdots, k_{max}^{'}$;
2 Create initial solution i using clustering heuristic;
3 **repeat**
4     $k = 1$;
5   **repeat**
6         $x' = Shake(N_k(i))$;
7   Apply local search to x'' =VND(x');
8   **if** $rnd > exp\left(\frac{-(f(x'')-f(x'))}{T}\right)$ **then**
9             $i = x''$;
10             $k = 1$;
11   **else**
12             $k = k + 1$;
13   **end**
14   **until** $k = k_{max}^{'}$;
15 **until** *runtime > MaxRuntime*;
16 **return** $BestNetwork$

criteria is matched. While no improving solutions are found, shaking operators increasingly modify the solution to reduce the risk of local minima.

Several problem specific adaptions must be performed to apply the VNS to the network design in liner shipping, see Algorithm 9. The key adjustments are the fitness function and the neighborhood structures. VNS implementations rely on good local search algorithms that help to identify the best solution within a neighborhood. In the scope of the vehicle routing problem, the 2-opt local search (see Croes (1958)) has been used in variable neighborhood search (see Chen et al. (2010) and Kytöjoki et al. (2007b)). This thesis uses the Variable Neighborhood Descending (VND) algorithm proposed by Mladenović and Hansen (1997) which is adapted to the liner network design problem and used in the VNS (see line 7 in Algorithm 9). The algorithm successively uses different operators, called neighborhood structures, in a specific order. If the solution improves, VND restarts with the first neighborhood structure. The algorithm terminates when a given runtime is reached and no neighboring solution lead to any improvement. The general concept is presented in Chapter 3. For details on the implementation of the structures in the LSNDP and their sequencing, see Hilger (2014). VND relies on the neighborhood structures presented in Table 5.9.

The operator *ChangeSinglePort*, which moves a single port between two services, is successfully used in vehicle routing problems as well (see Hemmelmayr et al.

Neighborhood Structure	Description
ChangeSinglePort	Changing a service's port
ChangePortSequence	Changing a sequence of ports
ChangeLinerService	Joining two or separating one liner service
ChangeVesselType	Changing a service's vessel type
ChangeVesselCount	Changing the number of deployed vessels
2-opt	Performs a 2-opt local search based on the services' distance and cargo flows

Table 5.9.: Neighborhood Structures used in Variable Neighborhood Descending.

(2009b)). Due to the vast solution space, several domain specific operators must be implemented. These operators rely on the preferable high utilization (see Agarwal and Ergun (2008)) and low bunker cost (see Wang and Meng (2012c)) in good liner networks.

The neighborhood structure *ChangePortSequence* defines all neighborhoods of a given network that can be created by moving a defined port sequence of service A to another service B. Similar operators are used for the vehicle routing problem as well (see Hemmelmayr et al. (2009b)). The operator selects the port sequence based on the leg utilization and the geographical distance of the ports. It is assumed that sequences with low utilization should be removed from a service and inserted into another service with less than average utilization. Feasibility of the modified services must be ensured by adjusting the vessel types due to draft limitations and increasing/decreasing the number of vessels to ensure a weekly round trip time based on the design speed.

The operator *ChangeLinerService* merges two services or splits one liner service into two. The operator is required to enable the extension or contraction of networks. A good selection of the splitting point can be achieved by selecting geographically close ports. The operator ensures the connectivity of the network by calling a port from both services that are created by the split-operation. In practical network most of the services are connected with each other to allow transhipments. Exceptions such as pendulum-services and networks with only one service must be considered in the operator as well.

The problem specific neighborhood *ChangeVesselType* changes the vessel type of one existing service in the network design. The operator is based on a preferring high service utilization. Services with low average utilization probably have a vessel type deployed where the capacity is too large for the transported cargo flows. Thus, the operator selects a smaller vessel type. When using a larger vessel type, the operator must ensure that all served ports' depths are not smaller than the new vessel's lightship draft.

The neighborhood *ChangeVesselCount* defines the neighborhood of all networks that can be created from a given network by increasing or decreasing the number of deployed vessels for all services. Services that steam with a high speed (obtained through the cargo allocation problem) may deploy another vessel to decrease the overall service speed and thereby decrease the bunker costs of this service. Similar, services that deploy too many vessels typically steam at their minimum speed and may use artificial port buffer to obtain the constant round trip time (recall that weekly services are assumed throughout the network design). For these services, decreasing the number of vessel may reduce the time charter cost of the service.

The last operator performs a 2-opt local search on all liner services in the network based on the distance and the transported cargo flows. The distance metric is important because ports within a region should be served together. However, cargo flows should be preferable routed on its shortest path from the origin to its destination. Thus, the 2-opt swap operation considers both the distance and the legs with high volume to perform the swap operations.

The described neighborhood structures can be called in arbitrary order. Hilger (2014) performs an analysis of different sequences and concludes that the differences are rather small. Nevertheless, the sequence presented in Table 5.9 performs best on the learning instances and is thus used in the remainder of this thesis.

Another method to improve the convergence of variable neighborhood search algorithms is the use of a Simulated Annealing (Kirkpatrick (1984)) acceptance criteria (see line 9 in Algorithm 9). This extension has been successfully used in vehicle routing problems (see for example Hemmelmayr et al. (2009b) and Chen et al. (2010)). Numerical results indicate that the use of the acceptance criteria lead to slightly improved convergence behavior in some instances of the LINER-LIB and is used in the remainder of this thesis to diversify the search algorithms.

Numerical experiments performed by Hilger (2014) indicate that larger liner network instances require a VNS shaking step that diversifies the neighborhood more than in smaller instances. Therefore, the implemented shaking procedure executes the *ChangeSinglePort* and *ChangeLinerService* operators with a probability of each 35%. Otherwise, no change on the network is done and the improvement is moved to the descending step.

5.2.5. Numerical Results

The proposed metaheuristics are evaluated in this section. First, the results for the hybrid evolutionary algorithm are presented. A detailed parameter tuning for the evolutionary algorithm can be found in Appendix D.2. Then, the results for the variable neighborhood search (VNS) are shown. Details on the VNS parameter tuning are presented in Hilger (2014).

Evolutionary Algorithm

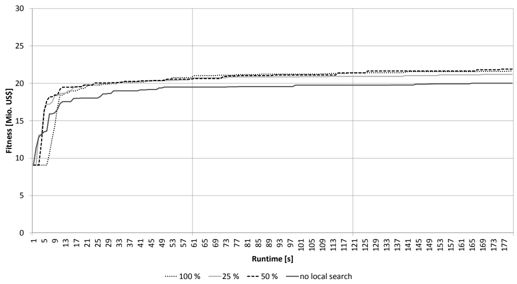
To simplify the evaluation of the evolutionary algorithm, the numerical results for the basic parameters are moved to Appendix D.2. The parameters listed in Table 5.10 are tested one after the other. Tools that automatically determine good parameters rely on many instances, preferable hundreds (see for example Ansótegui et al. (2009)), and are thus not applicable to the LSNDP using the LINER-LIB instances due to overfitting. The default values listed in Table 5.10 are evaluated on the WAF and Mediterranean instances (see Appendix D.2). Further information on the parameters is given in Section 5.2.4.

Parameter	Abbreviation	(Default) Values
Instance	I	Baltic, **WAF**, **Mediterranean**, Pacific, EuropeAsia, WorldSmall, WorldLarge
Runtime	RT	Instance dependent, 3, **5**, **20**, 30, 240, 240, 240 minutes
Crossover method^A	CX	**Informed Cycle (IC)**, Clustering (CL) and Cycle (CY) operator
Crossover select^A	CXS	**Binary Tournament (BT)**, Roulette Selection (RS)
Population size^A	PZ	10, 20, **50**
Pendulum%^{A}	PDL	0%, **10%**, 50%, 100%
Local search probability *	LSP	**0%**, 20%, 50%, 100%
Elitism^A	ELM	10%, **20%**, 50%
Mutation rate^F	MR	**1%**, 2.5%, 5%
Network injection^A	NI	**10**, 20, 50

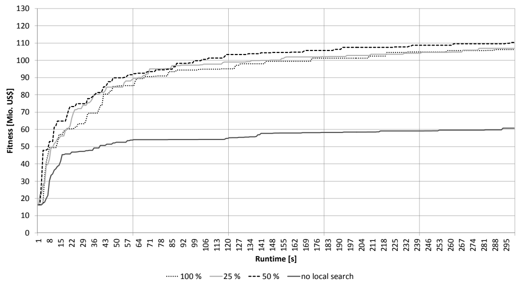
Table 5.10.: Parameters for the hybrid evolutionary algorithm. Parameters marked with A are evaluated in Appendix D.2. The local search probability is analyzed in this section. The bold parameters have been selected by the manual parameter tuning.

The following assumptions are made to simplify the algorithm evaluation for the LINER-LIB: First, no additional port buffer is used. This means, that the port duration is only determined by the pilotage and the duration resulting from the cargo movement. Second, the tolerance for the transit times is set to one day and no transhipment port duration is added. To deal with the randomness in the evolutionary algorithm, each configuration's result is averaged over five runs, always using five identical initial networks for different configurations.

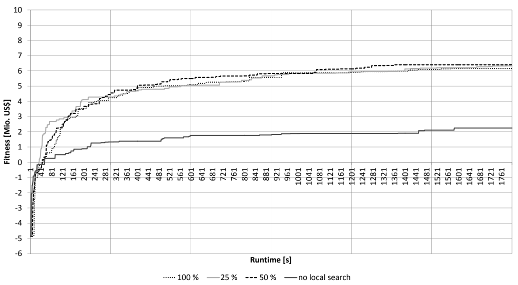
The EA uses a binary tournament selection strategy and the informed crossover operator. A medium population size of 50 individuals per generation is used. The initial population not only consists of networks created by the clustering heuristic, but also 10% pendulum service networks to diversify the population. To obtain convergence of the overall population, in every iteration the 20% best individuals



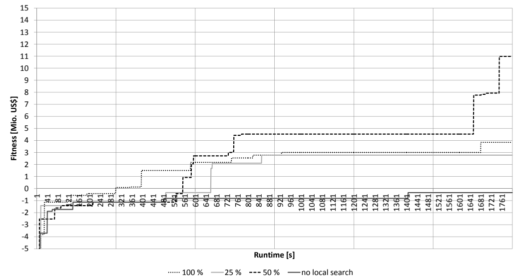
(a) Baltic instance.



(b) WAF instance.



(c) Mediterranean instance.



(d) Pacific instance.

Figure 5.14.: Convergence of the evolutionary algorithm in different LINER-LIB instances using the local search probabilities 0%, 25%, 50% and 100%.

(of the offspring and the parents) are selected for the next generation (referred to as an elitism strategy), and every 10 iterations without any objective improvement 25 random networks are added to the population and selected according to the elitism strategy.

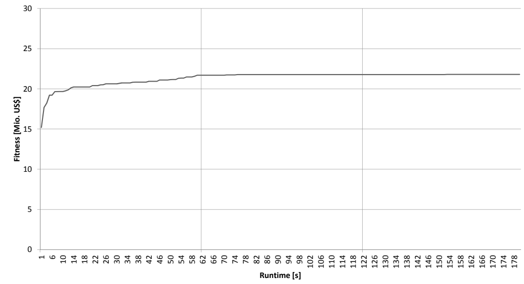
The main challenge of the population based algorithm is avoiding local minima by managing the randomness of the generation to get sufficient diversity. If too much randomness is introduced, the networks' fitness fluctuates too much. Thus, the algorithms decide to shrink the networks to decrease the costs. Not until a sufficient stable small network is found does the algorithm start to increase the network size. Thus, a relatively small mutation rate of 1% is used in the EA. Furthermore, it is important to start with relatively good, feasible, large networks to avoid the initial shrinkage phase of the algorithm. Its runtime is limited to 30 seconds, although the phase is often finished in less time for the small and medium sized instances. VND is executed in parallel for many individuals which reduces the runtime (see Hilger (2014)).

In Figure 5.14 the effects of different local search probabilities on the convergence is presented for four LINER-LIB instances. In all tested instances, the algorithm's convergence is worst when not using any local search algorithm. In the smallest instance, the improvement using a local search is relatively small (see Figure 5.14(a)). The best fitness found within three minutes is about 10% higher when applying a local search to 50% of the offspring. In contrast, the larger instances highly benefit from the local search (see Figure 5.14(b) - 5.14(d)). The best fitness found for the WAF instance is approximately 80% better and the Mediterranean's about 170%. On average, the algorithm cannot create a profitable network for the Pacific instance without the local search in 30 minutes. Using the local search with a low probability leads to profitable networks after five minutes and after nine minutes when using a medium local search probability. This can be explained by the time needed to execute VND for the individuals. The more often the VND is executed, the less iterations are performed by the hybrid evolutionary algorithm. This explains why performing a local search on only half of the population is better in all tested instances compared to running VND on all offspring. The local search probability is set to 50% for the remaining numerical results.

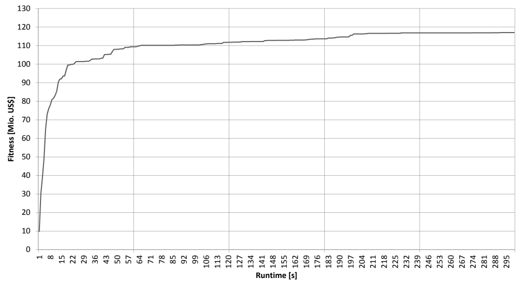
The LSNDP is by its nature very resilient to changes in its solution. For example changing the port rotation by repositioning one port cannot only lead to reduced profit, but also to invalid transit times and thus high penalty cost. Furthermore, transhipment operations may become highly ineffective because the cargo has be transported on many legs of the round trip. Removing a port can lead to the breakdown of most of the cargo flows, because a major transhipment port is required to distribute the cargo between different regions. Especially in the medium sized instances, it becomes important to induce a certain degree of diversification, meaning that not all of the offspring should be optimized by the local search.

Solving the mixed integer program in the previous section led to a lower bound of approximately 23.93 million US$ after 12 hours. The best solution found with the evolutionary algorithm after three minutes is 21.89 million US$ and has a gap of approximately 8.5% to the lower bound of the mixed integer program. This can be explained by the advantage of the mixed integer model of considering the interdependency between serving cargo and respecting the transit time. The evolutionary algorithm finds a network for the WAF instance with a profit of 110.35 million US$ after five minutes, compared to 72.59 million US$ after 24 hours using the mixed integer model, which is an increase of more than 50%. The best network found for the Mediterranean instance is 6.4 million US$, the Pacific instance's is 10.99 million US$. Within 24 hours, for none of the medium sized instances an integer solution could be found using the mixed integer model.

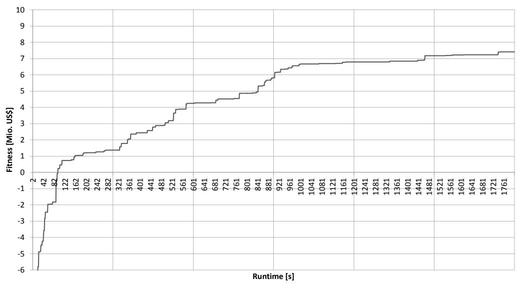
The results indicate that the medium sized instances can be solved using the hybrid evolutionary algorithm. It performs much better on the WAF instance compared to the lower bound resulting from the mixed integer model. However, no information on the solution quality can be stated for the Mediterranean and Pacific instance. Therefore, another metaheuristic is introduced to challenge the results of the hybrid evolutionary algorithm.



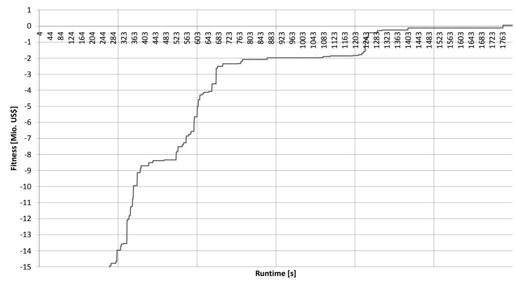
(a) Baltic instance.



(b) WAF instance.



(c) Mediterranean instance.



(d) Pacific instance.

Figure 5.15.: Convergence of variable neighborhood search in selected LINER-LIB instances.

Variable Neighborhood Search

The numerical results for the variable neighborhood search are presented in the remainder of this section. Similar to the evolutionary algorithm, the VNS has several parameters that can be configured. Details on the manual parameter tuning can be found in Hilger (2014). The configuration for the numerical results presented below uses a simulated annealing acceptance criteria and several parallel competing variable neighborhood descending steps. The shaking step is performed based on the *ChangeLinerService* and *ChangeSinglePort* neighborhood structures.

In Figure 5.15, the convergence of the VNS in selected LINER-LIB instances is shown. When compared to the evolutionary algorithm (EA), shown in Figure 5.14, one can observe that the evolutionary algorithm converges slower in the small Baltic and WAF instance compared to the VNS. Furthermore, the best fitness found in the Baltic instance using VNS is about 0.5% worse than the one found with the EA. The best network found for the WAF instance has a profit of 117.01 million US$ which is about 6% better than the best network found with the EA. In the Mediterranean instance, the evolutionary algorithm converges faster and reaches a fitness of close to 6 million US$. However, the best fitness found within 30 minutes is about 13% better using the VNS. In the Pacific instance, the convergence is again slower in the VNS compared to the EA, and results in marginal profitable networks after 30 minutes, whereas the EA has a fitness of about 11 million US$.

We conclude that both algorithms can solve the small instances within several minutes. Comparing the VNS with the EA, the VNS finds similar solutions regarding the profit in the same amount of time for the small instances. However, the differences are within a 6% range and could be explained by the stochastic components of the algorithms. In the Mediterranean instance, the convergence of the EA is faster compared to the VNS. However, the VNS finds an overall much better solution. In the Pacific instance, the EA outperforms the VNS in both the convergence and the best solution found. This can be explained by many individuals that diversify the whole population. The results of the hybrid evolutionary algorithm for the Pacific instance show that it becomes more important to execute local search operators more seldomly because they require a large runtime portion. The experiments indicate that an important factor is the number of iterations that the metaheuristics can perform. For the larger instances it can thus be relevant to further speed up the evaluation to increase the number of iterations.

5.3. Surrogate Extensions to Metaheuristics

The metaheuristics presented in the previous sections rely on a fitness function that is time-consuming to compute. Chapter 4.6 introduces several promising heuristics for the real fitness, such as limiting the number of cargo flows. These heuristic fitness functions can be used in the scope of metaheuristics. Heuristic (or approximate) fitness values are called surrogates (or proxies) and are used frequently in the engineering context (see for example Forrester et al. (2008), Shahrokhi and Jahangirian (2010) and Jin (2011)). In these applications, evaluating a complicated engineering model using simulation approaches is often too time-consuming to be included within metaheuristics. Thus, approximation functions with an adaptive level of accuracy are used. In the context of discrete optimization, the application of surrogate models is relatively rare. Successful applications can be found for example in Gendreau et al. (1996) where a surrogate has been used to quickly evaluate operators within a tabu-search which is an example of an informed operator (see Jin (2011)).

5.3.1. Metaheuristic Metacontrol

One approach to use surrogates is to replace the time-consuming fitness function with an approximation. The results in Section 5.3.2 indicate that this usually leads the heuristic to an optimal solution of the surrogate, called a *false optimum*. This observation is also found by Jin et al. (2000). They, together with Jin (2011), highlight the importance of managing the surrogate models by combining the surrogates with the real fitness function. Jin (2005) divides the strategies of applying surrogates in fixed and adaptive evolution control. Fixed evolution control can be divided into individual and generation based approaches. Individual-based approaches select individuals randomly or fitness-based and afterwards reevaluate them with the real fitness. Generation-based approaches reevaluate the whole population according to a strategy, such as a fixed number of generations.

The adaptive evolution control adapts the surrogate accuracy (or parameters) of the fixed evolution control (such as the number of individuals to reevaluate) during the solution process.

The strategies presented in Jin (2005) apply to evolutionary algorithms but can also be used within other metaheuristics, see Gendreau et al. (1996). Although the concept of using surrogates within metaheuristics is not new, relatively little attention has been paid to it in the context of discrete optimization so far. This can be explained by the following reasons:

1. The theoretical impact on the solution approach has not been analyzed in detail.

2. The impact of the surrogate quality (see Jin et al. (2003)) on the solution process has not been studied in detail.

3. Many problems, such as the distance based vehicle routing problem, provide a compact and easy way to calculate the fitness (objective) function.

Surrogates in discrete optimization problems are useful for problems with time-consuming fitness functions, such as multi-commodity flow problems with design decisions. Especially when no delta-evaluation of the solution changes is possible within a reasonable amount of time, the use of surrogates can help to explore the search space.

In Figure 5.16, a concept to manage different surrogate models in metaheuristics is given as a UML class diagram. Metaheuristics attach to a *MetaControl* class that handles the evaluation of one or more individuals, depending on whether the algorithm is population (genetic or evolutionary algorithms, ant-colony optimization etc.) or individual based (for example VNS, simulated annealing, tabu search etc.). The *MetaControl* must be configured to use a specific strategy to evaluate the solutions. In Figure 5.16, strategies for the fixed evolution control are given that can be either used in individual or population based metaheuristics and operate in one or two phases. The following strategies are implemented and evaluated in the numerical results:

Constant (CS) Evaluates (the/all) solution(s) according to a fixed surrogate.

Figure 5.16.: Concept for a heuristic meta control for population and individual based heuristics.

Fixed Generation (FGS) Evaluates the whole population using the real fitness after a fixed number of generations.

Fixed Population Percentage (FPS) Evaluates a fixed percentage of random individuals using the real fitness, the remaining individuals using the surrogate.

Best Reevaluation (BRS) This two phase strategy first evaluates the whole population using a constant surrogate. Afterwards, the best $b\%$ of the individuals in the population are reevaluated using the real fitness.

The use of surrogates can have the following advantages compared to always using the optimal or real fitness:

1. Speed-up of the fitness calculation and thereby a faster search space exploration.

2. Diversification of the search because the surrogate can lead to false but diversified decisions in metaheuristics.

Although there are advantages of using surrogates, it is reported that their application can lead heuristics to false optima (see Jin (2011)). Surrogates continuously evaluate a solution worse (or better) than the solution actually is. The impact of surrogates on the metaheuristic and surrogate quality measures are analyzed in Jin (2011). In the next section, we apply the (static) surrogate concept to the evolutionary algorithm and the variable neighborhood search. We show, that especially the larger instances in the VNS benefit from the use of surrogates.

5.3.2. Numerical Results

The surrogate concept presented above is applied to the proposed metaheuristics. The results for the hybrid evolutionary algorithm are presented first. The fitness function presented in this section is an optimization algorithm: the cargo allocation problem.

As presented in Section 4, three different approximation parameters exist which are used as surrogates here: reduce the amount of transportable cargo flows (CF), reduce the accuracy of the bunker cost function (SP) and decrease he maximum number of column generation (CG) iterations (It). The transportable cargo flows can be selected using different strategies: In the scope of this analysis we focus on the $cumQ * R$ and $Q * R$. $cumQ * R$ orders the cargo flows by their possible total revenue (quantity times revenue) and selects the cargo until a required percentage X (based on the overall possible revenue) is reached (see Chapter 4). In the $Q * R$ strategy, the cargo flows are ordered descending by their quantity times revenue. The $Q * R$ strategy takes X percent of all cargo flow definition, leading to a higher amount of cargo flows for the larger instance (see Appendix C.1). The surrogate strategies presented in the previous section are evaluated using different surrogate accuracies, denoted as (CF, SP, It). CF is the percentage of cargo flows, SP the percentage of support points (20 support points are referred to as 100%) and It the maximum number of column generation iterations.

Evolutionary Algorithm

The evolutionary algorithm provides several opportunities for static surrogates that are backed by the literature (see Jin (2011)).

The most simple usage of surrogates is to replace the real fitness (the cargo allocation problem) with a surrogate, using the constant surrogate (CS). This method has the largest benefit to speedup heuristics because only the surrogate is used. Figure 5.17 presents the results for this strategy for the WAF LINER-LIB instance for different cargo flow volumes, bunker cost accuracies and column generation iterations. The results are averaged on five runs to reduce the stochastic effects of the metaheuristics.

In general, one can observe that more accurate fitness functions also find better solutions. The number of column generation iterations combined with the cargo flow volume highly influence the quality of the solutions. The configuration with

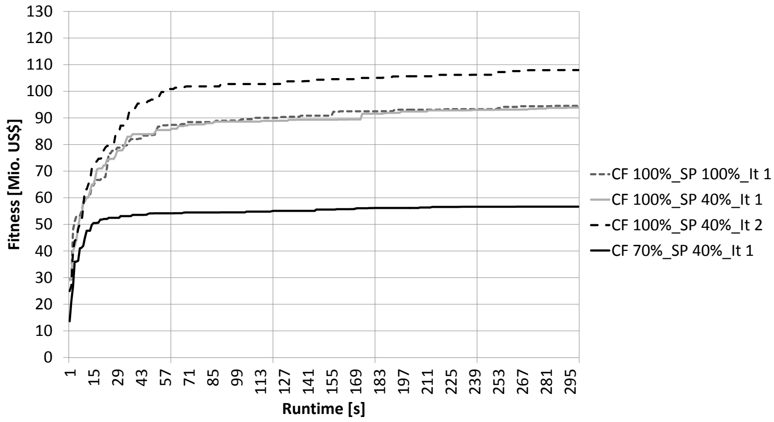


Figure 5.17.: Fitness convergence in the evolutionary algorithm on the WAF LINER-LIB instance using a constant surrogate with different accuracies.

100% cargo flows and 40% bunker cost discretization accuracy (CF 100%, SP 40%, *) performs better when two column generation iterations instead of one are performed. This can be explained by the high utilization of the improved networks that lead to beneficial alternative paths for the cargo flow due to the capacity constraints. On the other hand, modifying the accuracy of the bunker cost discretization does not influence the convergence clearly. Configuration $CF100\%, SP100\%, It1$ behaves almost identical to $CF100\%, SP40\%, It1$. This is backed by the observations in Chapter 4.6, where the bunker cost discretization is relatively good with few support points. Limiting the cargo flows' volume (for example to 70% most profitable cargo) clearly converges to a false global optimum (this also has been observed by Jin (2005)). The reason is that the opportunity to route the remaining 30% is missing and the heuristic focuses on transporting the most profitable cargo only.

The fitness values in Figure 5.17 refer to the surrogate values and are expected to be lower compared to evaluating the cargo allocation problem to optimality, i.e. using all cargo flows and an exact bunker cost function. In Figure 5.18 the average real fitness of the best solutions found after five minutes using the surrogate CS strategy is presented. The reevaluated networks have fitness values similar to those presented in Figure 5.17, whereas the 100% cargo volume and one column generation iteration are very similar. This indicates that the networks are designed to serve the cargo on the cheapest path. Allowing alternative paths does not increase the profit by using more than one CG iteration. Compared to the surrogate fitness, the fitness of configuration $CF70\%, SP40\%, It1$ increased most because more revenue can be gained through the available cargo flows.

To conclude the constant surrogate evaluation, the results are not better compared to the real fitness in the previous section. The evaluation of larger instances is

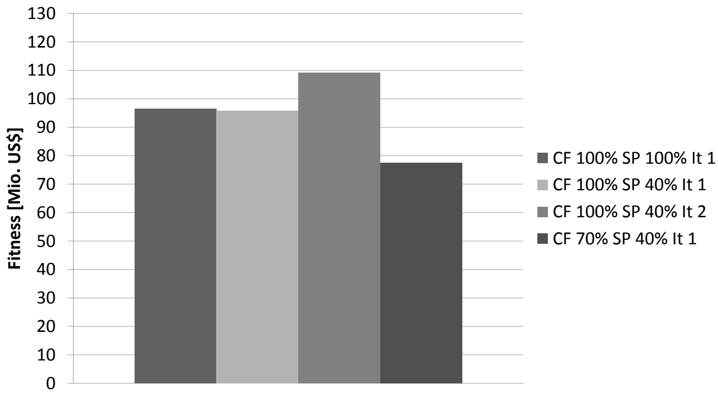


Figure 5.18.: Average real fitness for the best solutions found for different constant surrogates. Note that configuration $CF70\%, SP40\%, It1$ leads to one transit time invalid solution when reevaluated with the real fitness function.

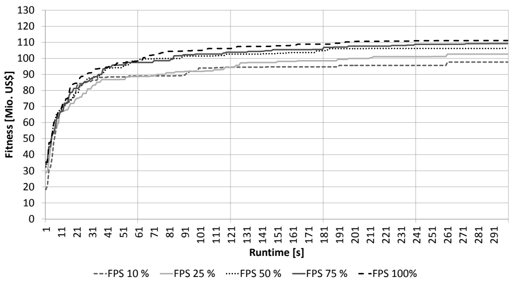
omitted because it is expected that algorithms that only rely on a constant surrogate show a similar undesirable behavior.

To avoid the problem to converge to a false global optima according to the surrogate, different strategies that make use of the real fitness can be used (see previous section). These are: Fixed Individual Percentage Strategy (FPS), Fixed Generation Strategy (FGS) and Best Reevaluation Strategy (BRS). The numerical results for these strategies are presented in the remainder of this section.

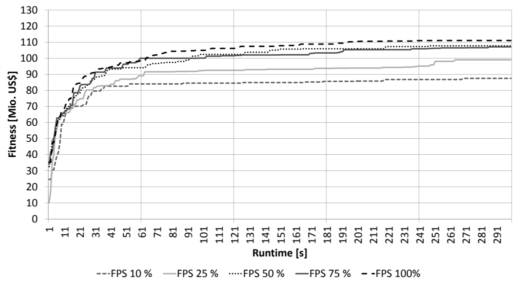
The convergence of the evolutionary algorithm in the WAF instance is shown in Figure 5.19. In each subfigure, the convergence when evaluating 10%, 25%, 50%, 75% and 100% random population's individuals using the real fitness is plotted for the fixed population percentage strategy (FPS). Because the evaluation runtime of the remaining individuals' should be decreased, the relatively inaccurate surrogate with one column generation iteration, 70% and 30% of cargo flows and 40% and 30% of bunker cost discretization support points are used. In the top row of Figure 5.19, the $cum.Q * R$, on the bottom the $Q * R$ cargo flow selection strategy are shown. The results indicate, the $cumQ * R$ strategy works slightly better than $Q * R$. The main drawback is that all results indicate that evaluating the whole population using the real fitness is superior compared to the surrogate model in the WAF instance.

Very similar results are available for the Mediterranean instance (see Figure 5.20). Nevertheless, a slightly faster convergence is observed at the beginning of the solution approach. This can be explained by the quicker fitness calculation.

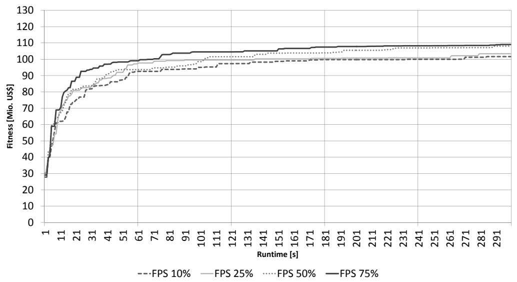
Overall, we can conclude that randomly picking individuals from the population that are evaluated exactly is not superior to the classic metaheuristic approach and, usually, even leads to solutions with worse objective values.



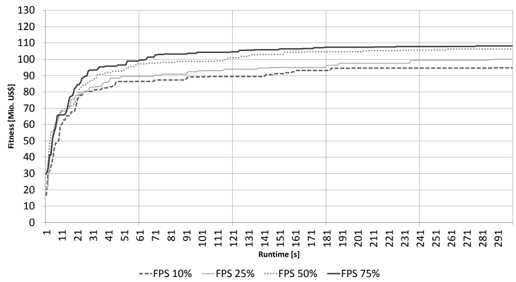
(a) 70% CF (cum. Q*R), 40% Support Points and 1 CG-Iteration.



(b) 30% CF (cum. Q*R), 30% Support Points and 1 CG-Iteration.

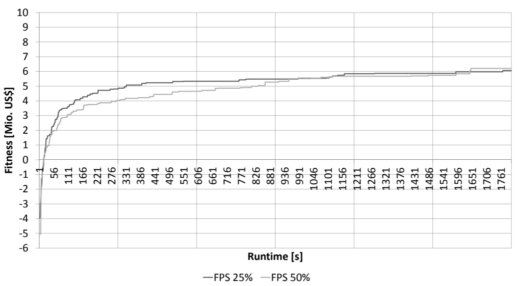


(c) 70% CF (Q*R), 40% Support Points and 1 CG-Iteration.

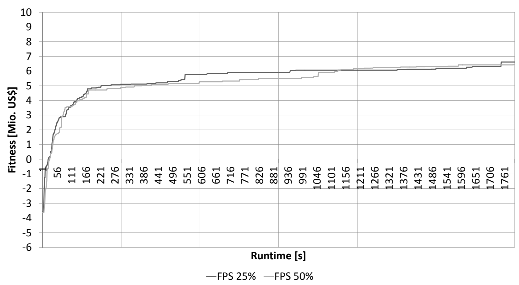


(d) 30% CF (Q*R), 30% Support Points and 1 CG-Iteration.

Figure 5.19.: Fitness convergence using different surrogate configurations for the FPS strategy in the evolutionary algorithm for the WAF instance.

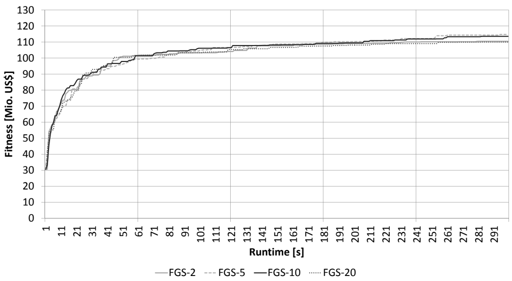


(a) 70% CF (cumQ*R), 40% Support Points and 1 CG-Iteration.

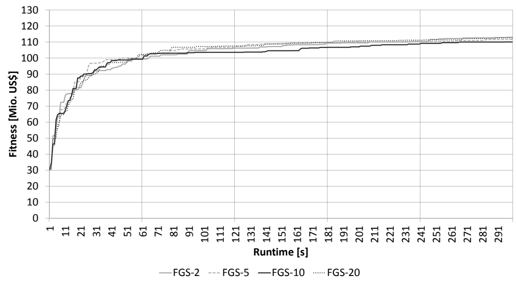


(b) 70% CF (Q*R), 40% Support Points and 1 CG-Iteration.

Figure 5.20.: Fitness convergence using different surrogate configurations for the FPS strategy in the evolutionary algorithm for the Mediterranean instance.



(a) $cumQ * R$ cargo flow selection.

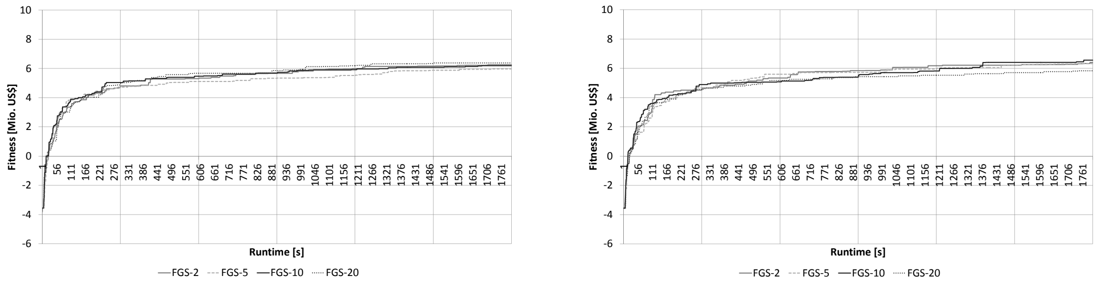


(b) $Q * R$ cargo flow selection.

Figure 5.21.: Evolutionary Algorithm fitness convergence using the Fixed Generation Strategy (FGS) with a 70% CF, 40% Support Points and 1 CG iteration surrogate with the $cumQ * R$ and $Q * R$ strategy for the WAF instance.

The second static surrogate is the fixed generation strategy (FGS) for which the numerical results are shown in Figure 5.21 for the WAF, in Figure 5.22 for the Mediterranean instance. For every instance and cargo flow selection strategy, every 2, 5, 10 and 20 iterations the whole population is evaluated using the real fitness function. The remaining individuals are evaluated with a surrogate that allocates 70% of the cargo flows, uses 40% of the support points and terminates after the first delayed column generation iteration. The $cum.Q * R$ cargo flow selection in Figure 5.21(a) converges better than the $Q * R$ strategy in Figure 5.21(b). Compared with the evolutionary algorithm using the real fitness, the best fitness found is about 4 - 5% better using the FGS surrogate strategy and the $cum.Q * R$ cargo flow selection. The $Q * R$ selection is about 3% better compared to the real fitness evaluation. The results for the Mediterranean instance indicate that in some configurations the runs with less exact evaluation, in others the ones with more exact evaluations perform better. A reason can be the diverse search space exploration of the evolutionary algorithm. This has been considered when choosing the crossover operators. In the case of evaluating the whole population approximately, the operators might not work as expected.

Finally, the results for the best reevaluation strategy (BRS) in the hybrid evolutionary algorithm are presented. The whole population is evaluated using a surrogate that routes 70% of the cargo flows according to the $cum.Q * R$ strategy, 40% of the support points and the column generation is terminated after the first iteration. In a second phase, the top 5%, 10%, $\cdots$, 75% of the population is reevaluated using the real fitness. The results are shown in Figure 5.23 for the WAF instance. Compared to the real fitness evaluation, the best fitness is 1-2% higher using the BRS surrogate on 75% of a population's individuals. Decreasing the percentage



(a) $cumQ * R$ cargo flow selection. (b) $Q * R$ cargo flow selection.

Figure 5.22.: Evolutionary Algorithm fitness convergence using the Fixed Generation Strategy (FGS) with a 70% CF, 40% Support Points and 1 CG iteration surrogate with the $cumQ * R$ and $Q * R$ strategy for the Mediterranean instance.

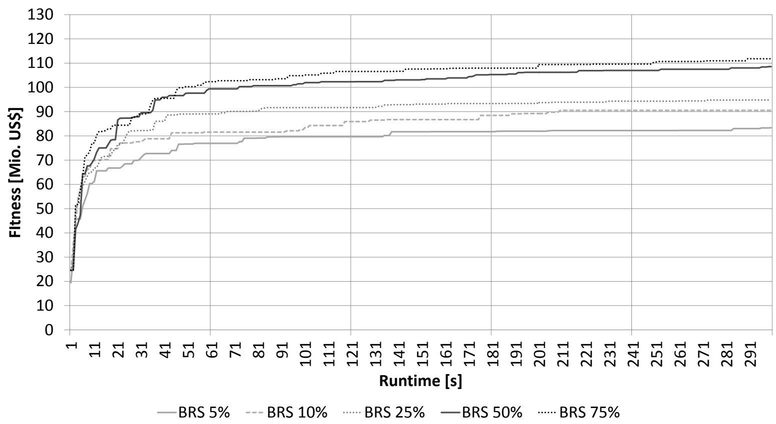


Figure 5.23.: Evolutionary Algorithm fitness convergence using the Best Individuals Reevaluation Strategy (BRS) with 70% CFs, 40% Support Points and 1 CG iteration surrogate with the $cumQ * R$ strategy for the WAF instance.

leads to solutions that are much worse. The numerical results are similar for the Mediterranean instance. In Figure 5.24, the best found fitness equals the fitness of the exact function.

The results of the surrogate fitness functions in the hybrid evolutionary algorithm are not significant. Although the best reevaluation strategy performs slightly better than the real fitness, the changes are within a one-digit percentage and are rather explained with the stochastic nature of the algorithms. One reason for the bad

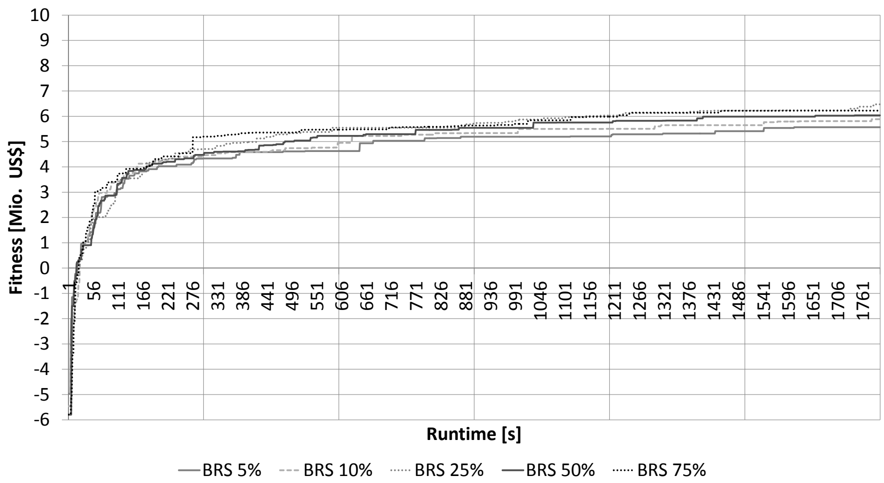


Figure 5.24.: Evolutionary Algorithm fitness convergence using the Best Individuals Reevaluation Strategy (BRS) with 70% CFs, 40% Support Points and 1 CG iteration surrogate with the $cumQ * R$ strategy for the Mediterranean instance.

performance of the surrogate fitness function can be the missing normalization between the surrogate and the exact fitness values. The best individuals are selected according to the elitism strategy that is based on the fitness. When some individuals are evaluated using a surrogate, they depend on the random part of the elitism function to survive the selection process and evolve different generations. Thus, the metaheuristic should rely on fitness values, evaluated with the same fitness method.

In the next section, the constant surrogate strategy is applied to the variable neighborhood search.

Variable Neighborhood Search

The variable neighborhood search (VNS) heavily relies on the local search algorithm variable neighborhood descent (VND) that is used in this thesis. Due to the high fitness sensitivity to network changes, the crossover operator might be the main drawback in the hybrid evolutionary algorithm. Thus, the VNS has been proposed that already indicated promising results for the medium sized instances. The drawback in larger instances is the relatively slow convergence. However, within the VNS one can easily adjust the VND surrogate accuracy to decrease the runtime. The results indicate that decreasing the accuracy can lead to false decisions within the whole algorithm.

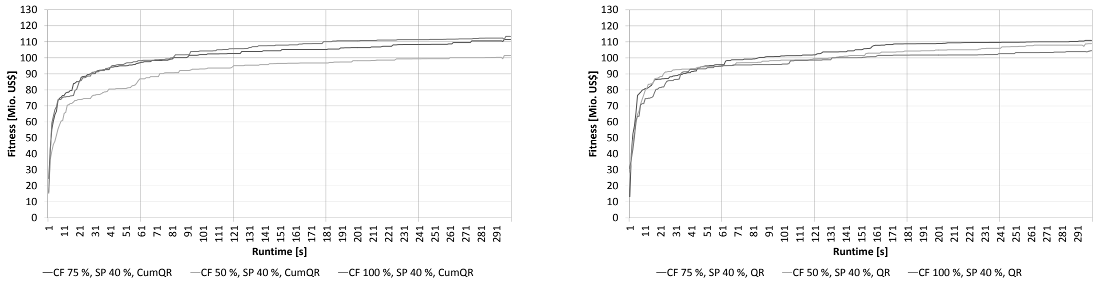
Regarding the acceptance decision, the quality of the surrogate is of great importance. The VNS metaheuristic differs in an important point to the evolutionary algorithm: Within the VND, a modified solution is accepted if its fitness is better than the previous solution. This means that the single decision that has to be supported by the surrogate is whether the solution is better (or worse) than before according to the real fitness.

Table 5.11 presents an overview of the surrogate quality (for a detailed evaluation, based on each single operator, see Appendix D.1 and D.2). The table shows the number of neighborhood structure executions for one VNS run for the Mediterranean instance. Beside the number of calls, the table indicates the number of correct decisions (according to the real fitness function), the number of wrong decisions and the number of no changes for different surrogate configurations. In all configurations, the number of iterations of the delayed column generation is fixed to one and only 40% of the support points are used. The different surrogate configurations limit the amount of transportable cargo flows to increase the speed of the evaluation. For example, the second configuration in Table 5.11, $CF75\% - Q * R$, indicates that 75% of the cargo flows, ordered descending by the most profitable cargo, is used in the surrogate. One can observe that on average 85% of the operations performed in the VNS are correct in the sense that evaluating them with the real fitness function would lead to the same decision. The right decision can be either to accept or reject a neighboring solution.

Configuration/ operator	$\sum$ **operations**	**correct decisions [%]**	**wrong decisions [%]**
CF 100%, Q*R	26907	88,26%	11,74%
CF 75%, Q*R	30177	84,90%	15,10%
CF 50%, Q*R	34461	84,60%	15,40%
CF 100%, cum. Q*R	25969	86,65%	13,35%
CF 75%, cum. Q*R	38989	83,99%	16,01%
CF 50%, cum. Q*R	52314	83,56%	16,44%

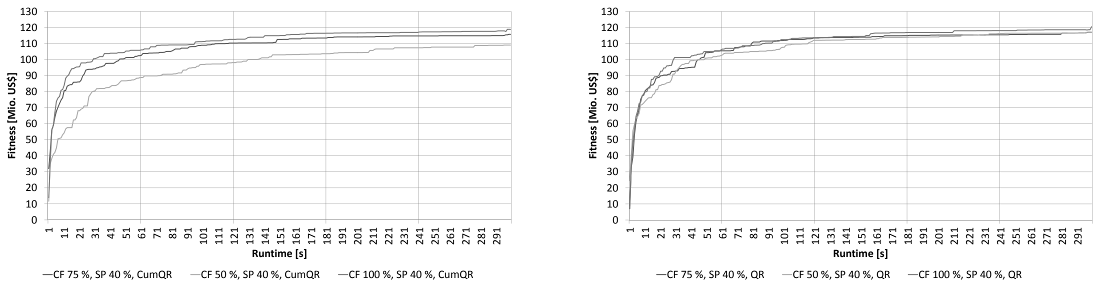
Table 5.11.: Accuracy of surrogate evaluation in the variable neighborhood descent solving the Mediterranean instance. The CG is aborted after the first iteration and 40% (8) support points for the bunker cost discretization is used. Data is recorded in five runs of the variable neighborhood descent.

The configuration *CF 100%, Q*R* and *CF 100% - cum. Q*R* indicates that the number of iterations and support points still have a strong effect on the correctness of the decisions. With the decreased number of cargo flows, the executed operations increase, especially for the $cum.Q * R$ cargo flow selection, which relies on taking the cargo flows that lead to X% of the possible overall revenue. Due to the different cargo flow distributions (see Appendix C.1), the speed up is larger compared to the $Q * R$ strategy and more than twice as many VNS iterations could be performed when using 50% of the CFs instead of 100% in the $cum.Q * R$ strategy. However,



(a) $cumQ * R$ cargo flow selection. (b) $Q * R$ cargo flow selection.

Figure 5.25.: Fitness convergence using different static surrogates in the VND local search in the WAF LINER-LIB instance. CG is terminated after the first iteration.



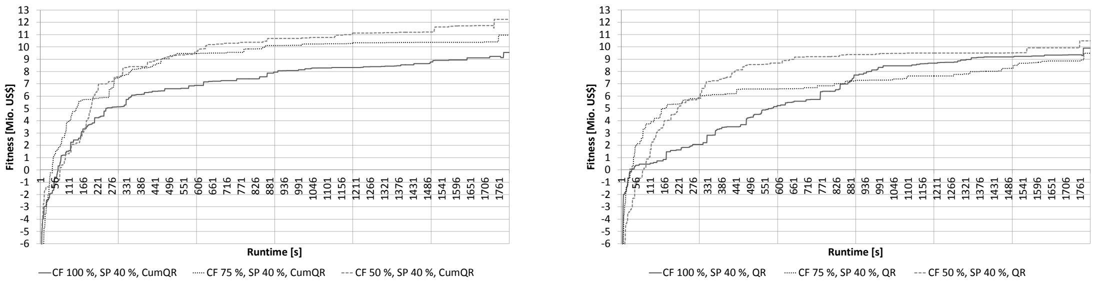
(a) $cumQ * R$ cargo flow selection. (b) $Q * R$ cargo flow selection.

Figure 5.26.: Fitness convergence using different static surrogates in the VND local search in the WAF LINER-LIB instance. CG is terminated after the second iteration.

the percentage of wrong decisions only increases slightly from 13.35% to 16.44%. The results indicate that the surrogate approximation in the variable neighborhood descending step is promising because the vast majority of decisions are still correct.

Figure 5.25 shows the convergence of the VNS with the configurations presented in Table 5.11 for the WAF instance. The results indicate that the convergence is highly influenced by the static surrogates: Limiting the surrogate to 50% of the cargo flows leads to objective values that are much worse than the best ones found using the real fitness function (compare Figure 5.15(b)). The best fitness found using the surrogate in the VND is about 4% worse compared to the real fitness for all cargo flow selection strategies.

The results change slightly when the accuracy of the surrogate is increased by limiting the number of column generation iterations to two. In Figure 5.26 the best



(a) $cumQ * R$ cargo flow selection.

(b) $Q * R$ cargo flow selection.

Figure 5.27.: Fitness convergence using different static surrogates in the VND local search in the Mediterranean LINER-LIB instance. CG is terminated after the first iteration.

objective found with two CG iterations and 40% of support points is more than 120 million US$ (about 2.5%, see Figure 5.15(b)). However, the differences in the surrogates are relatively small for the WAF instance.

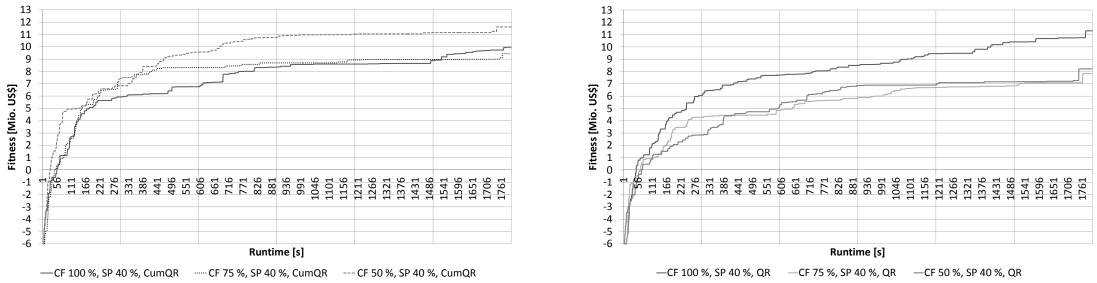
As expected, the advantage of faster evaluations in the VND local search increases with larger problem instances. In Figure 5.27, the VNS convergence is shown for the same surrogate configurations. Recall that the best fitness found for the Mediterranean instance using the real fitness was about 7 million US$ (see Figure 5.15(c)).

The results shown in Figure 5.27 indicate a much faster convergence for all surrogates for the maximum runtime of 30 minutes. The best fitness found is at least 30% better compared to the VNS using the real fitness VND evaluation. The best fitness found using the surrogate with the worst accuracy of 50% of cargo flows, 40% of support points and one delayed column generation iteration is about 70% better. In contrast to the much smaller WAF instance, increasing the number of column generation iterations to two slows the convergence down because the fitness calculation becomes too time consuming (see Figure 5.28).

These results indicate that the runtime improvements using surrogate models can be beneficial, especially in medium sized instances (see Figures 5.28(a) and 5.28(b)).

To back these findings, Figure 5.29 shows the same surrogate configurations for the Pacific instance. The best solution found with the VNS without a surrogate was on average 100,000 US$ after 30 minutes. Using the surrogate results in a fitness of more than 8 million US$ using the $cumQ * R$ strategy and more than 14 million US$ using the $Q * R$ strategy. Furthermore, the convergence of the VNS clearly improves because better solutions were found much faster. This holds for all tested LINER-LIB instances.

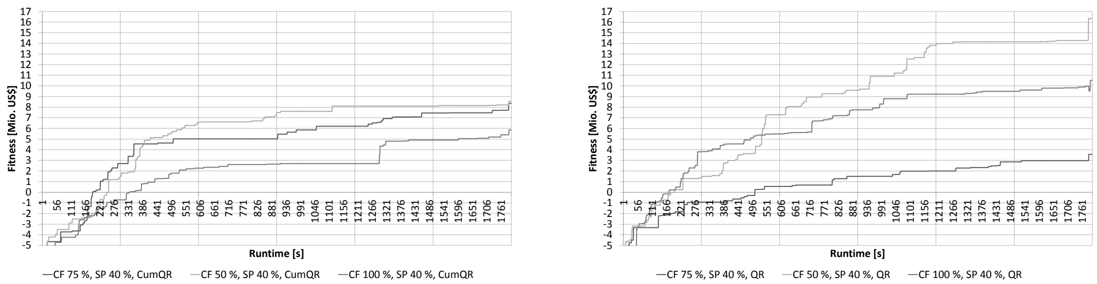
Compared to the hybrid evolutionary algorithm that found Pacific networks with a profit of 11 million US$, VNS improves the best solution found as well. It is



(a) $cumQ * R$ cargo flow selection. (b) $Q * R$ cargo flow selection.

Figure 5.28.: Fitness convergence using different static surrogates in the VND local search in the Mediterranean LINER-LIB instance. CG is terminated after the second iteration.

about 45% higher which clearly shows the superiority of the VNS using a constant surrogate strategy.



(a) $cumQ * R$ cargo flow selection. (b) $Q * R$ cargo flow selection.

Figure 5.29.: Fitness convergence using different static surrogates in the VND local search in the Pacific LINER-LIB instance. CG is terminated after the first iteration.

5.4. Interpretation of Results

Numerical results for the exact mixed integer model and metaheuristics for the liner shipping network design problem are presented in this chapter.

They indicate that the exact model can only be solved for very small instances of the liner shipping network design problem to optimality. For the smallest LINER-LIB instances large gaps are still observed after 24 hours. The main reason is the relative bad bounds that result from the formulation presented in this thesis which could be improved by introducing cutting plane techniques. This technique to solve the problem to optimality is also recognized by Plum et al. (2013b).

To be able to solve the problem for medium sized instances, two metaheuristics are presented: An hybrid evolutionary algorithm and a variable neighborhood search. Both methods are able to provide reasonable results up to the Pacific instance of the LINER-LIB benchmark suite. Using the default configuration, VNS works better for the small instances. Its drawback is the slow convergence in the medium sized instances where the hybrid evolutionary algorithm outperforms VNS.

To improve the convergence of both heuristics by speeding up the fitness calculation, the surrogate concept is applied to the liner shipping network design problem. The hybrid evolutionary algorithm provides a broad range for surrogate application strategies. However, none of the methods clearly outperforms the exact fitness function. The reason is that the population evolves based on the individuals fitness which is hard to normalize with difficult fitness functions.

The surrogate results for the VNS clearly outperforms the evolutionary algorithm and the VNS without surrogate in the medium sized Mediterranean and Pacific LINER-LIB instances. The reason lies in the acceptance criteria of the variable neighborhood local search. A solution that is evaluated with the surrogate is compared with a neighbor solution evaluated with the surrogate as well. The comparison shows that using the surrogate is sufficient to identify whether to accept or reject a candidate solution. Additionally, surrogates introduce diversification which is important for the heuristics as well.

In the remainder of this chapter, a sensitivity analysis is performed for the so far fixed bunker prices. Then, the methods are evaluated in a real-world case study.

5.5. Bunker Cost Uncertainty in the Tactical Planning Horizon

Fuel consumption is the biggest cost factor (compare Section 2.10). We analyze the effects of the bunker price fluctuations on the networks and discuss whether future research should be performed by including the uncertainties in the network design.

To analyze the effects of different bunker prices on the network design the LINER-LIB 2012 Baltic instance is used as an example. The reference bunker price of 600 US$ per metric ton has been varied by +/- 25%, leading to an upper price of 750 US$ and a lower one of 450 US$ per ton. All presented results are based on three runs of the variable neighborhood search using the surrogate with a maximum runtime of three minutes. Figure 5.30 shows the average best profit of the three runs for different bunker price per ton.

The bunker prices plotted on the x-axis represents a 5% change in the price per ton (based on the reference of 600 US$). The profit, shown on the y-axis, decreases with increased bunker price. The profit's difference within a 5% change is largest between 450US$ and 480 US$, meaning 2% profit loss. The overall difference between

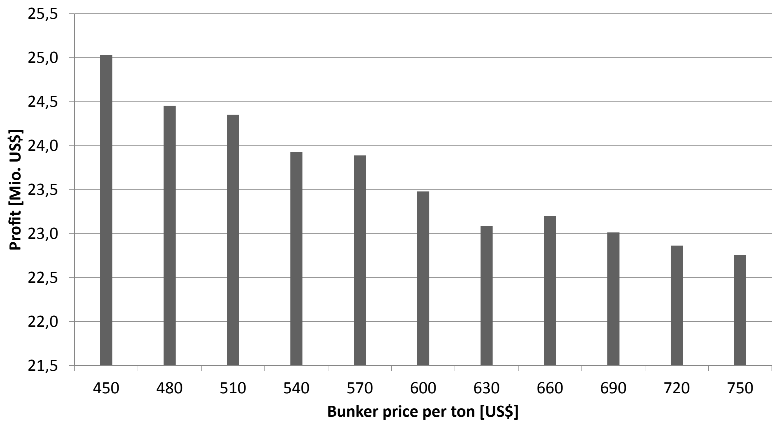


Figure 5.30.: Average best profit of three VNS runs for different bunker prices per ton.

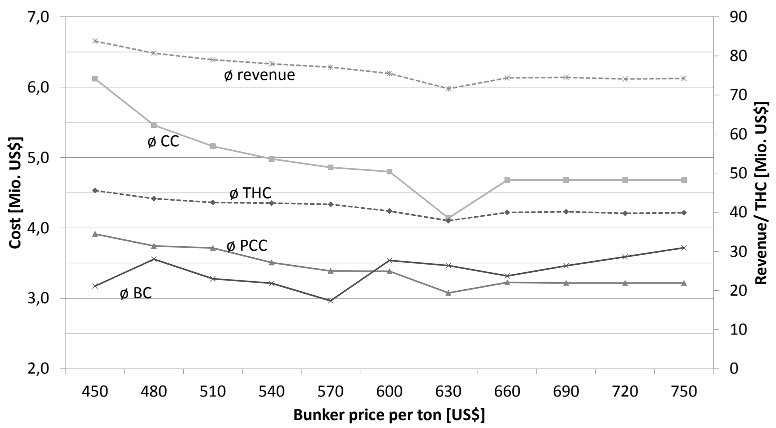


Figure 5.31.: Cost components and revenue in the best networks for different bunker prices per ton.

750US$ and 450US$ are decreased profits of about 9.1%. This of course assumes that the network can be always freely adjusted to the new parameter at no costs. The average profit at a bunker price of 630 US$ is slightly lower than expected. This can be explained by the stochastic component of the algorithm.

Figure 5.30 indicates that the networks adapt to the different bunker cost per ton, however, it does not indicate whether a structural change has taken place. Changes in the network design (such as modified vessel types or changed port rotation) means the method should consider the bunker cost variation directly. The first step to identify the reasons for profit changes is to look at the cost components.

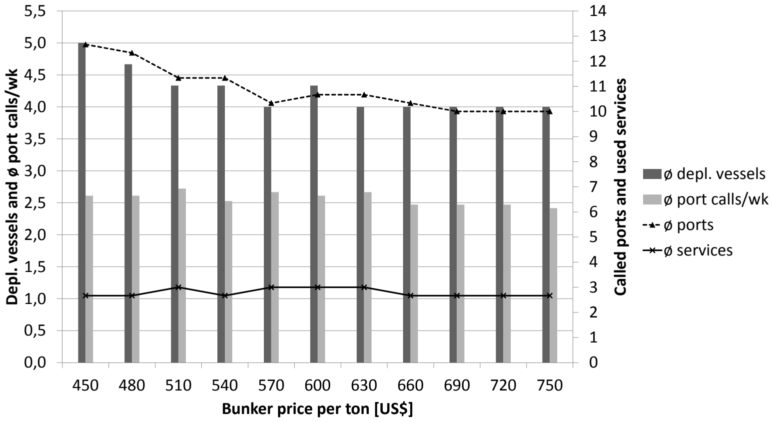


Figure 5.32.: Structural network changes with different bunker prices per ton.

In Figure 5.31, the average revenue and average costs of the best networks are shown. One can observe, that the average revenue, port call cost (PCC) and charter cost (CC) decrease with increased bunker cost. The terminal handling cost (THC) stay relatively constant. The decreased profit is caused by 6reduced port calls, which avoid the transportation of unprofitable cargo. The nearly constant terminal handling cost can be explained by an increased transhipment tendency in the network. The average total bunker cost (BC) increase with increased bunker cost per ton. The charter costs indicate that less vessels are deployed on average in the network. This is reasonable because a network with less vessels leads to an acceleration of the vessels to match the round trip time. Removing ports allows higher slack time at sea and a lower average speed.

The reduced number of called ports and deployed vessels can also be seen in Figure 5.32. The figure indicates a nearly steady number of services used throughout different bunker costs. In the Baltic instance, on average 2.5 services are deployed, meaning that the best networks found by the VNS contained two to three services. Similarly, on average six to seven ports are called per round trip, whereas slightly less ports are called per round trip under large bunker costs.

As indicated in Figures 5.31 and 5.32, the optimization reacts to the higher bunker cost by decreasing the called ports and deployed vessels. The average speed on the services is not clearly reduced (see Figure 5.33). The speed results from the port rotation and the number of deployed vessels. Because both are decreased with higher bunker prices, the model can avoid high speeds. Only in case of relatively low bunker prices are slightly higher speeds observed. This means that under all circumstances, the speed optimization is highly relevant for the shipping industry due to the cubic consumption function. The served cargo flows are also decreased in the high bunker price scenarios. The reason is that either the port of origin or

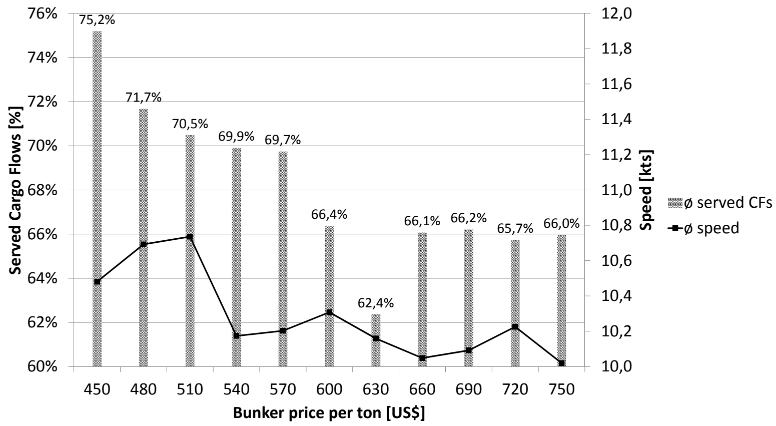


Figure 5.33.: Served cargo flows and average speed with different bunker prices per ton.

destination are not served anymore or the cargo becomes unprofitable. Vessels must increase their speed to compensate the duration to load and unload the cargo, which becomes unattractive for high bunker cost. The exception for the 630 US$ solutions can be seen again. VNS leads to networks that decrease the number of vessels and port calls more than similar solutions (see Figure 5.31). This results in less cargo served (see Figure 5.33).

Table 5.12 shows the effects on the networks' utilizations. The second column indicates the best peak utilization per service and per leg (BPU), the third the average worst utilization (WPU) and the last the maximum average utilization per service (BAU). The WPU remains at values between 78 - 87%, indicating that all legs have a good lower utilization bound. BPU shows that leg utilization up to 100% can be reached on nearly 40% of the networks. Due to the decreased served cargo flows and the reduced speed, unprofitable cargo is skipped in the optimization under relatively high bunker cost per ton. This is indicated by a slightly decreased average best utilization. However, since the network can perform adjustments to their structure, the BAU remains relatively high.

Concluding the results, structural network design changes occur for the port rotation and number of vessels. In the low bunker price scenario, the optimization model deploys more vessels and extends the geographical coverage. Additionally, more cargo is transported at a higher speed. With increased bunker prices per ton, less vessels and ports are served in the networks and the average speed is decreased. Typically, liner networks are designed and implemented for several years. In this time period, the bunker price is subject to uncertainty and the liner carriers have to adjust to the increased cost.

On the one hand, the sensitivity analysis indicates that a relatively high profit can

Bunker Cost US$	BPU%	WPU%	BAU%
450	100.00%	81.79%	100.00%
480	100.00%	86.71%	99.68%
510	100.00%	86.54%	99.68%
540	98.81%	79.13%	95.83%
570	97.63%	85.88%	97.31%
600	100.00%	87.37%	100.00%
630	100.00%	85.42%	99.36%
660	95.46%	79.17%	95.14%
690	95.46%	79.71%	95.46%
720	96.83%	79.67%	96.83%
750	95.46%	79.71%	95.46%

Table 5.12.: Average BPU, WPU and BAU for networks optimized to different bunker prices per ton.

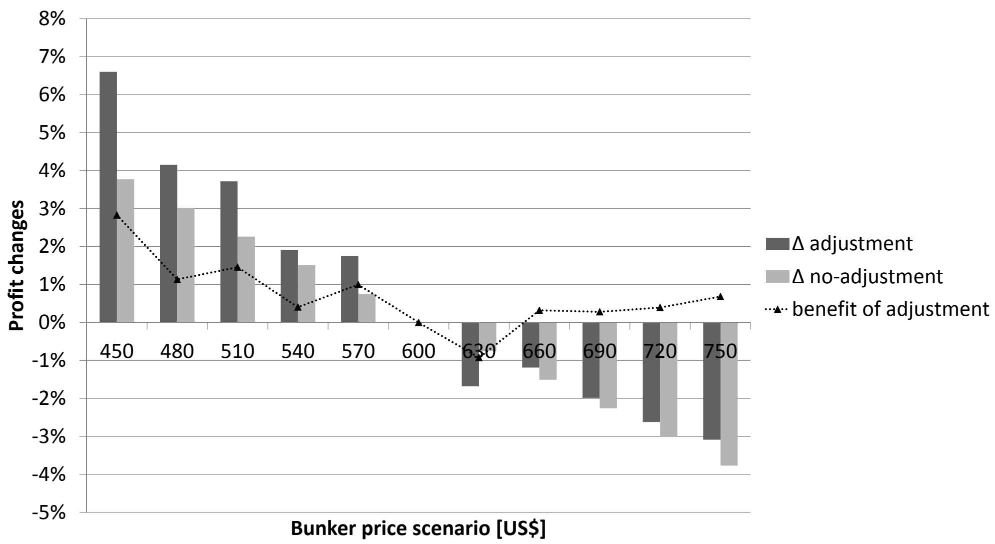


Figure 5.34.: Benefit of adjusting network for different scenarios that is optimized for a bunker price of 600 US$ per metric ton.

still be reached under high bunker cost scenarios by implementing slow steaming strategies and structurally adjusting the network. An increase in the bunker price by more than 60% leads to decreased profits of less than 10%. On the other hand, however, this adjustment is only possible if a carrier is able to implement the changes. Long term vessel charter contracts or long term cargo flow contracts would fix both the capacity and the required ports, which limits the freedom of action.

In Figure 5.34, the networks optimized with a bunker price of 600 US$ are used as a reference. On the x-axis different scenarios that vary the bunker cost with a step size of 5% is given. The y-axis indicates the profit change in percent with a basis of the profit at 600 US$. A liner carrier can either change the network (Δ adjustment) or remain in the network and use the slack time (both to the maximum and minimum speed) to react on the new bunker price (Δ no-adjustment).

When the bunker price decreases, the network is more profitable in both reactions due to the decreased total cost. The difference in the benefit slightly increases when the new bunker price increasingly differs from the planned price. Figure 5.34 indicates that the benefits gained from redesigning the network instead of keeping the network can be up to 3% for this relatively small feeder network. When the bunker price changes by more than 20%, the network should be redesigned or at least adjusted to the new parameters.

In times of increased bunker prices, liner carriers also benefit from redesigning their network and becomes more important with increasing prices. However, the benefit is smaller compared to decreased bunker prices. Figure 5.32 indicates that in these cases fewer ports per round trip are called. We argue that a critical network size is reached, where serving even less ports becomes less useful. Of course, the development of increased bunker cost indicated in Figure 5.31 will only work out as long as transporting the cargo flows is still profitable.

The sensitivity analysis in this section shows that it can be efficient to adjust liner networks to new bunker prices. In particular, in scenarios of large bunker price changes the profit can be increased when adjusting to the new price realization. However, before implementing these changes, further cost must be considered. Tierney and Jensen (2012) indicate that repositioning the vessels to service the adjusted liner services can lead to large costs. Therefore, the vessel repositioning cost must be considered before adjusting the network to the new bunker price.

The analysis also shows that the structure of liner shipping networks changes under different bunker prices. Especially during high volatile bunker price periods, such as in 2009 (see Chapter 2), the scenarios should be included to create solutions as stable as possible for different bunker prices. This could be an interesting aspect in future research.

5.6. Numerical Results from a Global Liner Carrier

The methods described in the previous chapters are applied to data of a global liner carrier. To challenge the algorithms, a single region within the global network has been selected. Table 5.13 shows the size of the problem instance. Approximately 40 ports (where in about 31 ports have cargo flow origins or destinations), 1400 legs and 1600 cargo flows of five different equipment types are selected: 20 and 40 foot dry and reefer as well as high cube containers. For the cargo flows, 140 transit time requirements between ports are given. The carrier operates a global network and

Set	Size
Ports	40
Cargo Flows	1600
Vessel Types	4
Resource Groups	3
Equipment Types	5
Legs	1400
Embargo Constraints	12
Transit Time	140

Table 5.13.: Approximate size of the problem instance used for the realistic case study.

cargo flows are transported into and out from the region to different destinations. To reduce the network size for optimization the port of loadings and destinations are set to the current transhipment ports outright the region South America for example. However, the model is still allowed to tranship containers between different services at all ports within the region.

The instance defines three available vessel types, ranging from nominal 800 TEU to 2800 TEU vessels. With an average weight of 14 tons per container, the capacity is nearly half. Each of the vessel types has a bunker profile attached and uses three capacity types, namely dry slots, reefer plugs and maximum weight. Each cargo flow and empty container utilizes the *weight* resource and either the slot resource. In case of reefer containers the plug resource is utilized as well.

The case study uses different embargo constraints that define ports that cannot be served within one liner service due to political reasons. The instance size is related to the LINER-LIB Pacific and Europe-Asia instance and thereby imposes a large computational challenge on the algorithms. For improving the given network structure and determining practical valid networks, some adjustments on the solution methods must be done (see Figure 5.35): The overall process starts with loading the instance's data from a database. Then, the cargo flows are aggregated to improve the solution time. The aggregation transforms all container types into 20 foot *Dry* and *Reefer* containers between the origin and destination and averages the revenue per container and the resources' utilization. Thus, we reduced the amount of cargo flows by removing the 40 foot and high cube containers. This aggregation reduces the amount of cargo flows by approximately 50%. The metaheuristics are used to improve the network. We use the carrier's current network as a starting point. The carrier pointed out that the maximum runtime can be set to several hours or even days due to the strategic decisions and the large potential of the optimization. However, we have limited the algorithm to terminate after eight hours. Then, the network must be reevaluated using the disaggregated cargo flows to obtain the optimal profit according to the cargo allocation (see Figure 5.35). Depending on the

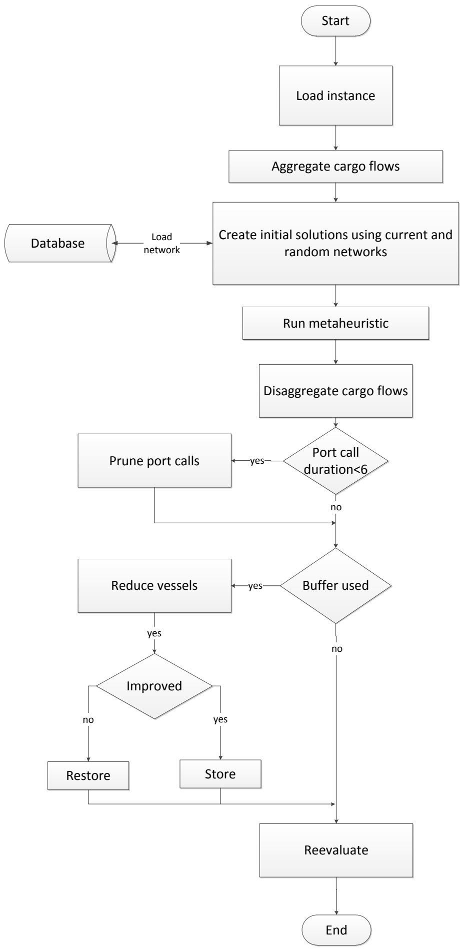


Figure 5.35.: Adjusted optimization process to handle practical constraints for the best liner shipping network found.

data, the gap between the actual cargo flows and the aggregation can be quite large. However, the tests indicate that the error concerning the evaluation of the current network is not more than 10%. After the aggregation, the network is modified to meet practical requirements:

- Remove ports that are not called more than six hours per visit.
- Adjust deployed vessels in case of services with large buffers.

First, the resulting network might still contain ports where the port call duration is less than six hours which is unreasonable for real-world networks. To remove these ports from the solution, we remove all ports that are called less than a specific amount of hours (if the pruning leads to an increased profit). Second, the process evaluates whether the cargo allocation determines a high additional buffer for all

Cons. Cargo Type	Utilization (TEU)	Utilization (Reefer)
Cargo Flows	75.09%	73.77%
Empty Containers	9.51%	
Overall	84.60%	

Table 5.14.: Current average utilization (with the optimal cargo allocation). The TEU utilization is split to laden and empty containers. Additionally. the average utilized reefer plugs are shown.

ports. This might be the case if the metaheuristic has too many vessels deployed on the service. Then, the number of vessels is tried to be decreased. Finally, the whole solution is reevaluated, the proforma schedules that contain the port durations (including pilotage, buffer and moving time), sea slack time (duration difference from the current to the maximum speed), detailed cargo flows as well as transhipment operations are printed out and the method terminates.

The results for this case study are presented in the remainder of this section. One specific trade has been selected for optimization. The specifics of that trade are: A high number of competitors, a high level of maturity of the provided liner services and a connection of two subregions. On the trade route, the operator uses different services, whereas some operate as feeders from the existing transhipment hubs but others serve cargo flows on the trade. Table 5.14 indicates the average utilization of the existing network using the cargo allocation method presented in Chapter 4 (note that this is a profit maximal cargo allocation that is not done in practice yet). It can be seen that the network's average utilization is about 84% (including dry, reefer and empty containers). The table shows that trade imbalances exist on the services, leading to an average utilization of about 10% due to empty containers. The utilization of reefer container plugs is less than 74%.

Figure 5.36 provides an overview of the cost structure of the carrier in the subnetwork. The bunker and container handling cost make up about 30% each. The third largest cost component are the fixed vessel costs at 19%, followed by the port call costs of 13%. The smallest component is the container depreciation at 5%.

The existing network is optimized using the variable neighborhood search presented in Section 5.2.4. To evaluate the intermediate results from the heuristic and be able to compare different solutions, the current system is compared with three alternatives: *Intermediate* is a network that results after a runtime of 1.5 hours using a random start solution. *Network A* and *Network B* are the two best solutions found within one run by the metaheuristic using the current network as initial solution and referred to as the *optimized networks*. The intermediate network uses one service more than the current, the optimized use as many services as before, but modified the services clearly.

Table 5.15 provides the utilization for the existing, intermediate and optimized

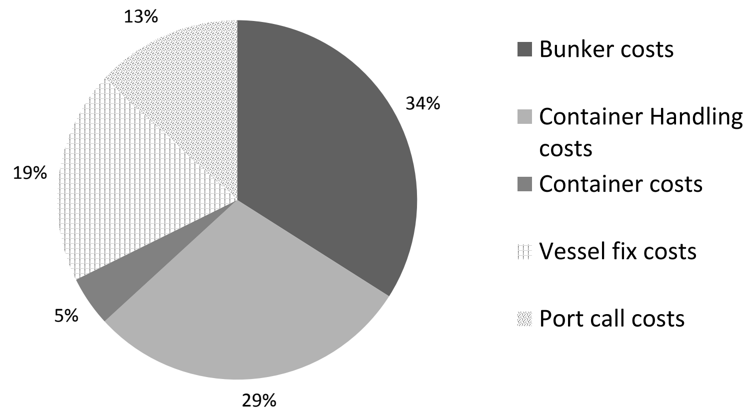


Figure 5.36.: Distribution of the average cost structure in the current liner network at hand.

networks. The average utilization of all services is increased by more than 10% on all optimized networks. The metaheuristic first tries to optimize the network according to its utilization, leading to a close to full utilization for the intermediate network. Afterwards, the network's utilization is decreased compared to the intermediate solution.

Type	**Current**	**Intermediate**	**Network A**	**Network B**
Laden containers	75.09%	93.52%	82.81%	85.03%
Empty containers	9.51%	5.85%	12.15%	13.02%
Overall	84.60%	99.37%	94.96%	98.04%

Table 5.15.: Average service utilization for all legs by laden (dry and reefer cargo flows) and empty containers

The utilization indicates that the modifications performed by the metaheuristic are suited to solve the problem. The differences of the solutions can be explained using Table 5.16: The provided utilization only considers cargo that can be transported in the network, however the volume of served cargo is relatively low in the intermediate solution. During the optimization process, the network is successively extended so that more ports and thereby more cargo flows can be served. The resulting utilization also depends on the capacity deployed in the network. Table 5.16 indicates that the intermediate solution serves more ports than the best networks found, however it still deploys relatively few capacity compared with the other networks. This leads to a high utilization, but only 62.48% of served cargo flows.

Another observation is made regarding the average speed on all service legs per network. The optimization clearly tries to modify the networks in such a way that the vessels can steam with a relatively low average speed on the services (shown in the first row of Table 5.16). The speed is decreased by more than 25% in the best solution found. The decreased speed can either be the result of more deployed vessels, less port calls or rearranged port sequences. The intermediate and network A still have less capacity (and fewer vessels) deployed, leading to a relatively high speed. Network B finds a suitable trade-off between network extension and compensation by more vessels. Table 5.16 indicates that increasing the number of port calls does not directly result in an increased speed. Instead, the port sequence is highly important.

Property	**Current System**	**Intermediate**	**Network A**	**Network B**
Served cargo flows [TEU]	96.18%	62.48%	95.90%	96.61%
∅ speed [kts]	16.15	15.55	14.36	13.14
Max. speed difference per service [kts]	1.20	10	3.70	4.00
Deployed capacity [%]	100.00%	46.15%	98.29%	103.42%
Port calls [%]	100.00%	123.53%	114.71%	111.76%
Total slack time [d]	23.92	43.51	24.16	41.59
∅ slack time [d]	5.98	8.70	6.04	10.40

Table 5.16.: Properties of the current, an intermediate and two alternative networks.

The results also indicate that the current system is designed to steam on a relatively constant speed on all legs, indicated by the row *Max. speed difference* that shows the maximum difference between the maximum and minimum leg speed of all services. Small values indicate that the overall network is designed for relatively constant speeds but still keeps the transit time. High values indicate that the vessels must accelerate to keep the required transit times, which can lead to high bunker cost. Especially the intermediate network has a very high speed difference value, indicating that the solution is not yet suited to reduce the overall average speed. Network A and B have a higher speed difference compared to the current network. This indicates that it is useful to let the network have different leg speeds to deal with the transit time. However, they must not be too large due to the cubic bunker consumption.

The last indicators presented in Table 5.16 are the total and average slack time at sea. These represent the total (or average) difference between the duration when steaming at the schedule's speed compared to the maximum speed. Thereby, the time can be used in operations as a buffer to hold the schedule in case of delays. The slack time depends on the length of the leg and the planned speed on the leg. The case study's current network has a total slack time of approximately 24 days, with six days on average per service. This value is highly increased in the intermediate and network B and indicates that the networks serve either longer distances (intermediate network) or implement a super slow steaming strategy (network B). The slack time of network A is relatively similar to the current network, indicating a relatively similar network structure regarding the distances. The solution methods presented in this thesis are designed to potentially increase the slack time because the bunker costs play the most important role in optimizing existing networks.

	Intermediate (%)	**Network A (%)**	**Network B (%)**
Profit	-47.91%	5.62%	7.66%
Revenue	-32.30%	0.22%	1.70%
Bunker cost	44.55%	-30.65%	-33.06%
Time charter cost	-39.06%	-1.74%	3.47%
Port call cost	-19.58%	7.53%	0.73%
Handling cost	-24.51%	-1.54%	1.93%
Container cost	-30.43%	0.20%	1.24%
∅ cost change	13.81%	-5.24%	-5.14%
Total cost change	9.39%	-7.70%	-7.05%

Table 5.17.: Changes in the profit, revenue and cost structure for the intermediate solution and the two alternative networks.

In Table 5.17 the changes of the profit, revenue and different costs for the three alternative networks are given in percent of the current system. Values less than zero indicate that the value is decreased compared to the current system. Values larger than zero indicate an improvement (on the revenue part) or a worsening (in case of the cost side). One can observe that the intermediate result's profit is nearly 50% worse compared to the current system. This can be explained by the low revenue and the increased bunker cost by more than 40%. The intermediate network deploys more than 50% less capacity in the network (and also smaller vessels), which leads to decreased port call and time charter costs. The handling and container cost are decreased due to the decreased served cargo flows. Overall, the intermediate network increases the cost by 9.39% compared to the existing network (due to the bunker cost).

The optimized networks found by the metaheuristic indicate an improvement of the current network between 5.6% to 7.7%. The networks transport slightly different cargo flows that are more profitable. However, the total number of cargo flows is only changed marginal (see Table 5.16). This can be explained by the fact that the current system already routes most of the cargo. The optimization tries to increase the cargo volume and decrease the costs. However, Table 5.17 indicates that the optimization mainly focuses on the bunker cost, which is decreased by more than 30%. The handling costs are only changed slightly because the revenue of the current system should be also reached. Overall, network A results in 7.7% decreased cost, whereas B improves the cost by 7.05%.

The case study shows that the developed methods can be used to optimize real-world liner shipping networks. In particular, it indicates that already relatively small subnetworks can provide optimization potential.

6. Integration into a Decision Support System

The scope of this chapter is to integrate the developed mathematical methods into a *decision support system* (DSS). Little (1970) defines a DSS as a "model-based set of procedures for processing data and judgments to assist a manager in decision making". Decision support systems can help planners to investigate more alternatives and get a higher level of confidence in their decision (see Sharda et al. (1988)). The purpose of the DSS presented in this chapter is a proof of concept that shows the technical possibility of supporting planners in adjusting and improving liner networks. The DSS is referred to as the *Liner Network Web Optimizer* (LinWo). Planners can manually create networks for different problem instances (such as the LINER-LIB) and evaluate their network designs using the proposed cargo allocation problem (see Chapter 4). The software checks model constraints that help to validate the manually planned networks (see Section 5.2.1). For example, the transit times and port drafts are checked automatically and the output is given via the web interface. The planners thereby get a direct and fast feedback about the global impacts of regional liner service changes. Furthermore, planners can automatically optimize liner networks using the metaheuristics presented in Chapter 5.

The chapter starts with applying a general DSS structure proposed in literature to the liner shipping network planning. A planning process is presented and implemented in the DSS. Finally, the software architecture and user interface are presented.

6.1. Decision Support System Components

Decision support systems in general consist of the following components (see (Turban and Aronson, 2007, p. 100)):

1. Data management
2. Model management
3. Knowledge-based management
4. User interface

These subsystems are shown in Figure 6.1 and presented for the Liner Network Web Optimizer in the next paragraphs. The *data management* subsystem reads and stores the data of the DSS. For LinWo, the data is read directly from the format

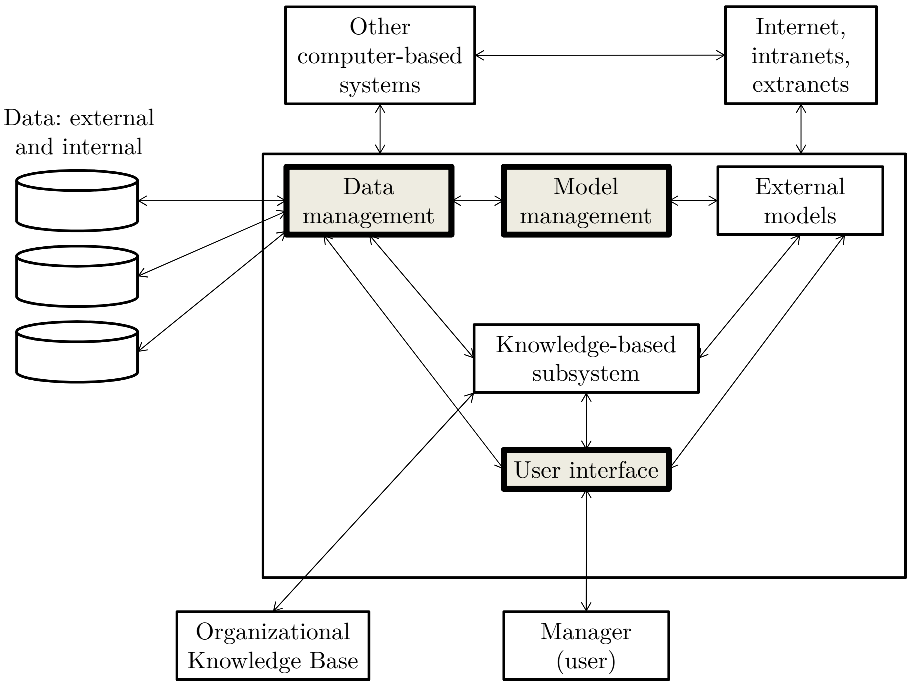


Figure 6.1.: Components of a decision support system, from (Turban and Aronson, 2007, p. 100). The implemented components for the DSS at hand are marked bold.

provided by the LINER-LIB[1]. However, the data can also be read from a database, for example to provide an external interface to the data.

The data management subsystem is used by the *model management* and the *user interface*. In the scope of the DSS developed in this thesis, the knowledge-based management subsystem is not implemented due to the missing business specific requirements. In general, the knowledge-based management component can either work independently or support the subsystems such as the organizational knowledge base with knowledge interfered from the quantitative models.

The model management component contains quantitative models to provide analytical capabilities for the overall system. For LinWo, all methods developed in this thesis are part of the model management component. The cargo allocation problem presented in Chapter 4 uses ports, demands and vessel types on a given network to evaluate it. The data is accessed by the data management subsystem. The cargo allocation problem is extended to calculate the *fitness* of a network (see Chapter

[1]To be able to frequently update the data, it is directly read from the comma separated files provided by the LINER-LIB, see `http:\\linerlib.org` and Brouer et al. (2013).

5.2.2). This extension can be used to provide information on the transit times and embargo constraints for the designed liner network. Finally, the optimization approaches presented in Section 5.2 are also integrated into LinWo. The methods are accessible by the user interface and the data management subsystem that handles the storage of intermediate results.

The users (planners and managers) access the decision support with the user interface. For the liner shipping network planning, users are at least the network planners. Depending on the level of organizational integration of a DSS *product*, the sales or vessel charter department can be involved as well. Users interact with the software through a web-based user interface. The communication is done via a web server that distributes the websites to the client. Implementation details are given in Section 6.3.

In the next section, the business processes arising within liner shipping network planning are described in detail. These processes are supported by the graphical user interface presented in Section 6.4.

6.2. Process Overview

The business process model[2] in Figure 6.2 provides an overview of the process to plan, evaluate and adjust liner networks using the decision support system.

The first step is to select a problem instance for which networks are designed. The problem instances, such as the LINER-LIB benchmark instances, contains all information required to evaluate and optimize a network (see Chapters 4 and 5). For example, port depths, costs, vessel types, cargo flows etc. are contained in each instance.

After selecting an instance, the network planner creates a new network or selects an existing from the database. Networks can be edited by changing the port rotation, number of vessels or the vessel type of an existing liner service. New liner services within a liner shipping network can be created as well. At each step, the network can be evaluated using the cargo allocation problem, which provides the user with information on the profit and costs of the configuration (see Section 6.4).

Based on random initial networks or a set of existing networks, automatic improvement methods can be executed. The user is informed on intermediate networks resulting from the mathematical optimization methods. The improvement process can be terminated at any point in time. Afterwards, he is able to manually adjust parts of the network and reevaluate them using the cargo allocation problem.

With the help of this iterative process, further business constraints can be respected. In the strategic network planning it is assumed that not all aspects can be covered by formal optimization methods. For example, a liner carrier can require to

[2] Business Process Model and Notation (BPMN) is a standard of the Object Management Group to model internal business procedures, see `http:\\www.bpmn.org`.

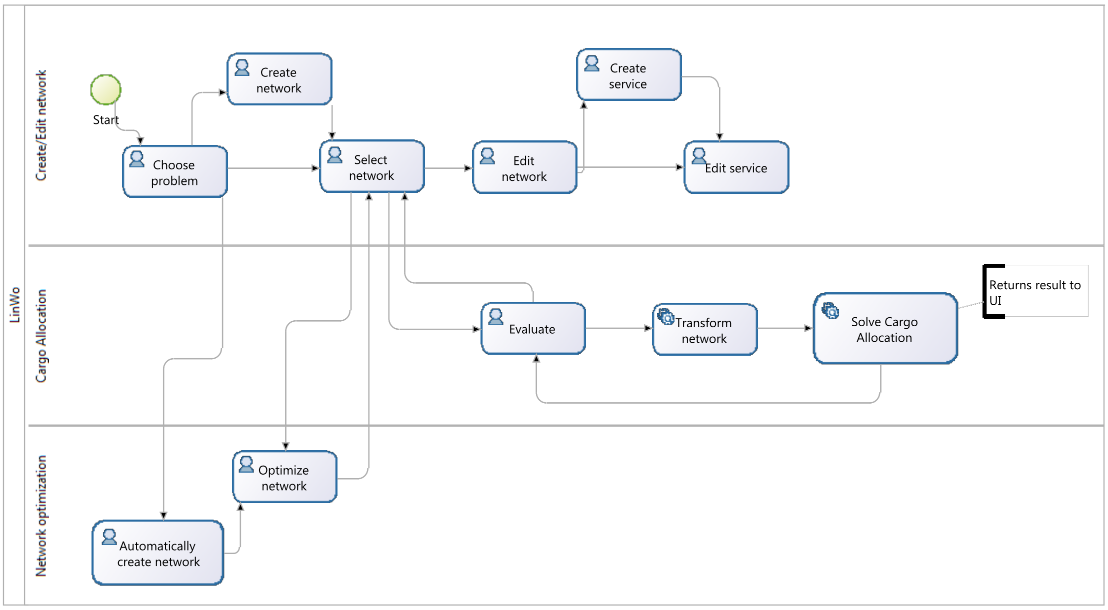


Figure 6.2.: User processes for the liner shipping network planning.

always serve a specific port within a service that has either not been explored by the metaheuristics or evaluated as not profitable to service. These manual adjustments can be easily done in the process presented in Figure 6.2.

6.3. Client-Server Communication

The software is implemented as a web-application, whereas data can be manipulated in the client's browser. This architecture allows the simple deployment of software changes in case of a multi-user environment. The software components are selected to support a wide range of web browsers and operating systems. Thereby, we successfully could run the application also on mobile devices. The cargo allocation and network optimizations are performed on the server side to reduce the clients' hardware requirements.

The used architecture is visualized in Figure 6.3 and uses the Model-View-Control (MVC) paradigm that distinguishes between the (business) model that stores the data, the view that presents this data and the connecting controller layer that is accessed by the view. The architecture differentiates relatively lean data that is sent to the web site, database objects used to persist networks and objects used by the models. Both, the web server and the client work and exchange data based on the same model definition. The server translates between the different models.

The view component in Figure 6.3 uses a single-page application that dynamically loads the data from the server. Unlike dynamic websites, the server provides static

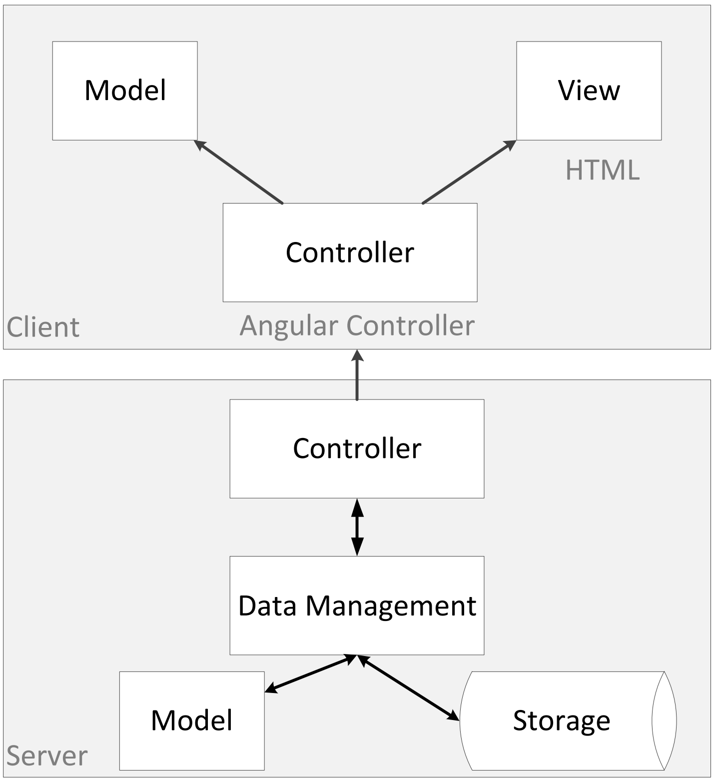


Figure 6.3.: Client-server communication of the decision support system.

HTML websites that are loaded by the client. The website is not dynamically rendered on the server to decrease the computation requirements on the server and to improve the duration to load the site, because clients can cache the static HTML pages. Inside the static HTML site, several partial views exist that are dynamically activated and filled with data from the server using the AngularJS framework from Google (see AngualarJS (2014)). Other JavaScript model view libraries such as KnockoutJS could be used instead of AngularJS. AngularJS 7 ensures the synchronization between JavaScript objects and the visualization in the view. Thus, complex DOM[3] manipulation is done automatically by Angular which reduces the required work to adjust the user interface (UI).

The client controller in Figure 6.3 implements functions used by the website's JavaScript and control functions in the user interface. The server side's controller is responsible to provide CRUD (create, read, update and delete) operations for each object in the model (such as a liner service, a network etc.).

The server provides interfaces based on the Representational State Transfer

[3]Document object model (DOM) is a specification to access HTML objects.

(REST) paradigm (see Fielding (2000)), implemented with the ASP[4] WebAPI framework. With the help of this API, HTTP requests can be done on unique URLs. For example, a call on */api/network/2* returns the existing network with the unique ID 2. The result format can be XML or JSON for example. The controller implements methods for the HTML methods GET, POST, PUT and DELETE. The methods are called asynchronously and a callback function on the client side is executed as soon as the method execution is finished. The client controller then processes the data and the view is updated. The advantage of this technology and communication is the operation system independence. For most of the platforms and programming language, libraries to retrieve and parse XML or JSON exist. Thereby, the server's data can also be retrieved by a thin clients (such as smart phones or tablet computers), by a Java Swing or .NET WCF client application. This provides a high degree of extensibility. In the scope of the proof of concept, messages have to be send from the server to the client, for example to inform about new optimization results or the solution progress. To simplify the asynchronous communication between the web browser and the server, SignalR[5] is used. It automatically uses the best technology available on the client side to send messages from the server. When recent browsers are used, the new technology Websockets is used.

The graphical user interface is implemented using the Twitter Bootstrap 3[6] framework. The framework provides a wide range of controls required for modern web applications. Besides, several extensions for smart phones and table computers exist that enhance the usability of the web application. Furthermore, the framework is adjusted to support all modern browsers. To simplify the CSS[7] handling, LESS 4 is used as preprocessor and Leaflet 5[8] with OpenStreetMap data to visualize the world map and the liner services. On the server side, a ASP.NET MVC 4 application is used that communicates with a MS SQL 2008 database using the Entity Framework 5. The ASP sites can be hosted on an IIS[9] server that supports server side events.

In Figure 6.4, an example liner service calling three ports is shown. The drawing of routes between two ports is not supported in OpenStreetMaps or in any other open source solution by the time of this thesis. Therefore, a system to calculate the routes between two ports had to be developed. The algorithm used is presented in the next section. Basically, the algorithm overlays the world map with a node grid of variable accuracy, removes the nodes that intersect with land masses and identifies the nodes closest to the ports. Afterwards, a shortest path between all

[4]Active server pages (ASP) is a technology developed by Microsoft to dynamically generate websites.
[5]SignalR can be downloaded at `http://signalr.net/`
[6]The Bootstrap framework is available at `http://getbootstrap.com`
[7]Cascading style sheets (CSS) is a standardized language to define the layout of documents, such as HTML websites.
[8]Leaflet is available at `http://leafletjs.com`
[9]Internet Information Services (IIS) is a web server and application platform from Microsoft.

port pairs is calculated using the algorithm of Dijkstra (1959). Each shortest path can be used to calculate the distance in nautical miles but also the way points on the world grid between two ports. The shortest paths between all port pairs are calculated once, stored in a database and automatically read when a liner service is manipulated on the website.

6.4. Graphical User Interface

An important feature of the decision support system is the graphical visualization of the networks. Figure 6.4 provides an overview of the user interface for the liner shipping network decision support system LinWo.

In Figure 6.4, the ports of the LINER-LIB Baltic instance are shown. The user can get information for these ports, such as depth, cost and UN location code, by clicking on the markers. Furthermore, the view point of the map can be arbitrarily changed with drag and drop. An example network has been created for the Baltic region that consists of two liner services: The east Baltic service (EBS) and the west Baltic service (EBS). The legs of the liner services are created using the method described in appendix E. The way points for the legs drawn in Figure 6.4 indicate the approximate routes of the vessels and are used to distinguish the legs. Note that the distances between the ports are calculated using a much higher resolution of the map.

An existing liner shipping network can be either modified or evaluated. Figure 6.5 shows the solution of the cargo allocation problem presented in Chapter 4. Different costs are shown to the user, providing an immediate feedback of the network quality on a monetary basis. The results of the cargo allocation can be transferred to other IT systems using the Microsoft Excel format. The document contains detailed information on the container paths through the network, transhipment volumes and vessel speeds per leg. This results in a proforma vessel schedule whose implementation can be discussed with other involved departments, such as the vessel charter department.

Figure 6.6 shows the graphical support to modify existing liner services. For example, the port of Gdynia in Poland should be added to the north bound direction of service EBS, before the call of Kaliningrad in Russia. Therefore, the user clicks on the edit button of the EBS service and selects Gdynia as new port. The port can be found using the UN location codes or port names. After selecting the new port, it is added to the last position of the selected service. Note that the legs are automatically refreshed on the website to provide an immediate feedback of the changes in the port rotation. The user can arbitrarily rearrange the ports within the port rotation using drag and drop. Again, the legs are automatically refreshed. Finally, network planners can commit their changes by saving them using the data management subsystem (see Figure 6.6).

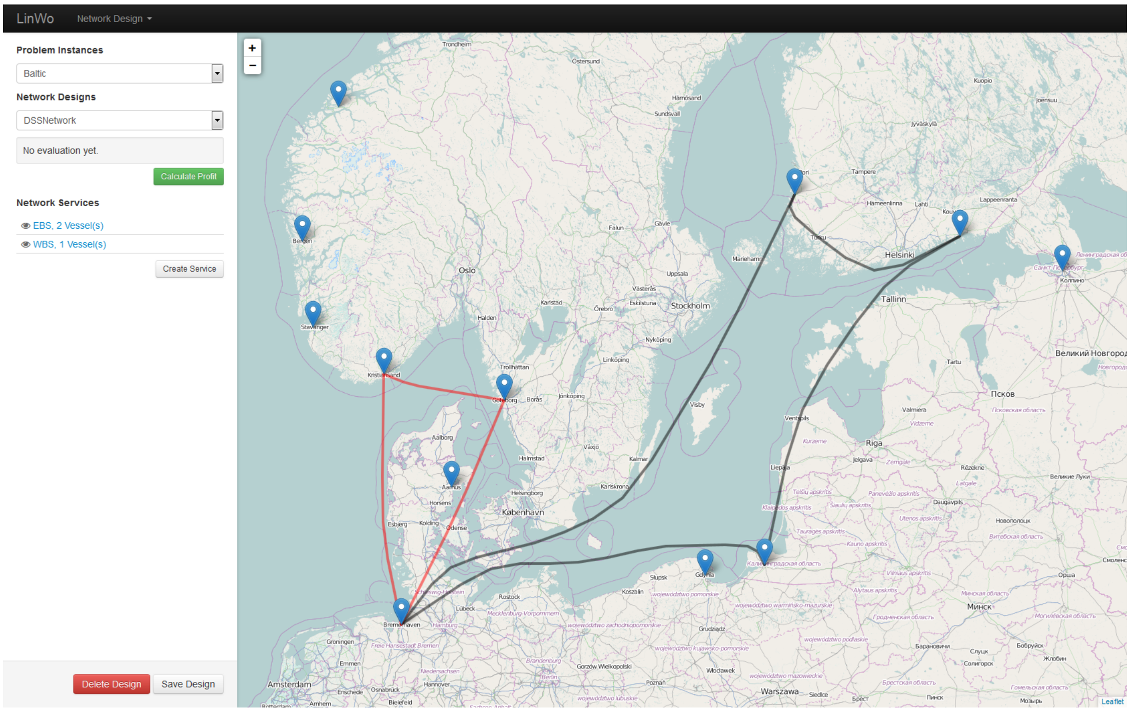


Figure 6.4.: Graphical user interface for the decision support system.

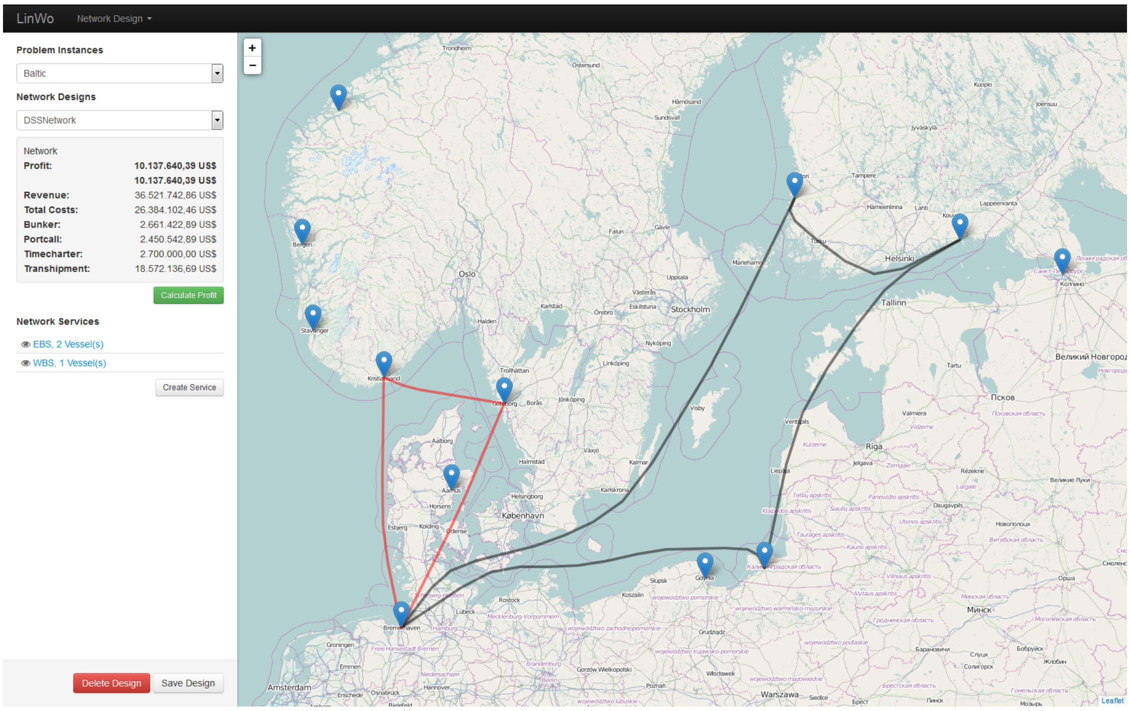
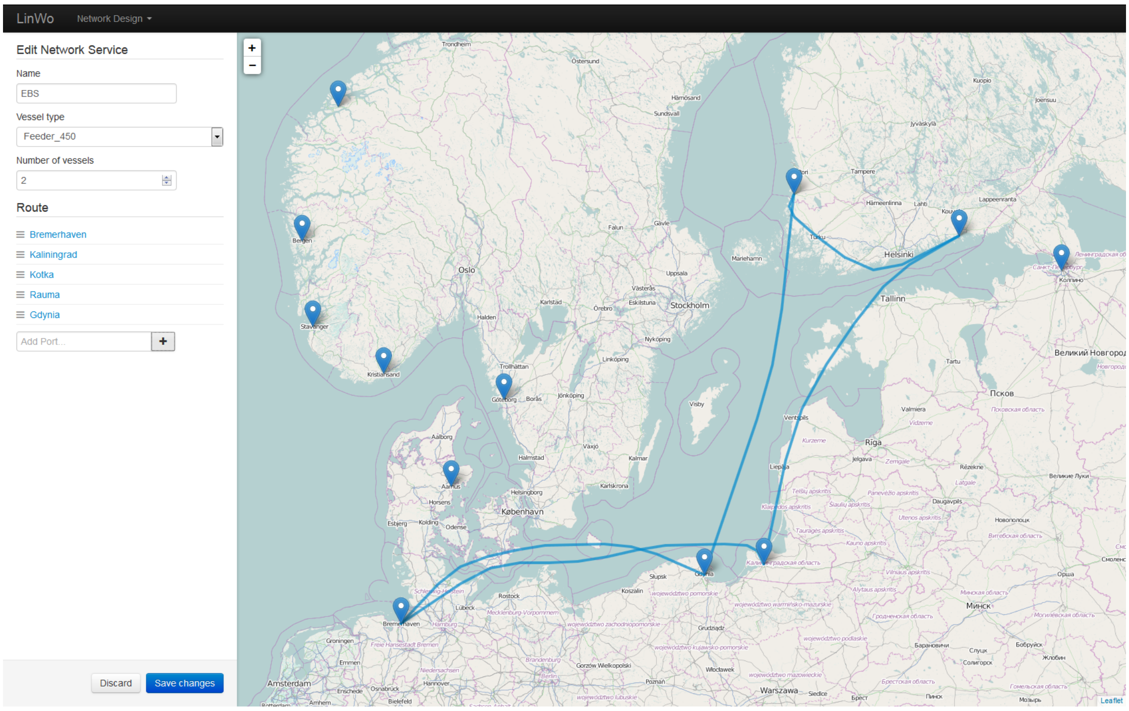
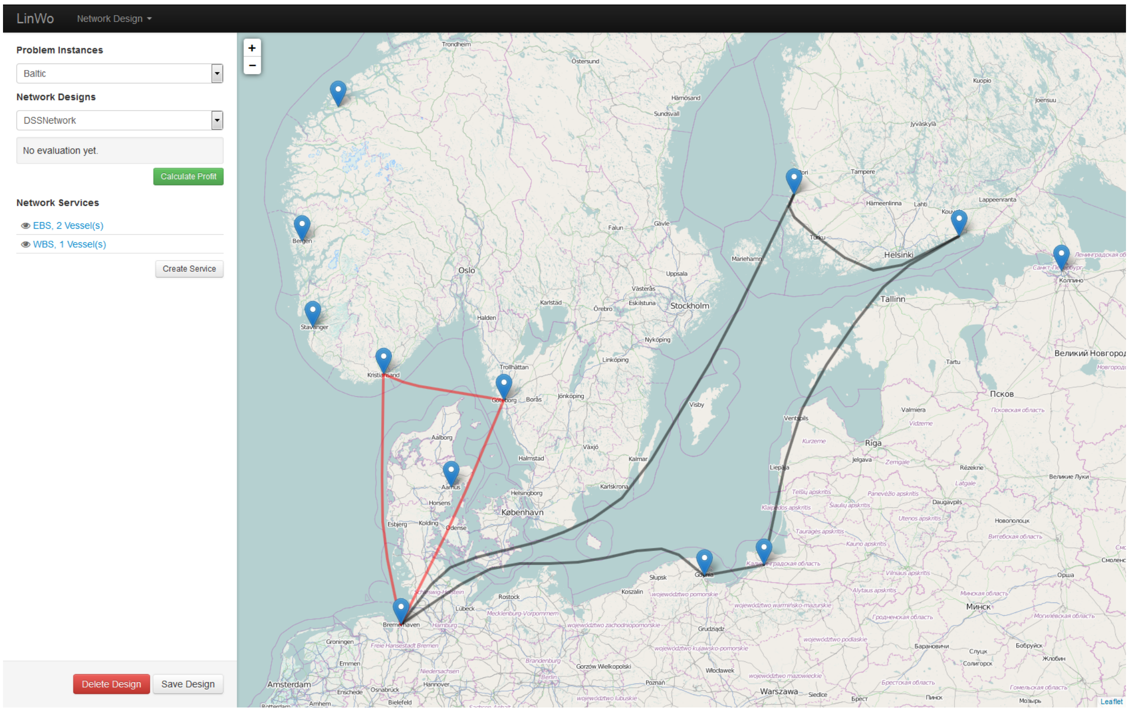


Figure 6.5.: Evaluating an existing liner shipping network.



(a) Port added.



(b) Modified liner shipping network (sequence changed).

Figure 6.6.: Modification of the port rotation of an existing liner service.

Editing a liner service can also involve changing the vessel type. Figure 6.7(a) shows the user interface to select another type using a combo box that provides the vessel types defined in the current problem instance. Again the user commits the modified service by saving the changes. Now, one can reevaluate the liner network using the cargo allocation. Note that extensive validation methods are implemented within this thesis to ensure the validity of the resulting liner service. As a result, the cargo allocation problem not only returns the solution and its profit and costs, but also results from the instance validation. In Figure 6.7(b) an validation error is shown: The selected vessel type's lightship draft (draft without any load, see Chapter 2) is too high to enter the port of Kaliningrad. This intermediate feedback on network changes is supposed to help planning liner shipping networks.

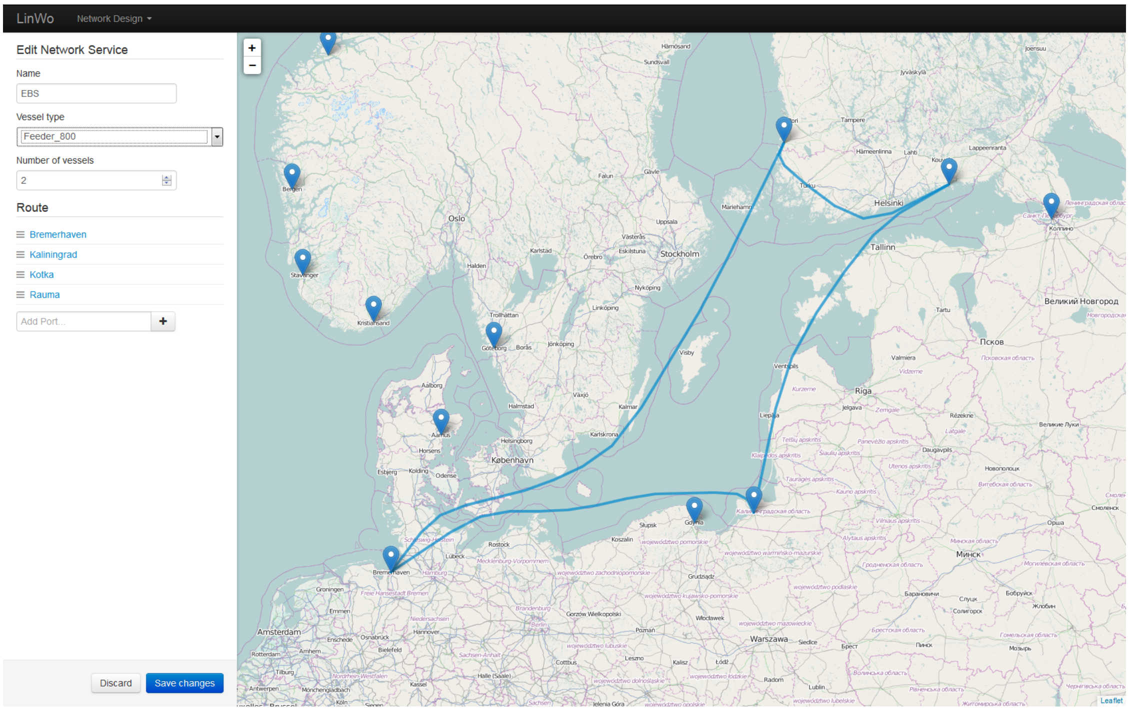
In the scope of this chapter, a proof of concept for a liner shipping decision support system has been developed. For a real-world application, several extensions should be considered: Providing timing and utilization aspects directly in the network drawn on the map or visualize the intermediate solutions from optimization algorithms. Furthermore, more planning problems from the literature, such as the empty container repositioning or bunker optimization, can be included in the decision support system.

The design of the architecture respects the scalability and extendability of the decision support system. The back-end is currently a Microsoft IIS Webserver and not able to perform concurrent network optimizations. It is expected that already two parallel optimizations will slow down the server. To avoid this problem, the architecture can be expanded to delegate the optimization requests to different optimization processes, preferably to different physical machines. This could be implemented using an agent-based back-end architecture, for example with a message queuing system for the communication. In .NET, the agents could use the library MassTransit[10] together with the messaging system RabbitMQ[11]. This allows high scalability in multi-user, multi-optimization scenarios. Optimization-agents could be started on demand and the messaging system could automatically perform a load balancing of the optimization requests.

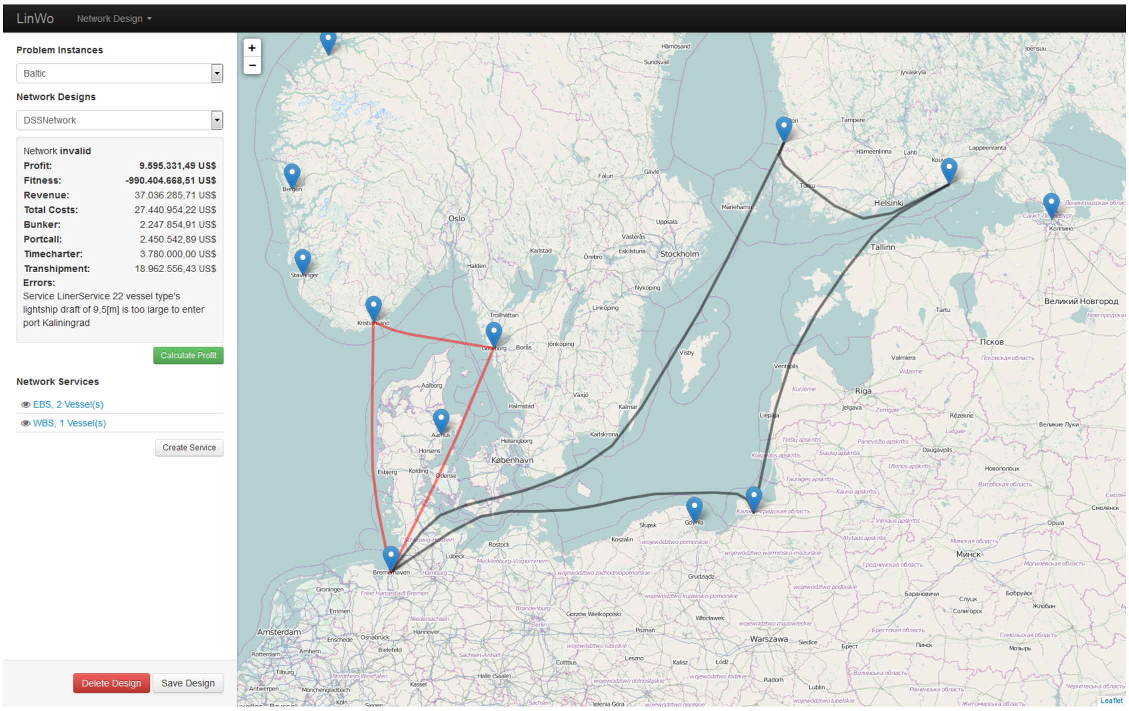
Regarding the process definition, further analysis on the different roles of the decision support system can be done. Planning liner shipping networks is a highly interwoven problem, connected with different liner carrier departments: Charter, sales and line managers provide data for this problem. This aspect can be integrated into a knowledge base proposed by Turban and Aronson (2007). It is expected that implementing such as decision support system for liner carriers can highly improve the planning and optimization process in practice.

[10]MassTransit is available for .NET using NuGet or at `http://masstransit-project.com`

[11]The message queuing system RabbitMQ can be downloaded at `http://www.rabbitmq.com`



(a) Changing vessel type.



(b) Reevaluating network.

Figure 6.7.: Changing a liner service's vessel type and reevaluating the whole network.

7. Conclusion

This chapter concludes the thesis by summarizing the work, performing a critical assessment of the goals and providing an outlook on future research opportunities.

7.1. Summary

Chapter 2 introduced real-world requirements to the liner shipping network design problem. These requirements were specified based on the current state-of-the-art in literature but also with experts from the liner shipping industry. Relevant planning aspects include complex route types, demand properties, timing aspects, cooperation in liner shipping and empty container repositioning. The chapter also defines the scope of the network design problem used in this thesis.

Chapter 3 reviews existing research. It showed that the network design problem is based on fundamental problems that exist in literature since decades. Following, work in the field of maritime research was analyzed. In particular, literature on the liner shipping network design, the cargo allocation (container routing), speed optimization and empty container repositioning problem were reviewed.

An increasing research interest in the liner shipping network design problem could be observed. The publications to tackle this problem are twofold: Some authors solve very small instances of the network design problem to optimality, others focus on optimizing large scale networks using metaheuristics without the possibility of evaluating the solution quality due to missing optimal solutions. Altogether, the methods still lack of practical applicability because important components, such as cargo transportation durations (transit times), speed optimization and partner network integration, are not considered yet.

Optimization methods to automatically improve liner shipping networks must deal with the cargo allocation (sub)problem. The container paths resulting from the cargo allocation problem determine some of the important costs, more specifically, the transhipment and fuel consumption costs. Chapter 3 showed that the state-of-the-art in cargo allocation already includes some of the real-world requirements (such as empty container repositioning) but also lacks of the integration of major aspects such as integrated speed optimization, draft constraints and partner integration. The chapter concludes the state-of-the-art by outlining the research gap and deriving the goals of this thesis.

Chapter 4 presented two mixed integer models with a linearized cubic fuel consumption function to solve the cargo allocation problem. The models integrated speed optimization, draft constraints, capacities of operated and partner services and empty container repositioning. To the best of our knowledge, this has not been done before in literature.

This thesis proved that the mathematical model can be solved to optimality using a linear programming relaxation of the mixed integer model. For further speed up, a novel column generation solution approach was proposed for the cargo allocation problem. The methods were evaluated on 140 networks of the LINER-LIB benchmark suite[1].

Chapter 5 introduced solution approaches for the more complex liner shipping network design problem. The approaches consider transit times between different ports to create competitive liner networks. To the best of our knowledge, multiple transit times across different services have not been considered in the liner network design before.

First, a mixed integer formulation to solve the liner shipping network design problem to optimality was presented. Second, two novel metaheuristics were proposed and evaluated on four of the LINER-LIB instances: A hybrid evolutionary algorithm (EA) and a variable neighborhood search (VNS). To further speed up the algorithms' convergence behavior, a fitness approximation (surrogate) approach known from the engineering context has been applied to the metaheuristics. To the best of our knowledge, surrogates have not been used in a linear optimization problem in the field of logistics or maritime research before. After the metaheuristics were evaluated, a sensitivity analysis to assess the impact of different bunker prices on the liner shipping networks design was performed. Closing Chapter 5, the solution approaches were evaluated in a real-world case study of a global liner carrier.

Finally, Chapter 6 integrated the solution approaches into a decision support system for the liner shipping network planning. Using the web system, networks can be adjusted using drag and drop functions. Then, using the cargo allocation problem, the adjustments can be evaluated. At any point of time, the network can be automatically optimized using the developed metaheuristics for the liner shipping network design problem. Afterwards, manual changes can be made on these optimized solutions to respect qualitative constraints.

7.2. Critical Assessment

At the beginning of this thesis in Section 1.1 we stated four research objectives that were defined in more detail in Section 3.5. Summing up, the goals were (1) to

[1] The LINER-LIB benchmark suite is introduced by Brouer et al. (2013) and contains data for ports, demands and vessel types based on Maersk Line

evaluate large scale, real-world liner networks, (2) to formalize practical requirements of liner networks and generate optimal solutions, (3) to develop metaheuristics to automatically optimize medium-sized liner networks in reasonable computational time, and (4) to integrate the methods into a prototypical decision support system. From our perspective these objectives have been achieved.

In Chapter 4 we presented a column generation based solution approach to tackle the first goal. The numerical results showed that the method can be used to solve large scale cargo allocation problems subject to real-world constraints in a few seconds to optimality. With the help of the proposed solution methods, planners are able to quickly assess a network's profit by determining the optimal cargo allocation and vessel speed. This highly relevant problem can be used by liner carriers to assess alternative networks very quickly. The container path generation subproblem, as part of the column generation method, is very flexible regarding future extensions, such as cabotage limitations.

With respect to the second and third goal, Chapter 5 presented optimal and heuristic approaches to solve the liner shipping network design problem. Practical requirements to optimize liner networks were successfully formalized as a mixed integer program and only very small instances were solved to optimality. Therefore, metaheuristics were developed to optimize medium-sized liner networks in a reasonable amount of computing time. In Chapter 5 is was shown that both, medium sized artificial instances from the LINER-LIB and real-world networks, could be optimized within several hours. The numerical results indicated that small networks in even relatively mature markets can still be optimized within several hours. The results of the solution methods were discussed and evaluated with network planners from a global liner carrier.

The developed planning methods were successfully integrated into a prototypical decision support system in Chapter 6 to realize goal four. We successfully developed an algorithm to enable the visualization of liner shipping networks on open source maps. This enables the use of the mathematical models to the real-world application and enhances the usability of the proposed methods.

Finally, we would like to conclude the goal achievement with a few critics. First, the mixed integer program was not able to solve the smallest instances of the LINER-LIB with a sufficient number of services and layers to optimality. The reasons have been discussed in Section 5.1.3. This made it difficult to assess the solution quality of our metaheuristics. However, with the help of the lower bounds and the application to the real-world network we claim that liner shipping networks can be optimized with the proposed methods.

Second, in reality there exist an interdependency between network changes and the available cargo flow. Calling different ports within a region can attract new cargo flows that are not provided as input for the model. Thus, once a new liner network has been designed the cargo flows should be reevaluated with experts from the liner carrier.

7.3. Future Research Opportunities

The solution methods in this thesis are developed for the context of liner shipping. Some of the aspects, for example the cargo allocation and speed optimization might be applicable to other industries. However, fuel consumption and speed optimization play a secondary role in many planning problems, such as public transportation. Not until fuel prices increase a substantial amount, decreasing the speed on roadways is of interest.

Extensions to the cargo allocation problem, such as time periods, berth windows and cargo flow dependent transit times would be an interesting future research. In operations, the port arrival and destination time is fixed and time window constraints exist. Furthermore, tide dependent drafts often impose a challenge on when to enter or leave a port to keep the schedule. These constraints require the modeling of liner service schedules with specific departure days. This enables the development of methods to create robust schedules.

First steps towards the consideration of cargo flow dependent transit times and integrated speed optimization are done by Guericke and Tierney (2014). They enable leg dependent speed optimization which might support the model to hold the transit times. Integrating their model into the fitness calculation of liner network is an interesting research opportunity and should be accompanied by developing heuristic solution methods.

The liner shipping network design problem could be solved with a reasonable objective value for medium sized (sub)networks of a global liner network in this thesis. Future research should improve the mixed integer model formulation or use advanced solution methods to solve the liner shipping network design problem with transit times to optimality. Additionally, large scale, real-world liner networks still provide a computational challenge for the future.

Another research opportunity is to further evaluate the fitness approximation approach. To obtain a more profound knowledge whether the approach is beneficial in general, it should be evaluated in other planning problems in the field of discrete optimization. Additional research can be done in the theoretical analysis of fitness approximations.

Bibliography

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

R. Agarwal and Ö. Ergun. Network design and allocation mechanisms for carrier alliances in liner shipping. *Operations research*, 58(6):1726–1742, 2010.

Alphaliner Newsletter, Volume 1, 2011. Operated Capacity January 2010 and 2011, 4th August 2014. URL `http://www.alphaliner.com`.

Alphaliner Newsletter, Volume 22. Carriers' Operating Margins vs. Revenue: 1st Quarter 2014, 13th August 2014. URL `http://www.alphaliner.com`.

Alphaliner Newsletter, Volume 26, 2013. East-West Carrier Alliances: How do they stack up?, 4th August 2014. URL `http://www.alphaliner.com/`.

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

M.W. Andersen. *Service Network Design and Management in Liner Container Shipping Applications*. PhD thesis, Department of Transport, Technical University of Denmark, 2010.

AngualarJS. Google inc., 4th August 2014. URL `https://angularjs.org`.

C. Ansótegui, M. Sellmann, and K. Tierney. A gender-based genetic algorithm for the automatic configuration of algorithms. In *Principles and Practice of Constraint Programming-CP 2009*, pages 142–157. Springer, 2009.

L.H. Appelgren. A column generation algorithm for a ship scheduling problem. *Transportation Science*, 3(1):53, 1969.

L.H. Appelgren. Integer programming methods for a vessel scheduling problem. *Transportation Science*, 5(1):64, 1971.

C. Archetti, M.G. Speranza, and A. Hertz. A tabu search algorithm for the split delivery vehicle routing problem. *Transportation Science*, 40(1):64–73, 2006.

C. Audet, V. Béchard, and S. Le Digabel. Nonsmooth optimization through mesh adaptive direct search and variable neighborhood search. *Journal of Global Optimization*, 41(2):299–318, 2008.

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

A.L. Ayre. *The Theory and Design of British Shipbuilding*. Thomas Reed and Co., LTD., 1885.

A.J. Baird. Containerization and the decline of the upstream urban port in europe. *Maritime policy and Management*, 23(2):145–156, 1996.

B.M. Baker and M.A. Ayechew. A genetic algorithm for the vehicle routing problem. *Computers & Operations Research*, 30(5):787–800, 2003.

R.H. Ballou. *Business Logistics Management*, volume 4. Prentice-Hall, 1999.

BARON. Baron nonconvex minlp solver, 4th August 2014. URL `http://archimedes.cheme.cmu.edu/?q=baron`.

C.B. Barrass. *Ship Stability for Masters and Mates*, volume 6. Butterworth-Heinemann, 2006.

J. Basedow, U. Magnus, and R. Wolfrum. *The Hamburg Lectures on Maritime Affairs 2009 and 2010*. Springer, 2012.

S. Basu. Tabu search implementation on traveling salesman problem and its variations: A literature survey. *American Journal of Operations Research*, 2(2):163–173, 2012.

J.E. Beasley and P.C. Chu. A genetic algorithm for the set covering problem. *European Journal of Operational Research*, 94(2):392–404, 1996.

T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219, 2006.

M. Bellmore and G.L. Nemhauser. The traveling salesman problem: a survey. *Operations Research*, 16(3):538–558, 1968.

H.B. Bendall and A.F. Stent. On measuring cargo handling productivity. *Maritime Policy and Management*, 14(4):337–343, 1987.

P.R. Bevington and D.K. Robinson. *Data reduction and error analysis for the physical sciences*, volume 2. McGraw-Hill, 1969.

M. Boile, S. Theofanis, A. Baveja, and N. Mittal. Regional repositioning of empty containers: Case for inland depots. *Transportation Research Record: Journal of the Transportation Research Board*, 2066(1):31–40, 2008.

A.J. Booker, J.E. Dennis Jr, P.D. Frank, D.B. Serafini, V. Torczon, and M.W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. *Structural optimization*, 17(1):1–13, 1999.

O. Bräysy. A reactive variable neighborhood search for the vehicle-routing problem with time windows. *INFORMS Journal on Computing*, 15(4):347–368, 2003.

M.R. Brooks. *Sea Change in Liner Shipping*. Pergamon, 2000.

B.D. Brouer and G. Desaulniers. A matheuristic for the liner shipping network design problem. In *Electronic Proceedings of LOGMS 2012*, 2012.

B.D. Brouer, D. Pisinger, and S. Spoorendonk. Liner shipping cargo allocation with repositioning of empty containers. *INFOR: Information Systems and Operational Research*, 49(2):109–124, 2011.

B.D. Brouer, J. Dirksen, D. Pisinger, C.E.M. Plum, and B. Vaaben. The vessel schedule recovery problem (vsrp)-a mip model for handling disruptions in liner shipping. *European Journal of Operational Research*, 2012.

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

Bunker Index. Bunker price index, 4th August 2013. URL `http://www.bunkerindex.com`.

Bunker Index. Bunker index: Price index, 3rd August 2014. URL `http://www.bunkerindex.com/prices/bixfree.php?priceindex_id=1`.

A. Busch and W. Dangelmaier. *Integriertes Supply Chain Management*. Gabler, 2002.

I.L. Buxton and G.H. Stephenson. Evaluating design for upgradeability: A simulation based approach for ships and marine products. In *Practical Design of Ships and Other Floating Structures. Proceedings of the Eighth International Symposium on Practical Design of Ships and Other Floating Strucutres*, volume 1, 2001.

R.H. Byrd, J. Nocedal, and R.A. Waltz. Knitro: An integrated package for nonlinear optimization. In *Large-scale nonlinear optimization*, pages 35–59. Springer, 2006.

Canal de Panamá. Panama canal expansion program progress report, 1st August 2014. URL `http://www.pancanal.com/eng/expansion/index.html`.

C. Chen and Q. Zeng. Designing container shipping network under changing demand and freight rates. *Transport*, 25(1):46–57, 2010.

P. Chen, H. Huang, and X.Y. Dong. Iterated variable neighborhood descent algorithm for the capacitated vehicle routing problem. *Expert Systems with Applications*, 37(2):1620–1627, 2010.

S.C. Cho and A.N. Perakis. Optimal liner fleet routeing strategies. *Maritime Policy and Management*, 23(3):249–259, 1996.

C. Chou, S. Kimbrough, J. Sullivan-Fedock, C.J. Woodard, and F.H. Murphy. Using interactive evolutionary computation (iec) with validated surrogate fitness functions for redistricting. In *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference*, pages 1071–1078. ACM, 2012.

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

M. Christiansen, K. Fagerholt, B. Nygreen, and D. Ronen. Ship routing and scheduling in the new millennium. *European Journal of Operational Research*, 228(3): 467–483, 2012.

V. Chvátal. *Linear programming*. Macmillan, 1983.

J.F. Cordeau, G. Laporte, J.Y. Potvin, and M.W.P. Savelsbergh. Transportation on demand. *Transportation, Handbooks in operations research and management science*, 14:429–466, 2004.

T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, et al. *Introduction to algorithms*, volume 2. MIT press Cambridge, 2001.

Couenne. Couenne nonconvex minlp solver, 4th August 2014. URL `https://projects.coin-or.org/Couenne`.

T.G. Crainic, M. Gendreau, and P. Dejax. Dynamic and stochastic models for the allocation of empty containers. *Operations Research*, 41(1):102–126, 1993.

G.A. Croes. A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812, 1958.

G.B. Dantzig and J.H. Ramser. The truck dispatching problem. *Management science*, 6(1):80–91, 1959.

M. Davarynejad, J. Vrancken, J. van den Berg, and C.A. Coello. A fitness granulation approach for large-scale structural design optimization. In *Variants of Evolutionary Algorithms for Real-World Applications*, pages 245–280. Springer, 2012.

A. Delgado, R.M. Jensen, K. Janstrup, T.H. Rose, and K.H. Andersen. A constraint programming model for fast optimal stowage of container vessel bays. *European Journal of Operational Research*, 220(1):251–261, 2012.

G. Desaulniers, J. Desrosiers, and M.M. Solomon. *Column generation*, volume 5. Springer, 2005.

M. Di Francesco, T.G. Crainic, and P. Zuddas. The effect of multi-scenario policies on empty container repositioning. *Transportation Research Part E: Logistics and Transportation Review*, 45(5):758–770, 2009.

E.W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

M. Dorigo and L.M. Gambardella. Ant colony system: A cooperative learning approach to the traveling salesman problem. *Evolutionary Computation, IEEE Transactions on*, 1(1):53–66, 1997.

R. Epstein, A. Neely, A. Weintraub, F. Valenzuela, S. Hurtado, G. Gonzalez, A. Beiza, M. Naveas, F. Infante, F. Alarcon, et al. A strategic empty container logistics optimization in a major shipping company. *Interfaces*, 42(1):5–16, 2012.

A.L. Erera, J.C. Morales, and M. Savelsbergh. Robust optimization for empty repositioning problems. *Operations Research*, 57(2):468–483, 2009.

EUROGATE Container Terminal Hamburg GmbH. Prices and conditions from 2013, 4th August 2014. URL `http://www1.eurogate.de/en/Terminals/Hamburg`.

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

K. Fagerholt. Designing optimal routes in a liner shipping problem. *Maritime Policy & Management*, 31(4):259–268, 2004b.

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

K. Fagerholt, G. Laporte, and I. Norstad. Reducing fuel emissions by optimizing speed on shipping routes. *Journal of the Operational Research Society*, 61(3): 523–529, 2009b.

L. Fan, M. Luo, and W.W. Wilson. Pricing joint products in liner shipping. In *International Forum on Shipping, Ports and Airports (IFSPA) 2012: Transport Logistics for Sustainable Growth at a New Level*, 2012.

C.E. Fayle. *A short history of the world's shipping industry.* Taylor & Francis US, 2005.

Federal Maritime Commission. The impact of the ocean shipping reform act of 1998. Technical report, Federal Maritime Commission, 2001.

C.M. Feng and C.H. Chang. Empty container reposition planning for intra-asia liner shipping. *Maritime Policy & Management*, 35(5):469–489, 2008.

R.T. Fielding. *Architectural styles and the design of network-based software architectures.* PhD thesis, University of California, 2000.

S.D.P. Flapper, J.A. van Nunen, and L.N. Van Wassenhove. *Managing closed-loop supply chains.* Springer, 2005.

L. Fleischer and M. Skutella. The quickest multicommodity flow problem. In *Integer Programming and Combinatorial Optimization*, volume 2337, pages 36–53. Springer, 2006.

R.W. Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962.

A. Forrester, A. Sobester, and A. Keane. *Engineering design via surrogate modelling: a practical guide.* John Wiley & Sons, 2008.

Frontline Solvers. Xpress solver engine, 4th August 2014. URL `http://www.solver.com/`.

R.A. Gatica and P.A. Miranda. Special issue on latin-american research: A time based discretization approach for ship routing and scheduling with variable speed. *Networks and Spatial Economics*, 11(3):465–485, 2011.

S. Gelareh and D. Pisinger. Fleet deployment, network design and hub location of liner shipping companies. *Transportation Research Part E: Logistics and Transportation Review*, 47(6):947–964, 2011.

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

S. Gelareh, R. Neamatian Monemi, P. Mahey, N. Maculan, and D. Pisinger. Single string planning problem arising in liner shipping industries: A heuristic approach. *Computers & Operations Research*, 40(10):2357–2373, 2013.

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

M. Gendreau, G. Laporte, and R. Séguin. A tabu search heuristic for the vehicle routing problem with stochastic demands and customers. *Operations Research*, 44(3):469–477, 1996.

P.C. Gilmore and R.E. Gomory. A linear programming approach to the cutting-stock problem. *Operations research*, 9(6):849–859, 1961.

F. Glover and G.A. Kochenberger. *Handbook in Metaheuristics*. Kluwer Academic Publishers, 2003.

M. Goetschalckx and C. Jacobs-Blecha. The vehicle routing problem with backhauls. *European Journal of Operational Research*, 42(1):39–51, 1989.

C.K. Goh and Y. Tenne. *Computational intelligence in expensive optimization problems*, volume 2. Springer, 2010.

J.F. Gonçalves, J.J. de Magalhães Mendes, and M. G.C. Resende. A hybrid genetic algorithm for the job shop scheduling problem. *European Journal of Operational Research*, 167(1):77–95, 2005.

S. Guericke and L. Suhl. Liner network design under consideration of transit times and partner networks. In *International Conference on Logistics and Maritime Systems 2013*, 2013.

S. Guericke and K. Tierney. Liner shipping cargo allocation with service levels and speed optimization. Technical report, University of Paderborn, DS&OR working paper no. 1403, Submitted to EJOR, 2014.

Gurobi. Gurobi optimization, 4th August 2014. URL `http://www.gurobi.com/`.

P. Hansen and N. Mladenović. *Variable neighborhood search*. Springer, 2003.

J.A. Hartigan and M.A. Wong. Algorithm as 136: A k-means clustering algorithm. *Applied statistics*, 28(1):100–108, 1979.

V.C. Hemmelmayr, K.F. Doerner, and R. F. Hartl. A variable neighborhood search heuristic for periodic routing problems. *European Journal of Operational Research*, 195(3):791–802, 2009a.

V.C. Hemmelmayr, K.F. Doerner, and R.F. Hartl. A variable neighborhood search heuristic for periodic routing problems. *European Journal of Operational Research*, 195(3):791–802, 2009b.

H. Hernández-Pérez and J.J. Salazar-González. The one-commodity pickup-and-delivery travelling salesman problem. In *Combinatorial Optimization—Eureka*, pages 89–104. Springer, 2003.

H. Hernández-Pérez and J.J. Salazar-González. Heuristics for the one-commodity pickup-and-delivery traveling salesman problem. *Transportation Science*, 38(2): 245–255, 2004.

H. Hernández-Pérez and J.J. Salazar-González. The multi-commodity one-to-one pickup-and-delivery traveling salesman problem. *European Journal of Operational Research*, 196(3):987–995, 2009.

I. Hilger. *Konzeption und Implementierung von Variable-Neighborhood-Suchstrategien für das Netzwerkdesignproblem in der Linienschifffahrt.* Master Thesis, University of Paderborn, 2014.

J.H. Holland. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence.* A Bradford Book, 1975.

J.H. Holland. *Adaptation in natural and artificial systems.* MIT Press, Cambridge, MA, 1992.

C.I. Hsu and Y.P. Hsieh. Direct versus terminal routing on a maritime hub-and-spoke container network. *Journal of Marine Science and Technology*, 13(3):209–217, 2005.

L.M. Hvattum, I. Norstad, K. Fagerholt, and G. Laporte. Analysis of an exact algorithm for the vessel speed optimization problem. *Networks*, 82(2):132–135, 2013.

IBM CPLEX. IBM ILOG CPLEX Optimizer, 4th August 2014. URL `http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer`.

G.B. Ihde. *Transport, Verkehr, Logistik*, volume 3. Franz Vahlen, 2001.

A. Imai, K. Sasaki, E. Nishimura, and S. Papadimitriou. Multi-objective simultaneous stowage and load planning for a container ship with container rehandle in yard stacks. *European Journal of Operational Research*, 171(2):373–389, 2006.

A. Imai, K. Shintani, and S. Papadimitriou. Multi-port vs. hub-and-spoke port calls by containerships. *Transportation Research Part E: Logistics and Transportation Review*, 45(5):740–757, 2009.

International Organization for Standardization. ISO 55.180.10 - general purpose containers, 4th August 2014. URL `http://www.iso.org`.

J.J. Jaw, A.R. Odoni, H.N. Psaraftis, and N.HM. Wilson. A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows. *Transportation Research Part B: Methodological*, 20(3):243–257, 1986.

Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft computing*, 9(1):3–12, 2005.

Y. Jin. Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation*, 1(2):61–70, 2011.

Y. Jin, M. Olhofer, and B. Sendhoff. On evolutionary optimization with approximate fitness functions. In *GECCO*, pages 786–793, 2000.

Y. Jin, M. Hüsken, and B. Sendhoff. Quality measures for approximate models in evolutionary computation. In *GECCO*, pages 170–173, 2003.

L. Jourdan, C. Dhaenens, and E.G. Talbi. Using datamining techniques to help metaheuristics: A short survey. In *Hybrid Metaheuristics*, pages 57–69. Springer, 2006.

B. Kallehauge, J. Larsen, O.B.G. Madsen, and M.M. Solomon. Vehicle routing problem with time windows. *Column Generation*, pages 67–98, 2005.

A.J. Keane. Experiences with optimizers in structural design. In *Proceedings of the conference on adaptive computing in engineering design and control*, volume 94, pages 14–27, 1994.

H.J. Kim. A lagrangian heuristic for determining the speed and bunkering port of a ship. *Journal of the Operational Research Society*, 65:747–754, 2013.

H.J. Kim and J.G. Kim. A heuristic algorithm for determining the speed and bunkering port of a ship considering greenhouse gas emissions. In *Proceedings of the Asia Pacific Industrial Engineering and Management Systems Conference*, 2012.

J. Kim and B.R. Moon. A genetic algorithm for linear ordering problem using an approximate fitness evaluation. In *Proceedings of the 2014 conference companion on Genetic and evolutionary computation companion*, pages 1461–1462. ACM, 2014.

S. Kirkpatrick. Optimization by simulated annealing: Quantitative studies. *Journal of statistical physics*, 34(5-6):975–986, 1984.

K.H. Kjeldsen. Classification of ship routing and scheduling problems in liner shipping. *INFOR: Information Systems and Operational Research*, 49(2):139–152, 2011.

K.H. Kjeldsen. *Routing and Scheduling in Liner Shipping.* PhD thesis, Department of Economics and Business, Aarhus University Denmark, 2012.

F. Klug. Logistics implications of global production networks in car manufacturing. In *Institute for Manufacturing (Hrsg.): Cambridge International Manufacturing Symposium, Symposium Proceedings, Cambridge*, pages 13–24. 2010.

J.W. Konings. *The Future of Intermodal Freight Transport: Operations, Design and Policy*. Edward Elgar Publishing, 2008.

R. Kruse, C. Borgelt, F. Klawonn, C. Moewes, M. Steinbrecher, and P. Held. Fundamental evolutionary algorithms. In *Computational Intelligence*, pages 227–274. Springer, 2013.

J. Kytöjoki, T. Nuortio, O. Bräysy, and M. Gendreau. An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers & Operations Research*, 34(9):2743–2757, 2007a.

J. Kytöjoki, T. Nuortio, O. Bräysy, and M. Gendreau. An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers & Operations Research*, 34(9):2743–2757, 2007b.

K.K. Lai, K. Lam, and W.K. Chan. Shipping container logistics and allocation. *Journal of the Operational Research Society*, 46(6):687–697, 1995.

E.L. Lawler, J.K. Lenstra, A.H.G.R. Kan, and D.B. Shmoys. *The traveling salesman problem: a guided tour of combinatorial optimization*, volume 3. Wiley, 1985.

M. Levinson. *The box: How the shipping container made the world smaller and the world economy bigger*. Princeton University Press, 2006.

T.C. Lirn, H.A. Thanopoulou, M.J. Beynon, and A.K.C. Beresford. An application of ahp on transhipment port selection: a global perspective. *Maritime Economics & Logistics*, 6(1):70–91, 2004.

Lloyd's List. The chinese conundrum, 30th July 2014. URL `http://www.lloydslist.com/ll/sector/containers/article445059.ece`.

J. Little. Models and managers: The concept of a decision calculus. *Management Science*, 16(8):466–485, 1970.

C. Liu, Z. Jiang, F. Chen, X. Liu, L. Liu, and Z. Xu. Empty container repositioning-a review. In *Intelligent Control and Automation (WCICA), 2010 8th World Congress on*, pages 3028–3033. IEEE, 2010.

H.A. Lu, C.W. Chu, P.Y. Che, et al. Slot allocation planning for an alliance service with ship fleet sharing. *Asia Pacific Management Reviews*, 15:325–339, 2010.

M.E. Lübbecke and J. Desrosiers. Selected topics in column generation. *Operations Research*, 53(6):1007–1023, 2005.

Y.H.V. Lun, K.H. Lai, and T.C.E. Cheng. *Shipping and Logistics Management*. Springer, 2010.

Maersk Line. The world's largest ship, 4th August 2014. URL `http://www.worldslargestship.com/facts/the-worlds-largest-ship-2/`.

M. Malchow and A. Kanafani. A disaggregate analysis of factors influencing port selection. *Maritime Policy & Management*, 28(3):265–277, 2001.

D.C. Mattfeld. *The Management of Transshipment Terminals:Decision Support for Terminal Operations in Finished Vehicle Supply Chains*. Springer, 2006.

Q. Meng and S. Wang. Liner shipping service network design with empty container repositioning. *Transportation Research Part E: Logistics and Transportation Review*, 47(5):695–708, 2011a.

Q. Meng and T. Wang. A scenario-based dynamic programming model for multi-period liner ship fleet planning. *Transportation Research Part E: Logistics and Transportation Review*, 47(4):401 – 413, 2011b.

Q. Meng, S. Wang, H. Andersson, and K. Thun. Containership routing and scheduling in liner shipping: overview and future research directions. *Transportation Science*, 2013.

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

U. Mhaky and C.Y. Lee. Service level assignment and container routing for liner shipping service networks. In *Proceedings of the International Multi-Conference of Engineers and Computer Scientists*, volume 2, 2014.

Z. Michalewicz and D.B. Fogel. *How to Solve It: Modern Heuristics*, volume 2. Springer, 2004.

R. Midoro and A. Pitto. A critical evaluation of strategic alliances in liner shipping. *Maritime Policy & Management*, 27(1):31–40, 2000.

R. Midoro, E. Musso, and F. Parola. Maritime liner shipping and the stevedoring industry: market structure and competition strategies. *Maritime Policy & Management*, 32(2):89–106, 2005.

C.E. Miller, A.W. Tucker, and R.A. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, 7(4):326–329, 1960.

S. Mitrovic-Minic and G. Laporte. The pickup and delivery problem with time windows and transshipment. *Infor-Information Systems and Operational Research*, 44(3):217–228, 2006.

N. Mittal. *Regional empty marine container management*. ProQuest, 2008.

N. Mladenović and P. Hansen. Variable neighborhood search. *Computers & Operations Research*, 24(11):1097–1100, 1997.

P. Moscato et al. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. *Caltech concurrent computation program, C3P Report*, 826:1989, 1989.

J. Mulder and R. Dekker. Methods for strategic liner shipping network design. *European Journal of Operational Research*, 235(2):367–377, 2013.

H. Nazif and L.S. Lee. Optimised crossover genetic algorithm for capacitated vehicle routing problem. *Applied Mathematical Modelling*, 36(5):2110–2117, 2012.

G.L. Nemhauser and L.A. Wolsey. *Integer and combinatorial optimization*. Wiley-Interscience series in discrete mathematics, 1999.

E.K. Norlund and I. Gribkovskaia. Reducing emissions through speed optimization in supply vessel operations. *Transportation Research Part D: Transport and Environment*, 23:105–113, 2013.

I. Norstad, K. Fagerholt, and G. Laporte. Tramp ship routing and scheduling with speed optimization. *Transportation Research Part C: Emerging Technologies*, 2010.

I. Norstad, K. Fagerholt, L.M. Hvattum, H.S. Arnulf, and A. Bjørkli. Maritime fleet deployment with voyage separation requirements. *Flexible Services and Manufacturing Journal*, pages 1–20, 2013.

T. Notteboom and P. Cariou. Fuel surcharge practices of container shipping lines: Is it about cost recovery or revenue-making. In *Proceedings of the International Association of Maritime Economists Conference, Copenhagen*, pages 24–26, 2009.

T. Notteboom and J.P. Rodrigue. Re-assessing port-hinterland relationships in the context of global commodity chains. *Ports, Cities and Global Supply Chains*, pages 51–66, 2007.

T. Notteboom and J.P. Rodrigue. Containerisation, box logistics and global supply chains: The integration of ports and liner shipping networks. *Maritime Economics & Logistics*, 10:152–174, 2008.

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

T.E. Notteboom and B. Vernimmen. The effect of high fuel costs on liner service configuration in container shipping. *Journal of Transport Geography*, 17(5):325 – 337, 2009.

A. Orden. The transhipment problem. *Management science*, 2(3):276–285, 1956.

I.H. Osman and G. Laporte. Metaheuristics: A bibliography. *Annals of Operations Research*, 63(5):511–623, 1996.

D. Pacino. An lns approach for container stowage multi-port master planning. In *Computational Logistics*, pages 35–44. Springer, 2013.

M. Padberg. Approximating seperable nonlinear functions via mixed zero-one programs. *Operations Research Letters*, 27(1):1–5, 2000.

P.M. Panayides and D.W. Song. Maritime logistics as an emerging discipline. *Maritime Policy & Management*, 40(3):295–308, 2013.

P.M. Panayides and R. Wiedmer. Strategic alliances in container liner shipping. *Research in Transportation Economics*, 2011.

V. Papapanagiotou, D. Weyland, R. Montemanni, and L.M. Gambardella. A sampling-based approximation of the objective function of the orienteering problem with stochastic travel and service times. *Lecture Notes in Management Science*, 5:143–152, 2013.

V. Papapanagiotou, R. Montemanni, and L.M. Gambardella. Objective function evaluation methods for the orienteering problem with stochastic travel and service times. *Journal of Applied Operational Research*, 6(1):16–29, 2014.

U. Pape. Ein heuristisches verfahren zur ermittlung von fahrplänen in der containerschiffahrt. *Operations-Research-Spektrum*, 1(3):169–181, 1980.

S.N. Parragh, K.F. Doerner, and R.F. Hartl. Variable neighborhood search for the dial-a-ride problem. *Computers & Operations Research*, 37(6):1129–1138, 2010.

C.E.M. Plum, D. Pisinger, J.J. Salazar-González, and M.M. Sigurd. The multi-commodity one-to-one pickup-and-delivery traveling salesman problem with path duration limits. In *Proceedings of the International MultiConference of Engineers and Computer Scientists*, volume 2, 2012.

C.E.M. Plum, D. Pisinger, J.J. Salazar-González, and M.M. Sigurd. Single liner shipping service design. *Computers & Operations Research*, 2013a.

C.E.M. Plum, D. Pisinger, and M.M. Sigurd. A service flow model for the liner shipping network design problem. *European Journal of Operational Research*, 2013b.

O. Polat. *Designing Liner Shipping Feeder Service Networks in the New Era of Mega Containerships*. PhD thesis, Technical University Berlin, 2013.

B.J. Powell and A.N. Perkins. Fleet deployment optimization for liner shipping: an integer programming model. *Maritime Policy & Management*, 24(2):183–192, 1997.

H.N. Psaraftis and C.A. Kontovas. Speed models for energy-efficient maritime transportation: A taxonomy and survey. *Transportation Research Part C: Emerging Technologies*, 26:331–351, 2013.

X. Qi and D.P. Song. Minimizing fuel emissions by optimizing vessel schedules in liner shipping with uncertain port times. *Transportation Research Part E: Logistics and Transportation Review*, 48(4):863–880, 2012.

K. Rana and RG Vickson. A model and solution algorithm for optimal routing of a time-chartered containership. *Transportation Science*, 22(2):83, 1988.

K. Rana and RG Vickson. Routing container ships using lagrangean relaxation and decomposition. *Transportation Science*, 25(3):201, 1991.

A. Ratle. Accelerating the convergence of evolutionary algorithms by fitness landscape approximation. In *Parallel Problem Solving from Nature—PPSN V*, pages 87–96. Springer, 1998.

C.R. Reeves. *Modern heuristic techniques for combinatorial problems.* John Wiley & Sons, Inc., 1993.

L.B. Reinhardt and D. Pisinger. A branch and cut algorithm for the container shipping network design problem. *Flexible Services and Manufacturing Journal*, 24(3):1–26, 2010.

L.B. Reinhardt, B. Kallehauge, A. Nørrelund, and A. Olsen. *Network Design Models for Container Shipping.* Centre for Traffic and Transport, Technical University of Denmark, 2007.

J. Renaud, G. Laporte, and F.F. Boctor. A tabu search heuristic for the multi-depot vehicle routing problem. *Computers & Operations Research*, 23(3):229–235, 1996.

Drewry Maritime Research. *Reefer Shipping Market Annual Review and Forecast 2013/14*. Drewry, 2013.

J.P. Rodrigue. *Factors impacting North American freight distribution in view of the panama canal expansion*. Van Horne Institute, 2010.

J.P. Rodrigue. *The Geography of Transport Systems*, volume 3. Routledge, 2013.

D. Ronen. The effect of oil price on the optimal speed of ships. *Journal of the Operational Research Society*, 33(11):1035–1040, 1982.

D. Ronen. Short-term scheduling of vessels for shipping bulk or semi-bulk commodities originating in a single area. *Operations Research*, pages 164–173, 1986.

D. Ronen. Ship scheduling: The last decade. *European Journal of Operational Research*, 71(3):325–333, 1993.

N Saeidi, H Jafari, A Ameli, and N Zaersoleymani. Container repositioning management in liner shipping industry. *Management Science Letters*, 3(6):1795–1804, 2013.

M. Şahin, G. Çavuşlar, T. Öncan, G. Şahin, and D. Tüzün Aksu. An efficient heuristic for the multi-vehicle one-to-one pickup and delivery problem with split loads. *Transportation Research Part C: Emerging Technologies*, 27:169–188, 2012.

Y. Sano and H. Kita. Optimization of noisy fitness functions by means of genetic algorithms using history of search. In *Parallel Problem Solving from Nature PPSN VI*, pages 571–580. Springer, 2000.

A. Schieck. *Internationale Logistik: Objekte, Prozesse und Infrastrukturen grenzüberschreitender Güterströme*. Oldenbourg Verlag, 2008.

G. Schmidt and W.E. Wilhelm. Strategic, tactical and operational decisions in multi-national logistics networks: a review and discussion of modelling issues. *International Journal of Production Research*, 38(7):1501–1523, 2000.

A. Schönknecht. *Maritime Containerlogistik: Leistungsvergleich von Containerschiffen in intermodalen Transportketten*. Springer, 2009.

A. Shahrokhi and A. Jahangirian. A surrogate assisted evolutionary optimization method with application to the transonic airfoil design. *Engineering Optimization*, 42(6):497–515, 2010.

J.S. Shang and C.K. Cuff. Multicriteria pickup and delivery problem with transfer opportunity. *Computers & industrial engineering*, 30(4):631–645, 1996.

R. Sharda, S.H. Barr, and J.C. MCDonnell. Decision support system effectiveness: a review and an empirical test. *Management science*, 34(2):139–159, 1988.

W.S. Shen and C.M. Khoong. A DSS for empty container distribution planning. *Decision Support Systems*, 15(1):75–82, 1995.

X. Sheng, L.H. Lee, and E.P. Chew. Dynamic determination of vessel speed and selection of bunkering ports for liner shipping under stochastic environment. *OR Spectrum*, 36(2):455–480, 2013.

K. Shintani, A. Imai, E. Nishimura, and S. Papadimitriou. The container shipping network design problem with empty container repositioning. *Transportation Research Part E: Logistics and Transportation Review*, 43(1):39–59, 2007.

Shipping Watch. Analyst firms examine cartel ban, 1st August 2014. URL `http://shippingwatch.com/carriers/article5709489.ece`.

H.K. Singh, T. Ray, and W. Smith. Surrogate assisted simulated annealing (sasa) for constrained multi-objective optimization. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–8. IEEE, 2010.

B. Slack. Containerization, inter-port competition, and port selection. *Maritime Policy and Management*, 12(4):293–303, 1985.

D. Song, J. Zhang, J. Carter, T. Field, J. Marshall, J. Polak, K. Schumacher, P. Sinha-Ray, and J. Woods. On cost-efficiency of the global container shipping network. *Maritime Policy & Management*, 32(1):15–30, 2005.

D.P. Song and J. Carter. Empty container repositioning in liner shipping. *Maritime Policy & Management*, 36(4):291–307, 2009.

D.P. Song and J.X. Dong. Effectiveness of an empty container repositioning policy with flexible destination ports. *Transport Policy*, 18(1):92–101, 2011.

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

M. Srinivas and L.M. Patnaik. Genetic algorithms: A survey. *Computer*, 27(6): 17–26, 1994.

M. Stopford. *Maritime economics*, volume 3. Taylor & Francis, 2009.

Suez Canal Authority. Canal characteristics, 1st July 2014. URL `http://www.suezcanal.gov.eg/sc.aspx?show=12`.

L. Suhl and T. Mellouli. *Optimierungssysteme: Modelle, Verfahren, Software, Anwendungen*, volume 3. Springer, 2013.

C. Sys. Is the container liner shipping industry an oligopoly? *Transport policy*, 16 (5):259–270, 2009.

H. Takagi and H. Iba. Interactive evolutionary computation. *New Generation Computing*, 23(2):113–114, 2005.

A.S. Tasan and M. Gen. A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries. *Computers & Industrial Engineering*, 62(3):755–761, 2012.

K. Tierney and R.M. Jensen. The liner shipping fleet repositioning problem with cargo flows. In *Computational Logistics*, volume 4555, pages 1–16. Springer, 2012.

K. Tierney, B. Áskelsdóttir, R. M. Jensen, and D. Pisinger. Solving the liner shipping fleet repositioning problem with cargo flows. *Transportation Science*, 2014.

P. Toth and D. Vigo. An exact algorithm for the vehicle routing problem with backhauls. *Transportation science*, 31(4):372–385, 1997.

P. Toth and D. Vigo. Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123(1):487–512, 2002.

N.K. Tran and H.D. Haasis. Literature survey of network optimization in container liner shipping. *Flexible Services and Manufacturing Journal*, pages 1–41, 2013.

E. Turban and J.E. Aronson. *Decision Support Systems and Intelligent Systems*, volume 7. Prentice Hall, 2007.

UN. United nations commodity trade 2012, 4th August 2014. URL `http://comtrade.un.org`.

UN Conference on Trade and Development. Review of maritime transport 2013. Technical report, United Nations, 2013.

P. Vansteenwegen, W. Souffriau, and D.V. Oudheusden. The orienteering problem: A survey. *European Journal of Operational Research*, 209(1):1–10, 2011.

C. Vilhelmsen, R. Lusby, and J. Larsen. Tramp ship routing and scheduling with integrated bunker optimization. *EURO Journal on Transportation and Logistics*, pages 1–33, 2013.

S. Wang. Essential elements in tactical planning models for container liner shipping. *Transportation Research Part B: Methodological*, 54:84–99, 2013.

S. Wang. A novel hybrid-link-based container routing model. *Transportation Research Part E: Logistics and Transportation Review*, 61:165–175, 2014.

S. Wang and Q. Meng. Schedule design and container routing in liner shipping. *Transportation Research Record: Journal of the Transportation Research Board*, 2222(-1):25–33, 2011.

S. Wang and Q. Meng. Liner ship route schedule design with sea contingency time and port time uncertainty. *Transportation Research Part B: Methodological*, 46 (5):615–633, 2012a.

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

S. Wang and Q. Meng. Sailing speed optimization for container ships in a liner shipping network. *Transportation Research Part E: Logistics and Transportation Review*, 48(3):701–714, 2012c.

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

S. Wang and Q. Meng. Liner shipping network design with deadlines. *Computers & Operations Research*, 41:140–149, 2014.

S. Wang, Q. Meng, and Z. Liu. Bunker consumption optimization methods in shipping: a critical review and extensions. *Transportation Research Part E: Logistics and Transportation Review*, 53:49–62, 2013a.

S. Wang, Q. Meng, and Z. Liu. Containership scheduling with transit-time-sensitive container shipment demand. *Transportation Research Part B: Methodological*, 54: 68–83, 2013b.

S. Wang, Q. Meng, and Z. Sun. Container routing in liner shipping. *Transportation Research Part E: Logistics and Transportation Review*, 49(1):1–7, 2013c.

D. Weyland, R. Montemanni, and L.M. Gambardella. A metaheuristic framework for stochastic combinatorial optimization problems based on gpgpu with a case study on the probabilistic traveling salesman problem with deadlines. *Journal of Parallel and Distributed Computing*, 73(1):74–85, 2013a.

D. Weyland, R. Montemanni, and L.M. Gambardella. An improved heuristic for the probabilistic traveling salesman problem with deadlines based on gpgpu. In *Computer Aided Systems Theory-EUROCAST 2013*, pages 332–339. Springer, 2013b.

V. Windeck. Environmental routing. In *A Liner Shipping Network Design*, pages 39–78. Springer, 2013.

L.A. Wolsey. Integer programming. *IIE Transactions*, 32(273-285):2–58, 2000.

P. Wong and C. Bamford. A study of market structure in liner shipping under the influence of government policies. In *Proceedings of 19th International Business Research Conference*, 2012.

World Port Source. World port source, 3rd August 2014. URL `http://www.worldportsource.com`.

World Shipping Council. Container ship design, 1st August 2014. URL `http://www.worldshipping.org/about-the-industry/liner-ships/container-ship-design`.

www.sea-distances.com. Sea distances - voyage calculator (nautical miles), 4th August 2014. URL `http://sea-distances.com/`.

www.sea-rates.com. Transit time/ distance calculator, 3rd August 2014. URL `http://www.searates.com/reference/portdistance/`.

Y. Xu, G. Lin, and H. Sun. Accounting for the china–us trade imbalance: An ownership-based approach. *Review of International Economics*, 18(3):540–551, 2010.

S. Yan, C.Y. Chen, and S.C. Lin. Ship scheduling and container shipment planning for liners in short-term operations. *Journal of marine science and technology*, 14 (4):417–435, 2009.

Z. Yao, S.H. Ng, and L.H. Lee. A study on bunker fuel management for the shipping liner services. *Computers & Operations Research*, 39(5):1160–1172, 2012.

H. Yi. *Topics in Contract Pricing and Spot Markets*. PhD thesis, Georgia Institute of Technology, 2008.

S. Zurheide and K. Fischer. A revenue management slot allocation model for liner shipping networks. *Maritime Economics & Logistics*, 14(3):334–361, 2012.

List of Figures

List of Tables

List of Algorithms

Appendix A.

Glossary

Bunker	Name for vessel fuel
Cargo flow	A container demand between two specific ports with a specific equipment type, revenue and resource utilization (such as weight)
Deadweight tonnage (tdw)	Measure how many tons a vessel can carry safely, including load, fuel, crew, ballast water
Feeder service	A service used for local or coastal transport and usually connected with main services
Forty foot equivalent unit (FEU)	Standardized container type of 12.19m length, 2.59m height and 2.44m width
Knot (kn)	Unit of speed, equals one nautical mile per hour
Leg	Connection between two ports
Lightship draft	Lightship draft refers to a vessel's draft without any payload
Liner service	Fleet of vessels operating between fixed ports on a regular basis. Consists of a port rotation and a vessel deployment
Long-haul operations	Transocean (such as transatlantic or transpacific) transportation
Longshoreman	Port employee to load and unload vessels

Nautical mile (nm)	Nautical distance unit, approximately 1.852 kilometer
Port call	Vessels call at different ports within a liner service
Port duration	Duration of a vessel's port call
Port rotation	Sequence of ports that perform a round trip. A port rotation can call a port more than once per round trip
Proforma schedule	Weekly port schedule with fixed arrival and departure days offered to the customer by the liner carrier
Shipper	The person or company who is usually the supplier or owner of commodities shipped
Short-haul operations	Local or coastal transportation
Subtour	Round trip that includes two disconnected sub round trips
Trade	A commercial link between two or more markets
Transit time	The duration to transport cargo from one port to another using the underlying network
Twenty foot equivalent unit (TEU)	Standardized container type of 6.06m length, 2.59m height and 2.44m width
Vessel deployment	Part of a liner service that specifies the vessel type and the number of vessels

Appendix B.

Transformation Algorithm for the Layered Network Structure

With the help of Algorithm 32, arbitrary liner services (defined by their leg list L) can be transformed in $\mathcal{O}(|L|)$ since the algorithm iterates through all legs and performs constant time operations inside the for loop. Note that $o(l), d(l)$ gives the origin and destination port of the leg l.

Algorithm 32 works as follows: first, the result list L^A and current origin and destination network layer l_o, l_d and a temporary set L^T, used to check whether a port is visited twice on one layer, is initialized. Afterwards all legs L of the service are iterated. Three different cases must be distinguished to create layered legs from leg i: first, the last visited leg's layer on the round trip must be connected to the first leg's origin layer, here zero. If the destination port of leg l is already visited on the current layer (see line 14), the leg connects the next layer with the current and a layer crossing leg l^C is created and inserted into the return list. Afterwards, the origin is set to the target to continue with legs on the same layer. In the last case (see line 26), a new leg is created on the layer indices l_o, l_d and added to the result list. Finally the result list is returned.

Algorithm 10: CreateLayeredLegs

Input: An ordered list of legs,
$L = \{(i, i+1) : i, i+1 \in P\}, (i,j), (k,l) \in L : j = k$

Output: The list of layered legs $L^L = \{(i,j,l,l') : (i,j) \in L\}$

$L^A = \emptyset$;
$l_o = l_d = 0$;
$L^T = \emptyset$;
for $i = 0; i < |L|; i++$ **do**
    $l = L_i$;
    if $i = 0$ **then**
        $L^T \leftarrow L^T \cup o(l)$;
    end
    if $i = |L_s| - 1$ **then**
        $l^B = (o(l), d(l), l_d, 0)$;
        $L^A \leftarrow L^A \cup \{l^B\}$;
        continue;
    end
    if $L^T \backslash \{d(l)\} \neq \emptyset$ **then**
        $l_d \leftarrow l_d + 1$;
        $l^C = (o(l), d(l), l_o, l_d)$;
        $L^A \leftarrow L^A \cup l^C$;
        $l_o = l_d$;
        $L^T = \{o(l)\}$;
        continue;
    end
    if $L^T \backslash \{o(l)\} \neq \emptyset$ **then**
        $L^T \leftarrow L^T \cup \{o(l)\}$;
    end
    $L^T \leftarrow L^T \cup \{d(l)\}$;
    $l = (o(l), d(l), l_o, l_d)$;
    if $l_o = l_d$ **then**
        $L^A \leftarrow L^A \cup \{l\}$;
    end
    $l_o = l_d$;
end
return L^A;

Appendix C.

Extended Numerical Results for the Integrated Cargo Allocation Problem

C.1. Problem Instances

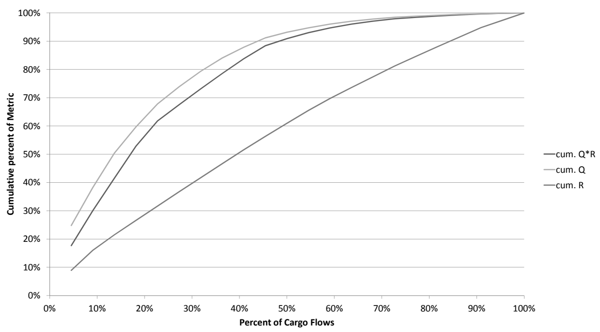


Figure C.1.: Distribution of the cargo flow quantity (cum. Q), revenue (cum. R) and quantity * revenue (cum. Q*R) for the LINER-LIB Baltic instance.

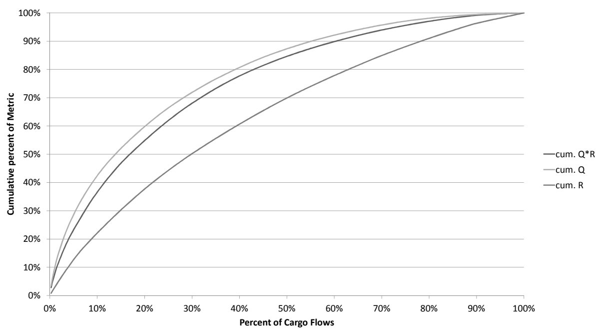


Figure C.2.: Distribution of the cargo flow quantity (cum. Q), revenue (cum. R) and quantity * revenue (cum. Q*R) for the LINER-LIB Mediterranean instance.

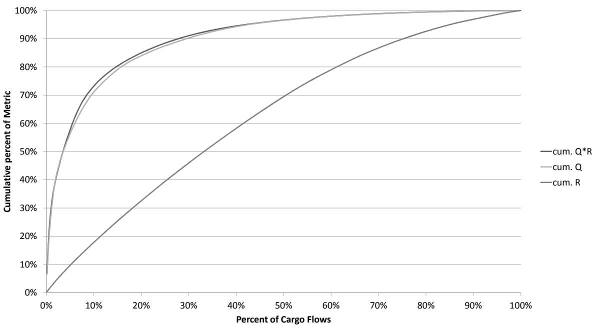


Figure C.3.: Distribution of the cargo flow quantity (cum. Q), revenue (cum. R) and quantity * revenue (cum. Q*R) for the LINER-LIB Pacific instance.

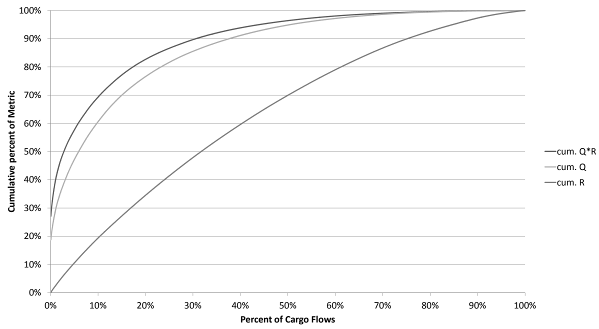


Figure C.4.: Distribution of the cargo flow quantity (cum. Q), revenue (cum. R) and quantity * revenue (cum. Q*R) for the LINER-LIB WorldSmall instance.

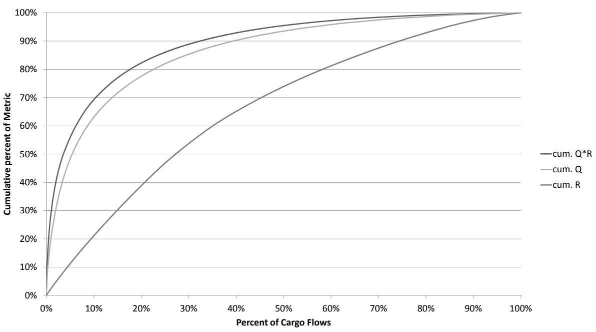


Figure C.5.: Distribution of the cargo flow quantity (cum. Q), revenue (cum. R) and quantity * revenue (cum. Q*R) for the LINER-LIB Europe-Asia instance.

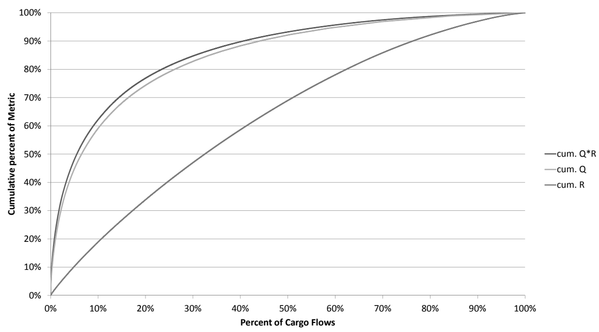


Figure C.6.: Distribution of the cargo flow quantity (cum. Q), revenue (cum. R) and quantity * revenue (cum. Q*R) for the LINER-LIB WorldLarge instance.

C.2. Cargo Allocation Approximations

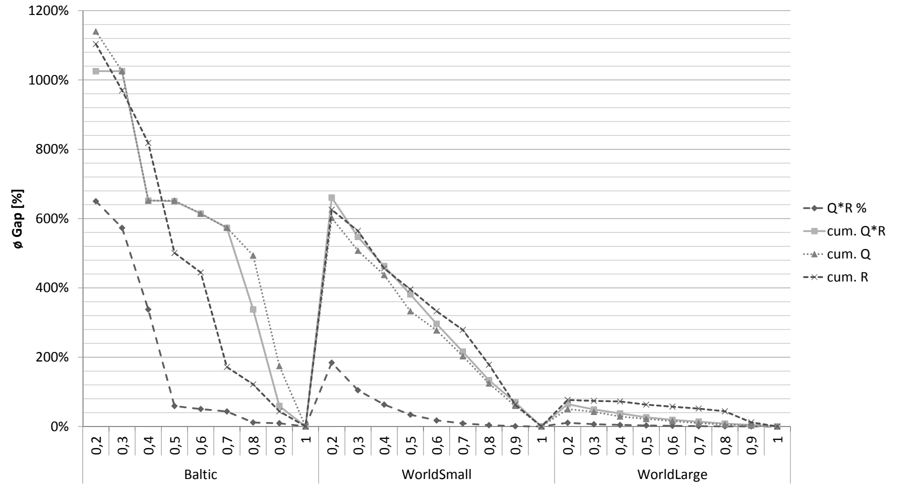


Figure C.7.: Gap in percent for varied cargo flow amount and different cargo flow selection strategies (for the Baltic, WorldSmall and WorldLarge).

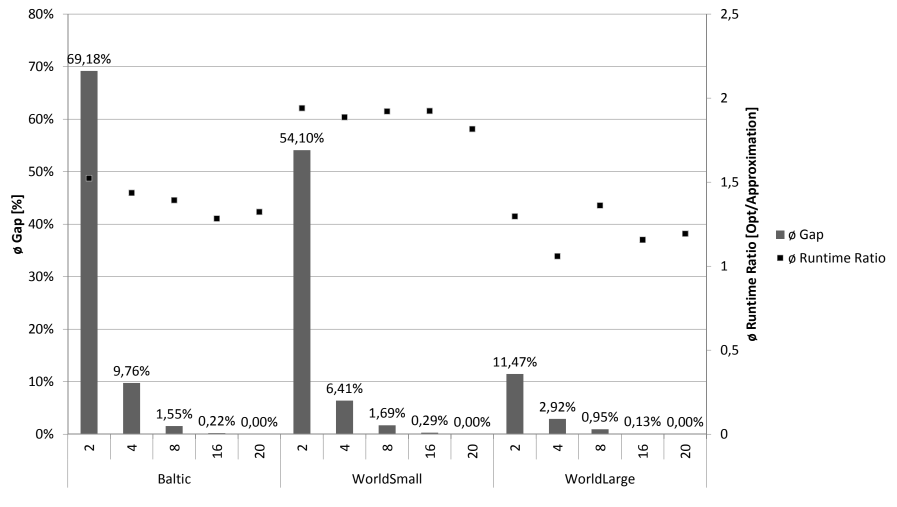


Figure C.8.: Effects of the support points on the objective function and the runtime (for the Baltic, WorldSmall and WorldLarge instances).

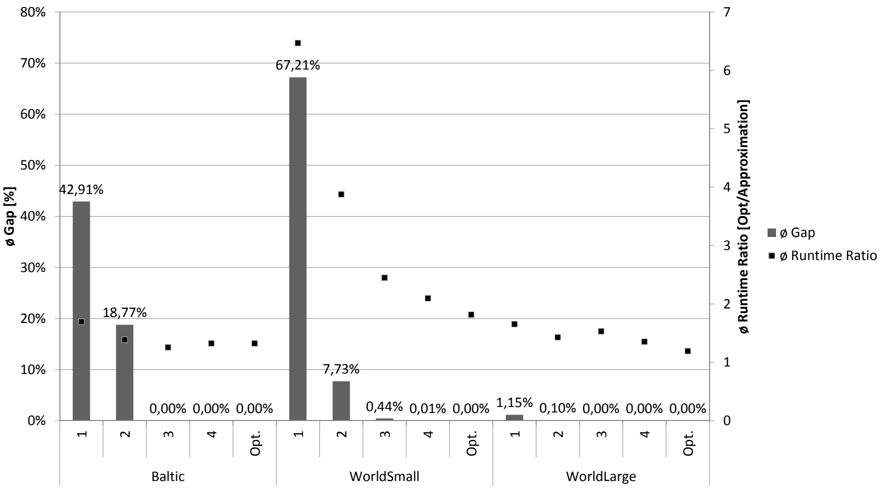


Figure C.9.: Gap and runtime improvement when terminating the Column Generation solution approach prematurely (for the Baltic, WorldSmall and WorldLarge instance).

Appendix D.

Extended Numerical Results for the Liner Shipping Network Design Problem

D.1. Liner Shipping Network MIP Formulation

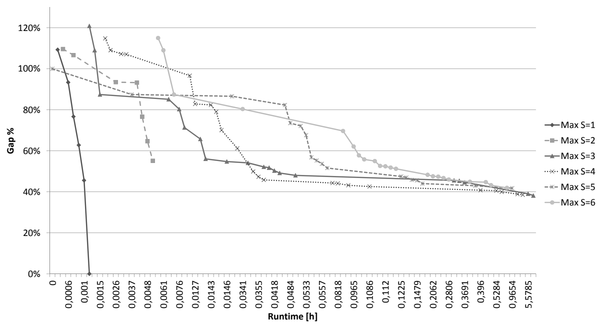


Figure D.1.: Baltic LINER-LIB 2012 instance with maximum of one vessel per service, no complex route type and no transit times.

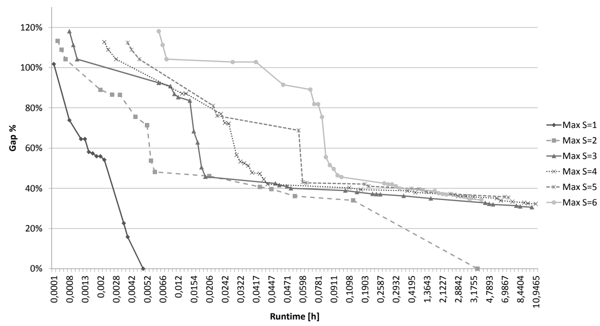


Figure D.2.: Baltic LINER-LIB 2012 instance with maximum of two vessel per service, no complex route type and no transit times.

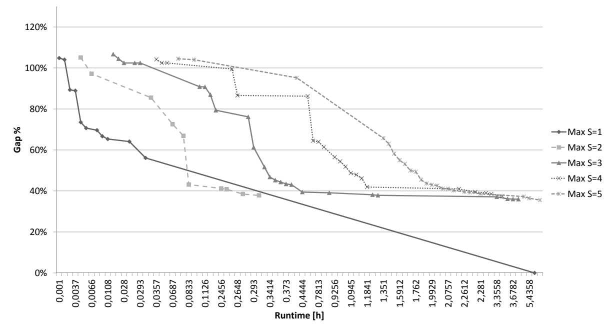


Figure D.3.: Baltic LINER-LIB 2012 instance with maximum of two vessel per service, complex route types allowed and no transit times.

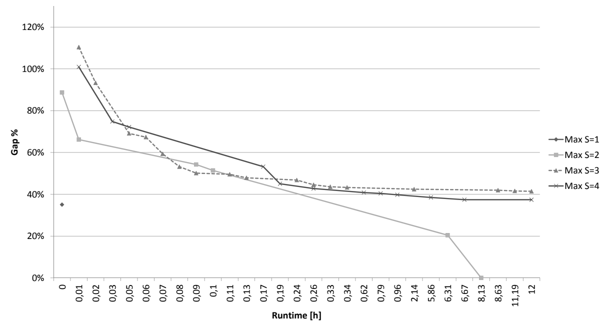
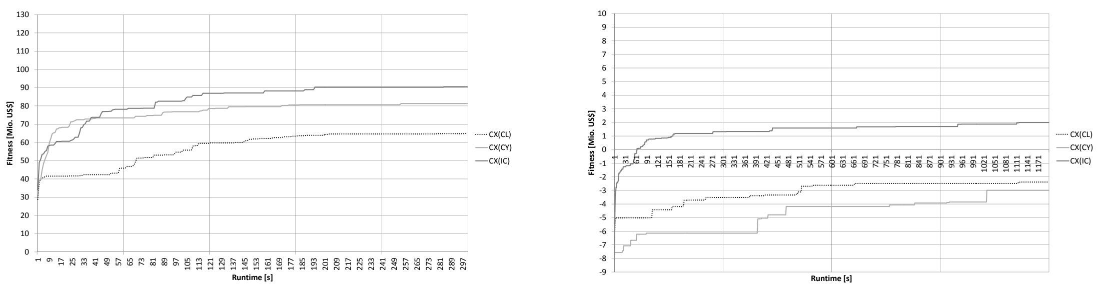


Figure D.4.: Baltic LINER-LIB 2012 instance with maximum of one vessel per service, no complex route type but transit times.

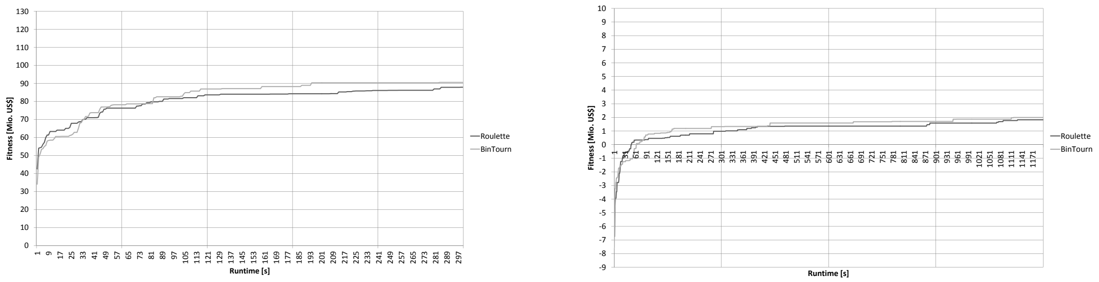
D.2. Parameter Tuning for the Evolutionary Algorithm



(a) WAF instance.

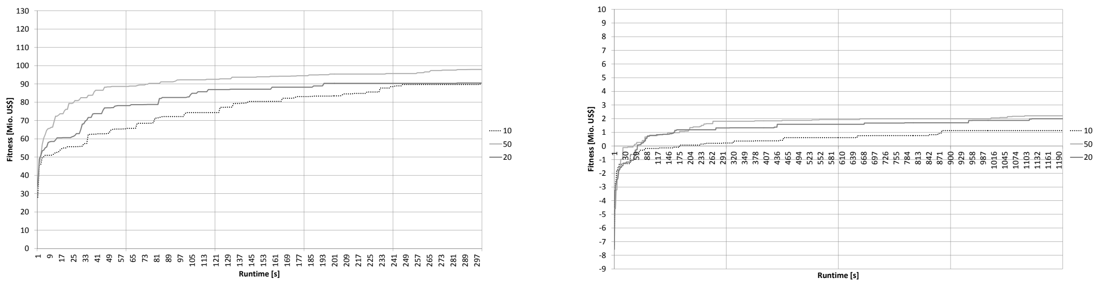
(b) Mediterranean instance.

Figure D.5.: Convergence of the evolutionary algorithm for LINER-LIB instances using the clustering CX(CL), the cycle CX(CY) and informed cycle CX(IC) crossover methods (averaged over five runs).



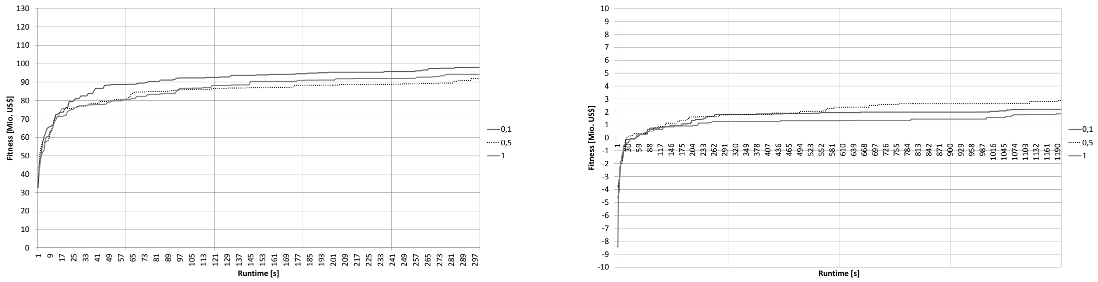
(a) WAF instance. (b) Mediterranean instance.

Figure D.6.: Convergence of Evolutionary Algorithm in different LINER-LIB instances using the roulette (RS) and binary tournament (BT) selection for the informed cycle crossover (IC) method (averaged over five runs).



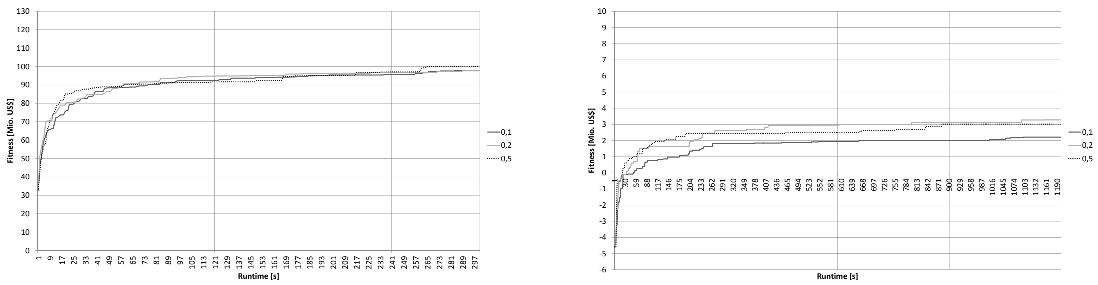
(a) WAF instance. (b) Mediterranean instance.

Figure D.7.: Convergence of Evolutionary Algorithm in different LINER-LIB instances using different population sizes (10, 20, 50 individuals per generation) for the informed cycle crossover method with a binary tournament strategy (averaged over five runs).



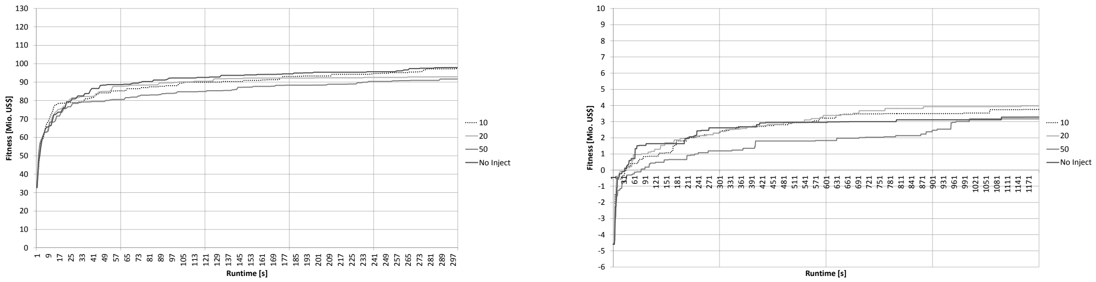
(a) WAF instance. (b) Mediterranean instance.

Figure D.8.: Convergence of Evolutionary Algorithm in the WAF instance using different initial pendulum percentage (10%, 50%, 100%) for the informed cycle crossover method with a binary tournament strategy and a population size of 50 (averaged over five runs).



(a) WAF instance. (b) Mediterranean instance.

Figure D.9.: Convergence of Evolutionary Algorithm in the WAF instance using different elitism ranges (10%, 20%, 50%) for the informed cycle crossover method with a binary tournament strategy, a population size of 50 and 10% pendulum services in the initial population (averaged over five runs).



(a) WAF instance.

(b) Mediterranean instance.

Figure D.10.: Convergence of Evolutionary Algorithm in different LINER-LIB instances injecting new networks after different iterations for the informed cycle crossover method with a binary tournament strategy, a population size of 50, 10% pendulum services in the initial population and an elitism of 10% (averaged over five runs).

D.3. Accuracy of VNS Surrogate Evaluation

Configuration/ operator	\sum operations	correct decisions [%]	wrong decisions [%]
CF 100%, Q*R			
Insert Port	4761	82,15%	17,85%
Change Vessel Type	3122	95,32%	4,68%
Change Vessel Count	3012	91,17%	8,83%
Change Single Port	5766	84,91%	15,09%
Change Port Sequence	3996	87,41%	12,59%
Change Liner Service	3507	90,08%	9,92%
2-opt	2743	93,58%	6,42%
sum/average	**26907**	**88,26%**	**11,74%**
CF 75%, Q*R			
Insert Port	5305	77,49%	22,51%
Change Vessel Type	3485	92,14%	7,86%
Change Vessel Count	3409	91,23%	8,77%
Change Single Port	6488	80,58%	19,42%
Change Port Sequence	4448	84,73%	15,27%
Change Liner Service	3956	87,26%	12,74%
2-opt	3086	88,72%	11,28%
sum/average	**30177**	**84,90%**	**15,10%**
CF 50%, Q*R			
Insert Port	6074	76,84%	23,16%
Change Vessel Type	4018	97,14%	2,86%
Change Vessel Count	3966	83,46%	16,54%
Change Single Port	7364	81,38%	18,62%
Change Port Sequence	5208	83,99%	16,01%
Change Liner Service	4552	86,49%	13,51%
2-opt	3279	90,58%	9,42%
sum/average	**34461**	**84,60%**	**15,40%**

Table D.1.: Accuracy of surrogate evaluation in the variable neighborhood descent solving the Mediterranean LINER-LIB instance. The column generation method is aborted after the first iteration and 40% (8) support points for the bunker cost discretization is used. Data is recorded in five runs of the variable neighborhood descent. The cargo flows are selected using the $Q * R$ strategy.

Configuration/ operator	\sum operations	correct decisions [%]	wrong decisions [%]
CF 100%, cum. Q*R			
Insert Port	4549	82,48%	17,52%
Change Vessel Type	3013	96,48%	3,52%
Change Vessel Count	2968	86,22%	13,78%
Change Single Port	5567	83,62%	16,38%
Change Port Sequence	3859	86,60%	13,40%
Change Liner Service	3434	84,97%	15,03%
2-opt	2579	91,82%	8,18%
sum/average	**25969**	**86,65%**	**13,35%**
CF 75%, cum. Q*R			
Insert Port	6760	75,00%	25,00%
Change Vessel Type	4555	96,90%	3,10%
Change Vessel Count	4501	87,20%	12,80%
Change Single Port	8074	79,96%	20,04%
Change Port Sequence	5932	82,82%	17,18%
Change Liner Service	5177	85,34%	14,66%
2-opt	3990	89,00%	11,00%
sum/average	**38989**	**83,99%**	**16,01%**
CF 50%, cum. Q*R			
Insert Port	8757	76,09%	23,91%
Change Vessel Type	6237	95,05%	4,95%
Change Vessel Count	6069	87,58%	12,42%
Change Single Port	10538	77,59%	22,41%
Change Port Sequence	8040	84,08%	15,92%
Change Liner Service	7163	85,44%	14,56%
2-opt	5510	86,26%	13,74%
sum/average	**52314**	**83,56%**	**16,44%**

Table D.2.: Accuracy of surrogate evaluation in the variable neighborhood descent solving the Mediterranean LINER-LIB instance. The column generation method is aborted after the first iteration and 40% (8) support points for the bunker cost discretization is used. Data is recorded in five runs of the variable neighborhood descent. The cargo flows are selected using the $cum.Q * R$ strategy.

Appendix E.

Calculating Waypoints and Sea Distances

To visualize liner networks with services on a website, using libraries such as the open source software OpenLayers[1], the path between two ports must be calculated. To the best of the author's knowledge, no free software exists that can draw the route between two ports by using seaways. This section describes a method that can be implemented easily and helps to visualize liner services.

The basic idea of the approach works as follows (see Figure E.1 for a graphical overview of the process): The network is initialized and the polygon data for the land masses loaded[2] (see E.1(b)). Afterwards, a graph is constructed and mapped to the polygon data, see E.1(c). The nodes that would lie on land are removed from the graph. In the next step, the nodes are connected with each neighbor horizontally, vertically and diagonal, see E.1(d). Afterwards, users can add further connection between nodes. This can be useful to add small canals to the graph or compensate inaccurate land polygons. The next step is to add cuts that remove connections, for example on headland (see E.1(f)). Now, the ports are added to the graph by inserting additional nodes and connecting them with edges to the closest two nodes. Now, either a shortest path algorithm between all port nodes (for example the Dijkstra algorithm, see Dijkstra (1959)), or a more specialized all-pairs shortest path (such as the Floyd-Warshall, see Floyd (1962)) can be used to calculate the distance and the way points. After this step, the unused nodes can be removed from the graph. The remaining edges are reduced to decrease the number of way points that have to be stored (see Figure E.1(i) and E.1(j)). Finally, the paths and distances can be smoothed and stored for later use (see E.1(k) and E.1(l)).

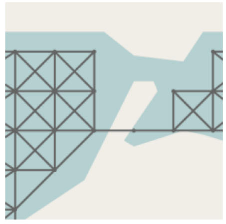
[1] Available under `http://openlayers.org/`

[2] Polygon data is available in different level of detail under `http://openstreetmapdata.com/data/land-polygons`

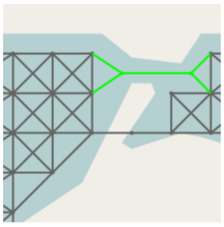
(a) Initial state.

(b) Loaded polygon data for land masses.

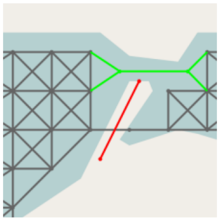
(c) Created network nodes for sea areas.



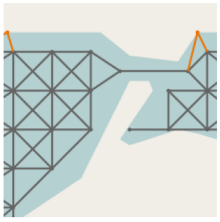
(d) Create connected graph.



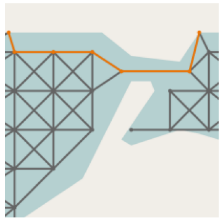
(e) Add connections between graph components.



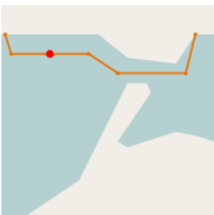
(f) Add cuts to the graph.



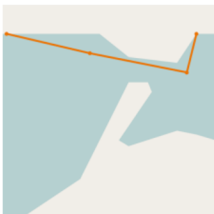
(g) Connect ports to closest nodes.



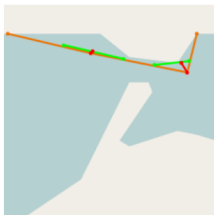
(h) All-pairs shortest paths for ports.



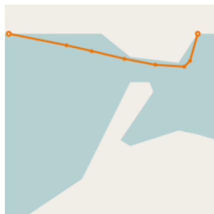
(i) Remove unused nodes and edges and prune graph on horizontal and vertical edges.



(j) Further path reduction.



(k) Optionally path smoothing.



(l) Store final way points between ports.

Figure E.1.: Visual process to calculate way points between ports for the user interface.